ASP.NET Core Interview Questions

This post is about ASP.NET Core Interview Questions. These questions are guidelines to assess the candidate about ASP.NET Core.These interview question covers basic to advance and will help you to prepare for the interviews, quick revision and provide strength to your technical skills.

Q. What is .NET Core?

Ans: .NET Core is a newer version of .NET, which is cross-platform, supporting Windows, MacOS and Linux, and can be used in device, cloud, and embedded/IoT

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

scenarios.

The following characteristics best define .NET Core:

- **Flexible deployment:** Can be included in your app or installed side-by-side user or machine-wide.
- **Cross-platform:** Runs on Windows, MacOS and Linux; can be ported to other OSes. The supported Operating Systems (OS), CPUs and application scenarios will grow over time, provided by Microsoft, other companies, and individuals.
- **Command-line tools**: All product scenarios can be exercised at the command-line.
- **Compatible**: .NET Core is compatible with .NET Framework, Xamarin and Mono, via the .NET Standard Library.

Q. How is it different from existing .NET framework?

Ans: Read Difference between .NET Core and .NET Framework

Q. What are .NET Platform Standards?

Ans: Read .NET Platform Standards

Q. What is ASP.NET Core?

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Ans: ASP.NET Core 1.0 is the next version of ASP.NET. It is open source and cross-platform framework (supports for Windows, Mac and Linux) suitable for building cloud based internet connected applications like web apps, IoT apps and mobile apps. ASP.NET Core apps can run on .NET Core or on the full .NET Framework.

Q. What are the newly introduced functionalities in ASP.NET Core?

Ans: Read Quick summary of what's changed in ASP.NET Core. ASP.NET Core 2.0 is already out and there are few changes and new things introduced. Read What's new in ASP.NET Core 2

Q. Why Use ASP.NET Core for Web Application Development?

Ans: ASP.NET Core is an robust, and feature-rich framework that provides features to develop super-fast APIs for web apps. ASP.NET Core should be prefered for following reason:

- ASP.NET Core is faster compare to traditional ASP.NET.
- **Cross-platform:** Runs on Windows, MacOS and Linux; can be ported to other OSes. The supported Operating Systems (OS), CPUs and application scenarios

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

will grow over time, provided by Microsoft, other companies, and individuals.

- **Flexible deployment:** Can be included in your app or installed side-by-side user or machine-wide. Runs on IIS or can be self-hosted in your own process.
- Built-in support for dependency injection.
- ASP.NET Core is cloud-ready and has improved support for cloud deployment.
- Provides Support for Popular JavaScript Frameworks.
- Unification Of Development Models which allows the MVC and Web API development models to use the same base class Controller.
- Razor Pages makes coding page-focused scenarios easier and more productive.
- Environment based configuration supported for cloud deployment.
- Built in logging support.
- In ASP.NET we had modules and handlers to deal with requests. In ASP.NET Core we have middleware which provides more control how the request should be processed as they are executed in the order in which they are added.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Q. What is ASP.NET Core Middleware and How it is different from HttpModule?

Ans: Read my post about How ASP.NET Core 1.0 Middleware is different from HttpModule

Q. What are the various JSON files in ASP.NET Core?

Ans: Read my post about Various JSON files in ASP.NET Core

Q. What is Startup.cs file in ASP.NET Core?

Ans: In ASP.NET, Global.asax (though optional) acts as the entry point for your application. Startup.cs, it is entry point for application itself. The Startup class configures the request pipeline that handles all requests made to the application. Read this read this excellent post The Startup.cs File in ASP.NET Core 1.0 – What Does It Do? to know more about startup.cs.

Q.What ConfigureServices() method does in Startup.cs?

Ans: This method is optional. It is the place to add services required by the application. For example, if you wish to use Entity Framework in your application then you can add in this method.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
public void ConfigureServices(IServiceCollection services)
{
    services.Configure<AppSettings>(Configuration.GetSubKey
    services.AddEntityFramework()
            .AddSqlServer()
            .AddDbContext<SchoolContext>();
    // Add MVC services to the services container.
    services.AddMvc();
}
```

Q.What Configure() method does in Startup.cs?

Ans: The `Configure` method is used to specify how the ASP.NET application will respond to HTTP requests. The request pipeline is configured by adding middleware components to an IApplicationBuilder instance that is provided by dependency injection. There are some built-in middlewares for error handling, authentication, routing, session and diagnostic purpose. Highlighted lines in below code, are built-in Middleware with ASP.NET Core 1.0.

```
public void Configure(IApplicationBuilder app, IHostingEnv
{
    loggerFactory.AddConsole(Configuration.GetSection("Log
    loggerFactory.AddDebug();

    app.UseApplicationInsightsRequestTelemetry();

    if (env.IsDevelopment())
    {
        app.UseBrowserLink();
        app.UseDeveloperExceptionPage();
    }
    else
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
14      {
15          app.UseExceptionHandler("/Home/Error");
16
17          // For more details on creating database during de
18          try
19          {
20              using (var serviceScope = app.ApplicationServi
21                  .CreateScope())
22              {
23                  serviceScope.ServiceProvider.GetService<Ap
24                          .Database.Migrate();
25              }
26          }
27          catch { }
28      }
29
30      app.UseIISPlatformHandler(options => options.Authentic
31      app.UseStaticFiles();
32      app.UseIdentity();
33
34      // To configure external authentication please see htt
35
36      app.UseMvc(routes =>
37      {
38          routes.MapRoute(
39              name: "default",
40              template: "{controller=Home}/{action=Index}/{i
41      });
42  }
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Q.What is the difference between app.Use vs app.Run while adding middleware?

Ans: Read app.Use vs app.Run in ASP.NET Core middleware.

Q.What is the difference between services.AddTransient and service.AddScope methods are Asp.Net Core?

Ans:ASP.NET Core out of the box supports dependency injection. These 3 methods allows to register the dependency. However, they offers different lifetime for registered service. **Transient** objects are created for every request (when requested). This lifetime works best for lightweight, stateless services. **Scoped** objects are the same within a request, but different across different requests. **Singleton** objects created the first time they're requested (or when `ConfigureServices` is run and an instance is specified with the service registration).

Q.What is Kestral?

Ans: Kestrel is a cross-platform web server for ASP.NET Core based on libuv, a cross-platform asynchronous I/O library. Kestrel is the web server that is included

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

by default in ASP.NET Core new project templates. If your application accepts requests only from an internal network, you can use Kestrel by itself.

If you expose your application to the Internet, you must use IIS, Nginx, or Apache as a reverse proxy server. A reverse proxy server receives HTTP requests from the Internet and forwards them to Kestrel after some preliminary handling. The most important reason for using a reverse proxy for edge deployments (exposed to traffic from the Internet) is security. Kestrel is relatively new and does not yet have a full complement of defenses against attacks.

Q.What is WebListener?

Ans: ASP.NET Core ships two server implementations **Kestral** and **WebListener**. WebListener is also a web server for ASP.NET Core that runs only on Windows. It's built on the Http.Sys kernel mode driver. WebListener is an alternative to Kestrel that can be used for direct connection to the Internet without relying on IIS as a reverse proxy server.

Q.What is ASP.NET Core Module (ANCM)?

Ans: ASP.NET Core Module (ANCM) lets you run ASP.NET Core applications behind IIS and it works only with Kestrel; it isn't compatible with WebListener. ANCM is a

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

native IIS module that hooks into the IIS pipeline and redirects traffic to the backend ASP.NET Core application. ASP.NET Core applications run in a process separate from the IIS worker process, ANCM also does process management. ANCM starts the process for the ASP.NET Core application when the first request comes in and restarts it when it crashes. In short, it sits in IIS and routes the request for ASP.NET Core application to Kestral.

Q.What are different ASP.NET Core diagnostic middleware and how to do error handling?

Ans: Read Various ASP.NET Core Diagnostics Middleware.

Q.What is a Host and what's the importance of Host in ASP.NET Core application?

Ans: ASP.NET Core apps require a host in which to execute. The host is responsible for application startup and lifetime management. Other responsibility of host's includes ensuring the application's services and the server are available and properly configured. **Don't confuse yourself with a Server.** The host is responsible for starting the app and its management, where the server is responsible for accepting HTTP requests. The host is configured to use a particular server; the server is unaware of its host.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

The host is typically created using an instance of a `WebHostBuilder`, which builds and returns a `WebHost` instance. The `WebHost` references the server that will handle requests.

```
1  public class Program
2      {
3          public static void Main(string[] args)
4          {
5              var host = new WebHostBuilder()
6                  .UseKestrel()
7                  .UseContentRoot(Directory.GetCurrentDirectory())
8                  .UseIISIntegration()
9                  .UseStartup<Startup>()
10                 .Build();
11
12             host.Run();
13         }
14     }
```

netcore1.1_program.cs hosted with ♥ by **GitHub** **view raw**

In ASP.NET Core 2.0, it is changed. It looks like this:

```
1  public class Program
2  {
3      public static void Main(string[] args)
4      {
5          BuildWebHost(args).Run();
6      }
7
```

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

```
8       public static IWebHost BuildWebHost(string[] args) =>
9           WebHost.CreateDefaultBuilder(args)
10              .UseStartup<Startup>()
11              .Build();
12  }
```

netcore2.0_program.cs hosted with ♥ by GitHub view raw

Read this post to know more.

Q. What does WebHost.CreateDefaultBuilder() do?

Ans: This method does the following things.

- Configure the app to use Kestrel as web server.
- Specify to use the current project directory as root directory for the application.
- Setup the configuration sub-system to read setting from appsettings.json and appsettings.{env}.json to environment specific configuration.
- Set Local user secrets storage only for the development environment.
- Configure environment variables to allow for server-specific settings.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

- Configure command line arguments (if any).
- Configure logging to read from the Logging section of the appsettings.json file and log to the Console and Debug window.
- Configure integration with IIS
- Configure the default service provider.

You can take a look at the code of this method here.

Q.What is the role of IHostingEnvironment interface in ASP.NET Core?

Ans: ASP.NET Core offers an interface named `IHostingEnvironment`, allows you to programmatically retrieve the current environment so you can have an environment-specific behaviour. By default, ASP.NET Core has 3 environments Development, Staging, and Production. Previously, the developers have to build the application differently for each environment (Staging, UAT, Production) due to dependency on config file sections and the preprocessor directive applicable at compile time. ASP.NET Core takes a different approach and uses `IHostingEnvironment` to retrieve the current environment. You can also define a custom environment like QA or UAT.

Q.How to handle 404 error in ASP.NET Core 1.0?

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Ans: Read How to handle 404 error in ASP.NET Core 1.0

Q.What is the use of UseIISIntegration?

Ans: This tells ASP.NET that IIS will be working as a reverse proxy in front of Kestrel. As if you expose your application to the Internet, you must use IIS, Nginx, or Apache as a reverse proxy server. When you wish to deploy your ASP.NET Core application on windows, you need to tell ASP.NET Core Host to use IIS integration. `UseKestrel` and `UseIISIntegration` must be used in conjunction as `UseKestrel` creates the web server and hosts the code. `UseIISIntegration` specifies IIS as the reverse proxy server.

```
var host = new WebHostBuilder()
          .UseKestrel()
          .UseContentRoot(Directory.GetCurrentDirectory())
          .UseIISIntegration()
          .UseStartup<Startup>()
          .Build();
```

Note that code making call to `UseIISIntegration()` does not affect code portability.

Q.What are the different ways for bundling and minification in ASP.NET Core?

Ans: There are different ways for doing bundling and minification in ASP.NET Core.

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

- Gulp – was the default choice for ASP.NET Core till beta versions. Later it was removed due to performance and speed issue and replaced with BundlerMinifier. Read this post to find out the reasons of this decision.
- BundlerMinifier – is a Visual Studio extension and it's default choice now. You should see bundleconfig.json file in your default template. Read this post to know more about bundling and minification in ASP.NET Core.
- ASP.NET Core Web Optimizer – ASP.NET Core middleware for bundling and minification of CSS and JavaScript files at runtime.
- Grunt – can also be used with ASP.NET Core.

For more Read Overview of Tools for bundling and minification in ASP.NET Core.

Q.What is launchsetting.json in ASP.NET Core?

Ans: Read Launchsetting.json in ASP.NET Core

Q.What is wwwroot folder in ASP.NET Core?

Ans: In ASP.NET Core, all the static resources, such as CSS, JavaScript, and image files are kept under the `wwwroot` folder (default). You can also change the name of this folder.

Q.What is "Razor pages" in ASP.NET Core?

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

Ans: "Razor Pages" is a new feature of ASP.NET Core and it was released with ASP.NET Core 2.0 release. Razor Pages are simple pages or views without controllers and introduced with the intent of creating page focused scenarios where there is no real logic is involved. You will find razor pages similar to ASP.NET Web Forms. They work on the convention and need to be placed in Pages folder and the extension is .cshtml. Razor pages uses handler methods to deal with incoming HTTP request. Read this post to find out more about handler methods.

Q.What is tag helper in ASP.NET Core?

Ans: Tag Helpers enable server-side code to participate in creating and rendering HTML elements in Razor files. For example, the built-in `ImageTagHelper` can append a version number to the image name. There are many built-in tag helpers like From, Image, Anchor and many others. Read How to use image tag helper in ASP.NET Core and How to use select tag helper in ASP.NET Core. Read more about Tag Helpers @ official documentation.

Q.What's new in .NET Core 2.1 and ASP.NET Core 2.1?

Ans: .NET Core 2.1 and ASP.NET Core 2.1 brings lots of new features on the table. Read New features of .NET Core 2.1 to know more about .NET Core 2.1. Where,

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

ASP.NET Core 2.1 introduces supports for SignalR, HTTPS by default, introduction of HttpClientFactory and many other. Read Quick summary of what's new in ASP.NET Core 2.1. Also read the migration guide to migrate ASP.NET Core 2.0 app to 2.1.

Q.What are the differences between ASP.NET MVC 5 and ASP.NET MVC Core 1.0?

Ans: Read Difference between ASP.NET MVC 5 and ASP.NET MVC Core 1.0.

Note: This page will be updated on timely basis. So keep visiting for new set of questions. Meanwhile please check others post to dive into ASP.NET Core.

PS: If you found this content valuable and want to return the favour, then

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD

RECENT POSTS

Add Authentication to Angular 7 App using ASP.NET Core 3

ASP.NET Core 3.0 App with .NET Core 3.0 preview 2 release

Visual Studio 2019 Preview 2 release new features

How You Can Create a .NET Core Application Using Entity Framework Core with Oracle

JOIN THE NEWSLETTER

Enter your email address to subscribe to this blog and receive notifications of new posts by email.

Email Address

Contact

Privacy Policy

FOLLOW US

Create PDF in your applications with the Pdfcrowd HTML to PDF API

PDFCROWD

7 Simple Hacks to Supercharge Your Angular 7 App

SUBSCRIBE

Create PDF in your applications with the Pdfcrowd HTML to PDF API PDFCROWD