ASP.NET CORE BASICS

TechRacine Solutions Academy

https://techracine.com

https://academy.techracine.com

Study Guide

Table of Contents

Introduction to ASP.NET Core

ASP.NET Core is a modern web development framework developed by Microsoft.

It is used to build web applications, APIs, enterprise portals, and cloud-based applications.

ASP.NET Core is open-source, cross-platform, and highly scalable.

Applications can run on Windows, Linux, and macOS.

ASP.NET Core is widely used in enterprise software development.

Features of ASP.NET Core

ASP.NET Core provides many modern development features.

1. Cross-platform support.

2. High performance.

3. MVC and API support.

4. Dependency Injection.

5. Middleware architecture.

6. Security and authentication support.

7. Cloud deployment compatibility.

Installing .NET SDK

.NET SDK is required for ASP.NET Core development.

The SDK includes runtime, libraries, and CLI tools.

Developers can verify installation using terminal commands.

```
dotnet --version
```

Understanding ASP.NET Core Architecture

ASP.NET Core commonly follows MVC architecture.

MVC stands for Model, View, and Controller.

Models manage data.

Views display UI.

Controllers process requests and responses.

Creating ASP.NET Core Project

Projects can be created using Visual Studio or terminal commands.

The dotnet CLI simplifies project creation.

```
dotnet new mvc -n AcademyProject
```

Project Folder Structure

Controllers folder stores controllers.

Models folder stores business models.

Views folder stores Razor pages.

wwwroot contains CSS, JavaScript, and images.

appsettings.json stores configuration settings.

Program.cs File

Program.cs is the startup file of ASP.NET Core applications.

It configures services and middleware.

```
var builder = WebApplication.CreateBuilder(args);

var app = builder.Build();

app.Run();
```

Controllers and Routing

Controllers process incoming HTTP requests.

Routing maps URLs to controller actions.

```
public class HomeController : Controller
{
    public IActionResult Index()
    {
        return View();
    }
}
```

Views and Razor Pages

Views display webpage content.

Razor syntax combines HTML and C# code.

```
<h1>@ViewBag.Title</h1>
```

Models in ASP.NET Core

Models represent business data.

Models are commonly used with databases.

```
public class Student
{
    public int Id { get; set; }

    public string Name { get; set; }
}
```

Working with appsettings.json

appsettings.json stores configuration details such as database connections.

```
{
  "ConnectionStrings": {
    "DefaultConnection":
    "Server=.;Database=AcademyDB;Trusted_Connection=True;"
  }
}
```

Dependency Injection

Dependency Injection improves maintainability and loose coupling.

ASP.NET Core includes built-in Dependency Injection.

```
builder.Services.AddControllersWithViews();
```

Middleware

Middleware processes HTTP requests and responses.

Middleware forms the application request pipeline.

```
app.UseAuthentication();

app.UseAuthorization();
```

Database Connection

ASP.NET Core commonly connects with SQL Server databases.

Connection strings are stored inside appsettings.json.

Entity Framework Core Basics

Entity Framework Core is an ORM framework.

It allows developers to interact with databases using C# classes.

```
dotnet ef migrations add InitialCreate
```

```
dotnet ef database update
```

Running ASP.NET Core Applications

Applications can be run using Visual Studio or terminal commands.

```
dotnet run
```

Using Visual Studio and VS Code

Visual Studio provides advanced ASP.NET Core development tools.

VS Code provides lightweight editing and integrated terminal support.

Basic CRUD Operations

CRUD stands for Create, Read, Update, and Delete.

Business applications commonly implement CRUD operations.

Best Practices

Use proper folder structure.

Keep controllers lightweight.

Use dependency injection properly.

Write reusable and maintainable code.

Protect sensitive configuration values.

Summary

ASP.NET Core is a powerful framework for modern backend development.

It supports APIs, MVC architecture, enterprise applications, and cloud deployment.

Learning ASP.NET Core provides strong skills for enterprise software development.