ASP.NET MVC 3 Tools Update Release Notes

[Overview 2](#_Toc289976488)

[Installation Notes 2](#_Toc289976489)

[Software Requirements 2](#_Toc289976490)

[Documentation 3](#_Toc289976491)

[Support 3](#_Toc289976492)

[Upgrading an ASP.NET MVC 2 Project to ASP.NET MVC 3 3](#_Toc289976493)

[Changes in ASP.NET MVC 3 Tools Update 5](#_Toc289976494)

["Add Controller" dialog box can now scaffold controllers with views and data access code 5](#_Toc289976495)

[Improvements to the "ASP.NET MVC 3 New Project" Dialog Box 8](#_Toc289976496)

[Project templates now include Modernizr 1.7 9](#_Toc289976497)

[Project templates include updated versions of jQuery, jQuery UI, and jQuery Validation 9](#_Toc289976498)

[Project templates now include ADO.NET Entity Framework 4.1 as a pre-installed NuGet package 9](#_Toc289976499)

[Project templates include JavaScript libraries as pre-installed NuGet packages 10](#_Toc289976500)

[Known Issues 10](#_Toc289976501)

[Disclaimer 12](#_Toc289976502)

# Overview

This document describes the release of ASP.NET MVC 3 Tools Update for Visual Studio 2010.

**Note**   This release is an update to the tools in Visual Studio that support ASP.NET MVC 3 RTM. There are no changes to run-time functionality for ASP.NET MVC 3 in this release; the *System.Web.Mvc.dll* assembly has not been updated.

ASP.NET MVC is a framework that uses the Model-View-Controller (MVC) pattern for developing Web applications. The ASP.NET MVC 3 Tools Update installer includes the following components:

* ASP.NET MVC 3 run-time components.
* ASP.NET MVC 3 Visual Studio 2010 tools.
* ASP.NET Web Pages run-time components.
* ASP.NET Web Pages Visual Studio 2010 tools.
* Microsoft Package Manager for .NET (NuGet) version 1.2.
* A hotfix for Visual Studio 2010 that enables support for Razor syntax.

The full set of release notes for each pre-release version of ASP.NET MVC 3 can be found on the ASP.NET website at the following URL:

<http://www.asp.net/learn/whitepapers/mvc3-release-notes>

# Installation Notes

**Important Note: Please make sure that all Visual Studio instances are closed before installing ASP.NET MVC 3 Tools Update.**

To install ASP.NET MVC 3 Tools Update using the Web Platform Installer (Web PI), visit the following page:

<http://www.microsoft.com/web/gallery/install.aspx?appid=MVC3>

Alternatively, you can download the installer for ASP.NET MVC 3 Tools Update for Visual Studio 2010 from the following page:

<http://go.microsoft.com/fwlink/?LinkID=208140>

ASP.NET MVC 3 Tools Update can be installed and run side-by-side with ASP.NET MVC 2 Tools for Visual Studio 2010. If you already have ASP.NET MVC 3 RTM installed, you can upgrade by running the ASP.NET MVC 3 Tools Update installer. There is no need to uninstall the existing ASP.NET MVC 3 installation.

# Software Requirements

The ASP.NET MVC 3 run-time components require the following software:

* .NET Framework version 4.

ASP.NET MVC 3 Visual Studio 2010 tools require the following software:

* Visual Studio 2010 or Visual Web Developer 2010 Express.

# Documentation

Documentation for ASP.NET MVC is available on the MSDN website at the following URL:

<http://go.microsoft.com/fwlink/?LinkId=205717>

Tutorials and other information about ASP.NET MVC are available on the MVC page of the ASP.NET website at the following URL:

<http://www.asp.net/mvc/>

# Support

This is a fully supported release. Information about getting technical support can be found at the [Microsoft Support website](http://support.microsoft.com/).

Also feel free to post questions about this release to the ASP.NET MVC forum, where members of the ASP.NET community are frequently able to provide informal support:

<http://forums.asp.net/1146.aspx>

# Upgrading an ASP.NET MVC 2 Project to ASP.NET MVC 3

To manually upgrade an existing ASP.NET MVC 2 application to version 3, do the following:

1. Create a new empty ASP.NET MVC 3 project on your computer. This project will contain some files that are required for the upgrade.
2. Copy the following files from the ASP.NET MVC 3 project into the corresponding location of your ASP.NET MVC 2 project. You'll need to update any references to the jQuery library to account for the new filename ( jQuery-1.5.1.js):

* /Views/Web.config
* /packages.config
* /scripts/\*.js
* /Content/themes/\*.\*

1. Copy the *packages* folder in the root of the empty ASP.NET MVC 3 project solution into the root of your solution, which is in the directory where the solution’s .sln file is located.
2. If your ASP.NET MVC 2 project contains any areas, copy the /Views/Web.config file to the Views folder of each area.
3. In both Web.config files in the ASP.NET MVC 2 project, globally search and replace the ASP.NET MVC version. Find the following:

System.Web.Mvc, Version=2.0.0.0

Replace it with the following:

System.Web.Mvc, Version=3.0.0.0

1. In Solution Explorer, delete the reference to System.Web.Mvc (which points to the DLL from version 2), then add a reference to System.Web.Mvc (v3.0.0.0).
2. Add a reference to System.WebPages.dll and System.Web.Helpers.dll. These assemblies are located in the following folders:

* %ProgramFiles%\ Microsoft ASP.NET\ASP.NET MVC 3\Assemblies
* %ProgramFiles%\ Microsoft ASP.NET\ASP.NET Web Pages\v1.0\Assemblies

1. In Solution Explorer, right-click the project name and select Unload Project. Then right-click the project name again and select Edit *ProjectName*.csproj.
2. Locate the ProjectTypeGuids element and replace {F85E285D-A4E0-4152-9332-AB1D724D3325} with {E53F8FEA-EAE0-44A6-8774-FFD645390401}.
3. Save the changes, right-click the project, and then select Reload Project.
4. In the application’s root Web.config file, add the following settings to the assemblies section.

<add assembly="System.Web.WebPages, Version=1.0.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35" />

<add assembly="System.Web.Helpers, Version=1.0.0.0, Culture=neutral, PublicKeyToken=31BF3856AD364E35" />

1. If the project references any third-party libraries that are compiled using ASP.NET MVC 2, add the following highlighted bindingRedirect element to the Web.config file in the application root under the configuration section:

<runtime>

<assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">

<dependentAssembly>

<assemblyIdentity name="System.Web.Mvc"

publicKeyToken="31bf3856ad364e35"/>

<bindingRedirect oldVersion="1.0.0.0-2.0.0.0" newVersion="3.0.0.0"/>

</dependentAssembly>

</assemblyBinding>

</runtime>

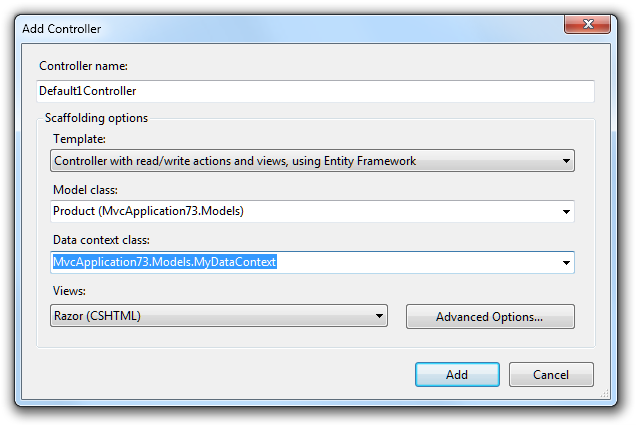
# Changes in ASP.NET MVC 3 Tools Update

This section describes changes made in the ASP.NET MVC 3 Tools Update release since the ASP.NET MVC 3 RTM release.

## "Add Controller" dialog box can now scaffold controllers with views and data access code

Scaffolding is a way of quickly generating a controller and views for your application. After the code has been generated, you can edit it to meet your project’s requirements.

To launch the Add Controller dialog box in ASP.NET MVC 3, right-click the Controllers folder in Solution Explorer , click Add, and then click Controller. The dialog box has been enhanced to offer additional scaffolding options.



### There are three scaffolding templates available by default.

### Empty Controller

This template generates an empty controller file. This template is equivalent to not checking Add actions for create, edit, details, delete scenarios in previous versions of ASP.NET MVC. If you choose this, no further options are available.

### Controller with empty read/write actions

This template generates a controller file that has all the required action methods but no implementation code in the methods. This template is equivalent to checking Add actions for create, edit, details, delete scenarios in previous versions of ASP.NET MVC. If you choose this, no further options are available.

### Controller with read/write actions and views, using Entity Framework

This template enables you to quickly create a working data-entry user interface. It generates code that handles a range of common requirements and scenarios, such as the following:

* *Data access*. The generated code reads and writes entities in a database. It works with the Entity Framework Code First approach if you choose an existing data context class or if you let the template generate a new DbContext class. It also works with the Entity Framework Database First or Model First approach if you choose an existing ObjectContext class.
* *Validation*. The generated code uses ASP.NET MVC model binding and metadata features so that form submissions are validated according to rules declared on your model class. This includes built-in validation rules, such as the Required and StringLength attributes, and custom validation rules.
* *One-to-many relationships*. If you define one-to-many foreign-key relationships between your model classes, the generated code will produce drop-down lists for selecting related entities. For example, you might define the following model classes following Entity Framework Code First conventions:

public class Product

{

public int ProductId { get; set; }

[Required]

public string Name { get; set; }

// Product belongs to Category

public int CategoryId { get; set; }

public virtual Category Category { get; set; }

}

public class Category

{

public int CategoryId { get; set; }

[Required]

public string Name { get; set; }

}

When you then scaffold a controller for the Product class, its views will allow users to choose a Category object for each Product instance.

This template enables additional options in the Add Controller dialog box. For Model class, you can choose any model class in your solution, which determines the type of data that users will be able to create or edit:

* If you want to use Entity Framework Code First, you can choose any model class.
* If you are using Entity Framework Database First or Entity Framework Model First, be sure to choose an entity class defined in your conceptual model.

For Data Context class, you can make these choices:

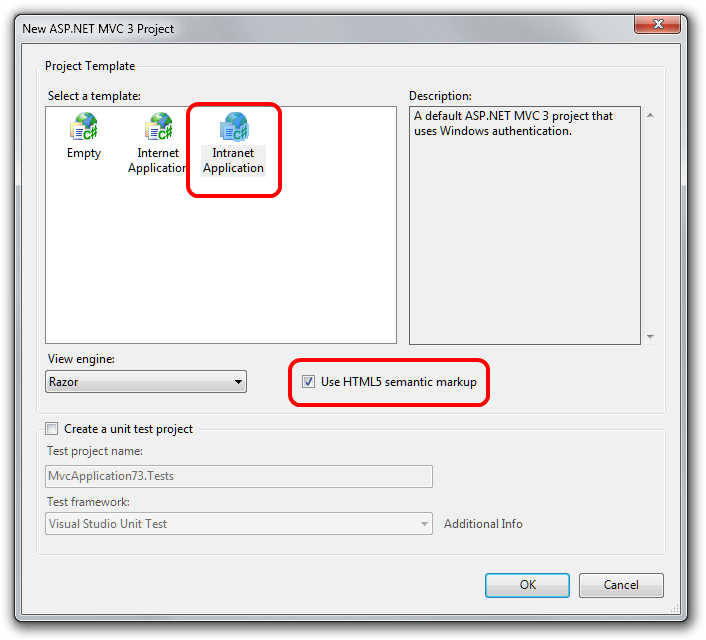
* If you want to use Code First and have no existing data context class, choose <New data context…>”. A data context class will then be generated for you.
* If you want to use Code First and have an existing data context class, choose it here. It will be updated to persist the model class you have selected.
* If you are using Database First or Model First, choose your object context class here.

For Views, choose the view engine you want to use, or choose None if you don't want to scaffold any views.

You can selectAdvanced Optionsto specify further options for the generated views. For example, you can choose the layout or master page to use.

## Improvements to the "ASP.NET MVC 3 New Project" Dialog Box

The dialog box you use to create new ASP.NET MVC 3 projects includes multiple improvements, as listed below.



### New "Intranet Project" Template

The Project Template list includes a new Intranet Application template. This template contains settings for building a web application using Windows authentication instead of forms authentication. Because an intranet application requires some IIS settings that can’t be encapsulated in a project template, the template includes a readme file with instructions for how to make the project template work in IIS. Documentation for the a new Intranet Application template is available on the MSDN website at the following URL:

<http://msdn.microsoft.com/en-us/library/gg703322(VS.98).aspx>

### Project templates are now HTML5 enabled

The new-project dialog box now contains an option to add HTML5-specific features to the project templates. Selecting the option causes views to be generated that contain the new HTML5 <header>, <footer>, and <navigation> elements.

Note that earlier versions of browsers do not support HTML5-specific tags. To address this limitation, the HTML5 project templates include a reference to the Modernizr library. (See the next section.)

## Project templates now include Modernizr 1.7

Modernizr is a JavaScript library that enables support for CSS 3 and HTML5 in browsers that do not yet support these features. This library is included as a pre-installed NuGet package in templates for ASP.NET MVC 3 projects. For more information about Modernizr, see <http://www.modernizr.com/>.

## Project templates include updated versions of jQuery, jQuery UI, and jQuery Validation

The project templates now include the following versions of the jQuery scripts:

* jQuery 1.5.1
* jQuery Validation 1.8
* jQuery UI 1.8.11

These libraries are included as pre-installed NuGet packages.

## Project templates now include ADO.NET Entity Framework 4.1 as a pre-installed NuGet package

The ADO.NET Entity Framework 4.1 includes the Code First feature. Code First is a new development pattern for the ADO.NET Entity Framework that provides an alternative to the existing Database First and Model First patterns.

Code First is focused around defining your model using POCO classes ("plain old CLR objects") written in Visual Basic or C#. These classes can then be mapped to an existing database or be used to generate a database schema. Additional configuration can be supplied using DataAnnotations attributes or using fluent APIs.

Documentation for using Code Firstwith ASP.NET MVC is available on the ASP.NET website at the following URLs:

<http://www.asp.net/mvc/tutorials/getting-started-with-mvc3-part1-cs>  
<http://www.asp.net/entity-framework/tutorials/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>

## Project templates include JavaScript libraries as pre-installed NuGet packages

When you create a new ASP.NET MVC 3 project, the project includes the JavaScript files mentioned previously (for example, the Modernizr library) by installing them using NuGet instead of directly adding the scripts to the Scripts folder in the project template contents. This enables you to use NuGet to update the scripts to the latest version when new versions of the scripts are released.

For example, given the frequency of new jQuery releases, the version of jQuery included in the project template will at some point be out of date. However, because jQuery is included as an installed NuGet package, you will be notified in the NuGet dialog box when newer versions of jQuery are available.

Because jQuery includes the version number in the file name, updating jQuery to the latest version also requires updating the <script> tag that references the jQuery file to use the new file name. Other included script libraries do not include the version number in the script name, so they can be more easily updated to their latest versions.

# Known Issues

* In some cases, installation may fail with the error message “Installation failed with error code (0x80070643)”. For information about how to work around this issue, see [KnowledgeBase article 2531566](http://support.microsoft.com/kb/2531566).
* The scaffolding for adding a controller does not scaffold entities that take advantage of entity inheritance support within Entity Framework. For example, given a base *Person* class that's inherited by a *Student* class, scaffolding the *Student* class will result in generated code that does not compile.
* Creating a new ASP.NET MVC 3 project within a solution folder causes a NullReferenceException error. The workaround is to create the ASP.NET MVC 3 project in the root of the solution and then move it into the solution folder.
* IntelliSense for Razor syntax does not work when ReSharper is installed. If you have ReSharper installed and want to take advantage of the Razor IntelliSense support in ASP.NET MVC 3, see the entry [Razor Intellisense and ReSharper](http://blogs.jetbrains.com/dotnet/2010/11/razor-intellisense-and-resharper/) on Hadi Hariri's blog, which discusses ways to use them together today.
* During installation, the EULA acceptance dialog box displays the license terms in a window that is smaller than intended.
* When you are editing a Razor view (.cshtml or .*vbhtml* file), selecting a code snippet for ASP.NET MVC will show snippets for .aspx views. ASP.NET MVC 3 does not include any snippets for Razor views.
* If you install ASP.NET MVC 3 for Visual Web Developer Express on a computer where Visual Studio is not installed, and then later install Visual Studio, you must reinstall ASP.NET MVC 3. Visual Studio and Visual Web Developer Express share components that are upgraded by the ASP.NET MVC 3 installer. The same issue applies if you install ASP.NET MVC 3 for Visual Studio on a computer that does not have Visual Web Developer Express and then later install Visual Web Developer Express.

Disclaimer

© 2011 Microsoft Corporation. All rights reserved. This document is provided "as-is." Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.