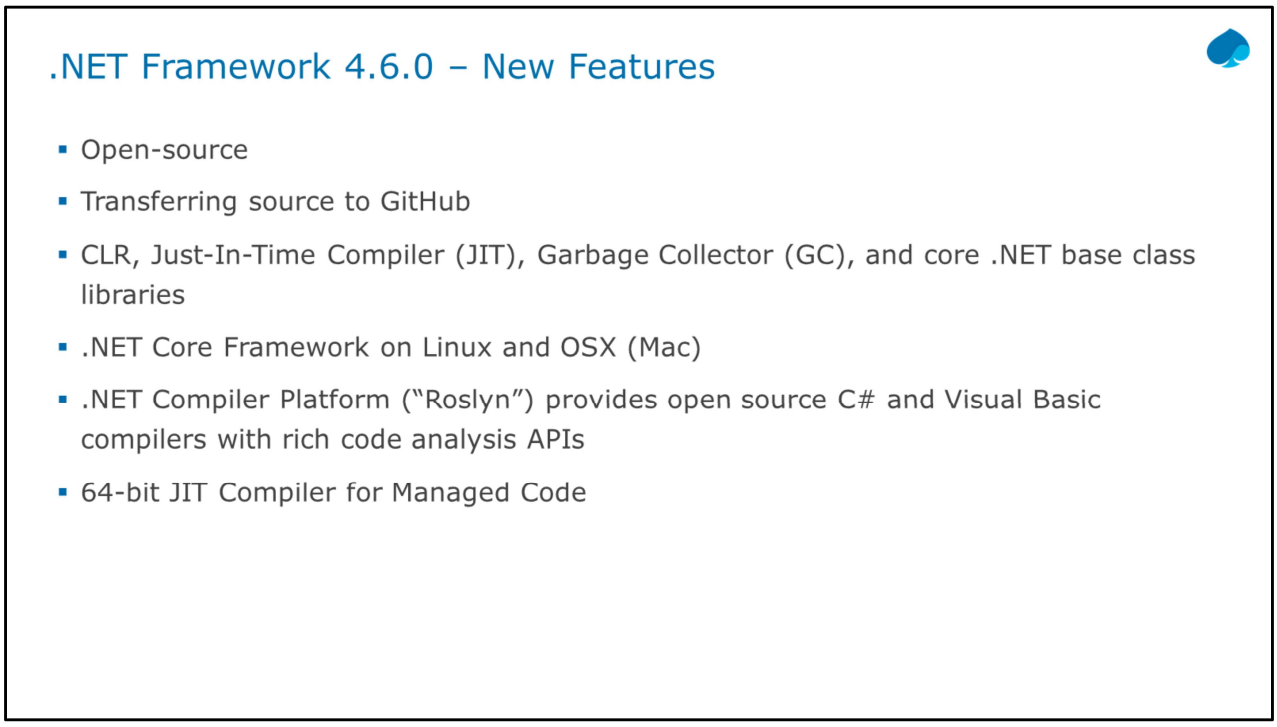
.NET Framework 4.6 New Features



.NET 2015 introduces the .NET Framework 4.6 and .NET Core. Some new features apply to both, and other features are specific to .NET Framework 4.6 or .NET Core. Now .NET Core has been made open source. Now it has been released with the MIT license. Right now Microsoft has released a few libraries code while the full Core code will be released at the time of final .NET release.
The .NET Compiler Platform ("Roslyn") provides open-source C# and Visual Basic compilers with rich code analysis APIs. You can build code analysis tools with the same APIs that Microsoft is using to implement Visual Studio!
The .NET Framework 4.6 features a new version of the 64-bit JIT compiler (originally code-named RyuJIT).

Please refer to below site for more information

https://docs.microsoft.com/en-us/dotnet/framework/whats-new/#whats-new-in-the-net-framework-462

.NET Framework 4.6.0 – Built-In Support for IoC & DI (continued…)

- Dependency Injection - Technique whereby one object supplies the dependencies of another object.
  - A dependency is an object that can be used (a service).
  - An injection is the passing of a dependency to a dependent object (a client) that would use it.
- DI Allows the Removal of Hard-Coded Dependencies
  - Which makes it possible to change them, at rum-time or compile-time.
- Built-In Support for Dependency Injection in ASP.NET Core
  - ASP.NET Core applications can leverage built in framework support for implementing dependency injection

Dependency Injection is the process of “injecting” the “dependency” whenever the dependency is requested by a client, where the dependency may be another service which the client (requester) may have no means of knowing how to create.

As an analogy, imagine a person (client) going to office carrying his lunch cooked by himself. In this scenario, the person has a “dependency” on food. But he had to know how to cook his food. But honestly, not everyone (client) knows to cook, but people do need food (dependency). This is where restaurants play the role of dependency Injection. They can supply food (“inject dependency”) to the people (client) without the person (client) needing to know how to cook.

ASP.NET core applications can leverage built in framework support for implementing dependency injection.

.NET Framework 4.6.1 – New Features

- Cryptography: Support for X509 certificates containing ECDSA
- ADO.NET Improvement
- WPF Improvement
- Profiling
- (NGEN) PDBs

The .NET Framework 4.6 added RSACng support for X509 certificates. The .NET Framework 4.6.1 adds support for ECDSA (Elliptic Curve Digital Signature Algorithm) X509 certificates.

Always Encrypted support for hardware protected keys ADO.NET now supports storing Always Encrypted column master keys natively in Hardware Security Modules (HSMs)

In WPF, Improved performance The delay in firing touch events has been fixed in the .NET Framework 4.6.1. Also typing in a **RichTextBox** control no longer ties up the render thread during fast input. Spell checking improvements.

The unmanaged Profiling API has been enhanced.

Cross-machine event tracing allows customers to profile a program on Machine A and look at the profiling data with source line mapping on Machine B

.NET Framework 4.6.2 – New Features

- ASP.NET Area Features
  - Improved support for localized error messages in data annotation validators
  - Async support for session-state store providers
  - Async support for output-cache providers
- Character Encoding - Unicode Standard, Version 8.0.0.
- Cryptography - Support for X509 certificates containing FIPS 186-3 DSA
- ADO.NET SQLClient - Connection pooling and timeouts with Azure SQL databases

Data annotation validators enable you to perform validation by adding one or more attributes to a class property. The attribute's **ValidationAttribute.ErrorMessage** element defines the text of the error message if validation fails. Starting with the .NET Framework 4.6.2, ASP.NET makes it easy to localize error message.

ASP.NET now allows task-returning methods to be used with session-state store providers, thereby allowing ASP.NET apps to get the scalability benefits of async.

Starting with the .NET Framework 4.6.2, task-returning methods can be used with output-cache providers to provide the scalability benefits of async

Characters in the .NET Framework 4.6.2 are classified based on the **Unicode Standard, Version 8.0.0**. In .NET Framework 4.6 and .NET Framework 4.6.1, characters were classified based on Unicode 6.3 character categories.

The .NET Framework 4.6.2 adds support for DSA (Digital Signature Algorithm) X509 certificates whose keys exceed the FIPS 186-2 1024-bit limit.

NET Framework Data Provider for SQL Server **(System.Data.SqlClient)** includes Connection pooling and timeouts with Azure SQL databases.

.NET Framework 4.6.2 – New Features (Continued)

- Windows Communication Foundation
  - WCF transport security support for certificates stored using CNG
  - Better support for multiple daylight saving time adjustment rules by the DataContractJsonSerializer class
  - Support for preserving a UTC time when serializing and deserializing with the XMLSerializer class
  - NetNamedPipeBinding best match
  - SSL 3.0 is not a default protocol

WCF transport security supports certificates stored using the Windows cryptography library (CNG).

Customers can use an application configuration setting to determine whether the **DataContractJsonSerializer** class supports multiple adjustment rules for a single time zone.

You can use an application configuration setting to determine whether the **XmlSerializer** preserves UTC time zone information when serializing and deserializing **DateTime** values.

WCF has a new app setting that can be set on client applications to ensure they always connect to the service listening on the URI that best matches the one that they request.

When using NetTcp with transport security and a credential type of certificate, SSL 3.0 is no longer a default protocol used for negotiating a secure connection.

.NET Framework 4.6.2 – New Features (Continued)

- Windows Presentation Framework
  - Soft keyboard support
  - Group sorting
  - Per-monitor DPI
- ClickOnce improvement - Support TLS 1.1 & TLS 1.2
- Converting Windows Forms and WPF apps to UWP apps

Soft Keyboard support enables focus tracking in a WPF applications by automatically invoking and dismissing the new Soft Keyboard in Windows 10 when the touch input is received by a control that can take textual input.

An application that uses a **CollectionView** object to group data can now explicitly declare how to sort the groups.

To support the recent proliferation of high-DPI and hybrid-DPI environments for WPF apps, WPF in the .NET Framework 4.6.2 enables per-monitor awareness.

ClickOnce has been updated to support TLS 1.1 and TLS 1.2 in addition to the 1.0 protocol, which it already supports. ClickOnce automatically detects which protocol is required; no extra steps within the ClickOnce application are required to enable TLS 1.1 and 1.2 support.

Windows now offers capabilities to bring existing Windows desktop apps, including WPF and Windows Forms apps, to the Universal Windows Platform (UWP). This technology acts as a bridge by enabling you to gradually migrate your existing code

base to UWP, thereby bringing your app to all Windows 10 devices.

.NET CORE

- NET Core is a general purpose development platform maintained by Microsoft and the .NET community.

- It is cross-platform, supporting Windows, macOS and Linux, and can be used in device, cloud, and embedded/IoT scenarios.

Following are the Features :-
Flexible deployment: Can be included in your app or installed side-by-side user- or machine-wide.
Cross-platform: Runs on Windows, macOS and Linux; can be ported to other operating systems.
Command-line tools: All product scenarios can be exercised at the command-line.
Compatible: .NET Core is compatible with .NET Framework, Xamarin and Mono, via the .NET Standard
Open source: The .NET Core platform is open source, using MIT and Apache 2 licenses.

.NET Core is composed of:-

- A .NET Runtime which provides a type system, assembly loading, a garbage collector, native interop and other basic services.
- The 'dotnet' app host, which is used to launch .NET Core apps. It selects the runtime and hosts the runtime, provides an assembly loading policy and launches the app. The same host is also used to launch SDK tools in much the same way.
- A set of framework libraries, which provide primitive data types, app composition types and fundamental utilities.

Difference in .NET Framework & .NET CORE

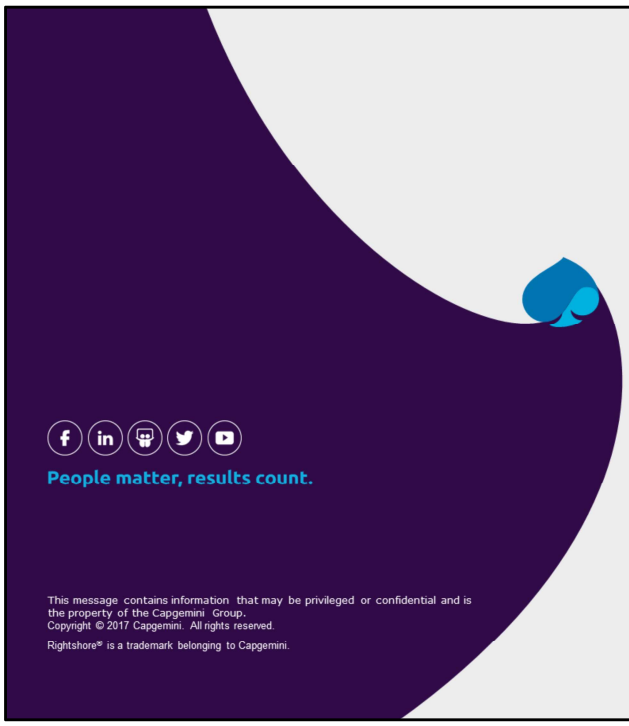
- App-Models
- API
- SubSystems
- Platforms
- Open Source

The major differences between .NET Core and the .NET Framework:
App-models -- .NET Core does not support all the .NET Framework app-models, in part because many of them are built on Windows technologies, such as WPF (built on top of DirectX). The console and ASP.NET Core app-models are supported by both .NET Core and .NET Framework.
APIs -- .NET Core contains many of the same, but fewer, APIs as the .NET Framework, and with a different factoring (assembly names are different; type shape differs in key cases). These differences currently typically require changes to port source to .NET Core. .NET Core implements the .NET Standard API, which will grow to include more of the .NET Framework BCL API over time.
Subsystems -- .NET Core implements a subset of the subsystems in the .NET Framework, with the goal of a simpler implementation and programming model. For example, Code Access Security (CAS) is not supported, while reflection is supported.
Platforms -- The .NET Framework supports Windows and Windows Server while .NET Core also supports macOS and Linux.
Open Source -- .NET Core is open source, while a read only subset of .NET Framework is open source.

Notes Pages

Use .NET Core for your server application when:
You have cross-platform needs.
You are targeting microservices.
You are using Docker containers.
You need high-performance and scalable systems.
You need side-by-side .NET versions per application.

Use .NET Framework for your server application when:
Your app currently uses .NET Framework
Your app uses third-party .NET libraries or NuGet packages not available for .NET Core.
Your app uses .NET technologies that aren't available for .NET Core.
Your app uses a platform that doesn’t support .NET Core.



People matter, results count.

About Capgemini

With more than 190,000 people, Capgemini is present in over 40 countries and celebrates its 50th Anniversary year in 2017. A global leader in consulting, technology and outsourcing services, the Group reported 2016 global revenues of EUR 12.5 billion. Together with its clients, Capgemini creates and delivers business, technology and digital solutions that fit their needs, enabling them to achieve innovation and competitiveness. A deeply multicultural organization, Capgemini has developed its own way of working, the Collaborative Business Experience™, and draws on Rightshore®, its worldwide delivery model.

Learn more about us at

www.capgemini.com