Configuration and Options in ASP.NET Core 6

Getting Started with Configuration

Steve Gordon

.NET Engineer and Microsoft MVP

@stevejgordon | www.stevejgordon.co.uk

Welcome!

Goals

- Focus on leveraging configuration and options to address real-world requirements
- Become productive as quickly as possible

Version Check

This course was created by using:

- ASP.NET Core 6
- .NET 6
- C# 10
- Visual Studio 2022

Version Check

This course is 100% applicable to:

- ASP.NET Core 6 and ASP.NET Core 7
- .NET 6 and .NET 7
- Visual Studio 2022 (any edition)

Most code will work fine with older supported versions of ASP.NET Core

Relevant Notes

A note on frameworks and libraries:

- A new version of .NET releases each year
- The configuration and options libraries change very little between versions
- Microsoft aim to maintain backward compatibility between releases

Overview

How configuration is structured

Define JSON configuration

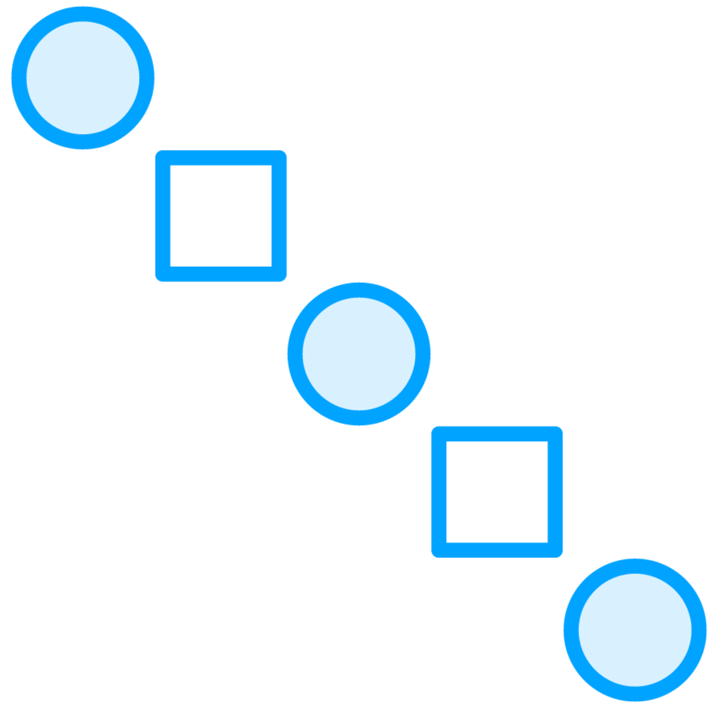
Access configuration at runtime

- Accessing by key
- Accessing by section

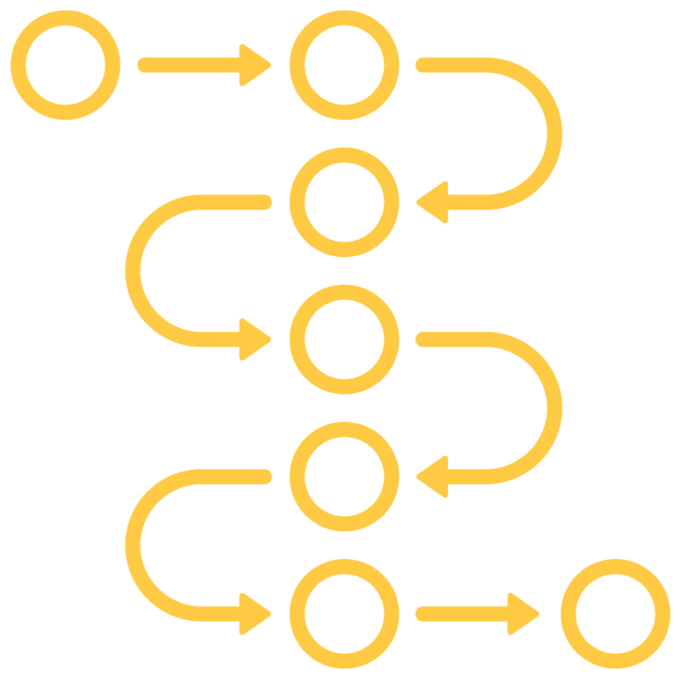
Bind configuration

Override configuration by environment

Later in This Course



Applying the Options Pattern



Working with Configuration Providers

Course Prerequisites

Beginner to intermediate knowledge of .NET

Experience with C# fundamentals

Follow Along

Download the exercise files

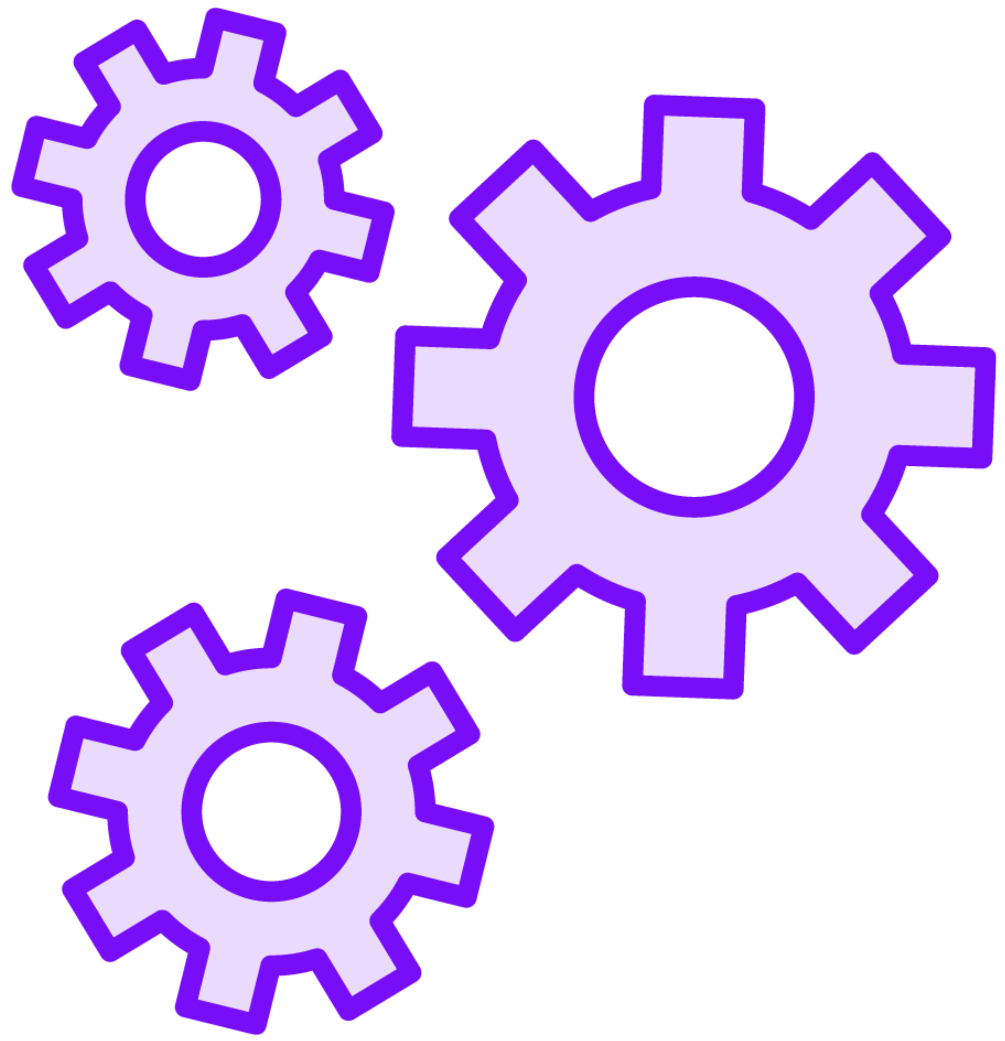
The solution requires the latest .NET 6.0.x SDK
http://dot.net

An IDE such as Visual Studio Community Edition or an editor such as Visual Studio Code

Let’s Get Started…

Application Configuration in ASP.NET Core

What is Configuration?



Mechanism by which we define and access settings

Provides initial settings for an application

Change behavior without recompilation

Defined in one or many sources

Accessed at runtime to control application behavior

Configuration in ASP.NET Core

Hosting

Logging

Configuration and Options

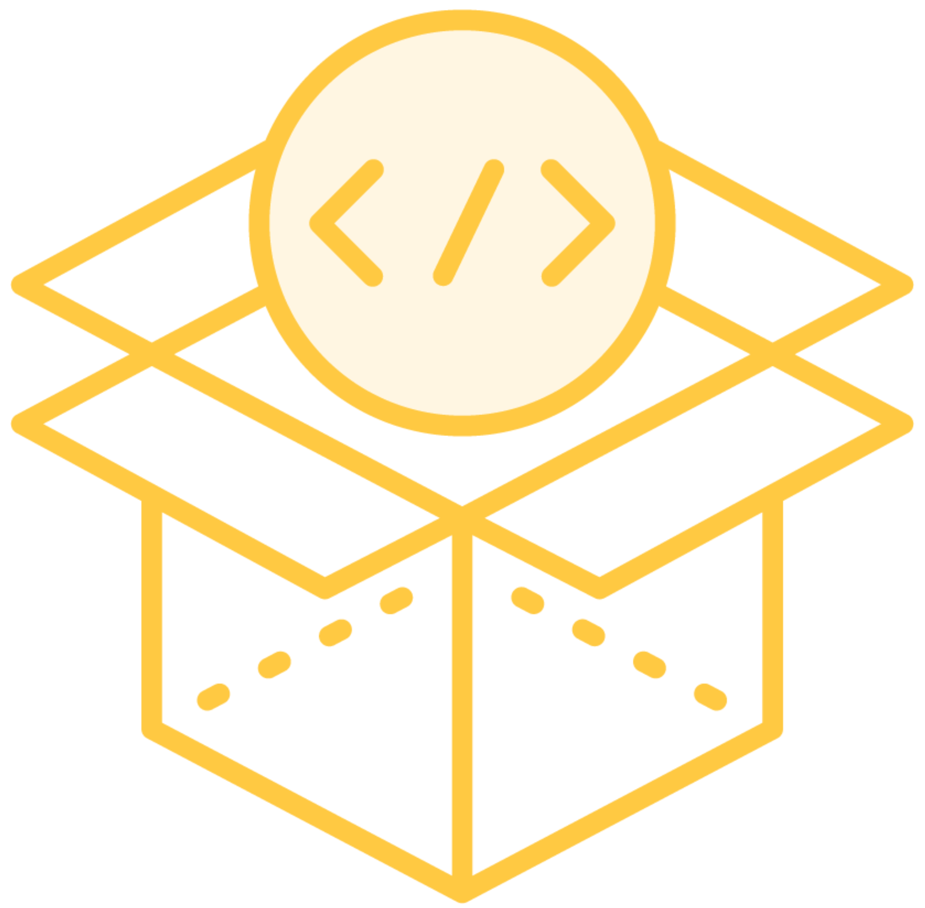
Dependency Injection

Configuration in ASP.NET Core

- Hosting
- Logging
- **Configuration and Options**
- Dependency Injection

Microsoft.Extensions.Configuration

Microsoft.Extensions.Configuration



Referenced by default in ASP.NET Core

Can be consumed in other .NET applications

- Console applications
- Worker service

Logical Configuration Structure

```
MyStringKey = “This is a string value”
MySecondStringKey = “This is another value”
MyBooleanKey = true
MyIntegerKey = 100
```

Configuration Hierarchy

```
Section1:KeyA = “My 1st value”
Section1:KeyB = “My 2nd value”
Section2:KeyA = “Doesn’t conflict”
Section2:SubSection:KeyA = “And another!!”
```

Introducing the Tennis Booking Application

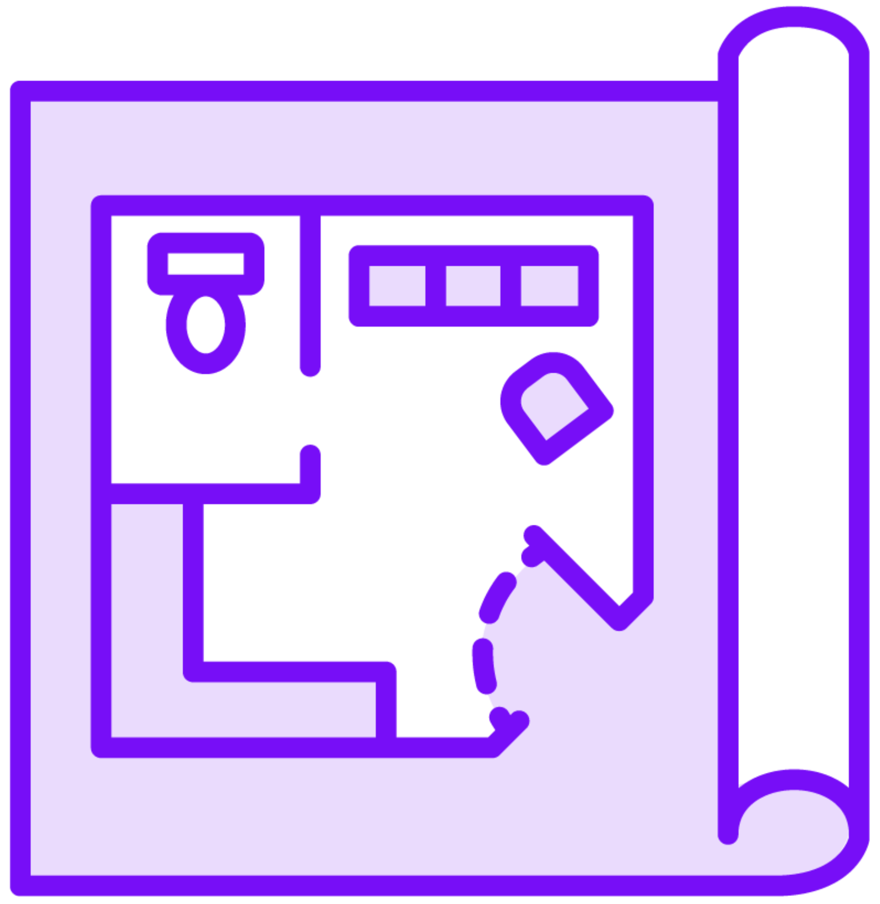
Getting started with configuration

Configuration in JSON files

Default configuration values

Defining additional configuration values

Plan of Attack



Improve the homepage

- Add a random greeting
- Add details of weather conditions

Define configuration flags to control features

JSON Configuration

The appsettings.json file is a source of configuration values

Read by the JSON configuration provider at startup

Accessing configuration at runtime

- Injecting and consuming configuration
- Conditionally rendering content

Accessing sections of configuration

Accessing string configuration values

Accessing configuration at startup

Conditionally registering services

How ASP.NET Core uses configuration

Learn More

Dependency Injection in ASP.NET Core 6

Steve Gordon

Downsides of Using IConfiguration Directly

Problems with Getting Values

Repetitive code

Fragile naming

Can lead to bugs

Binding configuration to objects

Binding Configuration

Make configuration values more accessible

Reduce noise

Reduce fragility

Loading configuration per environment

- Overriding development configuration

Per Environment Configuration

Development

```
cs="Server=(localdb)\\mssqlloc
aldb;Database=TennisBookings;"
```

Production

```
cs="Server=ProdDbServer;
Database=TennisBookings;"
```

Summary

Structure of configuration

Defined configuration in a JSON file

Accessed configuration at runtime

Bound configuration to an object

Loaded configuration per environment

Up Next:

Applying the Options Pattern