

## *Chapter 1: Introduction to .NET*

What is a design pattern?

Anti-patterns and code smells

Anti-patterns

Code smells

Understanding the web – Request/Response

Getting started with .NET

.NET SDK versus runtime

.NET 5 versus .NET Standard

Visual Studio Code versus Visual Studio versus the command-line interface (CLI).

Technical requirements

Summary

Questions

Further reading

## *Chapter 2: Testing Your ASP.NET Core Application*

Overview of automated testing and how it applies to ASP.NET Core

Test-driven development (TDD).

Testing made easy through ASP.NET Core

How do you create an xUnit test project?

Basic features of xUnit

How to organize your tests

How is it easier?

Summary

Questions

Further reading

## Chapter 3: Architectural Principles

### The SOLID principles

Single responsibility principle (SRP).

Open/Closed principle (OCP).

Liskov substitution principle (LSP).

Interface segregation principle (ISP).

Dependency inversion principle (DIP).

### Other important principles

Separation of concerns

Don't repeat yourself (DRY).

Summary.

Questions

## Section 2: Designing for ASP.NET Core

## *Chapter 4: The MVC Pattern using Razor*

### The Model View Controller design pattern

#### MVC using Razor

#### Directory structure

#### Structure of a controller

#### Default routing

#### Project: MVC

#### Conclusion

### View Model design pattern

#### Goal

#### Design

#### Project: View models (a list of students).

#### Project: View models (a student form).

#### Conclusion

#### Summary

#### Questions

## Further reading

## *Chapter 5: The MVC Pattern for Web APIs*

An overview of REST

Request HTTP methods

Response status code

Anatomy of a web API

Setting up a web API

Attribute routing

Returning values

C# features

Class conversion operators (C#).

Local functions (C# 7) and a static local function (C# 8).

The Data Transfer Object design pattern

Goal

Design

Project – DTO

API contracts

[Analyzing the DTO sample](#)

[Project – OpenAPI](#)

[Project – API contracts](#)

[Idea – Creating a typed client library](#)

[One last observation](#)

[Summary](#)

[Questions](#)

[Further reading](#)



## Chapter 6: Understanding the Strategy, Abstract Factory, and Singleton Design Patterns

### The Strategy design pattern

#### Goal

#### Design

#### Project: Strategy

#### Conclusion

### A brief look at a few C# features

#### Default literal expressions (C# 7.1).

#### Switch expressions (C# 8).

#### Discards (C# 7).

### The Abstract Factory design pattern

#### Goal

#### Design

#### Project: AbstractVehicleFactory

#### Project: MiddleEndVehicleFactory

Conclusion

The Singleton design pattern

Goal

Design

An alternate (better) way

Code smell: Ambient Context

Conclusion

Summary

Questions

## [Chapter 7: Deep Dive into Dependency Injection](#)

### [What is Dependency Injection?](#)

### [The composition root](#)

### [Extending IServiceCollection](#)

### [Object lifetime](#)

### [Code smell: Control Freak](#)

### [Using external IoC containers](#)

### [Revisiting the Strategy pattern](#)

### [Constructor injection](#)

### [Property injection](#)

### [Method injection](#)

### [Project: Strategy](#)

### [Revisiting the Singleton pattern](#)

### [The application state](#)

### [Project: Wishlist](#)

### [Tuples \(C# 7+\).](#)

[Understanding the Service Locator pattern](#)

[Project: ServiceLocator](#)

[Project: ServiceLocatorFixed](#)

[Conclusion](#)

[Revisiting the Factory pattern](#)

[Factory mixed with method injection](#)

[HomeViewModelFactory](#)

[Summary](#)

[Questions](#)

[Further reading](#)

## [Chapter 8: Options and Logging Patterns](#)

### [An overview of the Options pattern](#)

#### [Getting started](#)

#### [Project – CommonScenarios](#)

#### [Project – OptionsConfiguration](#)

#### [Project – OptionsValidation](#)

#### [Injecting options directly](#)

#### [Conclusion](#)

### [Getting familiar with .NET logging abstractions](#)

#### [About logging](#)

#### [Writing logs](#)

#### [Log levels](#)

#### [Logging providers](#)

#### [Configuring logging](#)

#### [Conclusion](#)

#### [Summary](#)

Questions

Further reading

## Section 3: Designing at Component Scale

## Chapter 9: Structural Patterns

### Implementing the Decorator design pattern

Goal

Design

Project: Adding behaviors

Project: Decorator using Scrutor

Conclusion

### Implementing the Composite design pattern

Goal

Design

Project: BookStore

Conclusion

### Implementing the Adapter design pattern

Goal

Design

Project: Greeter



Conclusion

Implementing the Façade design pattern

Goal

Design

Project: The façades

Conclusion

Summary

Questions

Further reading

## Chapter 10: Behavioral Patterns

### Implementing the Template Method pattern

#### Goal

#### Design

#### Project – Building a search machine

#### Conclusion

### Implementing the Chain of Responsibility pattern

#### Goal

#### Project – Message interpreter

#### Project – Improved message interpreter

#### Project – A final, finer-grained design

#### Conclusion

#### Summary

#### Questions

## Chapter 11: Understanding the Operation Result Design Pattern

Goal

Design

Project – Implementing different Operation Result patterns

The consumer

Its simplest form

A single error message

Adding a return value

Multiple error messages

Adding message severity

Sub-classes and factories

Advantages and disadvantages

Advantages

Disadvantages

Summary

Questions

Further reading

## Section 4: Designing at Application Scale

## Chapter 12: Understanding Layering

Introduction to layering

Splitting the layers

Layers versus tiers versus assemblies

To be or not to be a purist?

Sharing the model

The reality of small- to medium-sized enterprises

Responsibilities of the common layers

Presentation

Domain

Data

Abstract data layer

Shared rich model

Clean Architecture

Summary

Questions

## Further reading

## Chapter 13: Getting Started with Object Mappers

### Overview of object mapping

#### Goal

#### Design

#### Project: Mapper

#### Code smell: Too many dependencies

#### Pattern – Aggregate Services

#### Pattern – Mapping Façade

#### Project – Mapping service

#### Project – AutoMapper

#### Summary

#### Questions

#### Further reading



## Chapter 14: Mediator and CQRS Design Patterns

### A high-level overview of Vertical Slice Architecture

#### Implementing the Mediator pattern

##### Goal

##### Design

##### Project – Mediator (IMediator).

##### Project – Mediator (IChatRoom).

##### Conclusion

#### Implementing the CQRS pattern

##### Goal

##### Design

##### Project: CQRS

##### Code smell – marker interfaces

##### Conclusion

#### Using MediatR as a mediator

##### Project – Clean Architecture with MediatR

Conclusion

Summary

Questions

Further reading

## *Chapter 15: Getting Started with Vertical Slice Architecture*

### Vertical Slice Architecture

### What are the advantages and disadvantages?

### Anti-pattern: Big Ball of Mud

### Project: Vertical Slice Architecture

### Continuing your journey

### Summary

### Questions

### Further reading

## *Chapter 16: Introduction to Microservices Architecture*

What are microservices?

Cohesive unit of business

Own its data

Independence

Getting started with message queues

Conclusion

An overview of events

Domain events

Integration events

Implementing the Publish-Subscribe pattern

Message brokers

The event sourcing pattern

Example

Conclusion

Introducing Gateway patterns