MICROSOFT ARCHITECT
By Joydip Kanjilal, Columnist, InfoWorld
NOV 4, 2019 3:00 AM PST

How to work with cookies in ASP.NET Core

Take advantage of cookies to store and retrieve user-specific information in your ASP.NET Core web application

A cookie is a piece of data typically used to store information about the user and is stored on the user's computer. In most browsers each cookie is stored as a small file, but in Firefox they are stored all together in a single file. Cookies are represented as key-value pairs, and you can take advantage of the keys to read, write, or delete cookies.

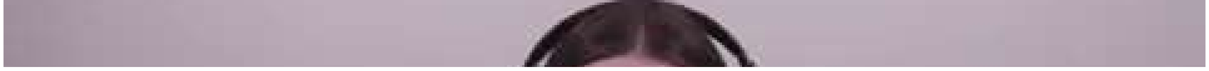
ASP.NET Core uses cookies to maintain session state; the cookie that contains the session ID is sent to the client with each request. This article presents a discussion of how we can work with cookies in ASP.NET Core.

[SQL databases are gaining NoSQL features. Discover 10 new tricks your old database can do. | Don't miss these 7 powerful features in MySQL and MariaDB. | Keep up with hot topics in programming with InfoWorld's App Dev Report newsletter.]

To work with the code examples provided in this article, you should have Visual Studio 2019 installed in your system. If you don't already have a copy, you can download Visual Studio 2019 here.

Create an ASP.NET Core MVC project in Visual Studio

First off, let's create an ASP.NET Core MVC project in Visual Studio 2019. Assuming Visual Studio 2019 is installed in your system, follow the steps outlined below to create a new ASP.NET Core MVC 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, specify the name and location for the new project.
6. Optionally, select the “Place solution and project in the same directory” check box.
7. Click Create.
8. In the “Create a New ASP.NET Core Web Application” window shown next, select .NET Core as the runtime and ASP.NET Core 2.2 (or later) from the drop-down list at the top.
9. Select “Web Application (Model-View-Controller)” as the project template to create a new ASP.NET Core MVC application.
10. Ensure that the check boxes “Enable Docker Support” and “Configure for HTTPS” are unchecked as we won’t be using those features here.
11. Ensure that Authentication is set to “No Authentication” as we won’t be using authentication here either.
12. Click Create.

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

Read a cookie in ASP.NET Core

You can read a cookie from the Request.Cookies collection. The following code snippet illustrates how you can read a cookie from the Request object in ASP.NET Core.

```
string cookie = Request.Cookies["Key"];
```

If you would like to specify the expiration time of the cookie, you can use the overloaded version of the Append method as shown in the code snippet given below.

```
CookieOptions option = new CookieOptions();
option.Expires = DateTime.Now.AddMilliseconds(10);
Response.Cookies.Append(key, value, option);
```

The CookieOptions class enables you to specify the following additional properties when creating a cookie:

- **Domain** — used to specify the domain associated with a cookie
- **Expiration time** — used to specify the expiration time of the cookie
- **Path** — used to specify the cookie path
- **Security policy** — used to specify if the cookie is accessible over HTTPS
- **HttpOnly** — used to specify if the cookie is available only to the server

Write a cookie in ASP.NET Core

To write a cookie you can take advantage of the Append method pertaining to the Request object. The following code snippet illustrates how this can be achieved.

```
Response.Cookies.Append(somekey, somevalue);
```

Delete a cookie in ASP.NET Core

To remove a cookie, you can use the Delete method of the Cookies collection pertaining to the Request object. The following code snippet shows how this can be achieved.

```
Response.Cookies.Delete(somekey);
```

Access HttpContext in ASP.NET Core

In this section we'll examine how we can work with cookie data in ASP.NET Core. We'll need to access the HttpContext to be able to access the Request object. You can access the HttpContext in ASP.NET Core using the IHttpContextAccessor interface. The HttpContextAccessor class implements this interface.

First you should register IHttpContextAccessor for dependency injection. The following code snippet illustrates how you can add a singleton service of type HttpContextAccessor in the ConfigureServices method of the Startup class.

```
public void ConfigureServices(IServiceCollection services)
        {
            services.AddSingleton<IHttpContextAccessor,
            HttpContextAccessor>();
            //Other code
        }
```

You can take advantage of dependency injection to get a reference to the IHttpContextAccessor instance. This will in turn provide you a reference to HttpContext.

The following code snippet illustrates how you can access the IHttpContextAccessor instance in the controller. Note that HomeController is created by default when you create a new ASP.NET Core MVC project in Visual Studio.

```
public class HomeController : Controller
{
  private readonly IHttpContextAccessor _httpContextAccessor;
  public HomeController(IHttpContextAccessor httpContextAccessor)
  {
     this._httpContextAccessor = httpContextAccessor;
  }
  //Write your action methods here
}
```

Write cookie data in your ASP.NET Core controller method

You can use the following method to write cookie data in your controller.

```
public IActionResult Write(string key, string value, bool isPersistent)
  {
       CookieOptions options = new CookieOptions();
       if (isPersistent)
           options.Expires = DateTime.Now.AddDays(1);
       else
           options.Expires = DateTime.Now.AddSeconds(10);
       _httpContextAccessor.HttpContext.Response.Cookies.Append
       (key, value, options);
       return View("WriteCookie");
  }
```

Read cookie data in your ASP.NET Core controller method

Once the cookie data has been written successfully, you can use the following method to read cookie data in your controller.

```
public IActionResult Read(string key)
  {
       ViewBag.Data =
       _httpContextAccessor.HttpContext.Request.Cookies[key];
       return View("ReadCookie");
  }
```

To check if a cookie has been written properly, you can inspect the cookie cache of your web browser. In a future post, we’ll examine how we can work with cookie-based authentication and authorization in ASP.NET Core.

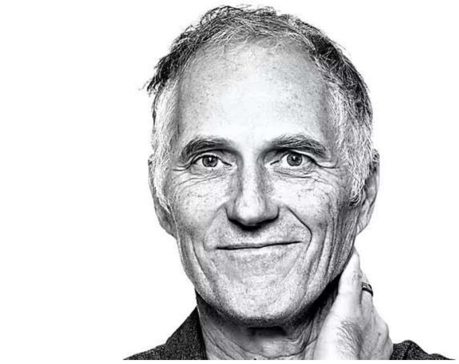
Joydip Kanjilal is a Microsoft MVP in ASP.Net, as well as a speaker and author of several books and articles. He has more than 20 years of experience in IT including more than 16 years in Microsoft .Net and related technologies.

Follow

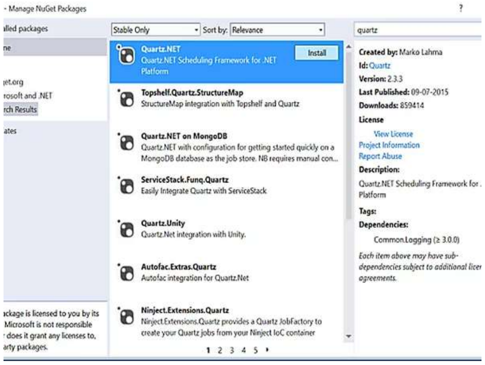
- **Stay up to date with InfoWorld’s newsletters for software developers, analysts, database programmers, and data scientists.**
- **Get expert insights from our member-only Insider articles.**

YOU MAY ALSO LIKE

Recommended by



Tim O'Reilly: the golden age of the programmer is over



Quartz.NET
Quartz.NET Scheduling Framework for .NET Platform

Topshelf.Quartz.StructureMap
StructureMap integration with Topshelf and Quartz

Quartz.NET on MongoDB
Quartz.NET with configuration for getting started quickly on a MongoDB database as the job store. NB requires manual con...

ServiceStack.Funq.Quartz
Easily Integrate Quartz with ServiceStack

Quartz.Unity
Quartz.Net integration with Unity.

Autofac.Extras.Quartz
Autofac integration for Quartz.Net

Ninject.Extensions.Quartz
Ninject.Extensions.Quartz provides a Quartz JobFactory to create your Quartz jobs from your Ninject IoC container

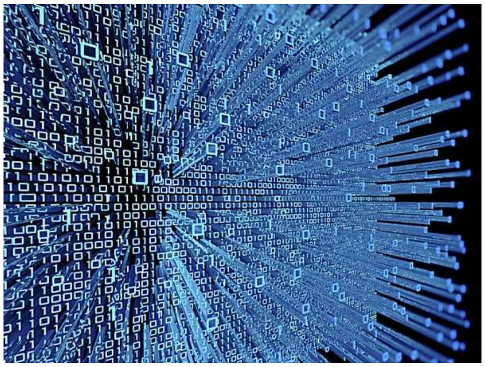
Created by: Marko Lahma
Id: Quartz
Version: 2.3.3
Last Published: 09-07-2015
Downloads: 859414
License
View License
Project Information
Report Abuse
Description:
Quartz.NET Scheduling Framework for . Platform
Tags:
Dependencies:
Common.Logging (≥ 3.0.0)
Each item above may have sub-dependencies subject to additional licer agreements.

ıckage is licensed to you by its Microsoft is not responsible does it grant any licenses to, arty packages.

How to work with Quartz.Net in C#

Edge computing archetypes are emerging, and they are not pretty

The 24 highest paying developer roles in 2020



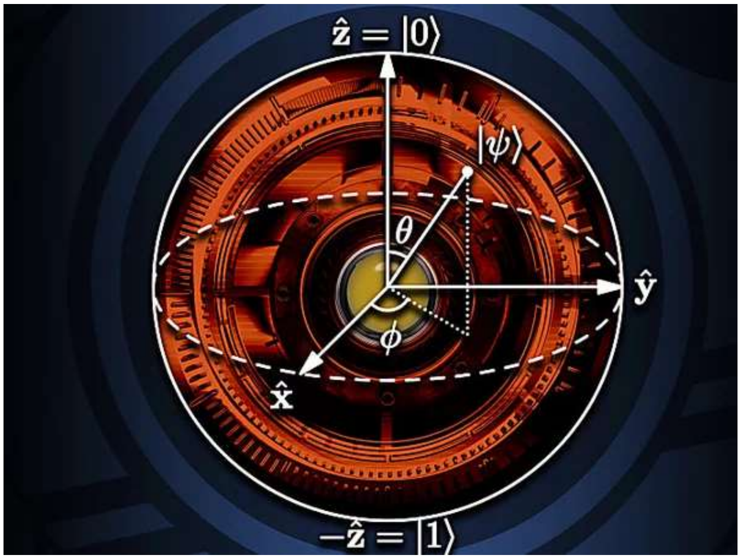
Mozilla spins out Pyodide Python-in-the-browser project

2 common cloud computing predictions for 2021 are wrong

Red Hat Enterprise Linux takes aim at edge

6 reasons to switch to managed Kubernetes

Red Hat OpenShift ramps up security and



Amazon Braket: Get started with quantum computing

Are industry clouds an opportunity or a distraction?

UNITED STATES