Working with Dependency Injection in Razor

Alex Wolf

www.alexwolfthoughts.com

To-Do List

What is Dependency Injection?

Demo: Managing Dependencies in View Components

Demo: Understanding Service Lifetimes

Demo: Injecting Dependencies into Views

Demo: Understanding Loosely Coupled Components

Demo: Alternative Dependency Injection Containers

Dependency Injection Overview

Components Should Depend on Abstractions

```
public class InventoryWidget : ViewComponent
{
    private IInventoryService _inventoryService;
    public InventoryWidget(IInventoryService inventoryService)
    {
        _inventoryService = inventoryService
    }
}
```

Declaring Dependencies

Classes usually declare their dependencies through a constructor

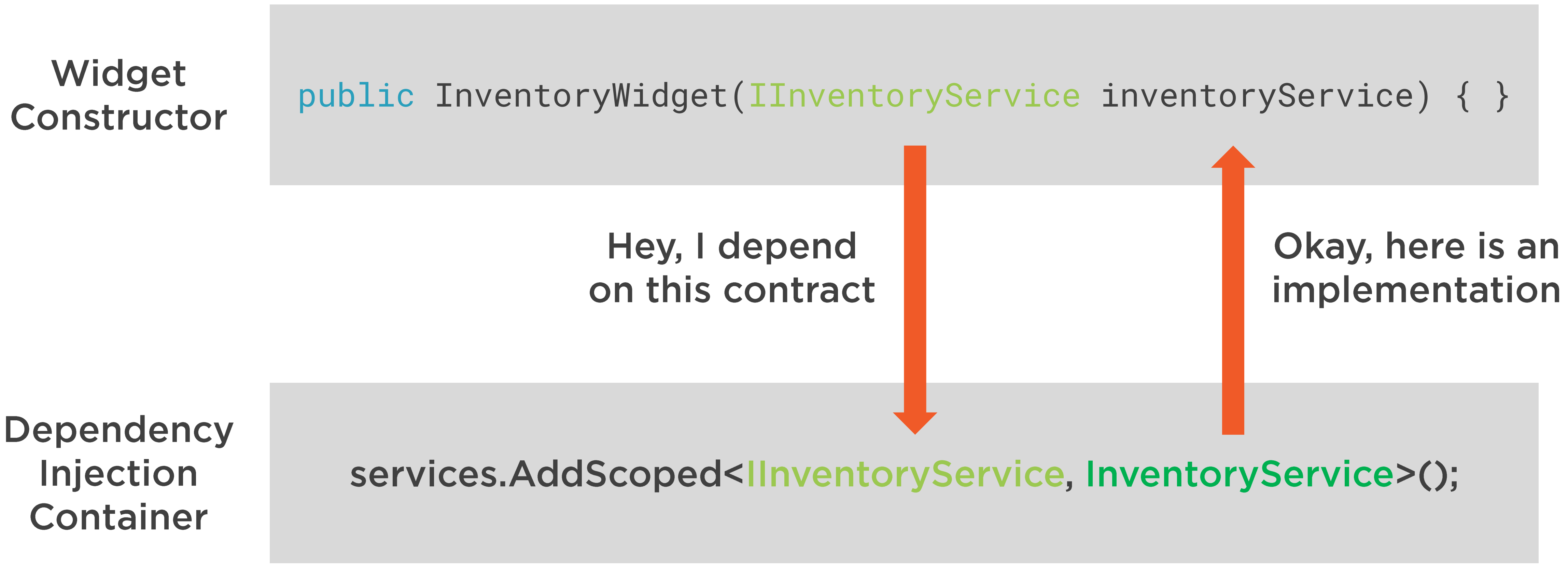
View Components follow this pattern

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddScoped<ISurveyService, SurveyService>();
    services.AddScoped<IInventoryService, InventoryService>();
}
```

Dependency Injection Containers

Containers provide instances of dependencies based on binding and configuration information

Providing Dependencies

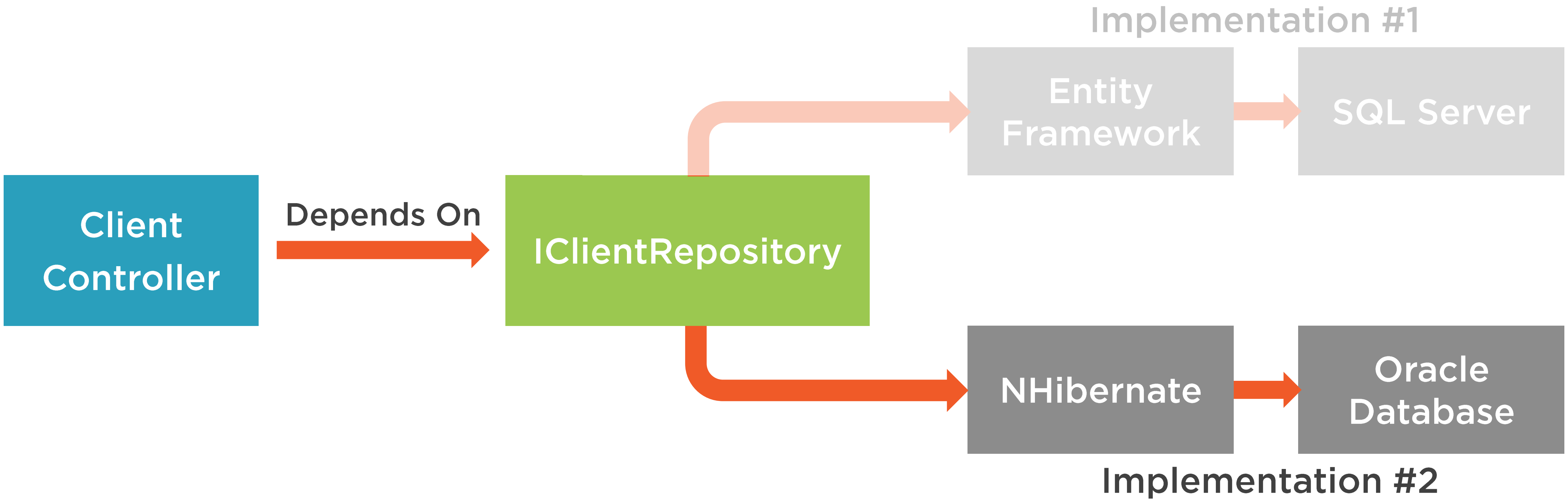


```
public InventoryWidget(IInventoryService inventoryService) { }
```

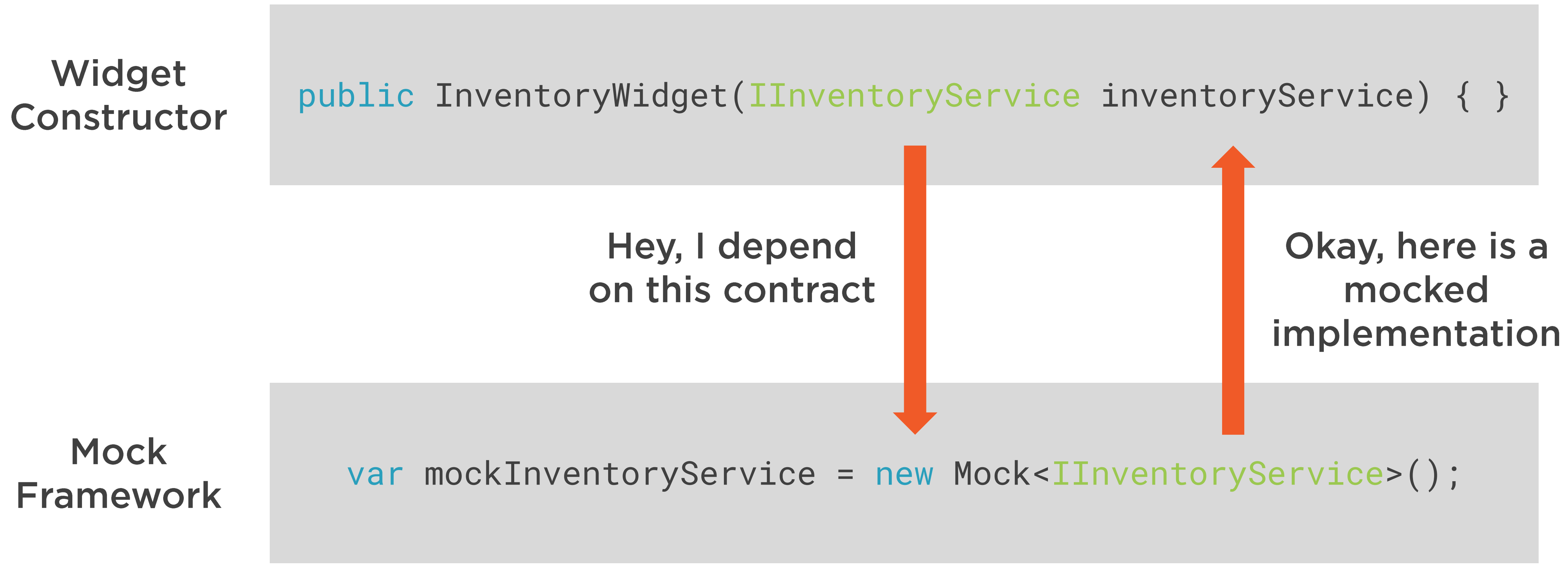
```
services.AddScoped<IInventoryService, InventoryService>();
```

Why Dependency Injection?

Benefits of Loose Coupling



Providing Dependencies for Testing



Widget Constructor

```
public InventoryWidget(IInventoryService inventoryService) { }
```

Hey, I depend on this contract

Okay, here is a mocked implementation

Mock Framework

```
var mockInventoryService = new Mock<IInventoryService>();
```

```
public class ClientController
{
    public ClientController(IClientService clientService, IEmailService
emailService, IWidgetService widgetService, IOrderService orderService)
    {
        // This Controller might do too many things
    }
}
```

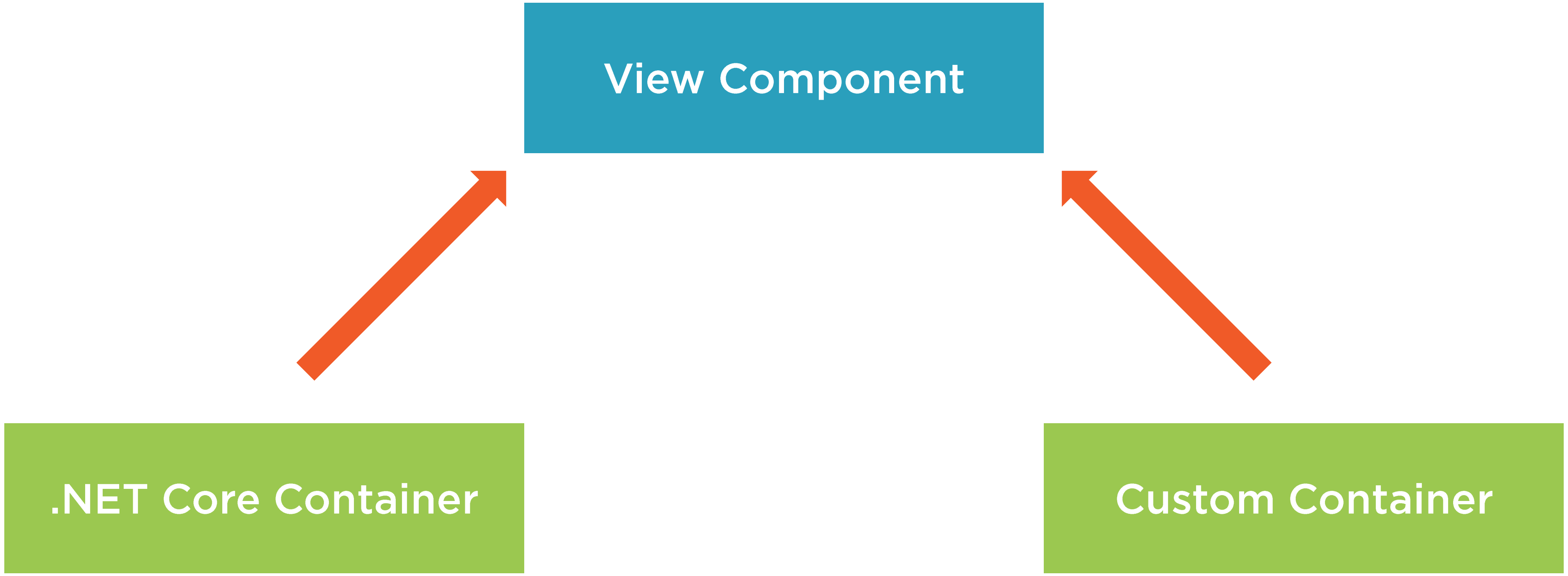
Improving Code Quality

Dependency injection can improve code readability and maintainability

Dependency Lifetime Management

Life Cycle Name	Result
AddTransient	Service created every time a dependency is requested
AddScoped	Service created once per HTTP Request
AddSingleton	Service is created for the lifetime of the application

View Components and Dependency Injection



Demo

Implementing Dependency Injection inside a View Component

Working with .NET Core Dependency Injection features

Demo

Managing service lifetimes inside View Components with Dependency Injection

Demo

Working with Dependency Injection inside of Views

Demo

Exploring loosely coupled components with Dependency Injection

Creating and consuming and alternative data source

Demo

Implementing an alternative Dependency Injection Container

Summary

Dependency Injection helps create loosely coupled components with better code quality

View Components are designed with strong support for Dependency Injection

Most Dependency Injection Containers offer features to manage the lifetime of services

Razor makes it easy to inject services directly into views, but use with caution

Alternative Dependency Injection Containers can be easily configured in .NET Core