Instructor example

Web University by Harsha Vardhan

What is the purpose of the appsettings.json file?

Appsettings.json contains all of the application's configuration settings, which allow you to configure your application behavior.

It includes with configuration settings related to logging, connection strings etc.

You can also write environment-specific configuration with “appsettings.Environment.json” file.

You can also load custom json files, custom INI files, InMemory configuration or Secrets manager to store configuration settings.

You have configuration values needed to access your application resources. Which configuration providers do you prefer for development, and which do you prefer for production?

Many cloud providers and Docker hosting platforms support environment variables, so environment variables make much sense for production environments. However, in case of sensitive information, I prefer the user secrets configuration provider for local development as there’s no way to add sensitive secrets to source control mistakenly. Finally, for non-sensitive data, I like JSON configuration since it’s one of the default configuration options, and it’s easy to add and manage into source control.

How do you use Options pattern in Asp.Net Core?

The configuration system in ASP.NET allows (actually, enforces) strongly typed settings using the IOptions<> pattern.

Services:

```
app.Services.Configure<Model>(builder.Configuration.GetSection(“ParentKey”));
```

Controller:

```
private readonly Model _options;

public ControllerName(IOptions<Model> options)
{
  _options = options.Value; //returns an instance of Model class that has configuration values from appropriate confi
}
```

How do you enable Secrets manager and why?

A feature in ASP.NET Core named User Secrets allows you to store user secrets outside your project tree in a JSON file, and can even be managed using a command-line tool called the Secrets Manager.

So as a benefit, you will not push sensitive values like passwords or API keys into your source control – so those values are kept secret to the same developer machine.

To enable & store secrets using “secret manager tool” (in PowerShell):

```
dotnet user-secrets init
dotnet user-secrets set “Key” “value”
dotnet user-secrets list
```