State Management In ASP.NET

The goal of this article is to understand State Management in ASP .NET. ASP .NET has several states including application state, session state, and view state.

Abhishek Jaiswal Mar 19 2019

24 30 734.3k

This article is all about how to maintain, clear or hold the states of your pages in ASP.NET applications. In this article I tried to briefly summarize the concept of State Management but I'll include Client-Side State Management only.

Agenda

The agenda of this article will be as follows,

- State Overview
- State Introduction
- State Outline
- State Management Types

cus
is the Fut

- State Management Scenario
- State Management Techniques
    - Client-side Techniques
        - Hidden
        - Cookies
        - Control State
        - Query Strings
    - Server-side Techniques
        - Session State
        - Application State

State Overview

As we all know, browsers are generally stateless.

Now the question arises here, what does stateless actually mean?

Stateless means, whenever we visit a website, our browser communicates with the respective server depending on our requested functionality or the request. The browser communicates with the respective server using the HTTP or HTTPs protocol.

But after that response, what's next or what will happen when we visit that website again after closing our web browser?

In this case HTTP/HTTPs doesn't remember what website or URL we visited, or in other words we can say it doesn't hold the state of a previous website that we visited before closing our browser, that is called stateless.

So our browsers are stateless.

State Introduction

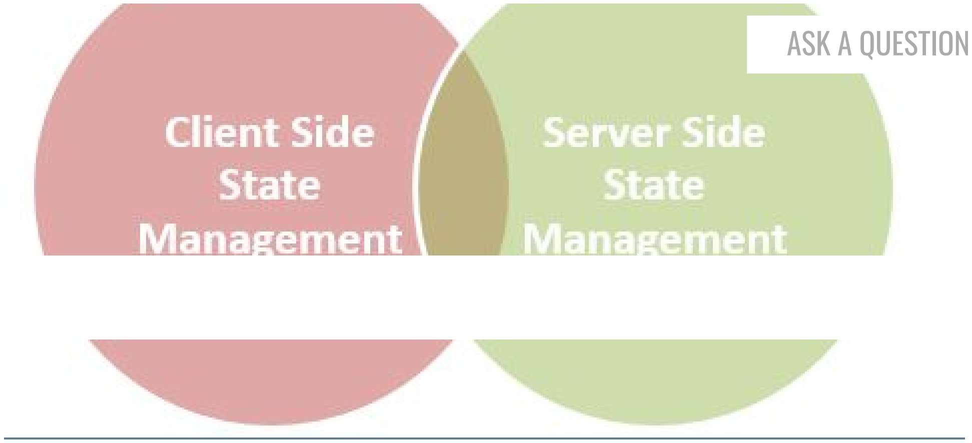
State Outline

As I said in the beginning, HTTP is a stateless protocol. It just cleans up or we can say removes all the resources/references that were serving a specific request in the past. These resources can be:

- Objects
- Allocated Memory
- Sessions ID's
- Some URL info
  and so on.

State Management Types

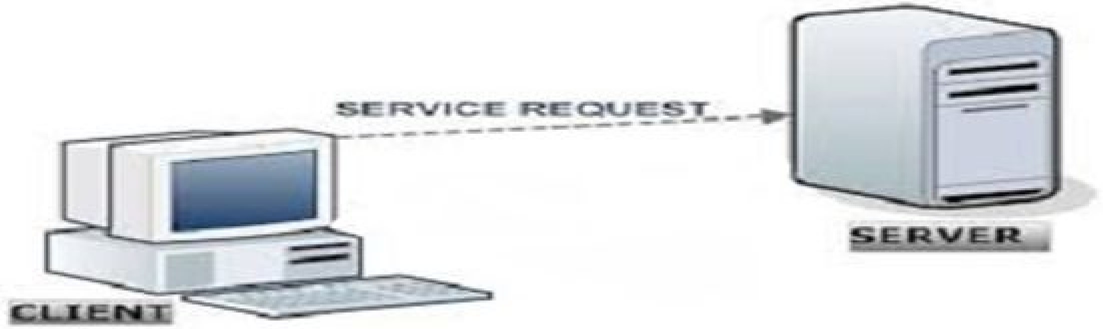
In ASP.NET there are the following 2 State Management methodologies:



Client-Side State Management

Whenever we use Client-Side State Management, the state related information will directly get stored on the client-side. That specific information will travel back and communicate with every request generated by the user then afterwards provides responses after server-side communication.

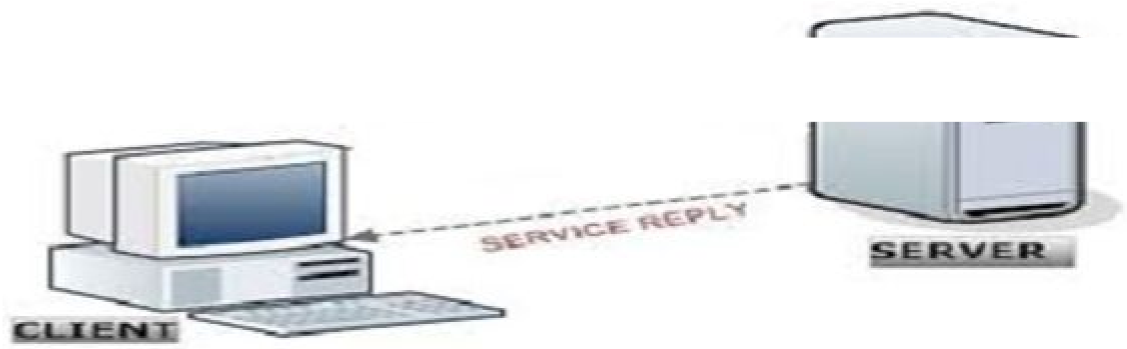
This architecture is something like the following,



Server-Side State Management

functionality there is more secure domains at the server side in comparison to Cli n

The structure is something like the following,

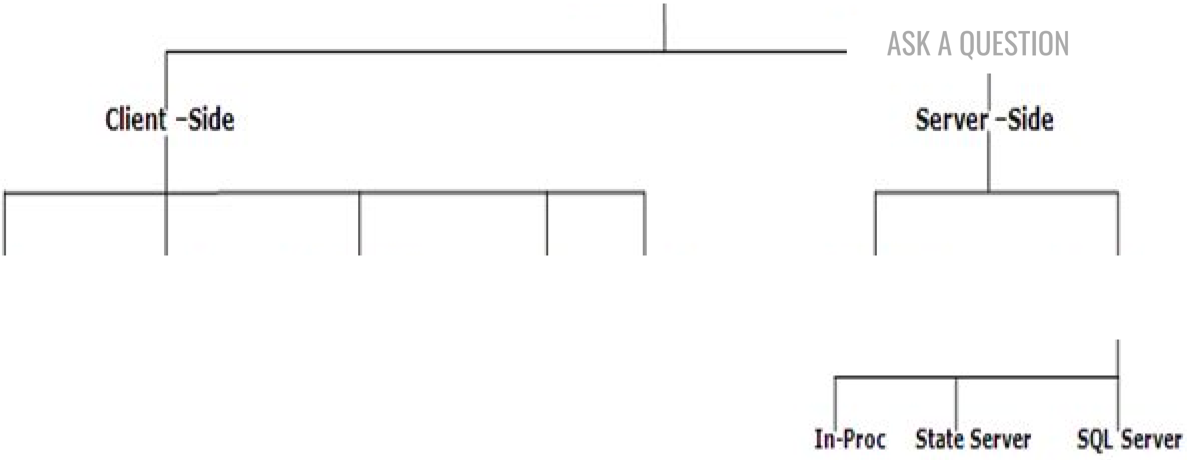


State Management Scenario

It will be a little difficult to directly evaluate what will be better for our application. We cannot directly say that we will use client-side or server-side architecture of State Management.

State Management Techniques

State Management techniques are based on client side and server side. Their functionality differs depending on the change in state, so here is the hierarchy:



Client-side | Techniques

Client-Side State Management techniques are,

- View State
- Hidden field
- Cookies
- Control State
- Query Strings

Server-side | Technique

Server-Side State Management techniques are,

- Session State
- Application State

Now I am defining each and every technique in detail with their reference example.

In general we can say it is used for storing user data in ASP.NET, sometimes in ASP.NET applications the user wants to maintain or store their data temporarily after a post-back.. In this case VIEW STATE is the most used and preferred way of doing that.

This property is enabled by default but we can make changes depending on our functionality, what we need to do is just

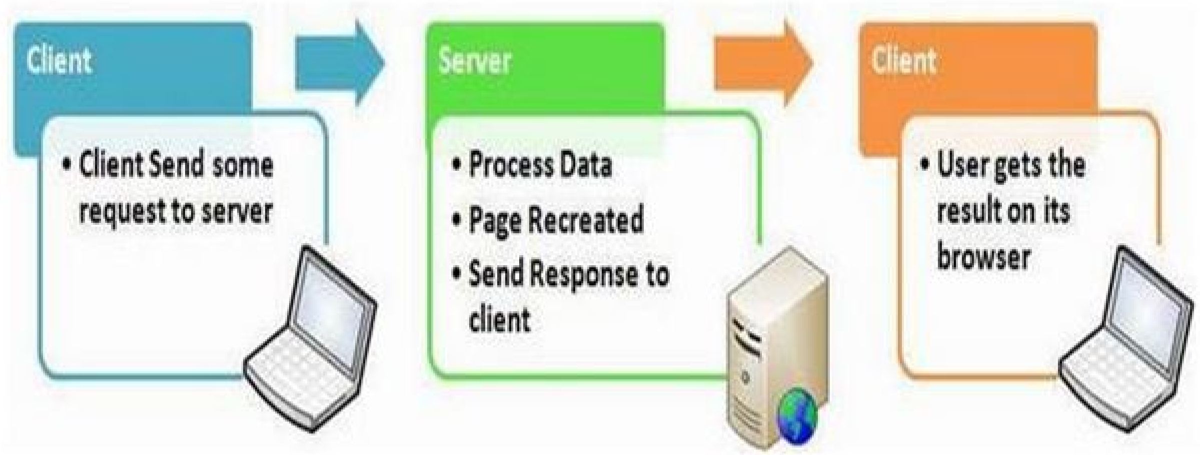

Client
• Client Send some request to server
Server
• Process Data
• Page Recreated
• Send Response to client
Client
• User gets the result on its browser

Figure: [View State Management]

```
// Page Load Event
protected void Page_Load(object sender, EventArgs e)
{
    if (IsPostBack)
    {
        if (ViewState["count"] != null)
        {
            int ViewstateVal = Convert.ToInt32(ViewState["count"]) + 1;
            View.Text = ViewstateVal.ToString();
            ViewState["count"]=ViewstateVal.ToString();
        }

        else
        {
        }
}

// Click Event
protected void Submit(object sender, EventArgs e)
{
        View.Text=ViewState["count"].ToString();
```

Points to Remember

Some of the features of view state are:

- It is page-level State Management
- Used for holding data temporarily
- Can store any type of data
- Property dependent

Hidden Field

A hidden field is used for storing small amounts of data on the client side. In most simple words it's just a container of some objects but their result is not rendered on our web browser. It is invisible in the browser.

It stores a value for the single variable and it is the preferable way when a variable's value is changed frequently but we don't need to keep track of that every time in our application or web program.

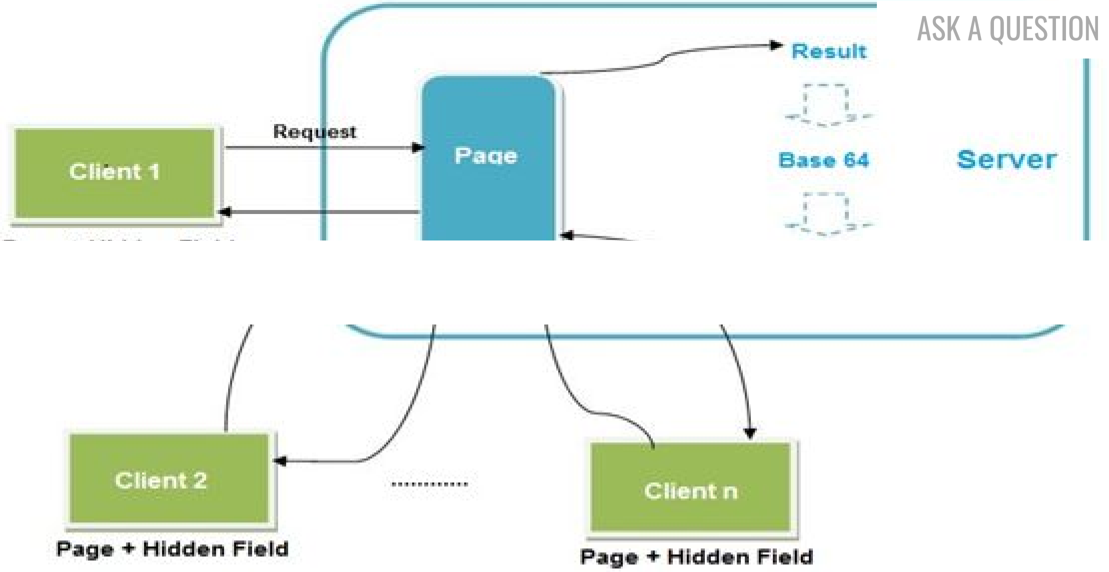


Figure: [Hidden Field Management]

```
// Hidden Field

int newVal = Convert.ToInt32(HiddenField1.Value) + 1;
HiddenField1.Value = newVal.ToString();
Label2.Text = HiddenField1.Value;
```

Points to Remember

Some features of hidden fields are:

- Contains a small amount of memory
- Direct functionality access

Cookies

A set of Cookies is a small text file that is stored in the user's hard drive using the client's browser. Cookies are just used for sake of the user's identity matching as it only stores information such as sessions id's, some frequent navigation or post-ba

Whenever we get connected to the internet for accessing a specific service, the co m our browser for identifying the user. The cookie access depends upon the life cycle or expiration of that specific cookie file.

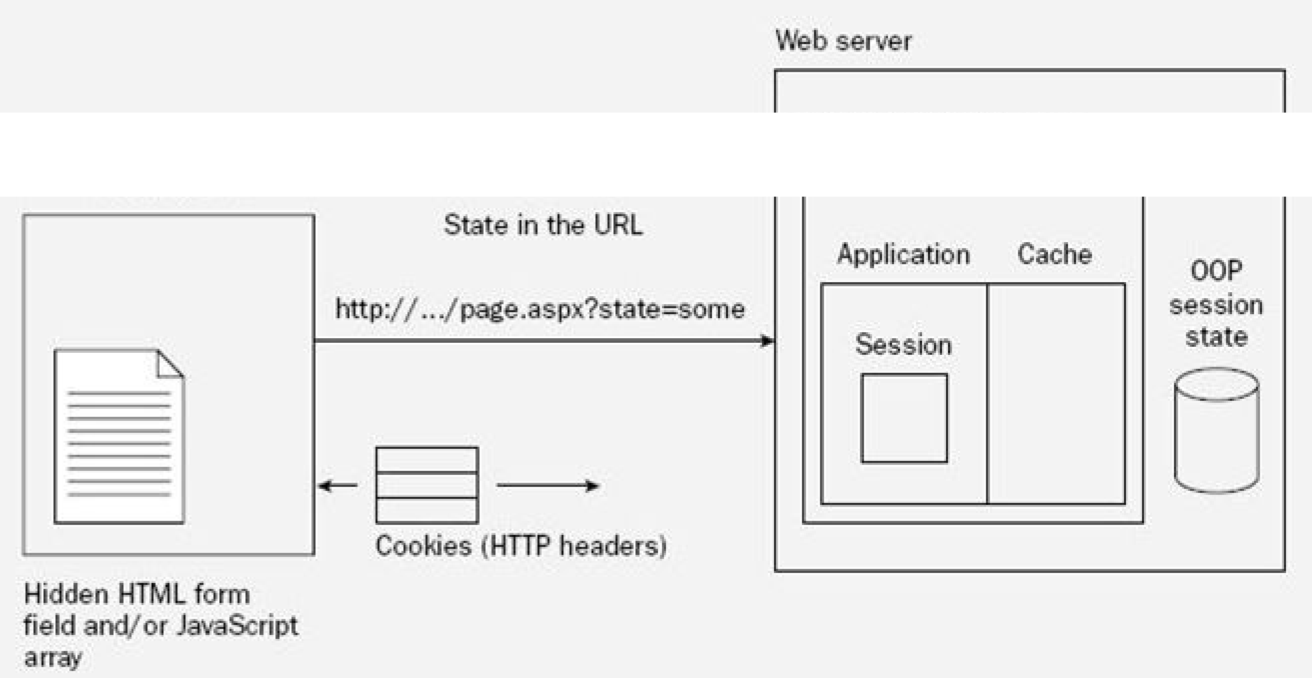
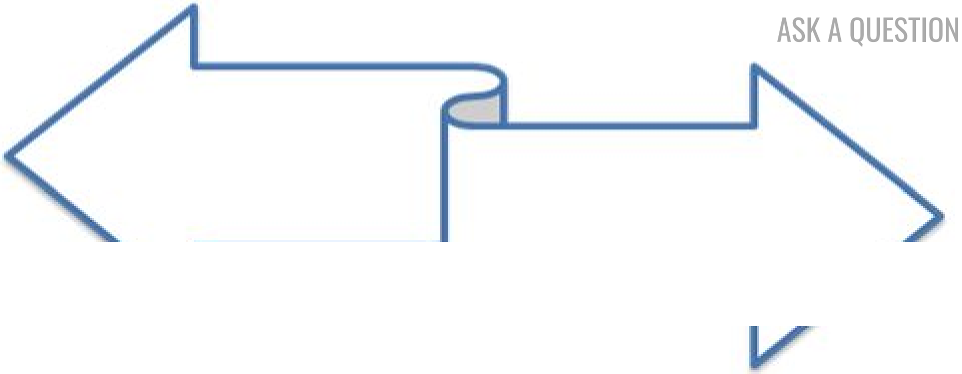


Figure: [Cookie Management]

```
int postbacks = 0;
if (Request.Cookies["number"] != null)
{
    postbacks = Convert.ToInt32(Request.Cookies["number"].Value) + 1;
}
// Generating Response
else
{
    postbacks = 1;
}
Response.Cookies["number"].Value = postbacks.ToString();

Result.Text = Response.Cookies["number"].Value;
```



Persistent Cookie

Cookies having an expiration date is called a persistent cookie. This type of cookie reaches their end as their expiration dates comes to an end. In this cookie we set an expiration date.

```
Response.Cookies["UserName"].Value = "Abhishek";
Response.Cookies["UserName"].Expires = DateTime.Now.AddDays(1);

HttpCookie aCookie = new HttpCookie("Session");
aCookie.Value = DateTime.Now.ToString();
aCookie.Expires = DateTime.Now.AddDays(1);
Response.Cookies.Add(aCookie);
```

Non-Persistent Cookie

Non-persistent types of cookies aren't stored in the client's hard drive permanently. It maintains user information as long as the user access or uses the services. Its simply the opposite procedure of a persistent cookie.

```
HttpCookie aCookie = new HttpCookie("Session");
aCookie.Value = DateTime.Now.ToString();
aCookie.Expires = DateTime.Now.AddDays(1);
Response.Cookies.Add(aCookie);
```

Points to Remember

- Store information temporarily
- It's just a simple small sized text file
- Can be changed depending on requirements
- User Preferred
- Requires only a few bytes or KBs of space for creating cookies

Control state is based on the custom control option. For expected results from CONTROL STATE we need to enable the property of view state. As I already described you can manually change those settings.

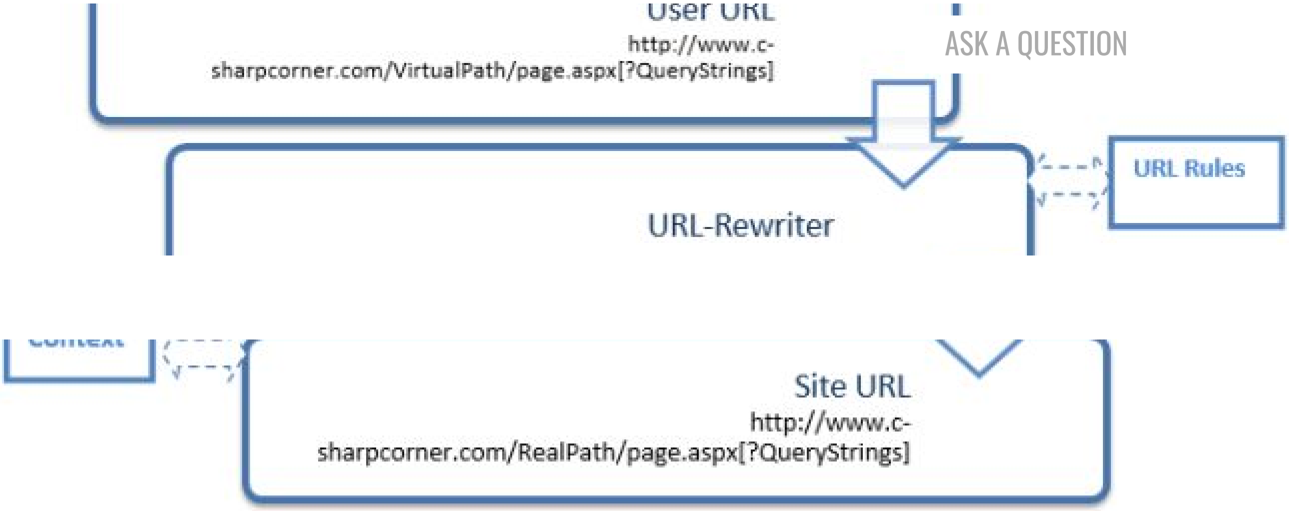
Points to Remember

Some features of query strings are:

- Used for enabling the View State Property
- Defines a custom view
- View State property declaration
- Can't be modified
- Accessed directly or disabled

Query Strings

Query strings are used for some specific purpose. These in a general case are used for holding some value from a different page and move these values to the different page. The information stored in it can be easily navigated to one page to another or to the same page as well.


User URL
http://www.c-sharpcorner.com/VirtualPath/page.aspx[?QueryStrings]
URL-Rewriter
URL Rules
Context
Site URL
http://www.c-sharpcorner.com/RealPath/page.aspx[?QueryStrings]

[Query Strings]

Figure: [Query Strings]

```
// Getting data
if (Request.QueryString["number"] != null)
{
    View.Text = Request.QueryString["number"];
}

// Setting query string
int postbacks = 0;

if (Request.QueryString["number"] != null)
{
    postbacks = Convert.ToInt32(Request.QueryString["number"]) + 1;
}
else
{
    postbacks = 1;
}

Response.Redirect("default.aspx?number=" + postbacks);
```

Some of the features are,

- It is generally used for holding values
- Works temporarily
- Switches info from one to another page
- Increase performance

I hope you have enjoyed reading this article. I'll write about Server-Side State Management very soon.

ASP.NET | Client Side Techniques | Server Side Techniques | State Management

Abhishek Jaiswal TOP 100

Geek | Blogger | Data Science Scholar | TechSavvy | Developer | Painter | Author | Trainer | Tech Evangelist | Philanthropist

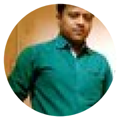
http://letustweak.com/

82 6.4m 3

View Previous Comments

24 30

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



Persistent Cookie and Non Persistent Cookie both examples are same . How can we set expiry period for Persistent Cookie?

Hidayat Sawardekar

Jan 31, 2019

1613 182 279

0 0 Reply

Control State points to remember, first point should be "Some feature of Control State" not "Some features of query strings "

Very useful thank you

Bhavani Sooriya Sep 19, 2017

1780 15 0 4 0 Reply

Very useful Explanation and when will u post server side management like this. Thank you

1774 21 0

When you will post server side state management system article .

Mahesh Kumar Dec 11, 2016

1793 2 0 2 0 Reply

Nice

Munesh Sharma May 22, 2016

156 12k 3.3m 3 0 Reply

Good and worth reading

Bhuvanesh Mohankumar Apr 27, 2016

154 12.1k 1.6m 3 1 Reply

Thanks man! :)

Abhishek Jaiswal Apr 30, 2016

82 20k 6.4m 2

Nice...

Humayun Kabir Mamun Nov 18, 2015

336 5.2k 0 3 1 Reply

Thanks buddy! :)

Abhishek Jaiswal Nov 22, 2015

82 20k 6.4m 3

Thank you guys! :)

cus
C# Corner
is the Fut

C#Corner ASK A QUESTION CONTRIBUTE

Good Job Sir

Mukesh Kumar

Nov 09, 2015

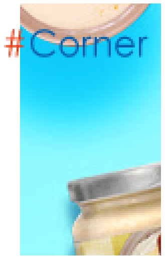
79 **20.3k** **7.2m**

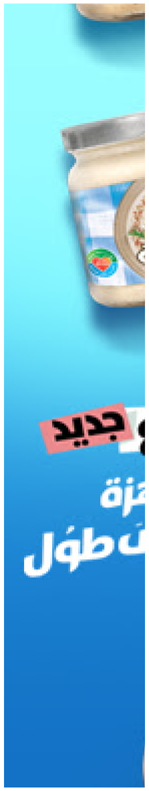
3 0 Reply

cus
C# Corner
: is the Fut

C# Corner

ASK A QUESTION CONTRIBUTE





TRENDING UP

01 Angular App With ASP.NET Core And Cosmos DB

02 The Future of .NET

Implement Microsoft And Twitter Based Authentication In ASP.NET Core 2.2

cus

C# Corner

: is the Fut

04 .NET 5 Is The Future Of .NET - What Every .NET Developer Must Know

05 The Top Seven Least-Known, Yet Important, C# Features

06 Performance Of Loops In C#

07 What Is .NET Core

08 What is Scrum? Who is Scrum Master In Agile Process?

09 Candidate Features For C# 9

10 Angular 8.0 - What's New And How To Upgrade

View All ○

cus

: is the Fut

C# Corner