Session-State Modes

<http://msdn.microsoft.com/en-us/library/ms178586(v=vs.100).aspx>

**.NET Framework 4**

[Other Versions](javascript:;)

26 out of 39 rated this helpful - [Rate this topic](http://msdn.microsoft.com/en-us/library/ms178586(v=vs.100).aspx#feedback)

ASP.NET session state supports several different storage options for session data. Each option is identified by a value in the [SessionStateMode](http://msdn.microsoft.com/en-us/library/system.web.sessionstate.sessionstatemode(v=vs.100).aspx)enumeration. The following list describes the available session state modes:

* **InProc** mode, which stores session state in memory on the Web server. This is the default.
* **StateServer** mode, which stores session state in a separate process called the ASP.NET state service. This ensures that session state is preserved if the Web application is restarted and also makes session state available to multiple Web servers in a Web farm.
* **SQLServer** mode stores session state in a SQL Server database. This ensures that session state is preserved if the Web application is restarted and also makes session state available to multiple Web servers in a Web farm.
* **Custom** mode, which enables you to specify a custom storage provider.
* **Off** mode, which disables session state.

You can specify which mode you want ASP.NET session state to use by assigning a [SessionStateMode](http://msdn.microsoft.com/en-us/library/system.web.sessionstate.sessionstatemode(v=vs.100).aspx) enumeration values to the **mode** attribute of the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) element in your application's Web.config file. Modes other than **InProc** and **Off** require additional parameters, such as connection-string values as discussed later in this topic. You can view the currently selected session state by accessing the value of the [HttpSessionState.Mode](http://msdn.microsoft.com/en-us/library/system.web.sessionstate.httpsessionstate.mode(v=vs.100).aspx)property.

[In-Process Mode](javascript:void(0))

In-process mode is the default session state mode and is specified using the **InProc** [SessionStateMode](http://msdn.microsoft.com/en-us/library/system.web.sessionstate.sessionstatemode(v=vs.100).aspx) enumeration value. In-process mode stores session state values and variables in memory on the local Web server. It is the only mode that supports the **Session\_OnEnd** event. For more information about the **Session\_OnEnd** event, see [Session-State Events](http://msdn.microsoft.com/en-us/library/ms178583(v=vs.100).aspx).

|  |
| --- |
| **Caution noteCaution** |
| If you enable Web-garden mode by setting the **webGarden** attribute to **true** in the [processModel](http://msdn.microsoft.com/en-us/library/7w2sway1(v=vs.100).aspx) element of the application's Web.config file, do not use **InProc** session state mode. If you do, data loss can occur if different requests for the same session are served by different worker processes. |

[State Server Mode](javascript:void(0))

**StateServer** mode stores session state in a process, referred to as the ASP.NET state service, that is separate from the ASP.NET worker process or IIS application pool. Using this mode ensures that session state is preserved if the Web application is restarted and also makes session state available to multiple Web servers in a Web farm.

To use **StateServer** mode, you must first be sure the ASP.NET state service is running on the server used for the session store. The ASP.NET state service is installed as a service when ASP.NET and the .NET Framework are installed. The ASP.Net state service is installed at the following location:

*systemroot*\Microsoft.NET\Framework\*versionNumber*\aspnet\_state.exe

To configure an ASP.NET application to use **StateServer** mode, in the application's Web.config file do the following:

* Set the **mode** attribute of the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) element to **StateServer**.
* Set the **stateConnectionString** attribute to**tcpip=***serverName***:42424**.

|  |
| --- |
| **NoteNote** |
| To improve the security of your application when using **StateServer** mode, it is recommended that you protect your**stateConnectionString** value by encrypting the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) section of your configuration file. For details, see [Encrypting Configuration Information Using Protected Configuration](http://msdn.microsoft.com/en-us/library/53tyfkaw(v=vs.100).aspx). |

The following example shows a configuration setting for **StateServer** mode where session state is stored on a remote computer namedSampleStateServer:

<configuration>

<system.web>

<sessionState mode="StateServer"

stateConnectionString="tcpip=SampleStateServer:42424"

cookieless="false"

timeout="20"/>

</system.web>

</configuration>

|  |
| --- |
| **NoteNote** |
| Objects stored in session state must be serializable if the mode is set to **StateServer**. For information on serializable objects, see the[SerializableAttribute](http://msdn.microsoft.com/en-us/library/system.serializableattribute(v=vs.100).aspx) class. |

To use **StateServer** mode in a Web farm, you must have the same encryption keys specified in the [machineKey](http://msdn.microsoft.com/en-us/library/w8h3skw9(v=vs.100).aspx) element of your Web configuration for all applications that are part of the Web farm. For information on how to create machine keys, see article 313091, "How to create keys by using Visual Basic .NET for use in Forms authentication," in the Microsoft Knowledge Base at http://support.microsoft.com.

[SQL Server Mode](javascript:void(0))

**SQLServer** mode stores session state in a SQL Server database. Using this mode ensures that session state is preserved if the Web application is restarted and also makes session state available to multiple Web servers in a Web farm.

|  |
| --- |
| **NoteNote** |
| Objects stored in session state must be serializable if the mode is SQL Server. For information on serializable objects, see the[SerializableAttribute](http://msdn.microsoft.com/en-us/library/system.serializableattribute(v=vs.100).aspx) class. |

To use **SQLServer** mode, you must first be sure the ASP.NET session state database is installed on SQL Server. You can install the ASP.NET session state database using the Aspnet\_regsql.exe tool, as described later in this topic.

To configure an ASP.NET application to use **SQLServer** mode, do the following in the application's Web.config file:

* Set the **mode** attribute of the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) element to **SQLServer**.
* Set the **sqlConnectionString** attribute to a connection string for your SQL Server database.

|  |
| --- |
| **NoteNote** |
| To improve the security of your application when using **SQLServer** mode, it is recommended that you protect your**sqlConnectionString** value by encrypting the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) section of your configuration file. For details, see [Encrypting Configuration Information Using Protected Configuration](http://msdn.microsoft.com/en-us/library/53tyfkaw(v=vs.100).aspx). |

The following example shows a configuration setting for **SQLServer** mode where session state is stored on a SQL Server named "SampleSqlServer":

<configuration>

<system.web>

<sessionState mode="SQLServer"

sqlConnectionString="Integrated Security=SSPI;data

source=SampleSqlServer;" />

</system.web>

</configuration>

|  |
| --- |
| **NoteNote** |
| If you specify a trusted connection to your SQL Server in the configuration file using the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) element's **sqlConnectionString**attribute, the [SessionStateModule](http://msdn.microsoft.com/en-us/library/system.web.sessionstate.sessionstatemodule(v=vs.100).aspx) will connect to SQL Server using SQL Server integrated security. The connection will be made using the ASP.NET process identity or the user credentials supplied for the [identity](http://msdn.microsoft.com/en-us/library/72wdk8cc(v=vs.100).aspx) configuration element, if they exist. You can specify that the IIS impersonated identity be used instead by specifying **<identity impersonate="true" />** and setting the **useHostingIdentity** attribute of the[sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) configuration element to **false**. For more information on the ASP.NET process identity, see [Configuring ASP.NET Process Identity](http://msdn.microsoft.com/en-us/library/dwc1xthy(v=vs.100).aspx)and [ASP.NET Impersonation](http://msdn.microsoft.com/en-us/library/xh507fc5(v=vs.100).aspx). |

To configure **SQLServer** mode for a Web farm, in the configuration file for each Web server, set the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) element's **sqlConnectionString**attribute to point to the same SQL Server database. The path for the ASP.NET application in the IIS metabase must be identical on all Web servers that share session state in the SQL Server database. For information on steps to resolve the issue when application paths differ between servers, see article 325056, "PRB: Session State Is Lost in Web Farm If You Use SqlServer or StateServer Session Mode," in the Microsoft Knowledge Base at http://support.microsoft.com.

Installing the Session State Database Using the Aspnet\_regsql.exe Tool

To install the session state database on SQL Server, run the Aspnet\_regsql.exe tool located in the*systemroot*\Microsoft.NET\Framework\*versionNumber* folder on your Web server. Supply the following information with the command:

* Thename of the SQL Server instance, using the **-S** option.
* The logon credentials for an account that has permission to create a database on SQL Server. Use the **-E** option to use the currently logged-on user, or use the **-U** option to specify a user ID along with the **-P** option to specify a password.
* The **-ssadd** command-line option to add the session state database.

By default, you cannot use the Aspnet\_regsql.exe tool to install the session state database on SQL Server Express. In order to run the Aspnet\_regsql.exe tool to install a SQL Server Express database, you must first enable the **Agent XPs**SQL Server option using Transact-SQL commands like the following:

EXECUTE sp\_configure 'show advanced options', 1

RECONFIGURE WITH OVERRIDE

GO

EXECUTE sp\_configure 'Agent XPs', 1

RECONFIGURE WITH OVERRIDE

GO

EXECUTE sp\_configure 'show advanced options', 0

RECONFIGURE WITH OVERRIDE

GO

You must run these Transact-SQL commands for any instance of SQL Server Express where the **Agent XPs** option is disabled.

By default, the Aspnet\_regsql.exe tool will create a database named ASPState containing stored procedures that support **SQLServer** mode. Session data itself is stored in the tempdb database by default. You can optionally use the **-sstype** option to change the storage location of session data. The following table specifies the possible values for the **-sstype** option:

|  |  |
| --- | --- |
| **Option** | **Description** |
| **t** | Stores session data in the SQL Server tempdb database. This is the default. If you store session data in the tempdb database, the session data is lost if SQL Server is restarted. |
| **p** | Stores session data in the ASPState database instead of in the tempdb database. |
| **c** | Stores session data in a custom database. If you specify the **c** option, you must also include the name of the custom database using the **-d** option. |

For example, the following command creates a database named ASPState on a SQL Server instance named "SampleSqlServer" and specifies that session data is also stored in the ASPState database:

**aspnet\_regsql.exe -S SampleSqlServer -E -ssadd -sstype p**

|  |
| --- |
| **NoteNote** |
| If you are running ASP.NET 1.0 or ASP.NET 1.1, you cannot use the Aspnet\_regsql.exe tool to configure ASP.NET to store session state in a persistent SQL Server database. However, you can obtain scripts to store session state in a persistent database. For details, see article 311209, "HOW TO: Configure ASP.NET for Persistent SQL Server Session State Management" in the Microsoft Knowledge Base at http://support.microsoft.com. As an alternative, Web servers running ASP.NET 1.0 or ASP.NET 1.1 can direct persistent session state to a SQL Server that has the ASP.NET 2.0 session state schema installed. |

In **SQLServer** mode, you can configure several computers running SQL Server to work as a failover cluster, which is two or more identical computers running SQL Server that store data for a single database. If one computer running SQL Server fails, another server in the cluster can take over and serve requests without session-data loss. To configure SQL Server mode for a failover cluster, you must specify **-sstype p** when you execute the Aspnet\_regsql.exe tool so that session state data is stored in the ASPState database instead of the tempdb database. Storing session state in the tempdb database is not supported for a SQL Server cluster. For more information about setting up SQL Server mode for a failover cluster, see article 323262, "How to use ASP.NET session state SQL Server Mode in a failover cluster" in the Microsoft Knowledge Base at http://support.microsoft.com.

[Custom Mode](javascript:void(0))

**Custom** mode specifies that you want to store session state data using a custom session state store provider. When you configure your ASP.NET application with a [Mode](http://msdn.microsoft.com/en-us/library/system.web.sessionstate.httpsessionstate.mode(v=vs.100).aspx) of **Custom**, you must specify the type of the session state store provider using the **providers** sub-element of the[sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) configuration element. You specify the provider type using an **add** sub-element and include both a **type** attribute that specifies the provider's type name and a **name** attribute that specifies the provider instance name. The name of the provider instance is then supplied to the**customProvider** attribute of the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) element to configure ASP.NET session state to use that provider instance for storing and retrieving session data.

The following example shows elements from a Web.config file that specify that ASP.NET session state use a custom session state store provider:

<configuration>

<connectionStrings>

<add name="OdbcSessionServices"

connectionString="DSN=SessionState;" />

</connectionStrings>

<system.web>

<sessionState

mode="Custom"

customProvider="OdbcSessionProvider">

<providers>

<add name="OdbcSessionProvider"

type="Samples.AspNet.Session.OdbcSessionStateStore"

connectionStringName="OdbcSessionServices"

writeExceptionsToEventLog="false" />

</providers>

</sessionState>

</system.web>

</configuration>

For more information on custom session state store providers, see [Implementing a Session-State Store Provider](http://msdn.microsoft.com/en-us/library/ms178587(v=vs.100).aspx).

|  |
| --- |
| **NoteNote** |
| A custom session state store provider will access any secured resource, such as SQL Server, using the ASP.NET process identity or the user credentials supplied to the [identity](http://msdn.microsoft.com/en-us/library/72wdk8cc(v=vs.100).aspx) configuration element, if they exist. You can specify that the IIS impersonated identity be used instead by specifying **<identity impersonate="true" />** and setting the **useHostingIdentity** attribute of the [sessionState](http://msdn.microsoft.com/en-us/library/h6bb9cz9(v=vs.100).aspx) configuration element to**false**. For more information on the ASP.NET process identity, see [Configuring ASP.NET Process Identity](http://msdn.microsoft.com/en-us/library/dwc1xthy(v=vs.100).aspx) and [ASP.NET Impersonation](http://msdn.microsoft.com/en-us/library/xh507fc5(v=vs.100).aspx). |

[See Also](javascript:void(0))

Reference

[providers Element for sessionState (ASP.NET Settings Schema)](http://msdn.microsoft.com/en-us/library/ms164669(v=vs.100).aspx)

Concepts

[ASP.NET Session State Overview](http://msdn.microsoft.com/en-us/library/ms178581(v=vs.100).aspx)

[ASP.NET State Management Overview](http://msdn.microsoft.com/en-us/library/75x4ha6s(v=vs.100).aspx)