ONE PAGE SUMMARY

IT & Software › Backend Development › ASP.NET Core

Introduction to ASP.NET Core

Unlock the Power of ASP.NET Core for Modern Web Development!

DEFINITION

> ASP.NET Core is a cross-platform framework for building web applications and APIs. It allows developers to create high-performance applications using C# and .NET.

KEY CONCEPTS

Cross-Platform

ASP.NET Core runs on Windows, macOS, and Linux, making it versatile for developers.

Middleware

Middleware components process requests and responses in the application pipeline, enabling custom functionality.

Dependency Injection

ASP.NET Core has built-in support for dependency injection, promoting loose coupling and easier testing.

Razor Pages

Razor Pages simplify page-focused scenarios by using a page-based programming model for web applications.

Configuration System

ASP.NET Core's configuration system allows for flexible settings management from various sources like JSON files.

EXAMPLES

→ Creating a RESTful API using ASP.NET Core controllers.

→ Building a simple web application with Razor Pages.

→ Implementing middleware for logging requests.

MEMORY TIPS

★ Remember 'CROSS' for Cross-Platform, Razor, and Middleware.

★ Think 'DIP' for Dependency Injection Principle.

★ Use 'CONFIG' to recall Configuration System features.

COMMON MISTAKES

✗ Neglecting to configure services in the Startup class.

✗ Forgetting to use 'await' with asynchronous methods.

✗ Overcomplicating middleware logic instead of keeping it simple.

QUICK RECAP

ASP.NET Core is a versatile framework for web development that supports cross-platform applications. Key features include middleware, dependency injection, and Razor Pages, which streamline the development process. Remember to configure services properly and keep middleware logic straightforward.