FullStack.Cafe - Tech Interview Plan

Q1: What is ASP.NET Core? ☆☆

Topics: ASP.NET

Answer:

ASP.NET Core is a brand new cross-platform web framework built with .NET Core framework. It is not an update to existing ASP.NET framework. It is a complete rewrite of the ASP.NET framework. It works with both .NET Core and .NET Framework.

Main characterestics of ASP.NET Core:

- DI Container which is quite simple and built-in. You can extend it with other popular DI containers
- Built-in and extensible structured logging. You can redirect output to as many sources as you want (file, Azure, AWS, console)
- Extensible strongly typed configuration, which can also be used to reload at run-time
- Kestrel – new, cross-platform and super fast web server which can stand alone without IIS, Nginx or Apache
- New, fully async pipeline. It is easily configured via middleware
- ASP.NET All meta package which improves development speed, and enables you to reference all Microsoft packages for ASP.NET Core and it will deploy only those that are being used by your code
- There is no *web.config*. We now use *appsettings.json* file in combination with other sources of configuration (command line args, environment variables, etc.)
- There is no *Global._asax – We have _Startup.cs* which is used to set up Middleware and services for DI Container.

Q2: Can ASP.NET Core work with the .NET framework? ☆☆

Topics: ASP.NET

Answer:

Yes. This might surprise many, but ASP.NET Core works with .NET framework and this is officially supported by Microsoft.

ASP.NET Core works with:

- .NET Core framework
- .NET framework

Q3: What is the difference between ASP.NET and ASP.NET MVC? ☆☆

Topics: ASP.NET

Answer:

ASP.NET, at its most basic level, provides a means for you to provide general HTML markup combined with server side "controls" within the event-driven programming model that can be leveraged with VB, C#, and so on. You define the page(s) of a site, drop in the controls, and provide the programmatic plumbing to make it all work.

ASP.NET MVC is an application framework based on the Model-View-Controller architectural pattern. This is what might be considered a "canned" framework for a specific way of implementing a web site, with a page acting as the "controller" and dispatching requests to the appropriate pages in the application. The idea is to "partition" the various elements of the application, eg business rules, presentation rules, and so on.

Think of the former as the "blank slate" for implementing a site architecture you've designed more or less from the ground up. MVC provides a mechanism for designing a site around a pre-determined "pattern" of application access, if that makes sense. There's more technical detail to it than that, to be sure, but that's the nickel tour for the purposes of the question.

Q4: What is `ViewState`? ☆☆

Topics: ASP.NET

Answer:

View State is the method to preserve the Value of the Page and Controls between round trips. It is a *Page-Level State Management* technique. View State is turned on by default and normally serializes the data in every control on the page regardless of whether it is actually used during a post-back.

A web application is stateless. That means that a new instance of a page is created every time when we make a request to the server to get the page and after the round trip our page has been lost immediately

Q5: What is a *postback*? ☆☆

Topics: ASP.NET

Answer:

A **postback** originates from the client browser. Usually one of the controls on the page will be manipulated by the user (a button clicked or dropdown changed, etc), and this control will initiate a *postback*. The state of this control, plus all other controls on the page (known as the View State) is Posted Back to the web server.

Q6: What is the meaning of Unobtrusive JavaScript? ☆☆☆

Topics: ASP.NET

Answer:

This is a general term that conveys a general philosophy, similar to the term REST (Representational State Transfer). Unobtrusive JavaScript doesn’t intermix JavaScript code in your page markup.

Eg : Instead of using events like onclick and onsubmit, the unobtrusive JavaScript attaches to elements by their ID or class based on the HTML5 data- attributes.

Q7: In which event of page cycle is the `ViewState` available? ☆☆☆

Topics: ASP.NET

Answer:

After the `Init()` and before the `Page_Load()`.

Q8: What are the different types of caching? ☆☆☆

Topics: ASP.NET

Answer:

ASP.NET has 3 kinds of caching :

1. Output Caching,
2. Fragment Caching,
3. Data Caching.

Q9: What is ViewState? How is it encoded? Is it encrypted? Who uses ViewState? ☆☆☆

Topics: ASP.NET

Answer:

View state is a kind of hash map (or at least you can think of it that way) that ASP.NET uses to store all the temporary information about a page - like what options are currently chosen in each select box, what values are there in each text box, which panel are open, etc. You can also use it to store any arbitrary information.

The entire map is serialized and encoded and kept in a *hidden variable* (__VIEWSTATE form field) that's posted back to the server whenever you take any action on the page that requires a server round trip. This is how you can access the values on the controls from the server code. If you change any value in the server code, that change is made in the view state and sent back to the browser.

Just be careful about how much information you store in the view state, though... it can quickly become bloated and slow to transfer each time to the server and back.

It's not encrypted at all. Just base encoded, which easily reversible.

Q10: What are the different validators in ASP.NET? ☆☆☆

Topics: ASP.NET

Answer:

Client-Side Validation: When validation is done on the client browser, then it is known as Client-Side Validation. We use JavaScript to do the Client-Side Validation.

**Server-Side Validation: ** When validation occurs on the server, then it is known as Server-Side Validation. Server-Side Validation is a secure form of validation. The main advantage of Server-Side Validation is if the user somehow bypasses the Client-Side Validation, we can still catch the problem on server-side.

Q11: What is the difference between ASP.NET Core Web (.NET Core) vs ASP.NET Core Web (.NET Framework)? ☆☆☆☆

Topics: ASP.NET

Answer:

- **ASP.NET Core on .NET Core** is cross-platform ASP.NET Core. It can run on Windows, Mac, and Linux (including Docker). The server doesn't need .NET Core installed - the dependencies can be bundled with the application.

  ASP.NET Core ships entirely as NuGet packages. This allows you to optimize your app to include only the necessary NuGet packages. In fact, ASP.NET Core 2.x apps targeting .NET Core only require a single NuGet package. The benefits of a smaller app surface area include tighter security, reduced servicing, and improved performance.

  It is not required to install .Net framework to run asp.net core with .net core application. An ASP.NET Core application with .net core is a console app that creates a web server in its Main method. It uses Kestrel web server to run the application.

- **ASP.NET Core on .NET Framework** is ASP.NET Core on the "full" or "desktop" .NET Framework (e.g. .NET Framework 4.6.2). These applications can only run on Windows, but everything else about ASP.NET Core behaves the same way. It also supports Aspx, WPF, WCF and WebServices.

Q12: Which type if caching will be used if we want to cache the portion of a page instead of whole page? ☆☆☆☆

Topics: ASP.NET

Answer:

Fragment Caching: It caches the portion of the page generated by the request. For that, we can create user controls with the below code:

```
<%@ OutputCache Duration="120" VaryByParam="CategoryID;SelectedID"%>
```

Q13: How we can force all the validation controls to run? ☆☆☆☆

Topics: ASP.NET

Answer:

The `Page.Validate()` method is used to force all the validation controls to run and to perform validation.

Q14: What are the different types of cookies in ASP.NET? ☆☆☆☆

Topics: ASP.NET

Answer:

- **Session Cookie** - Resides on the client machine for a single session until the user does not log out.
- **Persistent Cookie** - Resides on a user's machine for a period specified for its expiry, such as 10 days, one month, and never.

Q15: What is the difference between *classic* and *integrated* pipeline mode in IIS7? ☆☆☆☆

Topics: ASP.NET

Answer:

- **Classic mode** (the only mode in IIS6 and below) is a mode where IIS only works with ISAPI extensions and ISAPI filters directly. In this mode, ASP.NET is not much different from PHP or other technologies for IIS. This separation of the IIS and ASP.NET request-processing models results in duplication of some processing steps, such as authentication and authorization.

- **Integrated mode**, on the other hand, is a new mode in IIS7 where IIS pipeline is tightly integrated (i.e. is just the same) as ASP.NET request pipeline. ASP.NET can see every request it wants to and manipulate things along the way. In this mode, ASP.NET `HttpModules` basically have nearly as much power as an ISAPI filter would have had and ASP.NET `HttpHandlers` can have nearly equivalent capability as an ISAPI extension could have.

Q16: What is Katana? ☆☆☆☆

Topics: ASP.NET

Answer:

Katana is a fully developed framework made to make a bridge between current ASP.NET frameworks and OWIN specification. Katana is a set of components by Microsoft built using OWIN specifications. Some of these components include Web API, ASP.NET Identity and SignalR. vNext is the successor to Katana (which is why they look so similar) and ASP.NET vNext is now known as ASP.NET 5.

Q17: How to choose between ASP.NET 4.x and ASP.NET Core? ☆☆☆☆

Topics: ASP.NET

Answer:

- **ASP.NET Core** is an open-source, cross-platform framework for building cloud-based web apps on Windows, macOS, or Linux. It has higher performance than ASP.NET 4.x and uses .NET Core Runtime and there is no dependency on `System.Web`.

- **ASP.NET 4.x** is a mature framework that provides the services needed to build enterprise-grade, server-based web apps on Windows.

You should use .NET Core for your server application when:

- You have cross-platform needs.
- You are targeting microservices.
- You are using Docker containers.
- You need high performance and scalable systems.
- You need side by side of .NET versions by application.

You should use .NET Framework for your server application when:

- Your application currently uses .NET Framework (recommendation is to extend instead of migrating)
- You need to use third-party .NET libraries or NuGet packages not available for .NET Core.
- You need to use .NET technologies that are not available for .NET Core.
- You need to use a platform that doesn’t support .NET Core.

Q18: What is an `HttpHandler` in ASP.NET? Why and how is it used? ☆☆☆☆

Topics: ASP.NET

Answer:

In the simplest terms, an ASP.NET `HttpHandler` is a class that implements the `System.Web.IHttpHandler` interface.

ASP.NET HTTPHandlers are **responsible for intercepting requests** made to your ASP.NET web application server. They run as processes in response to a request made to the ASP.NET Site. The most common handler is an ASP.NET page handler that processes .aspx files. When users request an .aspx file, the request is processed by the page through the page handler.

ASP.NET offers a few default HTTP handlers:

- Page Handler (.aspx): handles Web pages
- User Control Handler (.ascx): handles Web user control pages
- Web Service Handler (.asmx): handles Web service pages
- Trace Handler (trace.axd): handles trace functionality

Q19: What is the difference between `<system.web>` and `<system.webServer>`? ☆☆☆☆

Topics: ASP.NET

Answer:

The **system.web** section is for configuring IIS 6.0, while the **system.webserver** version is used to configure IIS 7.0. IIS 7.0 includes a new ASP.NET pipeline and some configuration differences, hence the extra config sections.

The former is for Classic Mode. The latter is for Integrated Pipeline Mode (available in IIS7+).

Q20: What is `HttpModule` in ASP.Net? ☆☆☆☆

Topics: ASP.NET

Answer:

HTTP modules intercept incoming requests and inject pre-processing logic in the ASP.Net request processing pipeline.

There are two ways in which you can inject logic in the request pipeline of an ASP.NET application - `HttpHandlers` and `HttpModules`.

An `HttpModule` is a component that is part of the ASP.NET request processing pipeline and is called on **every request** that is made to your application. `HttpModules` can have access to the life cycle events of a request and hence they can be used to modify the response as well.