



LEARN & WIN \$300 GIFT CARD: Growth Mindset Virtual Conference featuring Live Panels, sessions, and music band

x



C# Corner

TRY CSharp.Live - 100s of Live Shows focused on learning and professional growth

Join a member

Login

Post

Ask Question



Session State in ASP.Net



Anupam Jain

Updated date Jun 10, 2019

328.8k

12

14

[Download Free .NET & JAVA Files API](#)

[Try Free File Format APIs for Word/Excel/PDF](#)

This article explains in detail the Session State Management technique and its modes in ASP.Net.

Let us start from the very beginning. Let's first try to understand why we need to maintain the state of our application or why we need State Management. As we all know, our web is **"Stateless"**, in other words a new instance of a web page class, is recreated each time the page is posted to the server. HTTP is a stateless protocol and it can't hold the client information on the page. For example, if the user inserts some information on one page and then moves to the next page then that inserted data will be lost from the first page and moreover the user will not be able to retrieve that information.

So basically here we need someone to hold the state of our application. Here is the privitage role of our **"session state"**. Basically a session is a variable used between the client and the server that is stored on the server side. Now it can be

stored either on an Internet Information Service (IIS) server that is by default our **"inproc"** mode or it can be stored in a state or SQL Server that is our **"outproc"** mode. We will discuss both, the inproc and outproc modes in detail later in the article.

So a session helps to maintain the user state and data all over the application by storing the information on the server memory. Also a session can store any kind of information or object on the server side and is accessible in the entire website.

Now let's discuss the entire scenario that happens when the user state and data is maintained using session state. First of all when the user requests a new application or page, first the "Application" start event fires in the get state and that application object is sharable in the entire website. After the application life cycle, the session start event fires for the specific user in the get state but when some other user again requests that page, no application start event will fire, only the session start event in the post state will fire for that specific user. Every object is stored in the application on the basis of the Key value. We can see both the application and session start up events by adding a **"Global.asax"** file in our project.

The process of maintaining the session state proceeds in the following manner. First the client hits the website and the information is stored in the session. Then a Session table will be made by default on the IIS server and in the session IDs of all the users visiting the website will be stored by the server. Now the next time the client requests some information with the unique session ID from the server, the server looks in the session providers and retrieves the serialized data from the state server and type casts the object.

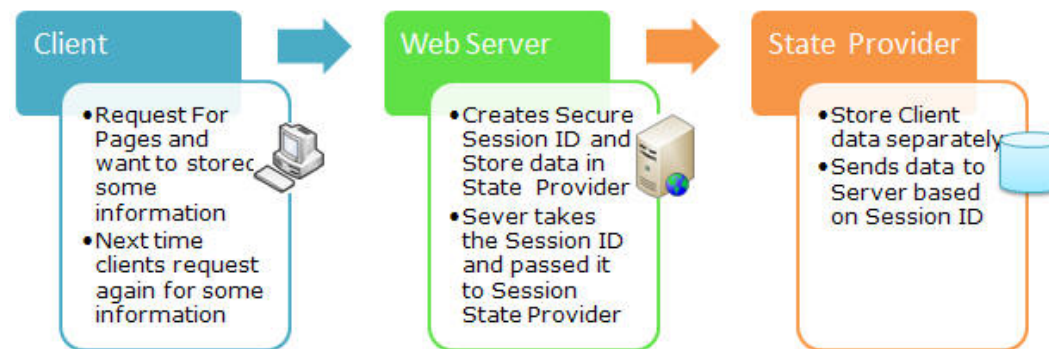


FIG: PROCESS FOR MAINTAINING THE SESSION STATE IN THE APPLICATION

Now let us see how to store and retrieve values in a session.

STORING AND RETRIEVING VALUES FROM SESSION

The following code is used for storing a value in a session:

```
01. // Storing Username in session.Session[ "UserName" ] = txtUser.Text;
02. // Retrieving
03. values
04. from
05. session.// Check whether session variable null
06. or not if(Session[ "UserName" ] != null) { // Retrieving UserName
07. from
08. session lblWelcome.text = "Welcome: +Session[" UserName "];
09. }
10. else
11. {
12. //Do something else
13. }
```

As we know all these values will be stored and retrieved from the session on the IIS server by default.

Now for maintaining the load balance we need to free the IIS server. So we require an **"Outproc"** mode of the session state. Now we will study both the **"Inproc"** and **"Outproc"** modes of the session state in detail.

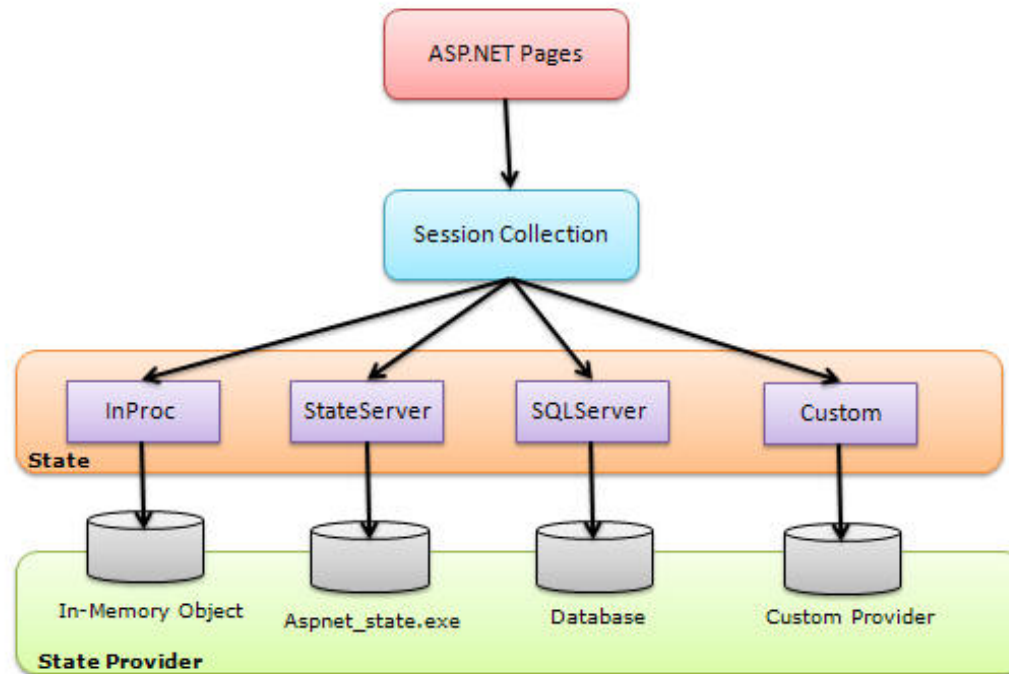


FIG: INPROC AND OUTPROC MODES IN SESSION STATE AND THEIR RESPECTIVE STATE PROVIDERS

INPROC SESSION MODE IN SESSION STATE

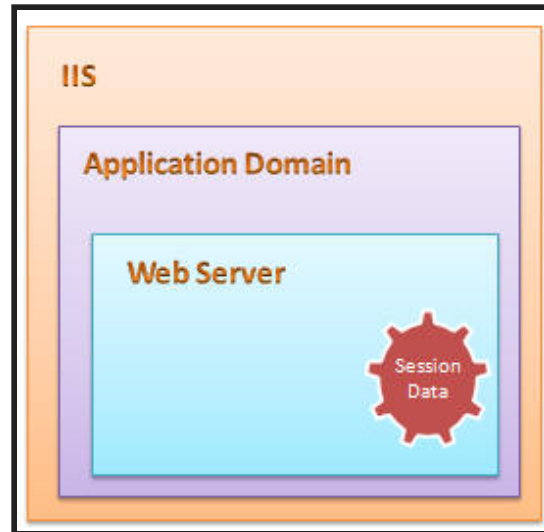


FIG: INPROC SESSION MODE

This is the default session mode in ASP.Net. It stores the information in a memory object in the current application domain. So it is easily and quickly available and is best suited for web application performance but the main disadvantage is that because all the information is stored on the server side in the same application domain, if we will restart the server all the data will be lost. When the client requests data, the State Provider reads the data from an in-memory object and return it to the client. **In web.config, we need to specify the session mode and also set the time out.**

```
<system.web>
  <sessionState mode="InProc" timeout="20"></sessionState>
</system.web>

/configuration>
```

Advantages

- It stores session data in a memory object of the current application domain. So accessing data is very fast and data is easily available.
- There is not a requirement for serialization to store the data in InProc session mode.
- Implementation is very easy, similar to using the ViewState.

Disadvantages

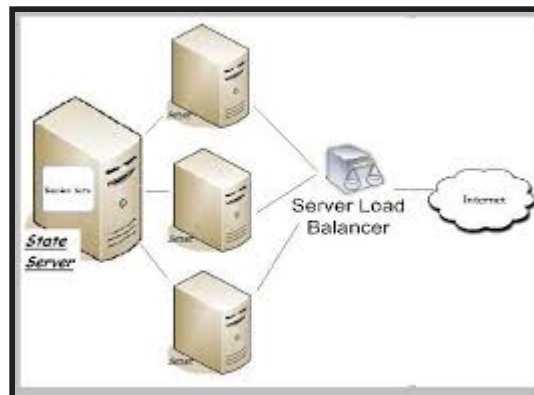
Although an InProc session is the fastest, common and default mechanism, it has many limitations as in the following:

- If the worker process or application domain is recycled, all session data will be lost.
- Though it is the fastest, more session data and more users can affect performance, because of memory usage.
- We can't use it in Web Garden scenarios.
- This session mode is not suitable for web Farm scenarios.

As in the preceding discussion, we can conclude that InProc is a very fast session storing mechanism but suitable only for small web applications. InProc session data will be lost if we restart the server, or if the application domain is recycled. It is also not suitable for Web Farm and Web Garden scenarios.

Now we will have a look at the other options available to overcome these problems. First is the StateServer mode.

STATESERVER MODE(OUTPROC MODE)



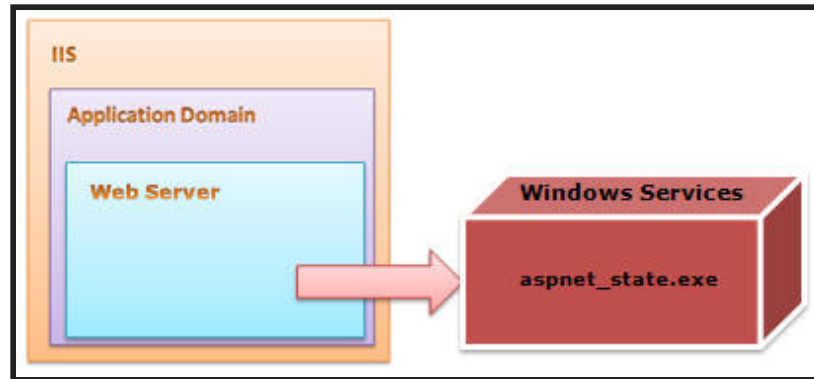


FIGURE: STATE SERVER MODE IN SESSION STATE

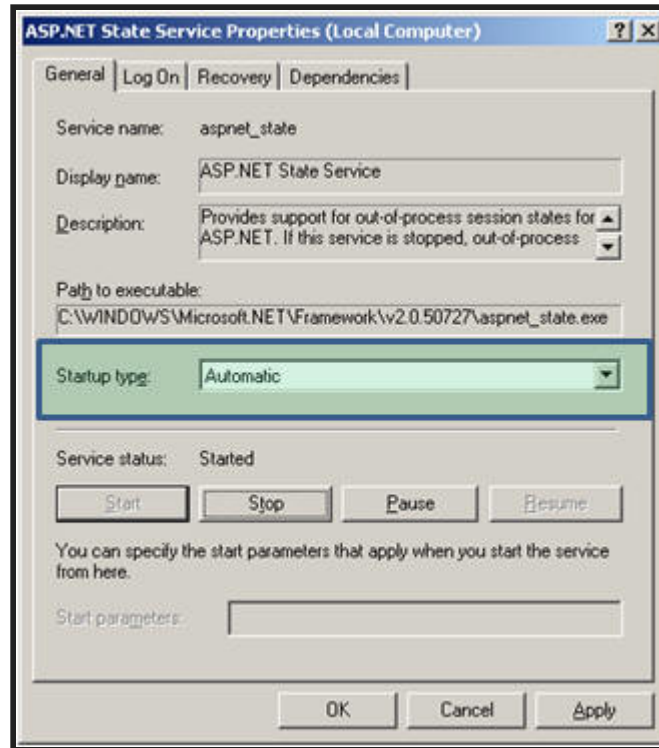
This is also called Out-Proc session mode. StateServer uses a stand-alone Windows Service that is independent of IIS and can also be run on a separate server. This session state is totally managed by *aspnet_state.exe*. This server may run on the same system, but it's outside of the main application domain where your web application is running. This means if you restart your ASP.NET process, your session data will still be alive. This approach has several disadvantages due to the overhead of the serialization and de-serialization involved, it also increases the cost of data access because every time the user retrieves session data, our application hits a different process.

Configuration for StateServer session mode

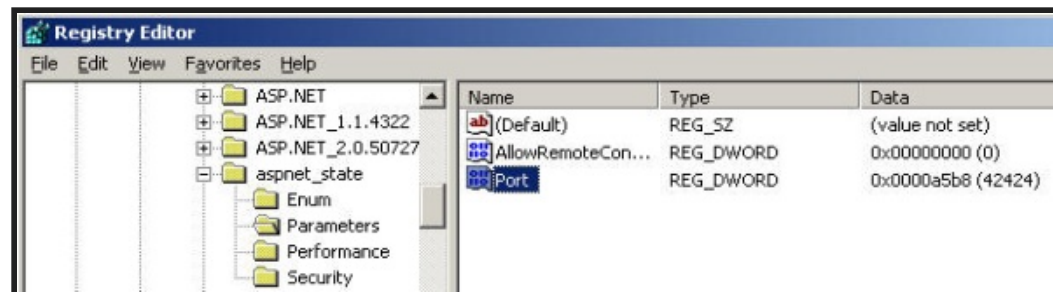
In StateServer mode, session data is stored in a separate server that is independent of IIS and it is handled by *aspnet_state.exe*. This process is run as a Windows Service. You can start this service from the Windows MMC or from the command prompt.

Services					
ASP.NET State Service					
Start the service					
Description: Provides support for out-of-process session states for ASP.NET. If this service is stopped, out-of-process requests will not be processed. If this service is disabled, any services that explicitly depend on it will fail to start.					
Name	Description	Status	Startup Type	Log On As	
.NET Runtime Optimization Service v2.0...	Microsoft ...		Manual	Local System	
Alert Notification Server		Started	Automatic	Local System	
Alerts	Notifies sel...		Disabled	Local Service	
Apache Content Server	Apache/2...		Automatic	Local System	
Apache Tomcat Application Server	Apache To...	Started	Automatic	Local System	
Application Experience Lookup Service	Processes ...	Started	Automatic	Local System	
Application Layer Gateway Service	Provides s...	Started	Manual	Local Service	
Application Management	Processes l...		Manual	Local System	
ASP.NET State Service	Provides s...		Manual	Network S...	
Automatic Updates	Enables th...	Started	Automatic	Local System	

By default, the "Startup Type" of the ASP.NET state service is set to Manual; we need to set it to Automatic.



From the command prompt, just type "net start aspnet_state". By default, this service listens to TCP port 42424, but we can change the port from the Registry editor as shown in the picture below:



Now have a look at the *web.config* configuration for the StateServer setting. For the StateServer setting, we need to specify the stateConnectionString. This will identify the system that is running the state server. By default, stateConnectionString uses IP 127.0.0.1 (localhost) and port 42424.


```

01. <configuration>
02.     <system.web>
03.         <sessionstate mode="StateServer"    time out="30" stateConnectionString="tcpip=127.0.0.1"
04.         "tcpip=localhost:42424"    </sessionstate>
05.     </system.web>
06. </configuration>

```

inproc and outproc mode in session state differences using code

Now let's us try to understand the difference between inproc and outproc mode in session state using code.

1. Open Visual Studio then create a new project. Select Web - ASP.Net Empty Web Application and let's name it "OutofProcSessionState".
2. Add new Web Form Login.aspx as in the following:
 - Add a new TextBox to get the username and a button to Login as in:
- 3.

Login.aspx

Enter User Name:

```

01. <asp:TextBox ID="txtUserName" runat="server"></asp:TextBox>
02. <asp:Button ID="btnSubmit" runat="server" Text="Go to Home Page" OnClick="btnSubmit_Click"/>

```

Login.aspx.cs: Add the handler code for the Submit button in the code behind as in the following:

- Store the User Name logged in by user in the session indexed by User Name.
- Redirect the user to the Home Page on click of the button.

```

01. protected void btnSubmit_Click(object sender, EventArgs e)
02. {
03.     Session["UserName"] = txtUserName.Text.ToString();
04.     Response.Redirect("Home.aspx");
05. }

```

- Add a new Web Form, Home.aspx:

1. Home.aspx: Add a label control in the Home Page to show the logged in User Name

01. | <label id="lblUserName" runat ="server"></label>

- Home.aspx.cs: In the Page Load event retrieve the User Name from the Session State.
 - Retrieve the User Name from the Session and display it.
 - If the User Name is not present in the session then show the user name as Anonymous User.

```
01. | protected void Page_Load(object sender, EventArgs e)
02. | {
03. |     if (Session["UserName"] != null && Session["UserName"] != "")
04. |     {
05. |         lblUserName.InnerText = Session["UserName"].ToString();
06. |     }
07. |     else
08. |     {
09. |         lblUserName.InnerText = "Anonymous User";
10. |     }
```

- Run the application with Login.aspx as the Start Page.
- So in the output we will find the UserName in the home page.
- So all is fine here so far since we are working by default in inproc mode and the session data is stored by default on the IIS server.
- Now let us stop our web development server and restart it again. Here we will find that all our session data is lost that was stored on the IIS server.
- So here we will be requiring the State Server mode in which all the session data will be stored on a separate server or Windows service that is totally independent of IIS and is managed by ASP.Net_state.exe as I said above.

I hope my article is useful for all those who are new in the field of ASP.Net and finding their future in the ASP.Net domain. Session State is quite useful for the purpose of interviews. In my next article I'll be describing SQL Server outproc mode of session state. I have tried my best to summarize Session State, still if anyone has questions then please provide me your valuable suggestions.

Asp.Net Session state

Asp.Net Page Life Cycle

Asp.Net SESSION STATE

Session state

Next Recommended Article

[View State Vs. Session State Vs. Application State](#)

OUR BOOKS



Anupam Jain

<https://www.c-sharpcorner.com/members/anupam-jain2>

1412

566.7k

[View Previous Comments](#)

14

12



Type your comment here and press Enter Key (Minimum 10 characters)

I am facing a difficulty with session state as "SQL Server". Can you please answer my below



question.<https://stackoverflow.com/questions/54117531/connect-to-aspstate-database-from-app-server-instead-of-web-server>

[Hari Teja Valavala V](#)

1985 5 0

Jan 16, 2019

0 0 Reply



Good one. I have used the StateServer in same applications.

[Bruno Péterson](#)

1912 78 0

Jun 18, 2015

0 0 Reply



Nice one.

[Manish Kumar Choudhary](#)

198 10.9k 4m

Apr 13, 2015

0 0 Reply



Good start

[Karthik Muthu Karuppan](#)

448 4.8k 610k

Apr 13, 2015

0 0 Reply



Good one

[Suresh Mogudala](#)

1847 143 2.4k

Apr 13, 2015

1 0 Reply



Great Start, keep it up

[Rahul Bansal](#)

149 15.1k 7.4m

Apr 13, 2015

1 0 Reply



nice

[Shubham Kumar](#)

414 5.2k 581.5k

Apr 13, 2015

1 0 Reply



goooooooooooooooood start anupam :)

[Nitin Pandit](#)

12 50.7k 29.5m

Apr 13, 2015

1 0 Reply



Good one :)

[Sibeesh Venu](#)

14 49.4k 8.6m

Apr 13, 2015

1 0 Reply

nice



Khargesh Rajput

379 5.6k 448.4k

Apr 13, 2015

1

0

Reply

FEATURED ARTICLES

What is Microsoft Mesh

How To Create SQL Server Database Project With Visual Studio

Angular 11 New Features

What Is Azure Functions? How to get started with Azure Functions?


Entity Framework Core 5.0 - An Introduction To What's New

[View All](#)

TRENDING UP

01 Read Excel File in WEB API using C#

02 Localization in Angular Application using Angular Locale

- 03 Build Restful API's With Node.js - Express - MySQL
- 04 How To Set Background Color Of A Selected Row Based On Checking/Unchecking Checkbox In Angular 10
- 05 Getting Started With .NET 6
- 06 Onion Architecture In .Net 5
- 07  Node.js API Authentication With JSON Web Tokens
- 08 CRUD Operation With .NET Core 3.1 And Entity Framework Core
- 09 Simplifying Use Of Static
- 10 What is Microsoft Mesh

[View All](#) 

[About Us](#) [Contact Us](#) [Privacy Policy](#) [Terms](#) [Media Kit](#) [Sitemap](#) [Report a Bug](#) [FAQ](#) [Partners](#)

[C# Tutorials](#) [Common Interview Questions](#) [Stories](#) [Consultants](#) [Ideas](#) [Certifications](#)

©2021 C# Corner. All contents are copyright of their authors.