(#2) What is ASP.NET Core | Asp.Net Core | Asp.Net Core tutorial

Что такое .Net Core

Dot Net Core-это кроссплатформенный

фреймворк с открытым исходным кодом, который используется для создания нескольких типов приложений.

Dot Net Core поддерживается корпорацией Майкрософт.

Что мы можем разработать

1) Web
   - *Asp.Net MVC Core*
   - *Web API*
   - *Razor Pages*
   - *Microservices*
2) Mobile
3) Console
4) Desktop
5) IOT (Internet of things)
6) ML (Machine learning)
7) Gaming
8) Cloud

Особенности

- Бесплатный (Никаких сборов или лицензий не требуется даже для коммерческих приложений)
- Открытый исходный код (доступен на GitHub https: //github.com/dotnet)
- Кросс-платформенный
- Поддерживается корпорацией Майкрософт
- Поддержка командной строки

Языковая поддержка

- C#
- Ф#
- Visual Basic (VB)

(#3) Setting up dot net core machine development | Asp.Net Core tutorial | Asp.Net Core 3.1 tutorial

Инструменты

- (Windows, Mac, Linux)
- Редактор (Рекомендуется Visual Studio, VS Code)
- Dot Net Core SDK

(#4) How to create asp.net core mvc web application (using Visual Studio 2019 & CLI) | .Net Core 3.

План

Как создать первый Asp.Net MVC Core-проект используя visual studio

Как создать первый Asp.Net MVC Core проект используя dotnet CLI

Запуск проекта с помощью Visual studio & dotnetCLI

(#5) What is MVC pattern (model view controller architecture) in .net core | Asp.Net Core Tutorial

План

Шаблон MVC

Роль каждого компонента

Как шаблон MVC работает в Asp.Net Core MVC

Подробности о каждом компоненте

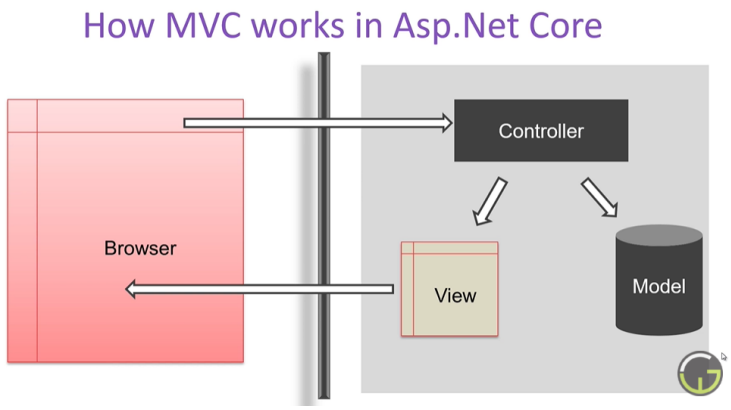
Преимущества использования шаблона MVC

Что такое MVC pattern?

- MVC расшифровывается как Model - View - Controller.
- MVC- это архитектурный шаблон проектирования. Это означает, что этот шаблон проектирования используется на уровне архитектуры приложения.
- Модель, представление и контроллер являются основными компонентами шаблона mvc.

Роль модели, представления и контроллера?

- Модель отвечает за данные.
- Представление отвечает за пользовательский интерфейс.
- Контроллер отвечает за поток приложения, принимая пользовательский ввод.



Понимание Контроллера

- Контроллер-это файл .cs (для c#), который имеет некоторые методы, называемые методами действия.
- Когда запрос поступает на контроллер, он фактически попадает в метод действия.
- Теперь все зависит от способа действия, что из него вернуть. Он может возвращать только вид, только данные или и то, и другое.

Понимание модели

- Модель в Asp.Net Core MVC - это простой класс, который имеет некоторые свойства.

- Модель используется для передачи данных от контроллера (метод действия) к представлению и наоборот, т. е. представление к контроллеру (метод действия).
- Модель также используется для проверки данных на стороне сервера. И с помощью некоторых методов мы также можем генерировать валидацию на стороне клиента.
- Не обязательно, чтобы каждый метод действия возвращал некоторую модель.

Понимание представления

- Представление - это комбинация HTML и C# (или F#, VB).
- Следовательно, для приложения C# расширение представления- это .CSHTML
- Все, что вы видите в своем браузере - это представление.
- Не обязательно, чтобы каждый метод действия возвращал представление.
- Когда представление (содержащее c#) визуализируется в браузере, весь его C# преобразуется в HTML. Это означает, что в браузере мы будем видеть только HTML и данные.

Преимущества MVC

- Разделение интересов.
- Каждый компонент имеет определенную работу, поэтому его легко отлаживать.

(#6) Convert console application to web application in asp.net core | WebHostBuilder in .net core

План

- Как преобразовать консольное приложение в веб-приложение
- Как настроить host builder для веб-приложения
- Хост
- Какие функции доступны по умолчанию в Host builder
- Методы CreateDefaultBuilder & ConfigureWebHostDefaults

Шаги по преобразованию консольного приложения в веб-приложение

- Обновление SDK в файле csproj
- Обновить тип вывода в csproj
- Добавить новый WebHostBuilder
- Добавить маршрутизацию
- Настройка маршрута по умолчанию

Что такое Хост

Хост - это объект, инкапсулирующий ресурсы приложения. Например:

- Инъекция зависимостей
- Регистрация
- Конфигурация
- Реализация IHostedService

(#7) Setup application on GitHub repository | ASP.NET Core Tutorial | .Net Core 3.1 MVC Tutorial

План

Создание пустого asp.net core application

Добавление удаленного репозитория (GitHub)

Push кода из локального репозитория в удаленный

(#8) Middleware in Asp.net Core | app.Use(), app.Next(), app.Map() | Http Pipeline | ASP.NET Core

План

Промежуточное программное обеспечение (middleware)

Как работает Middleware

Где доступен Middleware в приложении

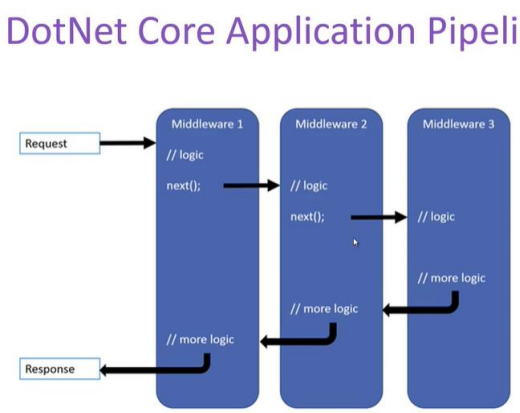
Как добавить новый middleware

Мказания к Middleware

Use(), Next() & Map() методы

Middleware

- Asp.Net Core создает конвейер HTTP-приложения, который обрабатывает запрос.
- Http Pipeline расположен в методе Configure в Startup.cs
- Весь запрос к приложению проходит через HTTP-конвейер.
- Middleware это часть кода (компонент), которая используется в Http-конвейере.
- В приложении мы используем несколько Middleware.
- Middleware имеет доступ ко всем запросам и ответам.



(#9) launchsettings.json in Asp.Net Core | Asp.Net Core tutorial

План

Что такое launchSettings.json файл .

Что все доступно в этом файле.

Как установить порт по умолчанию для приложения

(#10) environment variables in asp.net core | Asp.Net Core tutorial for beginners

План

Переменная среды.

Как установить переменные среды.

Как получить доступ к переменным среды.

Использование переменных среды.

ASP.NET Core Understanding the middleware pipeline, Startup configuration - Application Life Cycle