MICROSOFT ARCHITECT
By Joydip Kanjilal, Columnist, InfoWorld
JUL 29, 2019 3:00 AM PDT

How to work with session state in ASP.Net Core

Take advantage of the session middleware component in ASP.Net Core to store and retrieve user data

ASP.Net Core is an open source, cross-platform, lean, extensible and modular framework for building high-performance web applications. Session state in ASP.Net Core enables you to store user data and persist the data across requests from the same client. You can take advantage of the necessary middleware (available as part of the (Microsoft.AspNetCore.Session package) to work with session state in ASP.Net Core.

Middleware components in ASP.Net Core are used to customize the handling of requests and responses, and inspect, route, or modify the request and response messages that flow through the pipeline. Usually, you have a chain of middleware components in the application pipeline in ASP.Net Core. This article presents a discussion on how we can work with session state in ASP.Net Core.

Create an ASP.Net Core MVC project

First off, let's create an ASP.Net Core project in Visual Studio. Assuming Visual Studio 2017 or Visual Studio 2019 is installed in your system, follow the steps outlined below to create a new ASP.Net Core project in Visual Studio.

1. Launch the Visual Studio IDE.

2. Click on “Create new project.”
3. In the “Create new project” window, select “ASP.Net Core Web Application” from the list of templates displayed.
4. Click Next.
5. In the “Configure your new project” window shown next, specify the name and location for the new project.
6. Click Create.
7. In the “Create New ASP.Net Core Web Application” shown next, select .Net Core as the runtime and ASP.Net Core 2.2 (or later) from the drop-down list at the top.
8. Select “Web Application (Model-View-Controller)” as the project template to create a new ASP.NET Core application.
9. Ensure that the check boxes “Enable Docker Support” and “Configure for HTTPS” are unchecked as we won’t be using those features here.
10. Ensure that Authentication is set to “No Authentication” as we won’t be using authentication either.
11. Click Create.

You should now have a new ASP.NET Core project ready to go in Visual Studio. We’ll use this project in the subsequent sections of this article to work with session state.

Install the ASP.Net Core session state middleware

The next step is to install the necessary NuGet package for working with session state. The NuGet package we need is Microsoft.AspNetCore.Session. To add the Microsoft.AspNetCore.Session NuGet package to your project, right-click on the project in the Solution Explorer window, and click on “Manage NuGet Packages...” Then search for this package in the NuGet Package Manager and install it.

Alternatively, you can install this package from the NuGet Package Manager Console using the following command:

```
PM> Install-Package Microsoft.AspNetCore.Session
```

Configure the ASP.Net Core session state middleware

Now that the necessary NuGet package has been installed on your project, you can add the session middleware component to the ASP.Net Core pipeline. Note that to enable session state in ASP.Net Core you must add any of the IDistributedCache memory caches you want to use as a backing storage for session. You must also make a call to AddSession in the ConfigureServices method to add the session middleware to the ASP.Net Core pipeline. Lastly, you must make a call to the UseSession method in the Configure method of the Startup class.

The following code snippet shows how you can add the session middleware to the ASP.Net Core pipeline in the ConfigureServices method.

```
public void ConfigureServices(IServiceCollection services)
    {
        services.AddDistributedMemoryCache();
        services.AddSession(options =>
        {
            options.IdleTimeout = TimeSpan.FromSeconds(5);
            options.Cookie.HttpOnly = true;
            options.Cookie.IsEssential = true;
        });
        services.AddMvc()
            .SetCompatibilityVersion(CompatibilityVersion.Version_2_2);
    }
```

Once the session middleware component has been added to the pipeline, you must make a call to the UseSession method in the Configure method of the Startup class.

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
    {
        app.UseHttpsRedirection();
        app.UseStaticFiles();
        app.UseSession();
        app.UseHttpContextItemsMiddleware();
        app.UseMvc();
    }
```

Store and retrieve data to and from a session in ASP.Net Core

You can take advantage of the methods Set, SetInt32, and SetString to set session values. These methods accept a key and the data to be stored as arguments. The Set method accepts a byte array as an argument.

Similarly, you have the Get, GetInt32, and GetString methods to retrieve data from the session based on a key. The Get method accepts a key as an argument and returns a byte array. To use these methods you should add a reference to Microsoft.AspNetCore.Http in your code.

The following code snippet illustrates how you can add data to the session in your controller's action method.

```
public IActionResult Index()
{
   HttpContext.Session.SetString("Message", "Hello World!");
   HttpContext.Session.SetInt32("Year", 2019);
   return View();
}
```

And here is how you can retrieve those values in your controller's action method.

```
public IActionResult About()
{
    ViewBag.Message = HttpContext.Session.GetString("Message");
    ViewBag.Year = HttpContext.Session.GetInt32("Year");
    return View();
}
```

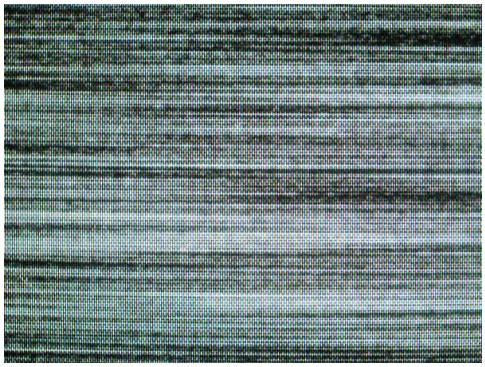
If you would like to set or get data pertaining to other types, you should create your own extension methods on ISession and implement the necessary serialization logic yourself. To store and retrieve a complex object to and from the session, you can serialize and deserialize the object to JSON. Instead of using the in-memory cache you can also store session data in SQL Server or Redis.

Joydip Kanjilal is a Microsoft MVP in ASP.Net, as well as a speaker and author of several books and articles. He has more than 20 years of experience in IT including more than 16 years in Microsoft .Net and related technologies.

Follow

What's new in Kubernetes 1.20

How IT priorities are shifting during the COVID-19 crisis



Python developers want static typing

The 24 highest paying developer roles in 2020

Microsoft Visual Studio 2022 preview is coming soon

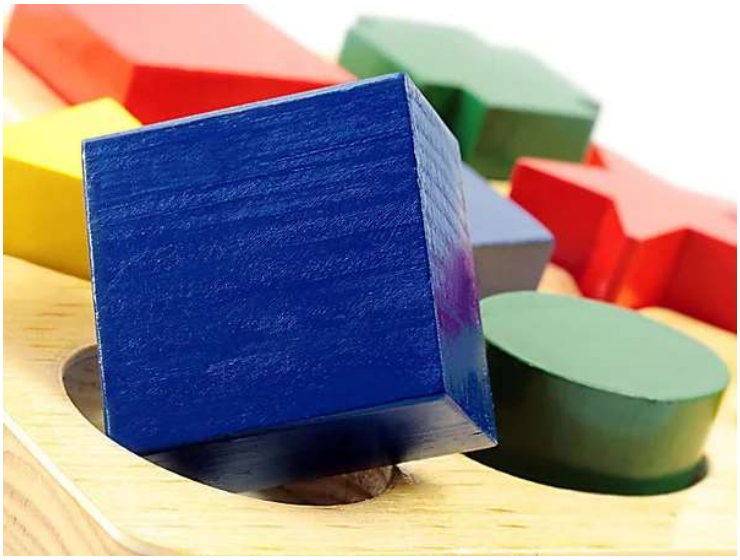
Deno Company forms to back Node.js rival

The decline of Heroku

3 enterprise AI success stories

3 cloud architecture mistakes we all make, but

6 ways Alibaba Cloud challenges AWS, Azure, and GCP



When Kubernetes is not the solution