EDUSUM

#1 Online Certification Guide

# 70-486

MCSA Web Applications

A Success Guide to Prepare-
Developing ASP.NET MVC Web Applications

edusum.com

## Table of Contents

Introduction to 70-486 Exam on Developing ASP.NET MVC Web Applications .. 2
Microsoft 70-486 Certification Details: .... 2
Microsoft 70-486 Exam Syllabus: .... 3
70-486 Sample Questions: .... 6
Answers to 70-486 Exam Questions: .... 8

# Introduction to 70-486 Exam on Developing ASP.NET MVC Web Applications

Use this quick start guide to collect all the information about Microsoft Developing ASP.NET MVC Web Applications (70-486) Certification exam. This study guide provides a list of objectives and resources that will help you prepare for items on the 70-486 Developing ASP.NET MVC Web Applications exam. The Sample Questions will help you identify the type and difficulty level of the questions and the Practice Exams will make you familiar with the format and environment of an exam. You should refer this guide carefully before attempting your actual Microsoft MCSA Web Applications certification exam.

The Microsoft Developing ASP.NET MVC Web Applications certification is mainly targeted to those candidates who want to build their career in Microsoft Visual Studio domain. The Microsoft Certified Solutions Associate (MCSA) - Web Applications exam verifies that the candidate possesses the fundamental knowledge and proven skills in the area of Microsoft MCSA Web Applications.

## Microsoft 70-486 Certification Details:

| | |
|---|---|
| Exam Name | Microsoft Certified Solutions Associate (MCSA) - Web Applications |
| Exam Code | 70-486 |
| Exam Price | $165 (USD) |
| Duration | 120 min |
| Number of Questions | 45-55 |
| Passing Score | 700 / 1000 |
| Books / Training | 20486B |
| Schedule Exam | Pearson VUE |
| Sample Questions | Microsoft Developing ASP.NET MVC Web Applications Sample Questions |
| Practice Exam | **Microsoft 70-486 Certification Practice Exam** |

# Microsoft 70-486 Exam Syllabus:

| Topic | Details | Weights |
|---|---|---|
| Design the application architecture | Plan the application layers<br>- Plan data access; plan for separation of concerns; appropriate use of models, views, and controllers; choose between client-side and server side processing; design for scalability<br><br>Design a distributed application<br>- Design a hybrid application (on-premises versus off-premises, including Azure), plan for session management in a distributed environment, plan web farms<br><br>Design and implement the Azure role life cycle<br>- Identify and implement Start, Run, and Stop events; identify startup tasks (IIS configuration [app pool], registry configuration, third-party tools)<br><br>Configure state management<br>- Choose a state management mechanism (in-process and out of process state management), plan for scalability, use cookies or local storage to maintain state, apply configuration settings in web.config file, implement sessionless state (for example, QueryString)<br><br>Design a caching strategy<br>- Implement page output caching (performance oriented), implement data caching, implement HTTP caching, implement Azure caching<br><br>Design and implement a WebSocket strategy<br>- Read and write string and binary data asynchronously (long-running data transfers), choose a connection loss strategy, decide a strategy for when to use WebSockets, implement SignalR<br><br>Design HTTP modules and handlers<br>- Implement synchronous and asynchronous modules and handlers, choose between modules and handlers in IIS | 15-20% |
| Design the user experience | Apply the user interface design for a web application<br>- Create and apply styles by using CSS, structure and lay out the user interface by using HTML, implement dynamic page content based on a design<br><br>Design and implement UI behavior<br>- Implement client validation, use JavaScript and the | 20-25% |

| Topic | Details | Weights |
|---|---|---|
| | DOM to control application behavior, extend objects by using prototypal inheritance, use AJAX to make partial page updates, implement the UI by using JQuery<br><br>Compose the UI layout of an application<br>- Implement partials for reuse in different areas of the application, design and implement pages by using Razor templates (Razor view engine), design layouts to provide visual structure, implement master/application pages<br><br>Enhance application behavior and style based on browser feature detection<br>- Detect browser features and capabilities; create a web application that runs across multiple browsers and mobile devices; enhance application behavior and style by using vendor-specific extensions, for example, CSS<br><br>Plan an adaptive UI layout<br>- Plan for running applications in browsers on multiple devices (screen resolution, CSS, HTML), plan for mobile web applications | |
| Develop the user experience | Plan for search engine optimization and accessibility<br>- Use analytical tools to parse HTML, view and evaluate conceptual structure by using plugs-in for browsers, write semantic markup (HTML5 and ARIA) for accessibility (for example, screen readers)<br><br>Plan and implement globalization and localization<br>- Plan a localization strategy; create and apply resources to UI, including JavaScript resources; set cultures; create satellite resource assemblies<br><br>Design and implement MVC controllers and actions<br>- Apply authorization attributes, global filters, and authentication filters; specify an override filter; implement action behaviors; implement action results; implement model binding<br><br>Design and implement routes<br>- Define a route to handle a URL pattern, apply route constraints, ignore URL patterns, add custom route parameters, define areas<br><br>Control application behavior by using MVC extensibility points<br>- Implement MVC filters and controller factories; control | 15-20% |

| Topic | Details | Weights |
|---|---|---|
| | application behavior by using action results, viewengines, model binders, and route handlers<br><br>Reduce network bandwidth<br>- Bundle and minify scripts (CSS and JavaScript), compress and decompress data (using gzip/deflate; storage), plan a content delivery network (CDN) strategy (for example, Azure CDN) | |
| Troubleshoot and debug web applications | Prevent and troubleshoot runtime issues<br>- Troubleshoot performance, security, and errors; implement tracing, logging (including using attributes for logging), and debugging (including IntelliTrace); enforce conditions by using code contracts; enable and configure health monitoring (including Performance Monitor)<br><br>Design an exception handling strategy<br>- Handle exceptions across multiple layers, display custom error pages using global.asax or creating your own HTTPHandler or set web.config attributes, handle first chance exceptions<br><br>Test a web application<br>- Create and run unit tests (for example, use the Assert class), create mocks; create and run web tests, including using Browser Link; debug a web application in multiple browsers and mobile emulators<br><br>Debug an Azure application<br>- Collect diagnostic information by using Azure Diagnostics API and appropriately implement on demand versus scheduled; choose log types (for example, event logs, performance counters, and crash dumps); debug an Azure application by using IntelliTrace, Remote Desktop Protocol (RDP), and remote debugging; interact directly with remote Azure websites using Server Explorer. | 20-25% |
| Design and implement security | Configure authentication<br>- Authenticate users; enforce authentication settings; choose between Windows, Forms, and custom authentication; manage user session by using cookies; configure membership providers; create custom membership providers; configure ASP.NET Identity<br><br>Configure and apply authorization<br>- Create roles, authorize roles by using configuration, | 20-25% |

| Topic | Details | Weights |
|---|---|---|
| | authorize roles programmatically, create custom role providers, implement WCF service authorization<br><br>Design and implement claims-based authentication across federated identity stores<br>- Implement federated authentication by using Azure Access Control Service; create a custom security token by using Windows Identity Foundation; handle token formats (for example, oAuth, OpenID, Microsoft Account, Google, Twitter, and Facebook) for SAML and SWT tokens<br><br>Manage data integrity<br>- Apply encryption to application data, apply encryption to the configuration sections of an application, sign application data to prevent tampering<br><br>Implement a secure site with ASP.NET<br>- Secure communication by applying SSL certificates; salt and hash passwords for storage; use HTML encoding to prevent cross-site scripting attacks (ANTI-XSS Library); implement deferred validation and handle unvalidated requests, for example, form, querystring, and URL; prevent SQL injection attacks by parameterizing queries; prevent cross-site request forgeries (XSRF) | |

## 70-486 Sample Questions:

**01. Why should you create a custom role provider?**
**a)** To use a data source not regularly supported
**b)** To use the SimpleRoleProvider
**c)** To use a database design different than .NET provides
**d)** To provide a special configuration file entry

**02. What type of authentication accepts login credentials that will be checked against the domain or local server and are sent in a hashed format?**
**a)** Basic authentication
**b)** Digest authentication
**c)** Forms authentication
**d)** Windows authentication

**03. How could you traditionally consume an ASMX web service from your application?**
**a)** Generate a proxy by selecting Add Reference In Visual Studio.
**b)** Create an HttpService and connect using Get(URL).
**c)** Generate a proxy by selecting Add A Service Reference in Visual Studio.
**d)** Create a WCF proxy class.

**04. What kind of helper methods does WebSecurity provide?**
**a)** Login
**b)** ResetPassword
**c)** CreateAccount
**d)** ChangePassword
**e)** DeleteAccount

**05. What do you need to do to use IntelliTrace from within Windows Azure?**
**a)** Publish the solution from any version of Visual Studio Professional 2012 or higher.
**b)** Select the Enable IntelliTrace check box before publishing the solution.
**c)** Ensure that you made all configuration changes in the Web.config file that will turn on IntelliTrace.
**d)** Download and view the IntelliTrace logs through a web browser.
**e)** Download and view the IntelliTrace logs through Visual Studio Ultimate 2012.

**06. How do you encrypt the <connectionStrings> section of the Web.config file?**
**a)** aspnet_regiis -pe "ConnectionStrings" -app "/MachineDPAPI" -prov "RsaProtectedConfigurationProvider"
**b)** aspnet_regiis -pe "Web.Config"-app "/MachineDPAPI" -prov "RsaProtectedConfigurationProvider"
**c)** aspnet_regiis -pd "ConnectionStrings" -app "/MachineDPAPI" -prov "RsaProtectedConfigurationProvider"
**d)** aspnet_regiis -pd "Web.Config" -app "/MachineDPAPI" -prov "RsaProtectedConfigurationProvider"

**07. What are common methods for detecting the type of browser running on a client?**
**a)** Use JavaScript to query for the userAgent header.
**b)** Use the window.addEventListener method.
**c)** Use the viewport <meta> tag.
**d)** Use the DisplayMode provider.

**08. As you design a sessionless state management system, what do you need to ensure that your application manages?**
**a)** Access to the state management system, whether it is a database, a web service, or other type of system
**b)** The HTTP headers
**c)** The session setting within the Web.config file
**d)** An identifier used by the server to identify the request

**09. What is the technique in which the client sends a request to the server, and the server holds the response until it either times out or has information to send to the client is?**
**a)** HTTP polling
**b)** HTTP long polling
**c)** WebSockets
**d)** HTTP request-response

**10. What are the primary differences between the AntiXSS Library and the default .NET Framework?**
**a)** The AntiXSS Library takes a blocked-list approach, whereas the .NET Framework takes an accepted-list approach.
**b)** The AntiXSS Library has be modified to realize performance gains.
**c)** The AntiXSS Library takes an accepted-list approach, whereas the .NET Framework takes a blocked-list approach.
**d)** The AntiXSS Library offers enhanced globalization capabilities.

## Answers to 70-486 Exam Questions:

| | | | | |
|---|---|---|---|---|
| Question: 01<br>Answer: a, c | Question: 02<br>Answer: b | Question: 03<br>Answer: b, c | Question: 04<br>Answer: a, b, c, d | Question: 05<br>Answer: b, e |
| Question: 06<br>Answer: a | Question: 07<br>Answer: a, d | Question: 08<br>Answer: a, d | Question: 09<br>Answer: b | Question: 10<br>Answer: b, c, d |

Note: If you find any typo or data entry error in these sample questions, we request you to update us by commenting on this page or write an email on feedback@edusum.com