

Глава 1. Введение в ASP.NET Core

ASP.NET Core - новая эпоха в развитии ASP.NET

Начало работы с ASP.NET Core

Проект ASP.NET Core в Visual Studio for Mac

Глава 2. Основы ASP.NET Core

Глава 3. Сервисы и Dependency Injection

Глава 4. Конфигурация

Глава 5. Состояние приложения. Куки. Сессии

Глава 6. Логирование

Глава 7. Маршрутизация

Глава 8. ASP.NET Core MVC

Глава 9. Контроллеры

Глава 10. Представления

Глава 11. Маршрутизация в ASP.NET Core MVC

Глава 12. Модели

Глава 13. HTML-хелперы

Глава 14. Tag-хелперы

Глава 15. View Component

Глава 16. Метаданные и валидация модели

Глава 17. Работа с данными в Entity Framework в MVC

Глава 18. Razor Pages

Глава 19. Web API

Глава 20. Фильтры

Глава 21. Аутентификация и авторизация

Глава 22. ASP.NET Core Identity

Глава 23. Клиентская разработка

Глава 24. Производительность и кэширование

Глава 25. Сервер и публикация приложения

Глава 26. Тестирование

Глава 27. URL Rewriting

Глава 28. Глобализация и локализация

Глава 29. SignalR Core

Глава 30. CORS и кросс-доменные запросы

Глава 31. Dapper

Глава 32. Работа с MongoDB

Глава 33. ReactJS

Глава 34. Сервисы gRPC

Глава 35. Дополнительные статьи

## Начало работы с ASP.NET Core

Последнее обновление: 15.12.2020

Удаленный доступ - Техническая поддержка 24x7

Uptime 99,9%. netrack.ru

ОТКРЫТЬ

Для разработки под ASP.NET Core мы можем использовать различный инструментарий. Если нашей рабочей платформой является Windows, то мы можем использовать полнофункциональную среду разработки Visual Studio. Если мы разрабатываем на Mac OS или Linux, то можем использовать расширенный редактор кода **Visual Studio Code**. Данный редактор также может работать и под Windows. В рамках данного руководства преимущественно будет использоваться среда Visual Studio 2019.

Программу для установки Visual Studio 2019 можно загрузить со страницы <https://www.visualstudio.com/downloads/>. В данном случае не важно, какой выпуск VS использовать - бесплатный Community или платные Professional или Enterprise. Все эти выпуски имеют встроенные средства для создания приложений на ASP.NET Core. В рамках этого руководства будет использоваться бесплатный выпуск VS 2019 Community.

Итак, загрузим установщик VS 2019 и запустим его. Вначале нам предлагается установить ряд опций. И так как мы будем работать с ASP.NET Core, то выбрать в программе для установке пункт **ASP.NET и разработка веб-приложений**:

Изменение — Visual Studio Community 2019 — 16.3.6

Рабочие нагрузки

Отдельные компоненты

Языковые пакеты

Веб-разработка и облако (4)

ASP.NET и разработка веб-приложений

Создание веб-приложений с использованием ASP.NET Core, ASP.NET, HTML/JavaScript и контейнеров, включа...

Разработка для Azure

Пакеты SDK Azure, средства и проекты для разработки облачных приложений и создания ресурсов с...

Разработка на Python

Редактирование, отладка, интерактивная разработка и система управления версиями для Python.

Разработка Node.js

Создавайте масштабируемые сетевые приложения с помощью Node.js, асинхронной среды выполнения...

Windows (3)

Расположение

C:\Program Files (x86)\Microsoft Visual Studio\2019\Community

Сведения об установке

Основной редактор Visual Studio

ASP.NET и разработка веб-приложений

Разработка на Python

Разработка Node.js

Разработка классических приложений .NET

Разработка классических приложений на ...

Разработка приложений для универсально...

Разработка мобильных приложений на .NET

Разработка игр с помощью Unity

Разработка мобильных приложений на яз...

Разработка игр на языке C++

Хранение и обработка данных

Приложения для обработки и анализа дан...

Разработка для Linux на C++

Кроссплатформенная разработка .NET Core

Включено

Средства разработки .NET Core

Средства разработки для платформы .NET Fr...

Необходимые компоненты для ASP.NET и ср...

IntelliCode

Также при выборе этого пункта в поле справа можно выбрать также необязательные компоненты, которые будут устанавливаться вместе с ASP.NET. Можно выбрать все компоненты.

Кроме того, чуть ниже также в программе установщика нам надо выбрать другой пункт **Кроссплатформенная разработка .NET Core**:

Изменение — Visual Studio Community 2019 — 16.3.6

Рабочие нагрузки

Отдельные компоненты

Языковые пакеты

Разработка надстроек для Office и SharePoint

Создание надстроек для Office 365 и SharePoint, решений SharePoint и надстроек VSTO на C#, VB и...

Разработка для Linux на C++

Создание приложений для Linux и их отладка.

Кроссплатформенная разработка .NET Core

Создание кроссплатформенных приложений с помощью .NET Core, ASP.NET Core, HTML, JavaScript и...

Расположение

C:\Program Files (x86)\Microsoft Visual Studio\2019\Community

Сведения об установке

Разработка классических приложений на ...

Разработка приложений для универсально...

Разработка мобильных приложений на .NET

Разработка игр с помощью Unity

Разработка мобильных приложений на яз...

Разработка игр на языке C++

Хранение и обработка данных

Приложения для обработки и анализа дан...

Разработка для Linux на C++

Кроссплатформенная разработка .NET Core

Включено

Средства разработки .NET Core

Средства разработки для платформы .NET Fr...

Необходимые компоненты для ASP.NET и ср...

IntelliCode

Отметив все необходимые нам опции, выполним установку Visual Studio.

После установки откроем Microsoft Visual Studio 2019 и при создании проекта выберем пункт **ASP.NET Core Web Application** - тип проекта для создания веб-приложения ASP.NET Core:

станок лазерн

Более высокая

резки мета

Более гл

режущая по

A Сер

Запрос се

Помощь сайту

YooMoney:  
410011174743222

Перевод на карту  
Номер карты:  
4048415020898850  
Номер карты:  
4890494751804113

□ ×

# Create a new project


Search for templates (Alt+S)

Clear all

C#

All platforms

All project types

 Console App (.NET Core)  
A project for creating a command-line application that can run on .NET Core on Windows, Linux and MacOS.


C#

Linux

macOS

Windows

Console

 ASP.NET Core Web Application  
Project templates for creating ASP.NET Core web apps and web APIs for Windows, Linux and macOS using .NET Core or .NET Framework. Create web apps with Razor Pages, MVC, or Single Page Apps (SPA) using Angular, React, or React + Redux.

C#

Linux


macOS

Windows

Cloud

Service

Web

 Blazor App  
Project templates for creating Blazor apps that run on the server in an ASP.NET Core app or in the browser on WebAssembly (wasm). These templates can be used to build web apps with rich dynamic user interfaces (UIs).

Next

На следующем шаге дадим какое-нибудь имя проекту, например, HelloApp, и определим для него местоположение на жестком диске:

## Configure your new project

ASP.NET Core Web Application 

C# Linux macOS Windows Cloud Service Web

Project name

HelloApp

Location

C:\Users\Eugene\Source\Repos\ASPNET\

Solution name

HelloApp

☐ Place solution and project in the same directory

Back


Create


После этого отобразится окно выбора шаблона нового приложения:


Create a new ASP.NET Core web application


.NET Core


ASP.NET Core 5.0


 **ASP.NET Core Empty**  
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.

 **ASP.NET Core Web API**  
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.

 **ASP.NET Core Web App**  
A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.

 **ASP.NET Core Web App (Model-View-Controller)**  
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.

 **ASP.NET Core with Angular**  
A project template for creating an ASP.NET Core application with Angular

 **ASP.NET Core with React.js**

**Authentication**  
No Authentication  
Change

**Advanced**  
☒ Configure for HTTPS  
☐ Enable Docker Support  
(Requires Docker Desktop)  
Linux

Author: Microsoft  
Source: Templates 5.0.1

Back

Create

Здесь нам доступно несколько типов проектов:

- **ASP.NET Core Empty:** пустой шаблон с самой минимальной функциональностью для создания приложений с нуля
- **ASP.NET Core Web API:** проект веб-приложения, который использует архитектуру REST для создания веб-сервиса
- **ASP.NET Core Web App:** проект, который для обработки запросов по умолчанию использует Razor Pages
- **ASP.NET Core Web App(Model-View-Controller):** проект, который использует архитектуру MVC
- **ASP.NET Core with Angular:** проект, предназначенный специально для работы с Angular 2+.
- **ASP.NET Core with React.js:** проект, который использует React.JS
- **ASP.NET Core with React.js and Redux:** проект, который использует React.JS и Redux

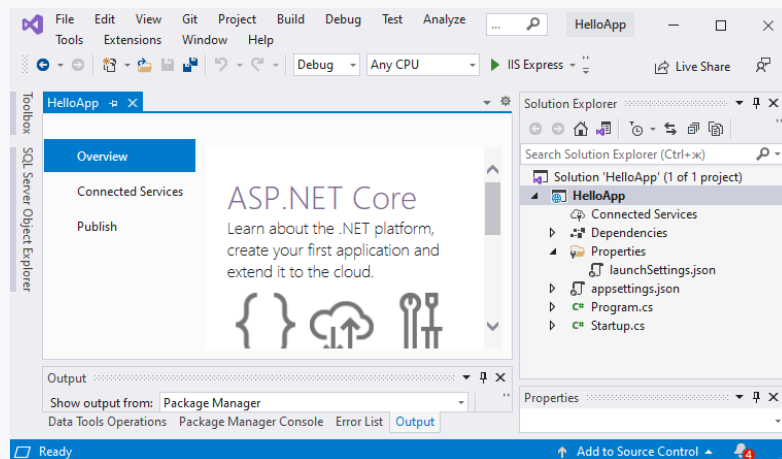
Кроме того, здесь мы можем указать версию ASP.NET Core в выпадающем списке, но в данном случае оставим значение по умолчанию - ASP.NET Core 5.0.

Также здесь можно указать тип аутентификации, который по умолчанию используется в проекте, и подключить контейнер Docker.

Также здесь есть флажок "Configure for HTTPS". При установке этого флажка проект при отладке и тестировании по умолчанию будет запускаться по протоколу HTTPS. В данном случае установка и снятие этого флажка не имеет значения. Кроме того, здесь есть возможность установить флажок "Enable Docker Support" (Requires Docker Desktop).

протоколу HTTPS. В данном случае установка и неустановка этого флажка не имеет значения. Кроме того, даже если мы установили эту отметку, то впоследствии через свойства проекта можно отменить запуск через HTTPS или, наоборот, заново установить.

Среди этих шаблонов выберем **ASP.NET Core Empty**, все остальные значения оставим по умолчанию и нажмем на кнопку OK. И Visual Studio создает новый проект:

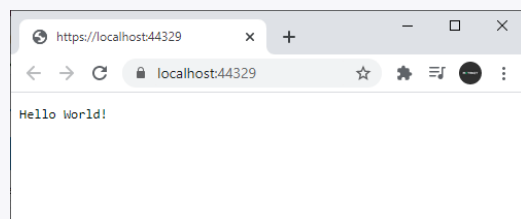


## Структура проекта ASP.NET Core

Рассмотрим базовую структуру стандартного проекта ASP.NET Core. Проект **ASP.NET Core Empty** содержит очень простую структуру - необходимый минимум для запуска приложения:

- **Connected Services:** подключенные сервисы из Azure
- **Dependencies:** все добавленные в проект пакеты и библиотеки, иначе говоря зависимости
- **Properties:** узел, который содержит некоторые настройки проекта. В частности, в файле launchSettings.json описаны настройки запуска проекта, например, адреса, по которым будет запускаться приложение.
- **appsettings.json:** файл конфигурации проекта в формате json
- **Program.cs:** главный файл приложения, с которого и начинается его выполнение. Код этого файла настраивает и запускает веб-хост, в рамках которого разворачивается приложение
- **Startup.cs:** файл, который определяет класс Startup и который содержит логику обработки входящих запросов

Данная структура, конечно, не представляет проект полнофункционального приложения. И если мы запустим проект, то в браузере увидим только строку "Hello World!", которая отправляется в ответ клиенту с помощью класса Startup:

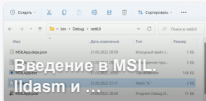
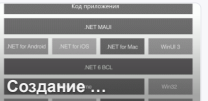
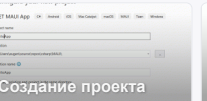


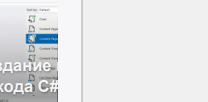


При создании других типов проектов ASP.NET структура будет отличаться, соответственно начальный проект будет иметь больше функционала, однако это тот каркас, от которого мы можем отталкиваться, добавляя в него какие-то свои файлы и папки.

[Назад](#) [Содержание](#) [Вперед](#)



ТАКОЕ НА METANIT.COM

 <p>2 месяца назад • 5 коммент...</p> <p>Введение в MSIL, компиляция кода на языке программирования C# в ...</p>	 <p>месяц назад • 9 комментариев</p> <p>Введение в MAUI (.NET Multi-platform App UI) и кроссплатформенную ...</p>	 <p>месяц назад • 2 комментариев</p> <p>Создание первого проекта на .NET MAUI и C# в Visual Studio, настройка ...</p>	 <p>месяц назад • 2 комментариев</p> <p>Контейнер AbsoluteLayout в .NET MAUI и C#, установка абсолютных ...</p>	 <p>месяц назад • 2 комментариев</p> <p>Страницы AppShell и MainPage в проекте .NET MAUI и C#, ...</p>	 <p>месяц назад • ...</p> <p>Создание ин кода C# в .NET MAUI и C#, ...</p>
---	--	--	---	---	---



Присоединиться к обсуждению...

войти с помощью

или через DISQUS ?



Имя



Анон · 6 лет назад

Здравствуйте.

Честно запутался с тем, как связаны .Net, .Net Core и DNX.

1) .Net Core представляет собой облегченный .Net, который имеет минимальный набор библиотек для работы ASP.NET приложения и так же среду CLR?

2) DNX - это одновременно и SDK и среда исполнения?

3) .Net Core и DNX - это одно и то же?

[Вконтакте](#) | [Telegram](#) | [Twitter](#) | [Канал сайта на youtube](#) | [Помощь сайту](#)

Контакты для связи: metanit22@mail.ru

Copyright © metanit.com, 2012-2022. Все права защищены.