

1. Which of these is not a built-in IoC container lifetime for DI?

Answer: (Please choose the correct answer)

- Singleton
- Transient
- Scoped
- Interface

Correct Answer: Interface

In ASP.NET Core, the built-in IoC (Inversion of Control) container supports the following lifetimes:

- **Singleton:** A single instance is created and shared throughout the application's lifetime.
- **Transient:** A new instance is created each time it is requested.
- **Scoped:** A new instance is created per each request (or scope).

An Interface is not a lifetime; it's a type used to define a contract that classes can implement.

2. Which of the below is a major limitation of Dependency Injection (DI)?

Answer: (Please choose the correct answer)

- Extensibility
- Testability
- Maintainability
- High Coupling

Correct Answer: High Coupling

Dependency Injection (DI) aims to reduce coupling between components by injecting dependencies rather than hard-coding them within classes. Among the given options, High Coupling is generally considered a limitation (although it's actually the opposite of what DI intends to solve). The correct answer in context is High Coupling, indicating that it would be the major limitation DI aims to address, making the question somewhat paradoxical.

3. Which one of the given options is false with respect to ConfigureServices?

Answer: (Please choose the correct answer)

- It is required to start up a Core App
- It is called by the host before the Configure method to configure the app's services.
- Configuration options are set by convention.

- Adding services to the service container makes them available within the app and in the Configure method.

Correct Answer: Configuration options are set by convention.

In the context of the ConfigureServices method in ASP.NET Core:

- **It is required to start up a Core App:** True. The ConfigureServices method is used to add services to the container, which are then available to the application.
- **It is called by the host before the Configure method to configure the app's services:** True. The ConfigureServices method is called by the runtime before the Configure method.
- **Configuration options are set by convention:** False. Configuration in ConfigureServices is not set by convention; rather, it is explicitly defined by the developer.
- **Adding services to the service container makes them available within the app and in the Configure method:** True. Services added in ConfigureServices are available throughout the app, including in the Configure method.

4. What happens when multiple configure method calls exist to process a pipeline without using a startup class?

Answer: (Please choose the correct answer)

- The last Configure call is used.
- The first Configure call is used.
- All the Configure calls are executed.
- Multiple Configure method calls cannot be made.

Correct Answer: The last Configure call is used.

In ASP.NET Core, if multiple Configure method calls exist to process a pipeline without using a startup class, the framework will use the last Configure method call.

5. Considering the following statements:

- **Statement 1:** The middleware added using ApplicationBuilder.Use may call the next middleware whereas the middleware added using ApplicationBuilder.Run never calls the next middleware.
- **Statement 2:** The configure method defines how the application will respond to each HTTP request.

Choose the correct answer based on the above 2 statements.

Answer: (Please choose the correct answer)

- Statement 1 is correct
- Statement 2 is correct
- Statement 1 is incorrect
- Statement 2 is correct

- Statement 1 is correct
- Statement 2 is incorrect
- Statement 1 is incorrect
- Statement 2 is incorrect

Correct Answer: Statement 1 is incorrect, Statement 2 is correct.

- **Statement 1 is incorrect:** In ASP.NET Core, middleware added using `ApplicationBuilder.Use` can decide whether to call the next middleware in the pipeline by invoking the next delegate. Middleware added using `ApplicationBuilder.Run` is typically terminal and does not call the next middleware.
- **Statement 2 is correct:** The `Configure` method defines how the application will respond to each HTTP request by setting up the request processing pipeline.

6. In the `Configure` method, we can configure the application (1) for your application using (2) instance that is provided by the built-in IoC container.

Answer: (Please choose the correct answer)

- (1) request pipeline (2) `IAApplicationBuilder`
- (1) result pipeline (2) `IAApplicationBuilder`
- (1) `IAApplicationBuilder` (2) request pipeline
- (1) `IAApplicationBuilder` (2) result pipeline

Correct Answer: (1) request pipeline (2) `IAApplicationBuilder`

In the `Configure` method, you configure the application (1) request pipeline using the (2) `IAApplicationBuilder` instance that is provided by the built-in IoC container.

7. Which one of these is not an inbuilt middleware found in .NET Core?

Answer: (Please choose the correct answer)

- `app.UseBrowserLink();`
- `app.UseStaticFiles();`
- `app.UseIdentity();`
- `app.EnableCaching();`

Correct Answer: `app.EnableCaching();`

In .NET Core, the following are inbuilt middleware:

- `app.UseBrowserLink();`
- `app.UseStaticFiles();`
- `app.UseIdentity();` (in earlier versions, now replaced with `app.UseAuthentication()` and `app.UseAuthorization()`)
- `app.EnableCaching();` is not an inbuilt middleware in .NET Core.

8. Which one of these Middleware comes at the earliest stage of the complete request processing pipeline for ASP.NET Core?

Answer: (Please choose the correct answer)

- ExceptionHandler
- Routing
- CORS
- Authentication

Correct Answer: CORS

In ASP.NET Core, CORS (Cross-Origin Resource Sharing) middleware typically comes early in the request processing pipeline to handle cross-origin requests before other middleware like authentication or routing are applied. This allows the server to respond appropriately to CORS preflight requests and handle cross-origin requests before further processing.

9. Which of the below is the default location of static files in an ASP.NET Core Application?

Answer: (Please choose the correct answer)

- inetpub
- ViewBag
- UseWebRoot
- wwwroot

Correct Answer: wwwroot

In an ASP.NET Core application, the default location for static files is the wwwroot folder.

10. Suppose you have created a script in your package.json file that runs lite-server named app-dev. Which of the following command-line scripts will run the respective script in your package.json file?

Answer: (Please choose the correct answer)

- npm app-dev
- node app-dev
- npm run app-dev
- node run app-dev

Correct Answer: npm run app-dev

To run a script named app-dev defined in your package.json file, you use the npm run command followed by the script name.

11. Which of these describes LibMan the best?

Answer: (Please choose the correct answer)

- Library Manager (LibMan) is a lightweight, client-side library acquisition tool.
- LibMan CLI is installed from the global package.
- The libman start command creates a libman.json file if one doesn't exist.
- LibMan does not support .NET CORE.

Correct Answer: Library Manager (LibMan) is a lightweight, client-side library acquisition tool.

LibMan (Library Manager) is designed for managing client-side libraries in web projects and is integrated into .NET Core applications.

12. Out of the given options, which of them cannot be minified?

Answer: (Please choose the correct answer)

- css files
- js files
- html files
- c# files

Correct Answer: c# files

Minification is typically applied to client-side files such as CSS, JavaScript, and HTML to reduce their size and improve load times. C# files are server-side code and are not minified; they are compiled into intermediate language (IL) code before being executed.

13. In ASP.NET Core 2.1 or later, add a new JSON file, named to enable bundling and minification.

Answer: (Please choose the correct answer)

- bundleconfig.json
- minifyconfig.json
- bundler.json
- bundler_config.json

Correct Answer: bundleconfig.json

In ASP.NET Core 2.1 or later, bundleconfig.json is used to configure bundling and minification of files.

14. Which of the below could be an advantage of including NuGet packages in ASP.NET Core 1.0?

Answer: (Please choose the correct answer)

- Slower performance and increased servicing

- Better performance and reduced servicing
- More memory availability
- No significant advantage

Correct Answer: Better performance and reduced servicing

Including NuGet packages in ASP.NET Core 1.0 can offer better performance and reduced servicing by allowing the use of optimized libraries and tools that can improve application efficiency and reduce the need for extensive manual maintenance.

15. How to print value from Controller to View in MVC?

Answer: (Please choose the correct answer)

- ViewBag.ECMDetail = "my message"; and in view ViewBag.ECMDetail
- ViewBag.ECMDetail = "my message"; and in view @ViewBag.ECMDetail
- ViewBag.ECMDetail = "my message"; and in view ViewBag.Title
- None of the above

Correct Answer: ViewBag.ECMDetail = "my message"; and in view @ViewBag.ECMDetail

In ASP.NET MVC, you use the ViewBag to pass data from the controller to the view. You set a value in the controller using ViewBag and then access it in the view using @ViewBag.PropertyName.

16. Which of these defines the ASP.NET Core Middleware?

Answer: (Please choose the correct answer)

- A class that you create in your project.
- A method that you create in your project.
- A software component that is assembled into an application pipeline to handle requests and responses.
- A built-in service that is used to interact with the database.

Correct Answer: A software component that is assembled into an application pipeline to handle requests and responses.

ASP.NET Core Middleware is a software component that is assembled into an application pipeline to handle requests and responses.

17. Which one of these is the right way to inject a dependency into a controller in ASP.NET Core?

Answer: (Please choose the correct answer)

- Add the dependency in the Configure method of the Startup class.

- Use the dependency directly in the controller without registration.
- Register the dependency in the ConfigureServices method of the Startup class and use constructor injection in the controller.
- Create a new instance of the dependency in the controller constructor.

Correct Answer: Register the dependency in the ConfigureServices method of the Startup class and use constructor injection in the controller.

In ASP.NET Core, you register dependencies in the ConfigureServices method of the Startup class and then use constructor injection in the controller to access those dependencies.

18. In an ASP.NET Core application, what is the purpose of the IWebHostBuilder?

Answer: (Please choose the correct answer)

- To configure and create an IWebHost instance.
- To manage the lifecycle of the application.
- To provide database connectivity.
- To define middleware components.

Correct Answer: To configure and create an IWebHost instance.

The IWebHostBuilder is used to configure and create an IWebHost instance, which is responsible for managing the web server and the application's startup process.

19. What is the role of the Configure method in the Startup class in ASP.NET Core?

Answer: (Please choose the correct answer)

- To define the application's data model.
- To set up middleware and the request processing pipeline.
- To configure the application's database connections.
- To handle user authentication and authorization.

Correct Answer: To set up middleware and the request processing pipeline.

The Configure method in the Startup class is used to set up middleware and the request processing pipeline in an ASP.NET Core application.

20. Which of the following is a benefit of using ASP.NET Core?

Answer: (Please choose the correct answer)

- It can only run on Windows.
- It has limited support for modern web development practices.
- It offers cross-platform support and high performance.
- It does not support dependency injection.

Correct Answer: It offers cross-platform support and high performance.

ASP.NET Core offers cross-platform support, allowing applications to run on Windows, macOS, and Linux, and it is designed for high performance.

21. Which command is used to create a new ASP.NET Core project from the command line?

Answer: (Please choose the correct answer)

- dotnet new mvc
- dotnet create mvc
- dotnet build mvc
- dotnet run mvc

Correct Answer: dotnet new mvc

The command `dotnet new mvc` is used to create a new ASP.NET Core MVC project from the command line.

22. How do you enable detailed error messages in an ASP.NET Core application during development?

Answer: (Please choose the correct answer)

- Set the environment variable to Production.
- Use the `DeveloperExceptionPage` middleware.
- Configure the application to hide errors.
- Enable exception logging only.

Correct Answer: Use the `DeveloperExceptionPage` middleware.

To enable detailed error messages in an ASP.NET Core application during development, you use the `DeveloperExceptionPage` middleware.

23. In ASP.NET Core, which service should be used to read configuration settings from `appsettings.json`?

Answer: (Please choose the correct answer)

- `IOptions<T>`
- `IConfiguration`
- `ILogger`
- `IHttpContextAccessor`

Correct Answer: `IConfiguration`

The `IConfiguration` service is used to read configuration settings from `appsettings.json` in an ASP.NET Core application.

24. Which of the following methods is used to add MVC services to the ASP.NET Core dependency injection container?

Answer: (Please choose the correct answer)

- `services.AddMvc()`
- `services.AddControllers()`
- `services.AddMvcServices()`
- `services.UseMvc()`

Correct Answer: `services.AddMvc()`

In an ASP.NET Core application, you use `services.AddMvc()` to add MVC services to the dependency injection container.

25. Which of the following middleware is used to serve static files in an ASP.NET Core application?

Answer: (Please choose the correct answer)

- `app.UseStaticFiles()`
- `app.UseMvc()`
- `app.UseEndpoints()`
- `app.UseRouting()`

Correct Answer: `app.UseStaticFiles()`

The `app.UseStaticFiles()` middleware is used to serve static files in an ASP.NET Core application.

40

regenerate from question number 26

ChatGPT

26. What does the `app.UseEndpoints` method do in an ASP.NET Core application?

Answer: (Please choose the correct answer)

- It configures middleware to serve static files.
- It sets up the application's request routing.
- It adds MVC services to the dependency injection container.
- It configures exception handling middleware.

Correct Answer: It sets up the application's request routing.

The `app.UseEndpoints` method is used to set up the application's request routing in an ASP.NET Core application.

27. In an ASP.NET Core MVC application, what attribute is used to specify that an action method should handle HTTP POST requests?

Answer: (Please choose the correct answer)

- [HttpGet]
- [HttpPut]
- [HttpPost]
- [HttpDelete]

Correct Answer: [HttpPost]

In an ASP.NET Core MVC application, the [HttpPost] attribute is used to specify that an action method should handle HTTP POST requests.

28. What is the purpose of the ConfigureServices method in the Startup class in ASP.NET Core?

Answer: (Please choose the correct answer)

- To set up middleware and the request processing pipeline.
- To configure the application's data model.
- To register services with the dependency injection container.
- To handle user authentication and authorization.

Correct Answer: To register services with the dependency injection container.

The ConfigureServices method in the Startup class is used to register services with the dependency injection container in an ASP.NET Core application.

29. Which of the following is a built-in logging provider in ASP.NET Core?

Answer: (Please choose the correct answer)

- NLog
- Serilog
- Log4Net
- Console

Correct Answer: Console

The Console logging provider is a built-in logging provider in ASP.NET Core. Other providers like NLog, Serilog, and Log4Net are third-party logging providers.

30. How do you inject a service into a view in ASP.NET Core?

Answer: (Please choose the correct answer)

- Use constructor injection.

- Use the ViewBag.
- Use the @inject directive.
- Use the @model directive.

Correct Answer: Use the @inject directive.

In ASP.NET Core, you use the `@inject` directive to inject a service into a view.

31. What is the purpose of the `appsettings.json` file in an ASP.NET Core application?

Answer: (Please choose the correct answer)

- To define middleware components.
- To configure logging providers.
- To store configuration settings.
- To register services with the dependency injection container.

Correct Answer: To store configuration settings.

The `appsettings.json` file is used to store configuration settings in an ASP.NET Core application.

32. How do you enable session state in an ASP.NET Core application?

Answer: (Please choose the correct answer)

- Use the `app.UseSession` middleware.
- Use the `services.AddSession` method.
- Use the `services.AddDistributedMemoryCache` method.
- All of the above.

Correct Answer: All of the above.

To enable session state in an ASP.NET Core application, you use the `services.AddSession` method to add session services to the dependency injection container, the `services.AddDistributedMemoryCache` method to configure the in-memory cache, and the `app.UseSession` middleware to enable session state.

33. In ASP.NET Core, what is the purpose of the `IApplicationBuilder` interface?

Answer: (Please choose the correct answer)

- To configure the application's services.
- To define the application's data model.
- To configure the application's request pipeline.
- To handle user authentication and authorization.

Correct Answer: To configure the application's request pipeline.

The `IApplicationBuilder` interface is used to configure the application's request pipeline in an ASP.NET Core application.

34. Which of the following methods is used to add MVC controllers with views to the ASP.NET Core dependency injection container?

Answer: (Please choose the correct answer)

- `services.AddMvc()`
- `services.AddControllers()`
- `services.AddControllersWithViews()`
- `services.UseMvc()`

Correct Answer: `services.AddControllersWithViews()`

In an ASP.NET Core application, you use `services.AddControllersWithViews()` to add MVC controllers with views to the dependency injection container.

35. What is the purpose of the `UseRouting` middleware in an ASP.NET Core application?

Answer: (Please choose the correct answer)

- To configure logging providers.
- To set up request routing.
- To handle static files.
- To enable session state.

Correct Answer: To set up request routing.

The `UseRouting` middleware is used to set up request routing in an ASP.NET Core application.

36. In ASP.NET Core, what is the `IWebHost` interface responsible for?

Answer: (Please choose the correct answer)

- Managing the application's request pipeline.
- Providing database connectivity.
- Hosting the application.
- Handling user authentication and authorization.

Correct Answer: Hosting the application.

The `IWebHost` interface is responsible for hosting the application in an ASP.NET Core application.

37. How do you register a custom middleware in an ASP.NET Core application?

Answer: (Please choose the correct answer)

- Use the `services.AddMiddleware` method.
- Add the middleware class to the `app.UseMiddleware` method.
- Configure the middleware in the `appsettings.json` file.
- Register the middleware in the `ConfigureServices` method.

Correct Answer: Add the middleware class to the `app.UseMiddleware` method.

To register a custom middleware in an ASP.NET Core application, you add the middleware class to the `app.UseMiddleware` method.

38. What is the purpose of the `IHostBuilder` interface in ASP.NET Core?

Answer: (Please choose the correct answer)

- To configure logging providers.
- To define the application's data model.
- To build and configure a host.
- To handle user authentication and authorization.

Correct Answer: To build and configure a host.

The `IHostBuilder` interface is used to build and configure a host in an ASP.NET Core application.

39. Which of the following methods is used to configure CORS in an ASP.NET Core application?

Answer: (Please choose the correct answer)

- `services.AddCors()`
- `app.UseCors()`
- `services.ConfigureCors()`
- `app.ConfigureCors()`

Correct Answer: `services.AddCors()` and `app.UseCors()`

In an ASP.NET Core application, you configure CORS by using `services.AddCors()` to add CORS services to the dependency injection container and `app.UseCors()` to enable CORS in the request pipeline.

40. How do you run an ASP.NET Core application from the command line?

Answer: (Please choose the correct answer)

- dotnet create
- dotnet build
- dotnet run
- dotnet publish

Correct Answer: dotnet run

To run an ASP.NET Core application from the command line, you use the `dotnet run` command.