Build a Classic ASP Website on IIS

Published: February 29, 2012

Updated: February 29, 2012

Applies To: Windows Server 2012

This document guides you through installing IIS and configuring a classic ASP website. Classic ASP is a server-side scripting environment that you can use to create and run dynamic web applications. With ASP, you can combine HTML pages, script commands, and COM components to create interactive web pages that are easy to develop and modify. Classic ASP is the predecessor to ASP.NET, but it is still in wide use today.

The Classic ASP server configuration adds IIS modules for ASP and ISAPI extensions to the default IIS installation.

In this document

Prerequisites

This guide was written for, and tested on, the following operating systems:

1. Windows Server® 2012
2. Windows® 8

Step 1: Install the IIS Web Server

You can use the Web Platform Installer (Web PI) to install IIS, and applications that run on IIS. Because the Web PI installs the latest versions of available Web Platform offerings, with just a few simple clicks you can download and install any new tools or updates. To learn more about the Web PI, see Learn more and install the Web PI.

You can also perform this procedure by using the Windows user interface (UI) or from a command line.

To install IIS on Windows Server 2012 by using the UI

1. On the **Start** page, click the **Server Manager** tile, and then click **OK**.
2. In **Server Manager**, select **Dashboard**, and click **Add roles and features**.
3. In the **Add Roles and Features Wizard**, on the **Before You Begin** page, click **Next**.
4. On the **Select Installation Type** page, select **Role-based or Feature-based Installation** and click **Next**
5. On the **Select Destination Server** page, select **Select a server from the server pool**, select your server, and click **Next**.
6. On the **Select Server Roles** page, select **Web Server (IIS)**, and then click **Next**.
7. On the **Select Features** page, note the preselected features that are installed by default, and then select the following additional role services:
    - ASP
    - ISAPI Extensions
8. Click **Next**.
9. On the **Web Server Role (IIS)** page, click **Next**.
10. On the **Select Role Services** page, note the preselected role services that are installed by default, and then click **Next**.

> **Note**
>
> You only have to install the IIS 8 default role services for a static-content web server.

11. On the **Confirm Installation Selections** page, confirm your selections, and then click **Install**.

12. On the **Installation Progress** page, confirm that your installation of the Web Server (IIS) role and required role services completed successfully, and then click **Close**.

13. To verify that IIS installed successfully, type the following into a web browser:

    http://localhost

    You should see the default IIS Welcome page.

To install IIS on Windows 8 by using the UI

1. On the **Start** page, type **Control Panel**, and then click the **Control Panel** icon in the search results.

2. In **Control Panel**, click **Programs**, and then click **Turn Windows features on or off**.

3. In the **Windows Features** dialog box, click **Internet Information Services**, note the preselected features that are installed by default, and then select the following additional role services:

    - ASP
    - ISAPI Extensions

4. Click **OK**.

5. To verify that IIS installed successfully, type the following into a web browser:

    http://localhost

    You see the default IIS Welcome page.

To Install IIS by using the command line

- Type the following command at a command prompt or into a script:

  Start /w pkgmgr /iu:IIS-WebServerRole;IIS-WebServer;IIS-CommonHttpFeatures;IIS-StaticContent;IIS-DefaultDocument;IIS-DirectoryBrowsing;IIS-HttpErrors;IIS-ApplicationDevelopment;IIS-ASP;IIS-ISAPIExtensions;IIS-HealthAndDiagnostics;IIS-HttpLogging;IIS-LoggingLibraries;IIS-RequestMonitor;IIS-Security;IIS-RequestFiltering;IIS-HttpCompressionStatic;IIS-WebServerManagementTools;IIS-ManagementConsole;WAS-WindowsActivationService;WAS-ProcessModel;WAS-NetFxEnvironment;WAS-ConfigurationAPI

Step 2: Add a Classic ASP Website

You can perform this procedure by using the user interface (UI), by running Appcmd.exe commands in a command-line window, by editing configuration files directly, or by writing WMI scripts.

To add a website by using the UI

1. Open IIS Manager.
   - For Windows Server 2012, on the **Start** page click the **Server Manager** tile, and then click **OK**. On the **Server Manager Dashboard**, click the **Tools** menu, and then click **Internet Information Services (IIS) Manager**.
   - For Windows 8, on the **Start** page type **Control Panel**, and then click the **Control Panel** icon in the search results. On the **Control Panel** screen, click **System and Security**, click **Administrative Tools**, and

then **click Internet Information Services (IIS) Manager**.

2. In the **Connections** pane, right-click the **Sites** node in the tree, and then click **Add Website**.

3. In the **Add Website** dialog box, type a friendly name for your website in the **Site name** box.

4. If you want to select a different application pool than the one listed in the **Application Pool** box, click **Select**. In the **Select Application Pool** dialog box, select an application pool from the **Application Pool** list and then click **OK**.

5. In the **Physical path** box, type the physical path of the Web site's folder, or click the browse button (**...**) to navigate the file system to find the folder.

6. If the physical path that you entered in step 5 is to a remote share, click **Connect as** to specify credentials that have permission to access the path. If you do not use specific credentials, select the **Application user (pass-through authentication)** option in the **Connect As** dialog box.

7. Select the protocol for the Web site from the **Type** list.

8. The default value in the **IP address** box is **All Unassigned**. If you must specify a static IP address for the Web site, type the IP address in the **IP address** box.

9. Type a port number in the **Port** text box.

10. Optionally, type a host header name for the Web site in the **Host Header** box.

11. If you do not have to make any changes to the site, and you want the Web site to be immediately available, select the **Start Web site immediately** check box.

12. Click **OK**.

To add a website by using the command line

- Use the following syntax at the command prompt or in a script:

> **Note**
>
> For this syntax to work, you either must be in the following directory, or have the directory in your path: `%windir%\system32\inetsrv`

appcmd add site /name: *string* **/id:** *uint* **/physicalPath:** *string* **/bindings:** *string*

The variable **name** *string* is the name, and the variable **id** *uint* is the unsigned integer that you want to assign to the site. The variables **name** *string* and **id** *uint* are the only variables that are required when you add a site in Appcmd.exe.

> **Note**
>
> When you add a site without specifying the values for the **bindings** and **physicalPath** attributes, the site will not be able to start.

The variable **physicalPath** *string* is the path of the site content in the file system.

The variable **bindings** *string* contains information that is used to access the site, and it should be in the form of *protocol***/***IP_address***:***port***:***host_header*. For example, a web site binding is the combination of protocol, IP address, port, and host header. A binding of **http/*:85:** enables a web site to listen for HTTP requests on port 85 for all IP addresses and domain names (also known as host headers or host names). On the other hand, a binding of **http/*:85:marketing.contoso.com** enables a web site to listen for HTTP requests on port 85 for all IP addresses and the domain name **marketing.contoso.com**.

To add a web site named **contoso** with an ID of 2 that has content in **c:\contoso**, and that listens for HTTP requests on port 85 for all IP addresses and a domain name of **marketing.contoso.com**, type the following at the command prompt, and then press ENTER:

appcmd add site /name: contoso /id:2 /physicalPath: c:\contoso /bindings:http/*:85:

/physical-path:c:\contoso /bindings:http/*:80:marketing.contoso.com

Step 3: Edit ASP Application Settings

IIS 8 provides default settings for ASP applications, but you can change those settings as needed. For example, you can enable client-side debugging on a test server to aide in troubleshooting issues during a test pass.

To edit ASP application settings by using the UI

1. Open IIS Manager and navigate to the level you want to manage.
2. In **Features View**, double-click **ASP**.
3. On the **ASP** page, edit settings as desired.
4. When finished, click **Apply** in the **Actions** pane.

To edit ASP application settings by using the command line

- **Specify default character set**

  To specify the default character set for an application, use the following syntax:

  appcmd set config /section:asp /codePage: *integerRange*

  The variable *integerRange* is the default character set. For example, to set the code page to a Latin character set used in American English and many European alphabets, type the following at the command prompt, and then press Enter:

  appcmd set config /section:asp /codePage: *1252*

- **Enable or disable buffering**

  To enable or disable buffering of ASP application

output, use the following syntax:

appcmd set config /section:asp /bufferingOn:True|False

True enables buffering whereas **False** disables buffering. The default value is **True**.

- **Enable or disable HTTP 1.1 chunked transfer encoding**

  To enable HTTP 1.1 chunked transfer encoding for the World Wide Web publishing service, use the following syntax:

  appcmd set config /section:asp /enableChunkedEncoding:True|False

  True enables HTTP 1.1 chunked transfer encoding whereas **False** disables HTTP 1.1 chunked transfer encoding. The default value is **True**.

- **Enable or disable HTML fallback**

  To enable or disable HTML fallback, use the following syntax:

  appcmd set config /section:asp /enableASPHTMLFallback:True|False

  True causes an .htm file that has the same name as the requested .asp file, if it exists, to be sent instead of the .asp file if the request is rejected because of a full request queue. The default value is **True**.

- **Enable or disable parent paths**

  To enable or disable paths relative to the current directory or above the current directory, use the following syntax:

  appcmd set config /section:asp /enableParentPaths:True|False

  True sets ASP pages to allow paths relative to the current directory or above the current directory. The default value is **True**.

- **Set client connection test interval**

  To set a time interval after which ASP will check to see whether the client is still connected before executing a request, use the following syntax:

executing a request, use the following syntax:

appcmd set config /section:asp /queueConnectionTestTime: *timeSpan*

The variable *timeSpan* sets the time interval (hh:mm:ss) after which ASP checks to see whether the client is still connected before executing a request. The default value is 00:00:03.

- **Set maximum requesting entity body limit**

  To specify the maximum number of bytes allowed in the entity body of an ASP request, use the following syntax:

  appcmd set config /section:asp /maxRequestEntityAllowed: *int*

  The variable *int* represents the maximum number of bytes allowed in the body of an ASP request. The default value is 200000 bytes.

- **Set request queue length**

  To specify the maximum number of concurrent ASP requests allowed into the queue, use the following syntax:

  appcmd set config /section:asp /requestQueueMax: *int*

  The variable *int* represents the maximum number of concurrent ASP requests that are allowed into the request queue. The default value is 3000.

- **Set request queue time-out**

  To specify the period that an ASP request can wait in the request queue, use the following syntax:

  appcmd set config /section:asp /queueTimeout: *timeSpan*

  The variable *timeSpan* represents the maximum time (hh:mm:ss) that an ASP request can wait in the request queue. The default value is 00:00:00.

- **Specify response buffering limit**

  To control the maximum number of bytes that an ASP page can write to the response buffer before a flush occurs, use the following syntax:

  appcmd set config /section:asp /bufferingLimit:

appcmd set config /section:asp /bufferingLimit: *int*

The variable *int* represents the maximum size, in bytes, of the ASP buffer. The default value is 4194304 bytes.

- **Set script time-out**

  To specify the default length of time that ASP pages let a script run before terminating the script and writing an event to the Windows Event Log, use the following syntax:

  appcmd set config /section:asp /scriptTimeout: *timeSpan*

  The variable *timeSpan* represents the maximum time (hh:mm:ss) that an ASP request can run before an event is written to the Windows Event Log. The default value is 00:01:30.

- **Specify threads per processor limit**

  To specify the maximum number of worker threads per processor that ASP can create, use the following syntax:

  appcmd set config /section:asp /processorThreadMax: *int*

  The variable *int* represents the maximum number of worker threads per processor that ASP can create. The default value is 25.

- **Specify default locale identifier**

  To define how dates, times, and currencies are formatted for an ASP application, use the following syntax:

  appcmd set config /section:asp /lcid: *int*

  The variable *int* represents the default locale identifier for an ASP application. The default value is 0.

- **Enable or disable automatic application restart**

  To enable or disable automatic restart of ASP applications whenever a configuration setting is changed, use the following syntax:

  appcmd set config /section:asp

/enableApplicationRestart:True|False

True enables ASP applications to be automatically restarted whenever a configuration setting is changed. The default value is **True**.

- **Enable or disable line number calculation**

  To enable or disable ASP to calculate and store the line number of each executed line of code to provide the number in an error report, use the following syntax:

  appcmd set config /section:asp /calLineNumber:True|False

  True enables line number calculation and storage. The default value is **True**.

- **Enable or disable COM component exception trapping**

  To enable or disable ASP pages to catch exceptions thrown by COM components, use the following syntax:

  appcmd set config /section:asp /exceptionCatchEnable:True|False

  True enables COM component exception trapping. If set to **False**, the Microsoft Script Debugger tool does not catch exceptions sent by the component that you are debugging. The default value is **True**.

- **Enable or disable client-side debugging**

  To enable or disable client-side debugging, use the following syntax:

  appcmd set config /section:asp /appAllowClientDebug:True|False

  True enables client-side debugging. The default value is **False**.

- **Enable or disable log error requests**

  To enable or disable the writing of ASP errors to the application section of the Windows event log, use the following syntax:

  appcmd set config /section:asp /logErrorRequests:True|False

True enables log error requests. By default, ASP errors are written to the client browser and the IIS logs. The default value is **True**.

- **Enable or disable server-side debugging**

- **Enable or disable Windows event logging of ASP errors**

  To enable or disable ASP debugging on the server, use the following syntax:

  appcmd set config /section:asp /appAllowDebugging:True|False

  True enables server-side debugging for ASP applications. The default value is **False**.

- **Run On End Functions Anonymously**

  To enable or disable **SessionOnEnd** and **ApplicationOnEnd** global ASP functions to run as the anonymous user, use the following syntax:

  appcmd set config /section:asp /runOnEndAnonymously:True|False

  True enables **SessionOnEnd** and **ApplicationOnEnd** global ASP functions to run as the anonymous user. The default value is **True**.

- **Specify script error message**

  To specify the error message to send to the browser if specific debugging errors are not sent to the client, use the following syntax:

  appcmd set config /section:asp /scriptErrorMessage: *string*

  The variable *string* represents the error message that is sent to the browser when specific debugging errors are not sent to the client. The default value is "An error occurred on the server when processing the URL. Please contact the system administrator".

- **Enable or disable sending errors to the browser**

  To enable or disable the writing of debugging specifics (file name, error, line number, and description) to the client browser in addition to logging them to the Windows event log, use the

logging them to the Windows event log, use the following syntax:

appcmd set config /section:asp /scriptErrorSentToBrowser:True|False

True enables the writing of debugging specifics to the client browser. The default value is **False**.

- **Specify default script language**

  To specify the default script language for all ASP applications that are running on the web server, use the following syntax:

  appcmd set config /section:asp /scriptLanguage: *string*

  The variable *string* represents the default script language. The default value is VBScript.

- **Specify cache directory path**

  To specify the name of the directory where ASP stores compiled ASP templates when the in-memory cache overflows, use the following syntax:

  appcmd set config /section:asp /diskTemplateCacheDirectory: *string*

  The variable *string* represents the cache directory path. The default value is *%windir%*\system32\inetsrv\ASP Compiled Templates.

- **Enable or disable type library caching**

  To enable or disable the caching of type libraries, use the following syntax:

  appcmd set config /section:asp /enableTypelibCache:True|False

  True enables the caching of type libraries. The default value is **True**.

- **Set maximum number of compiled ASP templates to store**

  To set the maximum number of compiled ASP templates that can be stored, use the following syntax:

  appcmd set config /section:asp

/maxDiskTemplateCacheFiles: *int*

The variable *int* represents the maximum number of compiled ASP templates to store. The default value is 2000.

- **Set maximum number of compiled ASP templates to store**

  To set the maximum number of precompiled script files to cache, use the following syntax:

  appcmd set config /section:asp /scriptFileCacheSize: *int*

  The variable *int* represents the number of precompiled script files to cache. If set to 0, no script files are cached. If set to 4294967295, all requested script files are cached. The default value is 500.

- **Set maximum number of scripting engines to cache**

  To set the maximum number of scripting engines that ASP pages keep cached in memory, use the following syntax:

  appcmd set config /section:asp /scriptEngineCacheMax: *int*

  The variable *int* represents the maximum number of scripting engines that are cached. The default value is 250.

- **Enable or disable COM+ side-by-side assemblies**

  To enable or disable side-by-side COM+ assemblies, which allow ASP applications to specify which version of a system DLL or classic COM component to use, use the following syntax:

  appcmd set config /section:asp /appServiceFlags:True|False

  True enables COM+ side-by-side assemblies. The default value is **False**.

- **Enable or disable COM+ tracker**

  To enable or disable COM+ tracker, use the following syntax:

  appcmd set config /section:asp

/enableTypelibCache:True|False

True enables COM+ tracker, which lets administrators or developers debug ASP applications. The default value is **False**.

- **Enable or disable multithreaded environments**

  To enable or disable ASP to run in a multithreaded environment, use the following syntax:

  appcmd set config /section:asp /executeInMta:True|False

  True enables ASP to run in a multithreaded environment. The default value is **False**.

- **Enable or disable thread model checking**

  To enable or disable whether IIS checks the threading model of any component that your application creates, use the following syntax:

  appcmd set config /section:asp /trackThreadingModel:True|False

  True enables thread model checking. The default value is **False**.

- **Specify COM+ partition ID**

  To specify the Globally Unique Identifier (GUID) of the COM+ partition, use the following syntax:

  appcmd set config /section:asp /partitionID: *string*

  The variable *string* represents the GUID of the COM+ partition. The default value is 00000000-0000-0000-0000-000000000000.

Note
You must also set the **appServiceFlags** flag to **True**.

- **Specify COM+ application**

  To specify the name of the COM+ application, use the following syntax:

  appcmd set config /section:asp /sxsName: *string*

The variable *string* represents name of the COM+ application.

> **Note**
>
> You must also set the **appSeviceFlags** flag to **True**.

- **Enable or disable COM+ partitioning**

  To enable or disable COM+ partitioning, use the following syntax:

  appcmd set config /section:asp /appServiceFlags :True|False

  True enables COM+ partitioning, which can be used to isolate applications in their own COM+ partition. The default value is **False**.

  > **Note**
  >
  > If set to **True**, you must also set a value for the **partitionID** property.

- **Enable or disable session state**

  To enable or disable session state persistence for an ASP application, use the following syntax:

  appcmd set config /section:asp /allowSessionState:True|False

  True enables session state persistence. The default value is **True**.

- **Set maximum number of concurrent sessions**

  To set the maximum number of concurrent sessions that ASP allows, use the following syntax:

  appcmd set config /section:asp /max: *int*

  The variable *int* represents the maximum number of concurrent sessions. The default value is -1.

- **Enable or disable secure session ID**

  To enable or disable the sending of a session ID as a secure cookie if assigned over a secure session channel, use the following syntax:

appcmd set config /section:asp /keepSessionIdSecure:True|False

True enables secure session ID. The default value is **True**.

- **Set session time-out**

  To specify the default time that a session object is maintained after the last request associated with the object is made, use the following syntax:

  appcmd set config /section:asp /timeout: *timeSpan*

  The variable *timeSpan* represents the maximum time (hh:mm:ss) that a session object is maintained after the last request associated with the object is made. The default value is 00:20:00.

Next Steps

Test your website thoroughly to ensure that it functions as expected. Then consider configuring the following features.

- To help you troubleshoot or optimize the performance of your web server, set up IIS logging. For instructions see, Configure Logging in IIS.
- To improve the security of your web server, configure request filtering. For instructions see, Configure Request Filtering in IIS.

See also

- Hosting-Friendly Web Server Platform (IIS): Scenario Overview
- Build a Static Website on IIS
- Build an ASP.NET Website on IIS
- Build a PHP Website on IIS
- Build an FTP Site on IIS

- Build a Web Farm with IIS Servers

Community Additions

Expand Application Development section

On step #7 under the "To install IIS on Windows Server 2012 by using the UI" section, you have to expand the Application Development item to see the "ASP" and "ISAPI Extensions."

Joe Rodgers
12/21/2012