**The ASP.NET Worker Process**

**Visual Studio 2005**

ASP.NET runs within a process known as the ASP.NET worker process. All ASP.NET functionality runs within the scope of this process.

A regular Web server contains only a single ASP.NET worker process. This is different from both Web farms and Web gardens:

* A Web farm contains multiple ASP.NET worker processes.

Each server in the group of servers handles a separate ASP.NET worker process.

* A Web garden contains multiple ASP.NET worker processes.

Each CPU in the SMP server handles a separate ASP.NET worker process.

**Choosing an ASP.NET worker process**

When a Web client connects to a Web farm or Web garden, one of the multiple ASP.NET worker processes is selected to run the request.

* In a Web farm, Network Load Balancing determines the ASP.NET worker process selected.
* In a Web garden, the ASP.NET worker process selected is determined by ASP.NET.

**State management with multiple ASP.NET worker processes**

When moving from a scenario with a single ASP.NET worker process (a normal Web server) to a scenario with multiple ASP.NET worker processes (a Web farm or Web garden), complications with state management are introduced.

Web pages are stateless, so a Web server must persist state through other means. Typical means to manage state on the Web server include Session State and the ASP.NET Cache.

**Note**Issues of persistence and state (within a single Web server) are discussed in detail in the section [Which Persistence Approach Should I Use with Crystal Reports?](http://msdn.microsoft.com/en-us/library/ms225433(v=vs.80).aspx).

Both Session and Cache are contained within the memory space of a single ASP.NET worker process. But in a Web farm or Web garden, multiple ASP.NET worker processes work together simultaneously. The Session or Cache within any individual ASP.NET worker process cannot manage state across multiple processes.

Therefore, an additional layer is required for state management: an out-of-process Session State server that stores and retrieves state information for every ASP.NET worker process in the Web farm or Web garden. See [Out-of-process Session State](http://msdn.microsoft.com/en-us/library/ms225449(v=vs.80).aspx).

# How to: Run the Worker Process Under a User Account

**Visual Studio 2013**

[Other Versions](javascript:;)

http://i.msdn.microsoft.com/Areas/Epx/Content/Images/ImageSprite.png

* [Visual Studio 2012](http://msdn.microsoft.com/en-us/library/bakfs900(d=printer,v=vs.110).aspx)
* [Visual Studio 2010](http://msdn.microsoft.com/en-us/library/bakfs900(d=printer,v=vs.100).aspx)
* [Visual Studio 2008](http://msdn.microsoft.com/en-us/library/bakfs900(d=printer,v=vs.90).aspx)
* [Visual Studio 2005](http://msdn.microsoft.com/en-us/library/bakfs900(d=printer,v=vs.80).aspx)

This topic has not yet been rated - [Rate this topic](http://msdn.microsoft.com/en-us/library/bakfs900(d=printer).aspx#feedback)

To set up your computer so that you can run the ASP.NET worker process (aspnet\_wp.exe or w3wp.exe) under a user account, follow these steps.

[Procedure](javascript:void(0))

### To run aspnet\_wp.exe under a user account

1. Open the machine.config file, located on your computer in the CONFIG folder under the path where you installed the runtime.
2. Find the <processModel> section and change the user and password attributes to the name and password of the user account you want aspnet\_wp.exe to run under.
3. Save the machine.config file.
4. On Windows Server 2003, IIS 6.0 is installed by default. The corresponding worker process is w3wp.exe.To run in IIS 6.0 mode with aspnet\_wp.exe as the worker process, you must follow these steps:
   1. Click Start, click Administrative Tools and then choose Internet Information Services.
   2. In the Internet Information Services dialog box, right-click the Web Sites folder and choose Properties.
   3. In the Web Sites Properties dialog box, choose Service.
   4. Select Run WWW service in IIS6.0 isolation mode.
   5. Close the Properties dialog box and Internet Services Manager.
5. Open a Windows Command Prompt and reset the server by running:

iisreset

— or —

net stop iisadmin /y

net start w3svc

1. Locate the Temporary ASP.NET Files folder, which should be in the same path as the CONFIG folder. Right-click the Temporary ASP.NET Files folder and choose Properties on the shortcut menu.
2. In the Temporary ASP.NET Files Properties dialog box, click the Security tab.
3. Click Advanced.
4. In the Advanced Security Settings for Temporary ASP.Net Files dialog box, click Add.

The Select User, Computer, or Group dialog box appears.

1. Type the user name in the Enter the object name to select box, and then click OK. The user name must follow this format: DomainName\UserName.
2. In the Permission Entry for Temporary ASP.NET Files dialog box, give the user Full Control, and then click OK to close the Entry for Temporary ASP.NET Files dialog box.
3. A Security dialog box will appear, and asks if you really want to change the permissions on a system folder. Click Yes.
4. Click OK to close the Temporary ASP.NET Files Properties dialog box.

**Out-of-Process Session State**

**Visual Studio 2005**

In Web farms and Web gardens, Session State must be shared across servers using some form of out-of-process Session State.

Two solutions are commonly used:

* Deploy the out-of-process Session State server that is provided with ASP.NET.
* Manually configure each Web server to store Session State data on a SQL Server.

**Serialization Requirement**

In both solutions, all objects to be stored must be serializable. Unless an object can be serialized, it cannot be persisted to an out-of-process state server.

**Note**Not all object models in the Crystal Reports SDK can be serialized. However, Crystal provides an upgrade solution for managing the state of reports with an out-of-process server. See [Serializable Object Models in Crystal Reports](http://msdn.microsoft.com/en-us/library/ms225448(v=vs.80).aspx).