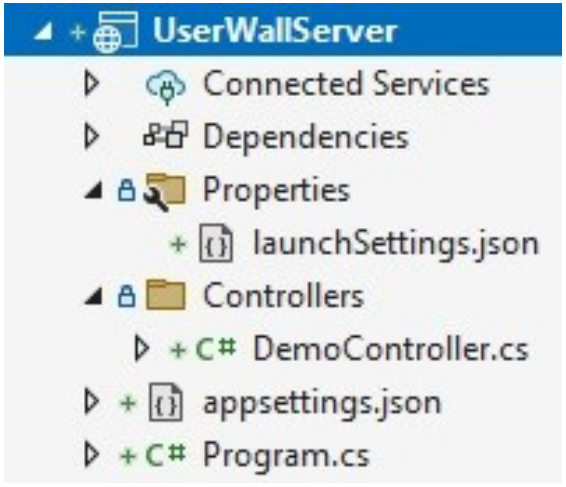
Промышленное программирование

Лекция 7

1. Структура проекта ASP.NET Core
2. Сервисы и контроллеры

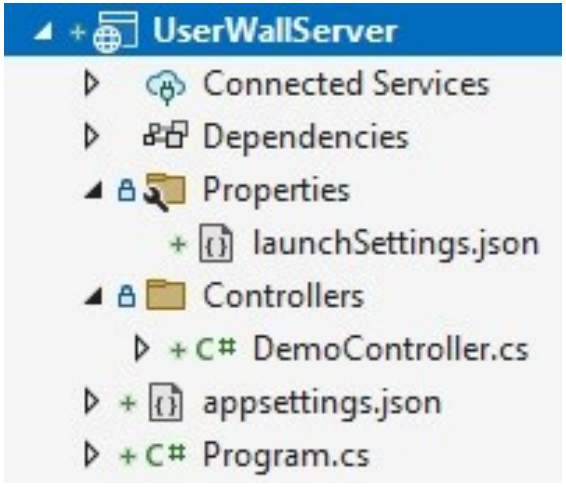
Файл проекта



1. **Файл проекта *.csproj**
2. Стартовый код Program.cs
3. Файл конфигурации appsettings.json
4. Файл запуска Properties/launchSettings.json
5. Пользовательский код DemoController.cs

```
<Project Sdk="Microsoft.NET.Sdk.Web">

    <PropertyGroup>
        <Nullable>enable</Nullable>
        <TargetFramework>net6.0</TargetFramework>
    </PropertyGroup>

</Project>
```

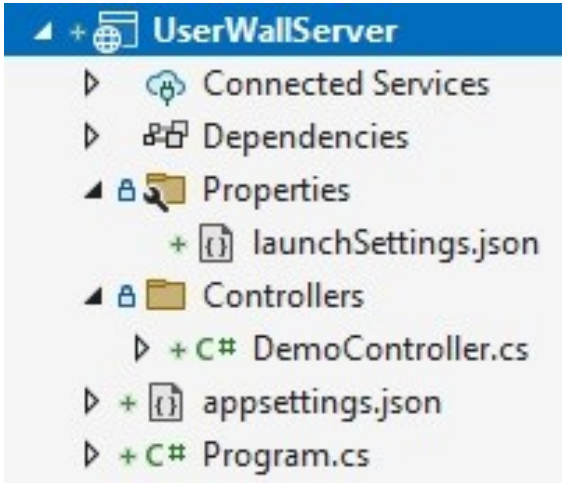
Стартовый код



1. Файл проекта *.csproj
2. **Стартовый код Program.cs**
3. Файл конфигурации appsettings.json
4. Файл запуска Properties/launchSettings.json
5. Пользовательский код DemoController.cs

```
using Microsoft.AspNetCore.Builder;
using Microsoft.Extensions.DependencyInjection;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllers();

var app = builder.Build();
app.MapControllers();

app.Run();
```

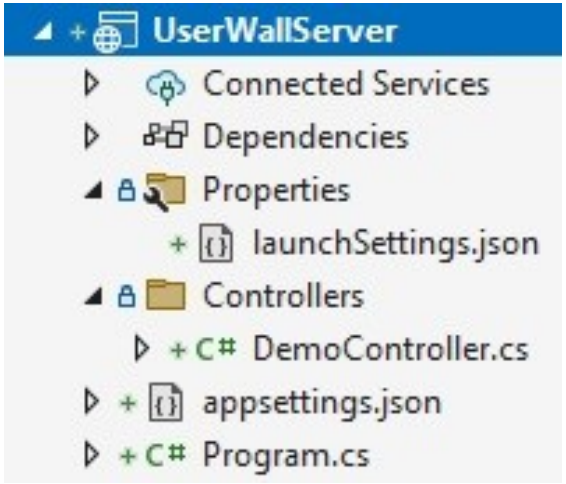
Файл конфигурации



1. Файл проекта *.csproj
2. Стартовый код Program.cs
3. **Файл конфигурации appsettings.json**
4. Файл запуска Properties/launchSettings.json
5. Пользовательский код DemoController.cs

```
{
  "AllowedHosts": "*",
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  }
}
```

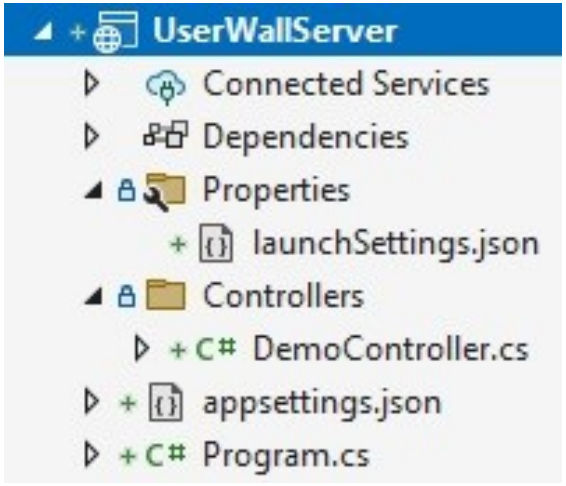
Файл запуска



1. Файл проекта *.csproj
2. Стартовый код Program.cs
3. Файл конфигурации appsettings.json
4. **Файл запуска Properties/launchSettings.json**
5. Пользовательский код DemoController.cs

```
{
  "$schema": "https://json.schemastore.org/launchsettings.json",
  "profiles": {
    "UserWallServer": {
      "commandName": "Project",
      "dotnetRunMessages": true,
      "launchBrowser": true,
      "launchUrl": "demo",
      "applicationUrl": "http://localhost:5273",
      "environmentVariables": {
        "ASPNETCORE_ENVIRONMENT": "Development"
      }
    }
  }
}
```

Пользовательский код



1. Файл проекта *.csproj
2. Стартовый код Program.cs
3. Файл конфигурации appsettings.json
4. Файл запуска Properties/launchSettings.json
5. **Пользовательский код DemoController.cs**

```
using Microsoft.AspNetCore.Mvc;

namespace UserWallServer.Controllers;

[ApiController]
[Route("api/[controller]")]
public class DemoController : ControllerBase
{
    [HttpGet]
    public int Get()
    {
        return 42;
    }
}
```

Service

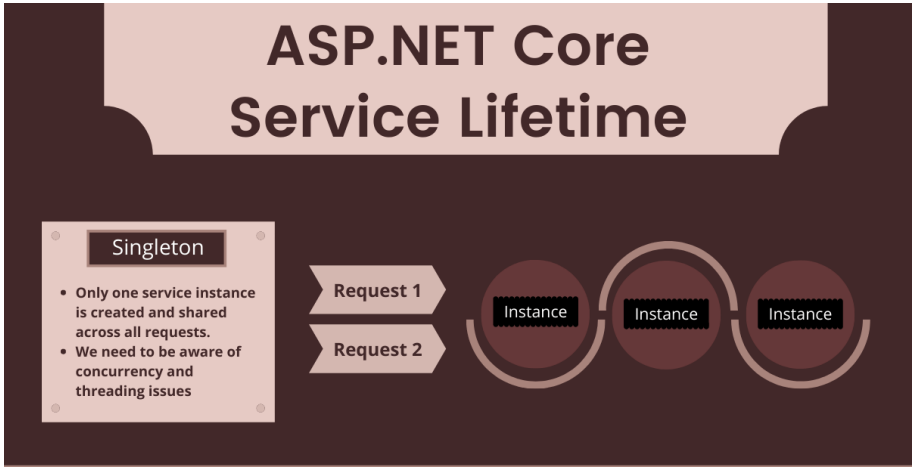
[I]CountService.cs

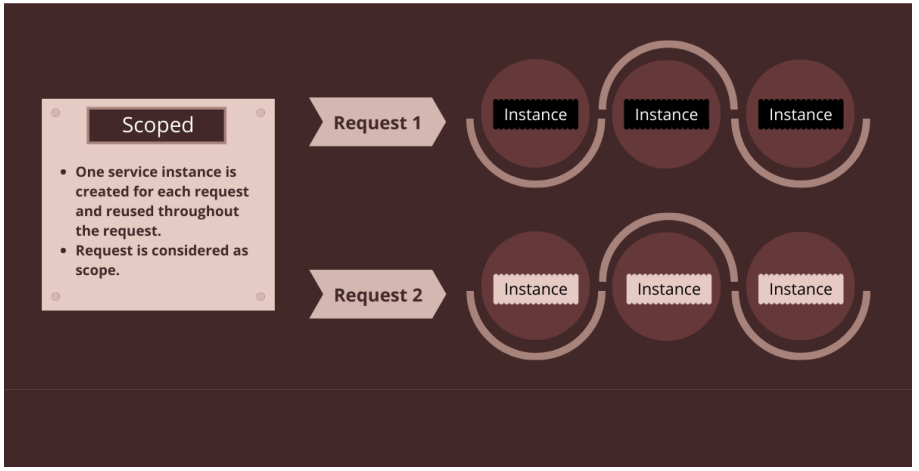
```
public interface ICountService
{
    public int Counter { get; set; }
}

public class CountService : ICountService
{
    public int Counter { get; set; }
}
```

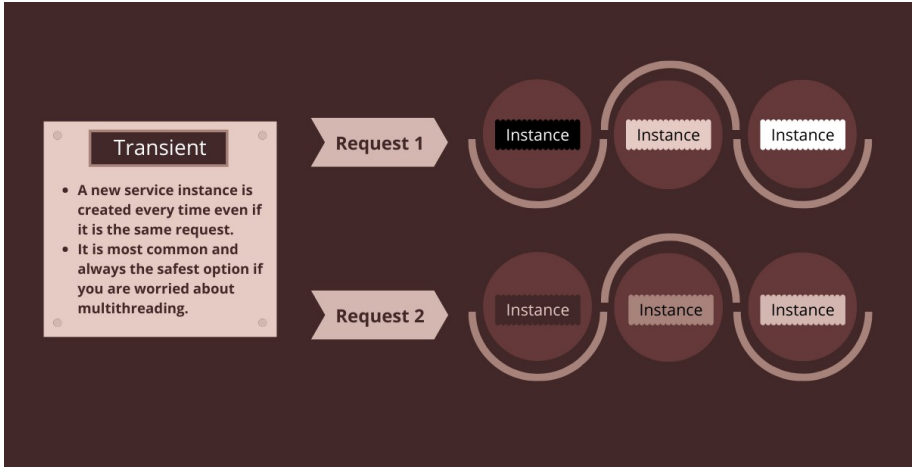
DemoController.cs

```
[ApiController]
[Route("api/[controller]")]
public class DemoController : ControllerBase
{
    private readonly ICountService _countService;

    public DemoController(ICountService countService)
    {
        _countService = countService;
    }

    [HttpGet]
    public int Get() => _countService.Counter;
}
```

Service Lifetime & Registration







[Dependency injection in ASP.NET Core]

Program.cs

```
builder.Services.AddSingleton<ICountService, CountService>();
```

Controller Methods

DemoController.cs

```
[ApiController]
[Route("api/[controller]")]
public class DemoController : ControllerBase
{
    private readonly ICountService _countService;

    public DemoController(ICountService countService)
    {
        _countService = countService;
    }

    [HttpGet]
    public int Get() => _countService.Counter;

    [HttpPost]
    public void Post(int counter) => _countService.Counter = counter;

    [HttpPut]
    public void Put() => ++_countService.Counter;

    [HttpDelete]
    public void Delete() => _countService.Counter = 0;
}
```

Controller Methods: Parameters

```
// [Route("api/[controller]")] DemoController
```

```
[HttpGet("{id:int}")]                    /api/demo/42          #1
int Get(int id);
```

```
[HttpGet]                                /api/demo?value=42    #2
int Get(int value);
```

```
[HttpGet("{id:int}")]                    /api/demo/42?value=42 #3
int Get(int id, int value);
```

```
[HttpGet("news/{id:int}")]               /api/demo/news/42     #4
int Get(int id);

[HttpGet("news/{id}")]                   /api/demo/news/today
int Get(string id);
```

```
[HttpGet]                                /api/demo             #5
IEnumerable<Entity> Get();

[HttpGet("{id:int}")]                    /api/demo/42
Entity Get(int id);
```

Parameter Binding in ASP.NET Web API

Параметры примитивных типов инициализируются:
1) из данных маршрутизации;
2) из строки запроса URI.

Параметр сложного типа инициализируется из тела запроса.

Controller Methods: Return Type

Controller action return types in ASP.NET Core web API

```
[HttpGet]
[ProducesResponseType(StatusCodes.Status200OK, Type = typeof(Entity))]
[ProducesResponseType(StatusCodes.Status404NotFound)]
ActionResult<Entity> Get(int id)
{
    if (id != 42)
        return NotFound();

    return new Entity();
}

[HttpPost]
[ProducesResponseType(StatusCodes.Status200OK)]
[ProducesResponseType(StatusCodes.Status400BadRequest)]
public ActionResult Post(int counter)
{
    if (counter == 0)
        return BadRequest();

    return Ok();
}
```