
ASP.NET Core
.Net Angular Juice
Ahmed Khalil

Edition V1.0

Published By: Ahmed Khalil {Student of Computer and Information Systems Department Mansoura University Egypt}

Using Technologies of Microsoft, Our beloved organization (.Net Core and Visual Studio and other tools}

Using Trading Mark of Microsoft and .Net Organization

Using Angular Trading Mark

Acknowledgement

Table of Contents

Abstract

Section 1 Introduction

Introduction

The technologies of development are daily in changes, for many reasons the technologies are provided to improve quality and decrease the time of works and projects, as the guide talk about tow technologies are in trends today but not for all the future, so these guides make just a realtime guides that developers spend much time for recognition and learning.

.Net Core Rest APIs become much usable as a backend tool to satisfy more productivity, as it is a way to resolve and module the system into many parts like Microservices Architecture and other Architecture, but the jobs really need someone who can do specialized thing like Backend developer or Frontend developer, because the fall of the Full stack development.

Someone specialized in .Net Core APIs and another one specialized in Angular is best than MVC developer as a performance of coding because of modular systems.

Angular is most popular frontend framework the can module the architecture of web pages into components and load each component when request by rewriting the content not to reload a new page so the performance is after loading SPA (Initial loading).

About this guide

This guide for every backend and frontend developer who need to merge the concepts and integrate the backend with frontend in best practices of Microsoft standards and Heuristics of clean code principles, the guide separate the two concepts so the reader can be only an Angular developer or a .Net Backend developer, so the most required concern here is separation of these concepts, and making Not a juice type, but like the water and oil to make sense.

This guide is working in a good example of application Market.IO is available on GitHub repos.

Focusing in tow technologies or frameworks that achieve backend and frontend concepts is the good way to start a good career based on the experience of work.

Versions

This guide has been revised to cover the .Net Core 3.x and Angular 9 plus many additional update related to tow "Waves" of technologies like .Net Core Packages provided by Nuget package manager and Angular 9 provided by NPM, and because of the existence of tow frameworks as cores so the versions begin with version 1.

What this guide does not cover

The Guide does not cover the advanced concepts of frontend development in angular like animation as this guide focus on the integrations between frontend and backend much, and what is the best practices about using middlewares of .net core, how you can create a great architecture, how to communicate with best practices and the guide will guide you to other advanced topics as references or additional resources.

Send me your feedback

You can send your feedback about my work in my repository on GitHub or compose e-mail to me at

- GitHub Repository
  - https://github.com/AhmedKhalil777/.Net-Angular-Juice
- E-mail
  - progeng_ahmed_khalil@outlook.com

Section 2 Introduction to .Net Core & Angular 9

.Net Core & Angular 9

.Net is a platform that provide the developer with tools, like its built-in tools, programming languages and libraries, that can build many types of Applications.

- Programming Languages C#, F#, Visual Basic
- Tools supported for Linux, Mac, Windows, and Docker
- Libraries working with Files, Strings, Databases, Cloud, Realtime and more

FIGURE 2-1 THE LOGOS OF ASPNET CORE AND ANGULAR

.Net Core is a cross platform and the successor of .Net framework, it is lightweight to run on many operating systems Mac, Windows, Linux, for many processor types X64, X86, ARM 32, ARM 64.

Angular is a frontend framework that consume Rest APIs to build web or mobile app, considered as the most common framework, other framework like Blazor, React, Vue Js, or the developers can use native Js development like ajax requests.

The Most important tool for web development is Asp.net that extends the functionalities of the .Net Platform to serve the developers in web development, and other tools for managing the services in Web Background.

Asp.Net Core

ASP.Net Core provide the good tools to build Backend-Applications like Rest APIs, GraphQL, gRPC, the provide the services over many protocols and this is the core study of book, this book will show different tools and libraries that developers can use to build good applications.

Tools that developer can use to process web requests in C# or F# and using such web-page templating syntax like razor but this is not our study, libraries for common web patterns MVC using adaptable technologies to register as a service like SPA, GraphQL etc.

This technology and Libraries are used by this book as a core technology to support the application as Backend Services.

Technologies and Tools

- .Net Core 3.1 Runtime
- .Net Core 3.1 SDK
- Visual Studio Code: For simple backend development
- Visual Studio: For Heavy Backend development
- Packages that will be explained in next chapters like
    - Aspnetcore.MVC
    - SignalR
    - EfCore
    - Swagger
    - Health Checks

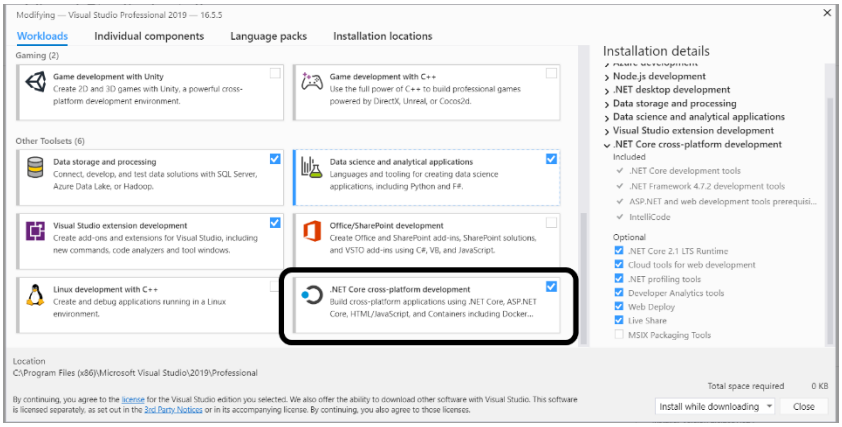


FIGURE 2-2 INSTALLING .NET CORE USING VISUAL STUDIO 2019

How to start

In this section, the developer must take a decision to use a good IDE, there are many IDEs commercial or free to use like visual studio 2019 the only version that support .Net core 3.x, visual studio code, rider and many other IDEs.

After downloading the IDE, Visual Studio support the development of .Net Core as the developer can choose it dynamically when installing by Visual Studio installer figured in Figure 2-2.

The second way you can download any IDE then developer can download the .Net Core SDK and Runtime from dot.net website illustrated in Figure 2-3.

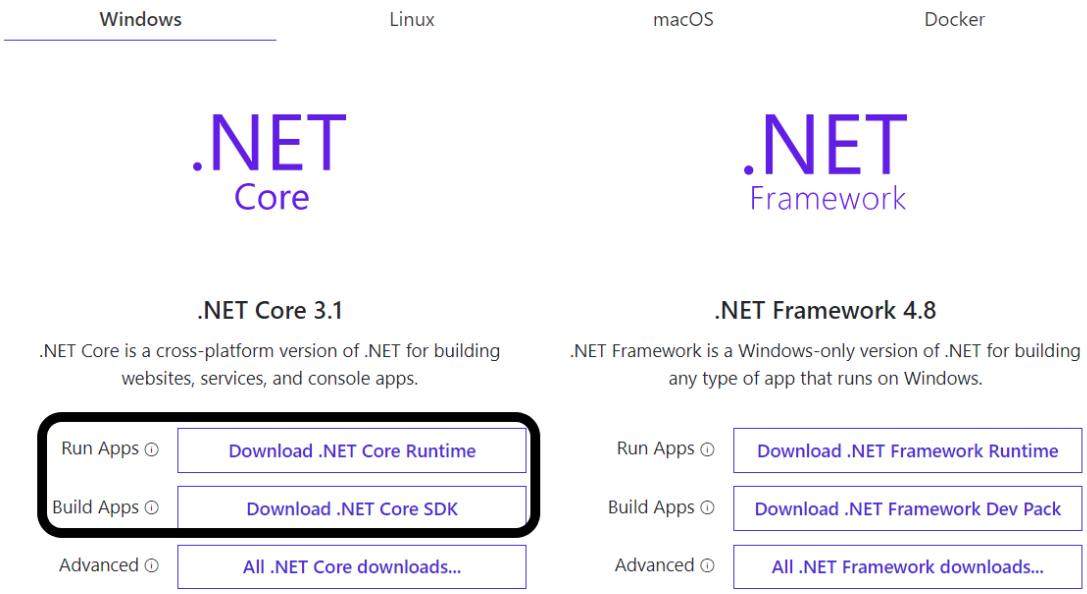


FIGURE 2-3 DOWNLOADING THE .NET CORE SDK AND THE RUNTIME FROM DOTNET WEBSITE

The developer can download them for any OS as shown in the header of the last figure, by installing them, the developer can use now the CLI to access the dotnet environment by the variable dotnet, now checking the environment version by writing command:

CLI => Command 2-1

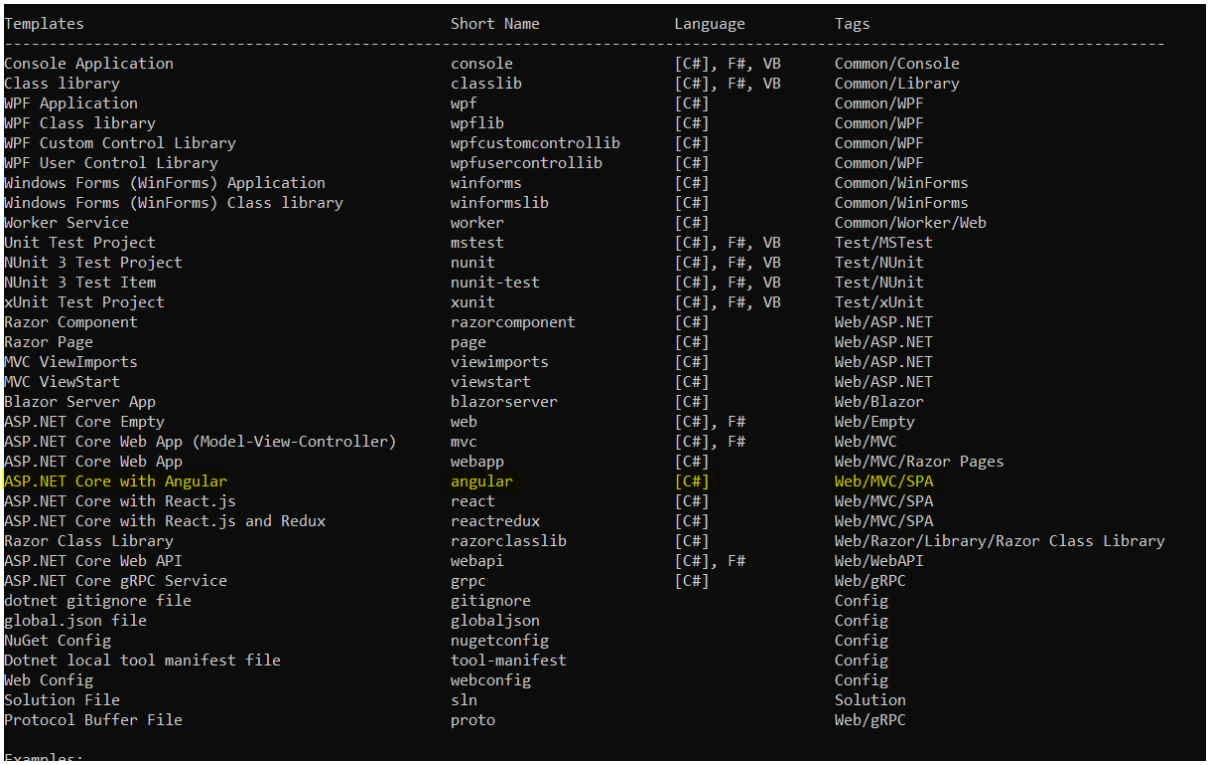
```
C:\Users\ProgE>dotnet --version

3.1.202
```

There are many templates to create web applications, which can be built-in the SDK or it can be created by the developers, but the first time, discovering the templates and listing them to choose the built-in template for Angular SPA development.

CLI => Command 2-2

```
C:\Users\ProgE>dotnet new
```

Now the CLI show all the Templates figured at Figure 2-4 and there is a template integrate the Angular SPA, and this is the guide interest to work.



Templates	Short Name	Language	Tags
Console Application	console	[C#], F#, VB	Common/Console
Class library	classlib	[C#], F#, VB	Common/Library
WPF Application	wpf	[C#]	Common/WPF
WPF Class library	wpflib	[C#]	Common/WPF
WPF Custom Control Library	wpfcustomcontrollib	[C#]	Common/WPF
WPF User Control Library	wpfusercontrollib	[C#]	Common/WPF
Windows Forms (WinForms) Application	winforms	[C#]	Common/WinForms
Windows Forms (WinForms) Class library	winformslib	[C#]	Common/WinForms
Worker Service	worker	[C#]	Common/Worker/Web
Unit Test Project	mstest	[C#], F#, VB	Test/MSTest
NUnit 3 Test Project	nunit	[C#], F#, VB	Test/NUnit
NUnit 3 Test Item	nunit-test	[C#], F#, VB	Test/NUnit
xUnit Test Project	xunit	[C#], F#, VB	Test/xUnit
Razor Component	razorcomponent	[C#]	Web/ASP.NET
Razor Page	page	[C#]	Web/ASP.NET
MVC ViewImports	viewimports	[C#]	Web/ASP.NET
MVC ViewStart	viewstart	[C#]	Web/ASP.NET
Blazor Server App	blazorserver	[C#]	Web/Blazor
ASP.NET Core Empty	web	[C#], F#	Web/Empty
ASP.NET Core Web App (Model-View-Controller)	mvc	[C#], F#	Web/MVC
ASP.NET Core Web App	webapp	[C#]	Web/MVC/Razor Pages
ASP.NET Core with Angular	angular	[C#]	Web/MVC/SPA
ASP.NET Core with React.js	react	[C#]	Web/MVC/SPA
ASP.NET Core with React.js and Redux	reactredux	[C#]	Web/MVC/SPA
Razor Class Library	razorclasslib	[C#]	Web/Razor/Library/Razor Class Library
ASP.NET Core Web API	webapi	[C#], F#	Web/WebAPI
ASP.NET Core gRPC Service	grpc	[C#]	Web/gRPC
dotnet gitignore file	gitignore		Config
global.json file	globaljson		Config
NuGet Config	nugetconfig		Config
Dotnet local tool manifest file	tool-manifest		Config
Web Config	webconfig		Config
Solution File	sln		Solution
Protocol Buffer File	proto		Web/gRPC

Examples:

FIGURE 2-4 THE LIST OF THE APPS THAT THE DEVELOPER CAN CHOOSE TO DEVELOP

Taking a look to a list item, The Template field describe what is the template (Asp.Net Core with Angular), then the Short Name is the way to call this template using CLI (angular), the supporting languages which by default is C# and the only for Angular, then the Tags.

Calling the angular template will create a new project as a blueprint from the origin template, the output is a project with some special concerns than the other applications or templates, crating a project using the next command.

CLI => Command 2-3

```
F:\MyBooks\Practical\Dotnet_and_Angular\src>dotnet new angular -n "MarketIO.Angular"
```

The result is a repository with the name of MarketIO.Angular, which contain the basic structure of Aspnetcore Rest API and, the Razor Pages provide the host with basic error pages when Angular app fails, but this is not our desire, and the main Angular app in the ClientApp folder, this means the backend of .Net Core is the manager of the Angular app, and the basic controller for routing and other concerns, the guide will explain all of concerns in next chapters, the structure of the Repository illustrated in Figure 2-5.

The basic template provide just a simple service that show the fake whether data and a counter and the home page, the Controller is the place where the Rest API Actions exist to provide data using Http methods get, post, delete, put, patch and others, and control basically the CRUD operations, but the guide will discuss best practices to implement the functionalities in good ways.

The wwwroot folder is the folder that is hosted when deployment and contains the files provided to users and static files, it can be the File Db Repository but this is not good practice for scalability and other concerns, the guide will discuss these concerns later in next sections.

FIGURE 2-5 THE BASIC APPLICATION STRUCTURE

Like any .Net App that can run, it contains the Program.cs file where the Main method start the app, but to create something understandable by developer there is a class of Startup that contain all the middlewares and the pipelines start at the application run or after hosting, this class is like how to drive a car, give the application the kit of configurations for concerns like gas and other tools to run the car by the key which give the application (car) the power but not to go but when host it can go.

This sample is a good simple way to present the .Net core integrated to Angular but thinking about scratch work is the best way to create app and explain.

Steps of making it like a scratch

- Delete the controller exist WhetherForecastController.cs and its model class in the root WheatherForecast.cs.
- Delete the ClientApp Repository to make Angular app from scratch.

The next steps the guide will explain the Angular and How to create the app.

Angular 9

Angular is a platform for building mobile and desktop web applications this is the book frontend application technology, serve as SPA, this paradigm of technology use one page to control all other functionalities of system by calling something called components, these components shown by programming functionalities that provided by initial loading files and some requests to backend.

Single Page Application Paradigm

A single-page application (SPA) is a web application or website that interacts with the web browser by dynamically rewriting the current web page with new data from the web server, instead of the default method of the browser loading entire new pages. The goal is faster transitions that make the website feel more like a native app. (WikiPedia, 2019)

Is JavaScript being the only way to build SPAs

The developers thought that the technology of SPA, the only way to implement is to write JS code or a framework build on JS like Angular and developer write TypeScript (not native) but compiled into JS natively, and the Figure 2-6 explain the importance and the features of TypeScript over JavaScript.

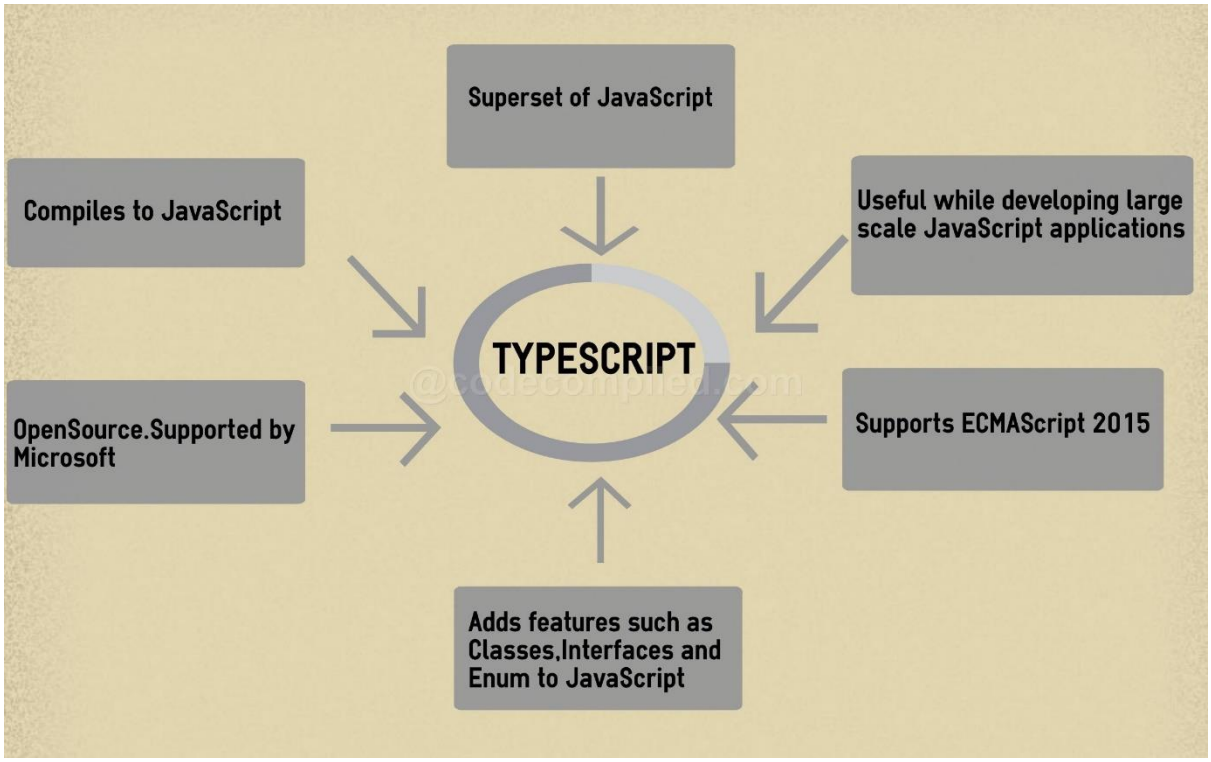


FIGURE 2-6 FEATURES OF TYPESCRIPT TO CODING, HTTPS://WWW.CODECOMPILED.COM/TYPESCRIPT-TUTORIALS/TYPESCRIPT-TUTORIAL-LEARN-TYPESCRIPT-BASICS/

.Net Team provide a new technology called Blazor to create a SPA using C# language as they provide the WebAssembly written to the DOM of Web, Blazor components compiled and run together then update the page, JavaScript interpret function then update the page directly change the DOM by Wrappers illustrated in Figure 2-7 .

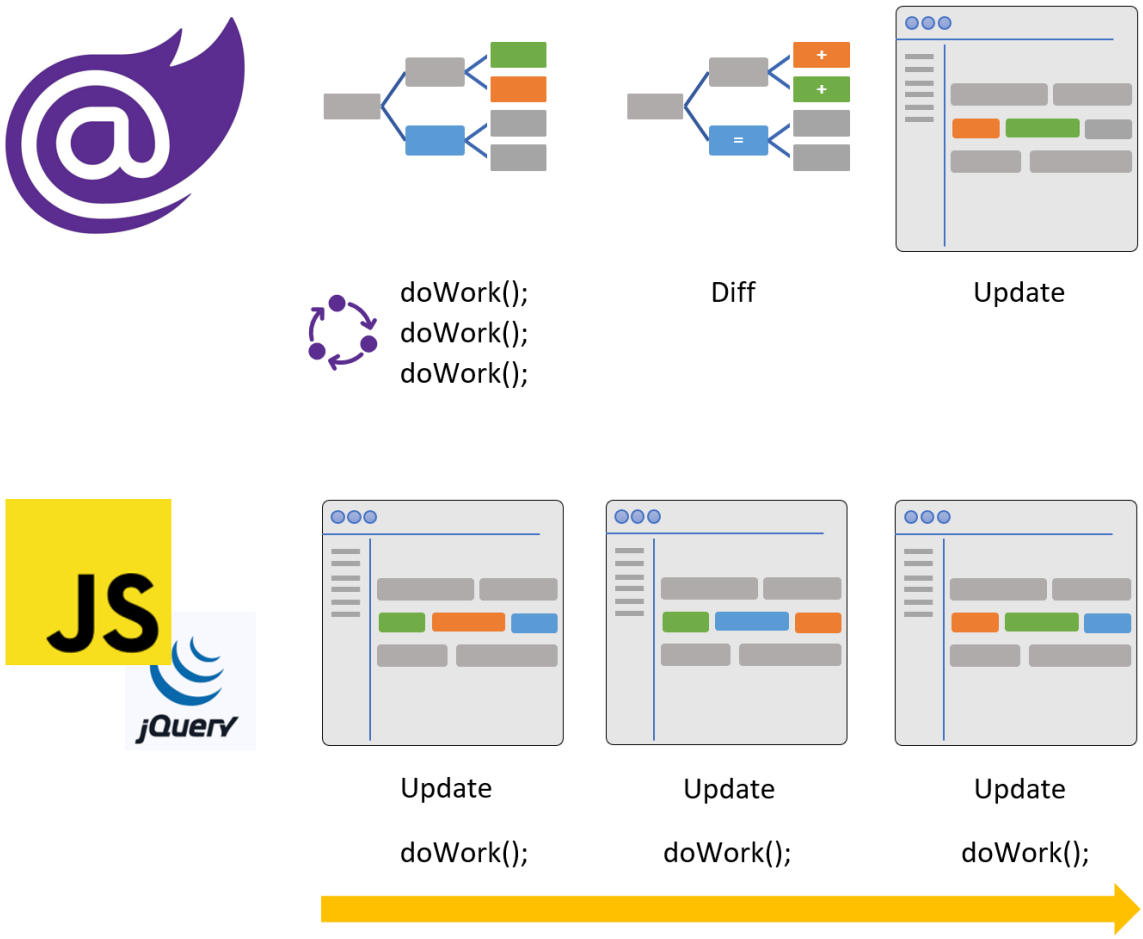


FIGURE 2-7 BLAZOR VS JS EXECUTING MODELS HTTPS://WWW.TELERIK.COM/BLOGS/COMPARING-NATIVE-BLAZOR-COMPONENTS-TO-WRAPPED-JAVASCRIPT-COMPONENTS

Bibliography

Flanagan, D. (2006). *JavaScript - The Definitive Guide.*