Technical University Of Sofia

Course Project

Computing

ASP.Net And .Net Core Features

Sevcan ALKAN Prof.Roumen Trifonov

Sofia

2017

Content

1. INTRODUCTION AND SUMMARY OF TOPICS
2. GENERAL INFORMATION ABOUT PROGRAMMING LANGUAGES
3. INTRODUCTION TO C# AND .NET
   1. What Is C#, History Of C# And Why Use C#
   2. What Is .Net And How Does .Net Work
4. WHAT IS ASP.NET AND HISTORICAL DEVELOPMENT OF ASP.NET
   1. MVC Design Pattern
   2. What Is ASP.Net Web Api
   3. ASP.Net Some Good Integrations
      1. AutoFac
      2. Entity Framework
      3. .Net Identity
      4. SignalR
      5. Elmah
5. WHAT IS ASP.NET CORE
   1. ASP.Net Core Features
6. WHICH ONE CHOOSE: ASP.NET VS. ASP.NET CORE
   1. How Do Create ASP.Net Core WebApi And Basic Get, Post, Put, Delete Operations with EntityFramework(MS SQL)
7. Concussion
8. Refefances
9. Introduction And Summary Of Topics

I will focus mainly on ASP.NET after I have general information about programming languages and C # .Net. Finally, the sample application will be written with ASP.Net Core.

1. General Information About Programming Languages

The programming languages is a standardized note that the programmer uses to express a certain algorithm. Programmer can uses different programming languages for write some commands. Programming languages allow the porgrammer to tell which data conditions to process on the computer and which operations to perform under which conditions. More than 2500 programming languages have been made so far.

Programming languages divided into three groups.

**Low Level Languages:** This programming languages is very close to the machine code. Machine managment is very developet. People learning these languages it must be information about the microprocessor. (Ex. Assembly Programming language)

**Medium Level Languages:** These languages, which are very flexible, can do both upper and lower level programming. It is a little clearer than lower-level languages. (Ex. C Programming language)

**High Level Languages:** They are also called event-based programming languages, but these programming languages only work around certain functions and reduce the programming skills. The fastest and most effective programming languages are this category. As it is easier to learn and apply than other programming languages, the most suitable languages for beginners are high-level programming languages.

The program written in high-level programming languages must be translated into machine language in order to be able to run. For this, the program is written in which high-level language the compiler is used. Thus the source program written in the high-level programming language is converted into the program purpose in the machine language. It is possible to change the content of the source program, but the compiled purpose has no way of interfering with the content of the program.

1. Introduction To C# And .Net
   1. What Is C#, History Of C# And Why Use C#

C# is a simple, modern, general-purpose, object-oriented programming language designed by Microsoft. Microsoft is also a new generation programming language developed for the .NET platform. More than 40 programming languages are available on the .NET platform. Some sections assume C# is the main language of the .NET platform. Especially related information based programming concept.

C#, which started its development in 2002, continues to develop into nowdays. Versions of C#;

* **C# 1.0/1.2** (2002) - Modern, object-orıented, type safe programming language.
* **C# 2.0** (2005) - Generics, partical class, anonymous types, iterators, nullable type, static classes, delegate interface.
* **C# 3.0** (2007) - Implicit types, object/collection initilizers, auto-implemented properties, extention methods, lambda expression, expression trees, partical methods.
* **C# 4.0** (2010) - Dynamic binding, named and optimal arguments, generic covariance and contavariance.
* **C# 5.0** (2013) - Async methods, caller info attributes, tuples.
* **C# 6.0** (2015) – Roslyn, await in catch/finaly, auto property initalizer, string interpolation, nameof operator, dictionary initalizer.
* **C# 7.0** (2016) – Tuples, pattern matching, decomposition, improved out variables, ref returns, local functions, literal impovements.

Why Use C#: C# is an elegant, simple, type-safe, object-oriented language that allows enterprise programmers to build a breadth of applications. C# also gives you the capability to build durable system-level components by virtue of the following features:

* Full COM/Platform support for existing code integration.
* Robustness through garbage collection and type safety.
* Security provided through intrinsic code trust mechanisms.
* Full support of extensible metadata concepts.

You can also interoperate with other languages, across platforms, with legacy data, by virtue of the following features:

* Full interoperability support through COM+ 1.0 and .NET Framework services with tight library-based access.
* XML support for wWeb-based component interaction.
* Versioning to provide ease of administration and deployment.

C#’s usage areas;

* Console Application
* Winform Application for Windows
* ASP.NET  for WebSite, WebApi and WCF services
* Cross-Platform Mobile Application with Xamain(Mono)
* UWP(Universal Windows Platform one application supported for all windows systems: Windows 10, Windows 10 Mobile, Windows 10 IOT, Xbox.
* Ardino Programming with 3.rd libary
* Azure programming
  1. What Is .Net And How Does .Net Work

.Net Microsoft is a platform that aims to develop applications independent of the programming language and the system to be run. It is not a programming language; on the contrary, it is an environment that allows many programming languages to develop applications. If you do not develop an application, this platform is a simple tool that works for you in the background. If you develop an application, you can develop powerful programs or web applications using Visual Studio with one of the programming languages supported by the platform. There is a special relationship with the C# 'in runtime environment, the .NET Framework. C# was designed to develop code for the .NET Framework, and the libraries used by C# were libraries defined by the .NET Framework.



An application you write on the .Net platform is independent of the platform and platform. But how? At this point, the components of the platform are important. There are many components in the platform, which seems quite complicated for beginners. For example, a code you write with C # is adapted to the Common Language Infrastructure (CLI) with the compiler. From there, the Common Intermediate Language (CIL) sends the Common Language Runtime (CLR) to the platform to convert the platform to the appropriate language. CLR's function is to convert the code sent by CIL into machine language. Thus, the code written in C # (with other platform languages) is compiled and made executable.

1. WHAT IS ASP.NET AND HISTORICAL DEVELOPMENT OF ASP.NET

ASP.Net is a web application development framework. It was developed with significant improvements over classic ASP developed by Microsoft. ASP.Net is a modern technology and allows the creation of web pages, web applications and XML web services. ASP.Net programming language is not.

Version History Of ASP.Net;

**Asp.Net 1.0**

**Release Date:** 16 January, 2002  
**.Net Framework:** .Net 1.0 with Visual Studio .Net  
**Features:**

* Use of dll class libraries
* Support for object-oriented web application development

**Asp.Net 1.1**

**Release Date:** 24 April, 2003  
**.Net Framework:** .Net 1.1 with Visual Studio .Net 2003  
**Features:**

* Built-in support for databases and ODBC.
* ASP.NET Mobile controls
* Internet Protocol version 6 (IPv6) support

**Asp.Net 2.0**

**Release Date:** 07 November, 2005  
**.Net Framework:** .Net 2.0 with Visual Studio 2005  
**Features:**

* Some New Server Controls were introduced in ASP.NET 2.0:
* ImageMap Control
* BulletedList Control
* HiddenField Control
* File Upload Control
* Wizard Control
* Localize Control
* MultiView and View Controls
* Substitution Control
* Data Controls - GridView Control, DetailsView Control, FormView Control
* Data Source Controls - SqlDataSource Control, AccessDataSource Control, ObjectDataSource Control, XmlDataSource Control, SiteMapDataSource Control, SiteMapPath Control
* Navigation Controls - SiteMapPath Control, Menu Control, TreeView Control
* Login Controls - Login Control, LoginView Control, PasswordRecovery Control, LoginStatus, LoginName, ChangePassword
* CreateUserWizard
* Web Parts
* Control State
* Master Pages
* Themes
* Skins
* Roles and Personalization
* Profiles
* Localization and Globalization
* Automatic Compilation
* Compiled Deployement and Source Protection
* Improvements in Code Behind Model
* Membership Service
* Cross Page Postback
* Validation Groups
* Role Management
* ASP.NET Configuration API

**Asp.Net 3.5**

**Release Date:** 19 November, 2007  
**.Net Framework:** .Net 3.5 with Visual Studio 2008  
**Features:**

* New Controls:
* Data Control - ListView Control, DataPager Control
* Data Source Control - LinqDataSource Control, EntityDataSource Control
* LINQ Support
* Silverlight Support
* Integrated ASP.NET AJAX
* Multi Targeting Framework Support
* JavaScript Debugging and Intellisense
* ASP.NET Merge Control for merging precompiled assemblies
* Tight Integration with IIS 7.0
* New CSS Design Tool

**Asp.Net 4.0**

**Release Date:** 12 April, 2010  
**.Net Framework:** .Net 4.0 with Visual Studio 2010  
**Features:**

* Setting Meta Tags with the Page.MetaKeywords and Page.MetaDescription Properties
* Enabling View State for Individual Controls
* Support for Recently Introduced Browsers and Devices
* Browser Capabilities Providers
* Routing
* ClientIDMode Property
* Persisting Row Selection in Data Controls
* Enhancements in FormView and ListView controls
* New filter control QueryExtender Control

**Asp.Net 4.5**

**Release Date:** 15 August, 2012  
**.Net Framework:** .Net 4.5 with Visual Studio 2012  
**Features:**

* Async Support
* Support for asynchronous modules and handlers
* Model Binding
* Strongly Typed Data Controls
* Unobtrusive Validation
* Bundling and Minification
* Support for WebSocket Protocol
* OAuth Support for login through other social sites
* HTML5 features enhancements
* Friendly URL
* Value Providers
* ASP.NET Web API
* Filtering by values from a control
* HTML Encoded Data-Binding Expressions

**Asp.Net 4.5.1**

**Release Date:** 17 October, 2013  
**.Net Framework:** .Net 4.5.1 with Visual Studio 2013  
**Features:**

* One ASP.NET
* ASP.NET Scaffolding
* ASP.NET Identity
* Bootstrap
  1. MVC Design Pattern

ASP.NET MVC is a framework developed by Microsoft to add MVC pattern to ASP.NET. To understand what ASP.NET MVC is, it is useful to first examine what MVC is.

MVC is one of the most important architectural patterns in application development (especially web application development). Today, MVC is the Microsoft ASP.NET MVC Framework that comes to mind, whereas it has been in the software world since 1979 (Microsoft was founded in 1975).

It consists of the initials of MVC, Model, View, Controller words, and each word represents a different layer of MVC.

**Model:** In the MVC world, the location of the model application data or state is usually in the database or xml / json file format. The model isolates the data layer (database, xml, json file, etc.) from the application, so you do not need to know where the data layer is on other layers. The model layer is often the Entity Framework, Nhibernate, LLBLGen, etc.

**View:** View is created with the layer containing the interface the client has seen, usually using the data in the Model layer. By separating the View layer from the Model and Controller layers, interface changes can be made without having to change other layers of the application.  
It is possible to use the latest version technologies such as HTML5 and CSS3 in the View layer. With HTML5 and CSS3, it is very easy to develop applications that can work on desktop and mobile browsers. In fact, HTML5 and CSS3 technologies can be used to develop Windows Store applications.

**Controller:** The controller fulfills tasks such as being bridged between the Model and View layers, manipulating the request from the client. There can be one or more Actions in the Controller, usually each Action is used to generate a web page. Another important building block of MVC is Routing Mechanism.

**Routing:** Routing is a mechanism that directs the client to the appropriate Controller and Action requested by the application. The client sends the request to a specific address of the application, and through the routing mechanism, the most appropriate Controller for the address and its Action are detected and executed.

**And other features:**

* Razor View engine,
* Script bundling,
* Action Filter,
* IoC(Inversion of Control),
* Session,
* .Net Identity,
* TempData, VewBag, ViewData (for transfer data View to Controller or Controller to View)
  1. What Is ASP.Net Web Api

Before we go to Asp .Net Web Api, let's talk about Api. API expansion is “Application Programming Interface”. Is an interface that we define within the framework of certain rules to open the services or data we have to the outside world and make it available for other applications-platforms.

Asp .Net Web Api can be defined as a framework used to create services that can be communicated over HTTP protocol, which can be consumed by numerous clients (browsers, mobile phones, tablets, pc, etc.). It shows some similarities with Asp.net MVC because they have common features like routing, controllers, action results, filters, model binders, but they are not part of the MVC Framework. Asp .net Web Api is part of Core Asp .Net and can be used with MVC or other web application types. At the same time, all of these can be used as stand-alone Web services applications.

**API Features:**

* Http supports CRUD operations because it can work with Get, Post, Put and Delete methods,
* The response contains the HttpStatusCode and Accept Header parameters,
* The responses can be formatted by the type of MediaTypeFormatter the user desires,
* There is support for the ODATA and Query writing is very easy,
* It can be hosted in an application or on IIS,
* It takes some features of MVC (routing, controllers, action results, filters, model binders)
* WebAPI more ligther and faster than ASP.Net MVC
  1. ASP.Net Some Good Integrations
     1. AutoFac

Autofac is an addictive Inversion of Control container for .NET Core, ASP.NET Core, .NET 4.5.1+, Universal Windows apps, and more.

*More information for visit: https://autofac.org/*

* + 1. EntityFramework

Writing and managing ADO.Net code for data access is a tedious and monotonous job. Microsoft has provided an O/RM framework called "Entity Framework" to automate database related activities for your application.

Microsoft has given the following definition of Entity Framework:

*The Microsoft ADO.NET Entity Framework is an Object/Relational Mapping (ORM) framework that enables developers to work with relational data as domain-specific objects, eliminating the need for most of the data access plumbing code that developers usually need to write. Using the Entity Framework, developers issue queries using LINQ, then retrieve and manipulate data as strongly typed objects. The Entity Framework's ORM implementation provides services like change tracking, identity resolution, lazy loading, and query translation so that developers can focus on their application-specific business logic rather than the data access fundamentals.*

Entity framework is an Object/Relational Mapping (O/RM) framework. It is an enhancement to ADO.NET that gives developers an automated mechanism for accessing & storing the data in the database.

Entity framework is useful in three scenarios. First, if you already have existing database or you want to design your database ahead of other parts of the application. Second, you want to focus on your domain classes and then create the database from your domain classes. Third, you want to design your database schema on the visual designer and then create the database and classes.

The following figure illustrates the above scenarios.

[](http://www.entityframeworktutorial.net/Images/EF-overview.png)

As per the above figure, EF creates data access classes for your existing database, so that you can use these classes to interact with the database instead of ADO.Net directly. EF can also create the database from your domain classes, thus you can focus on your domain-driven design. EF provides you a model designer where you can design your DB model and then EF creates database and classes based on your DB model.

What is ORM?

ORM is a tool for storing data from domain objects to relational database like MS SQL Server, in an automated way, without much programming. O/RM includes three main parts: Domain class objects, Relational database objects and Mapping information on how domain objects map to relational database objects (tables, views & storedprocedures). ORM allows us to keep our database design separate from our domain class design. This makes the application maintainable and extendable. It also automates standard CRUD operation (Create, Read, Update & Delete) so that the developer doesn't need to write it manually. A typical ORM tool generates classes for the database interaction for your application as shown below.

What are the Entity Framework pluses and minuses;

**Pluses;**

* Supported Object Orianted Programming.
* It offers a structure independent of the database.
* Without having to use SQL or JDBC, we can write an application that uses the database by writing a little code.

**Minuses:**

* Performance may be inadequate in some situations.
* The control of the operation is not exactly in the hands of the software developer.
* It takes a lot of time to learn to use ORM tools.
  + 1. .Net Identity

New authorization engine for Asp.net.  
It offers new features according to the old structure and is the most featured feature in new versions of ASP.NET.  
  
Token based Claim-based authorization support. Role managemet supports.  
  
Azure active directory and social networking and login support. Two-step verification etc.

*More information for visit: https://docs.microsoft.com/en-us/aspnet/identity/overview/getting-started/introduction-to-aspnet-identity*

* + 1. SignalR

ASP.NET SignalR is a new library for ASP.NET developers that makes developing real-time web functionality easy. SignalR allows bi-directional communication between server and client. Servers can now push content to connected clients instantly as it becomes available. SignalR supports Web Sockets, and falls back to other compatible techniques for older browsers. SignalR includes APIs for connection management (for instance, connect and disconnect events), grouping connections, and authorization.

* + 1. Elmah

Elmah basicly error logging integration for ASP.Net. Easly integrate with Nuget package manager.

1. WHAT IS ASP.NET CORE

By 2014, we are seeing a significant transformation of the .NET Framework, Microsoft's application development platform.

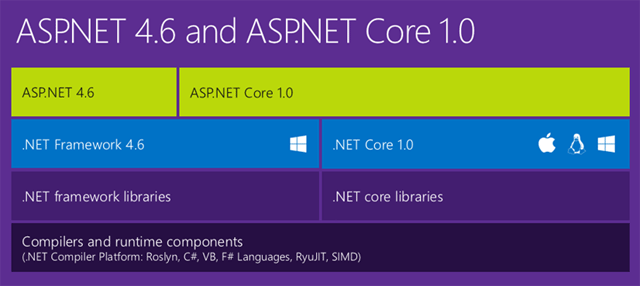
  The core release of the .NET Framework, which is not open source and runs only on Windows operating systems as the official release, announced that .NET Core will be open source and platform independent. In the past, a Mono named project provided the .NET Framework to work on Linux systems, though it was not official.

With this move by Microsoft, we are now officially releasing a .NET Framework that is open source and platform independent from the core version. So with .NET Core you can develop applications and run applications on Linux and MacOS operating systems.

In the meantime, it has come to the point of completing .NET Core development. We can say that it is especially important for ASP.NET Core, which is built on this platform and used to develop web applications. So, Microsoft, Linux and MacOS users have been passing these code warmups and Visual Studio Code named code editor for months so that they can be easily adapted. According to Microsoft's road map, .NET Core and ASP.NET Core are expected to be released as RTM and official releases in 2016.

The main purpose of .NET Core is to provide web applications development with ASP.NET Core (at least for now). With .NET Core, only web applications and console applications can be developed. Windows Forms, Windows Service, WPF etc. We can not develop applications in this environment for now.

Simplified core version of .NET Framework for .NET Core. .NET Core is made up of class libraries named CoreFX and runs on the runtime named CoreCLR. It is useful to say that you have a managed environment, that the JIT compiler and the Garbage Collector (GC) are kept exactly the same. CoreFX Many systems, such as System.IO, System.Collections. And Microsoft. Contains its components. For details you can review the CoreFX Github repo.  
Because the core is a .NET Framework version, application developers who have previously used versions of the .NET Framework 4, 4.5, etc. may not be able to find some libraries on .NET Core. It is also worth emphasizing that .NET Core will be deployed over NuGet, because some System. You can also download your libraries from NuGet and use them in your projects.

[](http://devnot.com/wp-content/uploads/2016/02/aspnet4.6-aspnetcore1.0.png)

The ASP.NET Core project structure has significant differences from the classic ASP.NET projects. Instead of web.config, it is necessary to have a different setup file with JSON extension, to have a Global.asax file instead, to have a Startup file in a similar structure, to run the applet at its simplest at startup and to use the components (static file, directory browsing, The first differences are multiplied by the fact that you can use it by recording one by one.

Finally, let's highlight what is not supported in .NET Core. Currently we can only develop with C # language, ie VB.NET etc. We can not use .NET languages. We can develop MVC and Web API application with ASP.NET Core, WebForms is not supported at present.

**ASP.NET Core, Does ASP.NET 4.5 Continue?**

No it is not. ASP.NET Core is working on .NET Core. .NET Core is a .NET deployment created from basic libraries in the .NET Framework and made much simpler for different platforms. Therefore, we should say that despite the use of ASP.NET Core and the classic ASP.NET libraries are different. ASP.NET Core will continue to evolve over the classic ASP.NET .NET Framework, not as a continuation of classic ASP.NET applications.

**What Will Classic ASP.NET Get?**

Microsoft's main development environment is still the .NET Framework, so we expect it to continue where ASP.NET is being developed. Classic ASP.NET bitecek will not go on, it will not go far ... the rhetoric is far from reality at the moment. ASP.NET 5 is actually the previous name of ASP.NET Core. In a nutshell, classic ASP.NET stands where it stands, and will continue to evolve, although not as accelerated as ASP.NET Core.

* 1. ASP.Net Core Features
* Integration with many client-side technologies. (Gulp, Grunt, TypeScript, Knockout, Angular.js)
* Web.config independent and flexible Configuration structure. (The main theme of someone from our next post)
* Built-in Dependency Injection. (We can say it is a lightweight DI integration even from AutoFac)
* Modular construction and its lightweight pipeline.
* Harness self-hosting of IIS
* Cross-platform formation
* Open source development

1. WHICH ONE CHOOSE: ASP.NET VS. ASP.NET CORE

Currently active ASP.NET 4.6 version of the classic ASP.NET, the .NET Framework is better equipped, stable and useful. So when we think of technical equipment and things to do, classic ASP.NET is obviously ahead. In addition to its powerful and well-structured structure, the most important weaknesses of ASP.NET Core are its performance and performance on Windows platforms only.

At this stage, ASP.NET Core should only be selected for ar-ge purpose, because it is a bit risky to develop an application on a structure which is not yet stable version or even RTM version. Release is now available with peace of mind, even if the application you will be developing in the style of MVC or Web API should be your first choice. Apart from the performance advantages offered by classic ASP.NET, you can also reduce your server costs by hosting your applications on a Linux server. If your servers are Windows and you are working on Windows plarforms, classic ASP.NET will be a better choice for you, but start by examining ASP.NET Core and keeping up with developments.

* 1. How Do Create ASP.Net Core WebApi And Basic Get, Post, Put, Delete Operations with EntityFramework(MS SQL)

Concussion

Referances

* Programlama Dilleri Ve Özellikleri(Programming Languages And Features) <http://www.robotiksistem.com/programlama_dilleri_ozellikleri.html> (28 February 2017)
* Enes Taha Selek (13 February 2009). C# Nedir?(What is C#?) <https://csharpturkcesi.wordpress.com/2009/02/13/csharp-ve-dotnet/> (28 February 2017)
* Bikesh Srivastava (30 January 2017). History Of C# Programming Language. <http://www.c-sharpcorner.com/blogs/history-of-c-sharp-programming-language> (28 February 2017)
* Çağlar Gül (27 November 2016). C# Ve .Net’e Giriş(Introduction to C# And .Net) <http://www.caglargul.com/2016/11/c-sharp-ve-net-giris.html> (28 February 2017)
* (July 2016). C# Diline Ve .Net Framework’e Giriş(Introduction to C# And .Net Framework) <https://msdn.microsoft.com/tr-tr/library/z1zx9t92.aspx> (28 February 2016)
* Yücel Karaca. ASP.Net Nedir Ve Ne İşe Yarar?(What Is ASP.Net And What Makes It Work?). https://www.teknologweb.com/asp-net-nedir-ne-ise-yarar (28 February 2017)
* Manish Banga (01 March 2015). Brief Version History Of ASP.NET With Features. http://www.dotnetstudy.com/brief-version-history-of-aspnet-with-features?id=6 (28 February 2017)
* Bora Kasmer (19 July 2014). ASP.NET MVC Nedir, Ne İşe Yarar(What is ASP.NET MVC, What it Does) http://www.borakasmer.com/asp-net-mvc-nedir-ne-ise-yarar/ (28 February 2017)
* http://www.entityframeworktutorial.net/what-is-entityframework.aspx (28 February 2017)
* Uğur Umutluoğlu (22 February 2016). ASP.NET Core 1.0 ve ASP.NET’in Geleceği(ASP.NET Core 1.0 and the Future of ASP.NET) http://devnot.com/2016/asp-net-core-1-0-ve-asp-netin-gelecegi/ (28 February 2017)