

# AMV11: ASP.NET MVC 3 in 75 Minutes

Paul Litwin

Fred Hutchinson Cancer Research Center /  
Deep Training

[paul@deeptesting.com](mailto:paul@deeptesting.com)

[twitter.com/plitwin](https://twitter.com/plitwin)



# Paul Litwin

- **Developer**

- Focus: ASP.NET, ASP, C#, SQL Server, Reporting Services
- MCSD
- Microsoft MVP
- Programmer Manger with Fred Hutchinson Cancer Research Center (Seattle)

- **Co-Founder and Senior Trainer**

- Deep Training
  - [www.deeptraining.com](http://www.deeptraining.com)

- **Conference Chair/Speaker**

- Chair, Microsoft ASP.NET Connections
- Member INETA Speakers Bureau

- **Author of over a dozen books & courses, including...**

- *Agile ASP.NET Unleashed (writing...)*
- *AppDev SQL Server 2005 & 2008 Reporting Services Courses*
- *ASP.NET for Developers*
- *Access Cookbook*
- *Access 2002 Desktop/Enterprise Dev Handbook*



# OpenSpaces at DevConnections

- Free-form “sessions”
- Come to present, discuss, or listen
- Anyone can lead a session
  
- Tonight from 7:45-9:45
- Free Beer & Pizza
- Sponsored by Microsoft Community

# Slides & Samples Download

- You can download them from:
  - [www.deeptraining.com/litwin](http://www.deeptraining.com/litwin)

# ASP.NET MVC 3 in 75 Minutes Agenda

- **ASP.NET MVC Defined**
- MVC 3 Tools Update
- Building a Database-First MVC App
- Validation
- Where to Go From Here

# MVC Design Pattern

- Conery, Hanselman, Haack, Guthrie define it as...
  - An architectural pattern used to separate an application into three main aspects
    - **Model**. Set of classes that represent data and business rules for how data can be changed and manipulated
    - **View**. Application's user interface
    - **Controller**. Set of classes that handles communication from user, overall application flow, and application-specific logic

# What's Wrong with Web Forms?

- Nothing...but...
  - Web forms don't support as precise control over "separation of concerns"
    - Too much of an application ends up in the "code behind"
  - High level of abstraction
    - Not as "close to the metal" as some devs want
  - Web forms are not as well suited for testing
    - Testing the "code behind" code is difficult
  - Search engine optimization (SEO) issues
    - ASP.NET 4.0 addresses this

# What Web Forms & ASP.NET MVC Have in Common

- Both use Visual Studio
- Both can use .aspx pages
  - Both can use Master pages and user controls
  - though you can use a different view engine if you'd like in ASP.NET MVC (*Razor*, *NHaml*, *Spark*, *Brail*, *NVelocity*, etc.)
- Both can use any data access framework (ADO.NET, LINQ, Entity Framework, etc.)
- Both built on ASP.NET
  - ASP.NET runtime, localization, HTML encoding, ...
- Both run on IIS

# What's New in MVC 3

- Razor View Engine
- New ViewBag Property
- New ActionResult Types
- Global ActionFilters
- More HtmlHelpers
- JavaScript / Ajax Improvements
- Validation Improvements
- NuGet Integration

# Highlights of New Stuff 1

- Razor View Engine Option
  - Clean, concise language
- Web Forms (ASPX) View Engine still there
  - Which view engine should you choose?
    - ASPX may be most natural
    - Razor is being pushed hard by Microsoft

# Demo

## Razor vs. ASPX Views

# Highlights of New Stuff 2

- **JavaScript / Ajax Improvements**
  - New jQuery-template-based “Unobtrusive JavaScript”
- **Validation Improvements**
  - Out-of-box support for client-side validation
  - Support for remote validation

# Highlights of New Stuff 3

- **NuGet Packer Manager**
  - Easy way to install add-ins and additional functionality to projects

# ASP.NET MVC 3 in 75 Minutes Agenda

- ASP.NET MVC Defined
- **MVC 3 Tools Update**
- **Building a Database-First MVC App**
- **Validation**
- **Where to Go From Here**

# ASP.NET MVC 3 Tools Update

- This update improves Visual Studio's support for MVC 3
- Add Controller dialog supports better scaffolding
- New Intranet Project template (Windows Authorization)
- New jQuery, Modernizr libraries
- NuGet, Entity Framework 4.1

# Getting Current with VS 2010

- Purchase & install VS 2010
  - You can use free VS Web Developer Express
- Install VS Service Pack 1
- Install ASP.NET MVC3 Tools Update
- Install Web Standards Update for VS
- Install NuGet 1.5

See [weblogs.asp.net/paullitwin](http://weblogs.asp.net/paullitwin) for links and details

# ASP.NET MVC 3 in 75 Minutes Agenda

- ASP.NET MVC Defined
- MVC 3 Tools Update
- **Building a Database-First MVC App**
- **Validation**
- **Where to Go From Here**

# Developing Data-Centric Apps using MVC & EF 4.1

- **Database-First**
  - You create a db schema in SQL Server or other db
  - You generate model from db
- **Model-First**
  - You design model using ORM designer first
  - You generate db from model
- **Code-First**
  - You create classes
  - You generate model & db from classes

EF 4.1 Database can be SQL Server or other provider  
(MySQL, Oracle, etc.)

# Database-First Development using MVC 3

- Supported since EF 1
- Steps
  - Create database
  - Add ADO.NET Entity Data Model to Models
  - Select “Generate from database”
  - EF creates model to match database
- New for EF 4.1
  - Simplified DbContext means less code; easier to write code against
  - You can use DbContext irrespective of whether you are doing Code-First development
  - Requires “Add Code Generation Item...”

# Steps to Build Database-First MVC 3 App

- I. Create EF model from db
- II. Add controller using EF scaffolding
- III. VS automatically creates supporting views

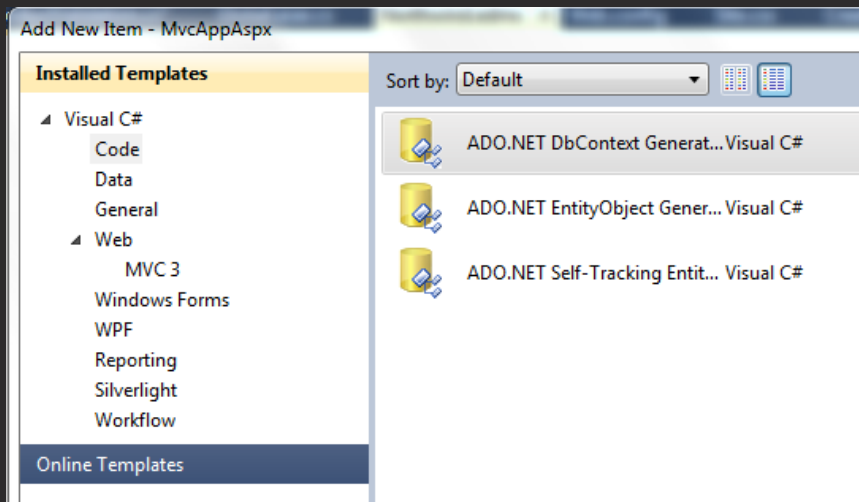
# I. Creating Model

- Step 1
  - Add Entity Framework model to Models folder
    - From Add New Item dialog, select **Data|ADO.NET Entity Data Model**
    - Select **Generate from database**

# I. Creating Model

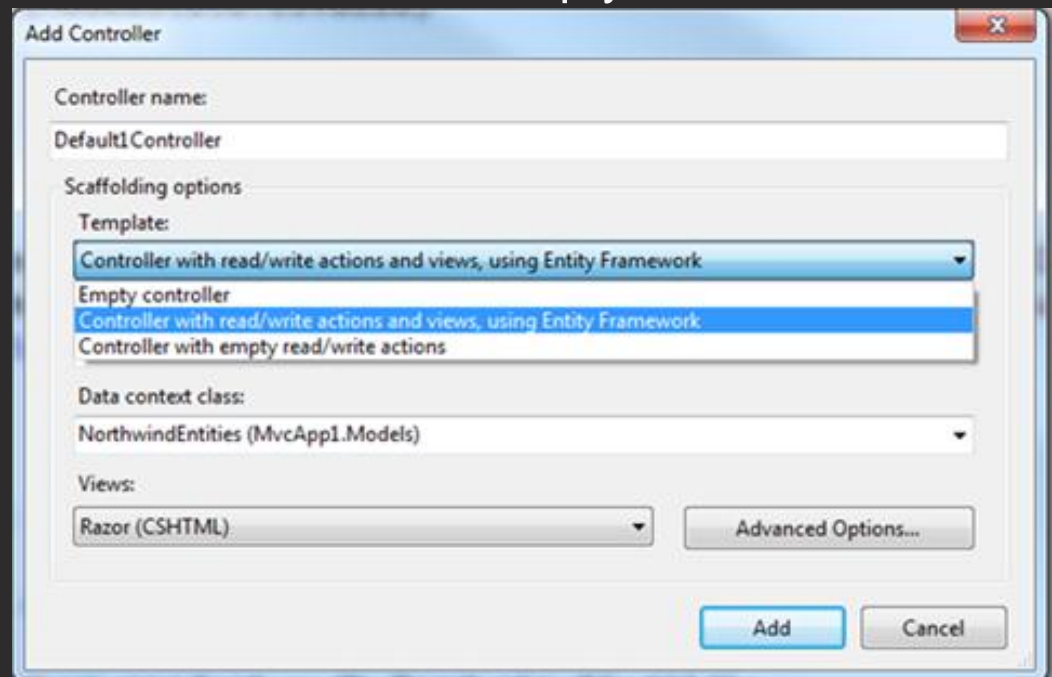
- Step 2

- Change EF code generation to **DbContext** using **Add Code Generation Item...** pop-up menu



## II. Creating Controller

- Right-click on Controllers folder and select **Add|Controller...**
  - Scaffolding options
    - **Empty Controller** – exactly as it sounds (“real programmer” option)
    - **Controller with r/w actions using EF** – new for Tools Update
    - **Controller with empty r/w actions** – creates empty skeleton
  - Model class
    - **EF entity class**
  - Data context class
    - **EF data context**
  - Views
    - **Razor or ASPX**



# III. Creating Views

- If you chose to scaffold controller using “Controller with r/w actions using EF” template, then VS will scaffold CRUD views
- Otherwise, you can individually scaffold out each view by right-clicking on controller method and selecting **Add View**
- Or you can build each view from scratch (“*real programmer*” option)

# Improving the Model

- Add repository classes to insulate controller from model
- Another option is to use better scaffolding tool that will generate the repository classes for you
  - Check out my *Lightning Development with MVC Scaffolding* talk later today at 3:45 PM

# Adding Repository Classes to Model

- Add CRUD methods to repository class that call the model
- Controller now calls the repository class instead of directly calling the model

# ASP.NET MVC 3 in 75 Minutes Agenda

- ASP.NET MVC Defined
- MVC 3 Tools Update
- Building a Database-First MVC App
- **Validation**
- **Where to Go From Here**

# Validation

- Add as model attributes
  - Add validation using attributes in partial class

```
[MetadataType(typeof(Product_Validation))]  
public partial class Product  
{  
}  
  
public class Product_Validation  
{  
    [Required(ErrorMessage = "Product Name required.")]  
    [StringLength(40,  
        ErrorMessage = "Must be 40 characters or less.")]  
    public string ProductName { get; set; }  
    ...  
}
```

# Validation

- Errors flow through to view
  - Displayed using ValidationMessageFor helper

```
<div class="editor-field">  
  @Html.EditorFor(model => model.ProductName)  
  @Html.ValidationMessageFor(model => model.ProductName)  
</div>
```

# What if You Need More than Attribute-based Validation Provides?

- Have class support `IValidatableObject`
- Good news is you can use both

```
public partial class Participant : IValidatableObject
{
    public IEnumerable<ValidationResult> Validate(ValidationContext vc)
    {
        if (CellPhone == null && DaytimePhone == null &&
            EveningPhone == null)
        {
            yield return new ValidationResult(
                "Please supply at least one valid phone number.",
                new string[] { "DaytimePhone" }
            );
        }
    }
}
```

# You Can Also Validate in Controller

- Use `ModelState.AddModelError` method

```
public ActionResult Lookup(FormCollection formValues)
{
    int? id = participantRepository.FindByEmail(User.Identity.Name);

    if (id == null)
    {
        ModelState.AddModelError("EmailAddress", "Email not found. ");
        return View("Lookup");
    }
}
```

# Remote Validation

- Part 1

- Add remote validation attribute

```
Remote("ValidateProductName", "Home",  
    AdditionalFields = "ProductID",  
    ErrorMessage = "A product with this name  
    already exists. Product names must be unique.")]  
public string ProductName { get; set; }
```

- Points to controller method that returns JsonResult

# Remote Validation

- Part 2
  - Create controller method

```
public ActionResult ValidateProductName(string
ProductName, int? ProductID)
{
    return Json(
        pr.IsProductNameUnique(ProductName,
        ProductID), JsonRequestBehavior.AllowGet);
}
```

- Calls method back in model that returns bool

# Remote Validation

- **Part 3** (this would be in controller if not using repository)
  - Repository method that validates to true/false

```
public bool IsProductNameUnique(string ProductName, int? ProductID)
{
    if (ProductID != null)
        return (db.Products.Where(p => p.ProductName == ProductName &&
            p.ProductID != ProductID).Count() == 0);
    else
        return (db.Products.Where(p => p.ProductName ==
            ProductName).Count() == 0);
}
```

- Needs to handle both inserts and edits

# ASP.NET MVC 3 in 75 Minutes Agenda

- ASP.NET MVC Defined
- MVC 3 Tools Update
- Building a Database-First MVC App
- Validation
- **Where to Go From Here**

# Where to Go From Here?

- **nuget is your friend**
  - MVC Scaffolding package
  - Lots of other cool stuff that can save you an incredible amount of time
  - Scott Hanselman's blog features a ton of nuget packages
- **Inversion of control**
- **Unit tests**

# Slides & Samples Download

- You can download them from:
  - [www.deeptraining.com/litwin](http://www.deeptraining.com/litwin)

# Resources

- <http://asp.net/mvc>
  - Starting place for everything on ASP.NET MVC
- <http://weblogs.asp.net/scottgu/>
  - Msft Genius Dev Leader Scott Guthrie's blog
- <http://haacked.com/>
  - Msft MVC PM Phil Haack's blog
- <http://hanselman.com/blog/>
  - Msft Community Guru Scott Hanselman's blog
- <http://blog.stevensanderson.com/>
  - Msft PM for MVC Scaffolding
- <http://blogs.msdn.com/adonet/>
  - ADO.NET Team blog on EF

# Additional Sessions By Me

- *Lightning Development with MVC Scaffolding*
  - Today at 3:45 PM
- *What's So Funny About Peace, Love, and Server Controls?*
  - Tomorrow at 9:30 AM

# OpenSpaces at DevConnections

- Free-form “sessions”
- Come to present, discuss, or listen
- Anyone can lead a session
  
- Tonight from 7:45-9:45
- Free Beer & Pizza
- Sponsored by Microsoft Community

# Your Feedback is Important

Please fill out a session evaluation form drop it off at the conference registration desk.

Thank you!