devone

.NET Core

and .NET Standard

Rainer Stropek

software architects gmbh

Web http://www.timecockpit.com
Mail rainer@timecockpit.com
Twitter @rstropek

time cockpit
Saves the day.

Abstract

In the last years, Microsoft has radically changed its .NET platform. Rewrite of the compiler, switch to open source, making it real cross-platform, harmonize various .NET flavors into one .NET Standard library – .NET Core had been a long and partly bumpy journey for us developers but with the launch of Visual Studio 2017, the entire .NET Core stack has become RTM. In his session, Rainer Stropek, long-time Microsoft MVP and MS Regional Director, speaks about the current state of .NET Core.

- Where is Microsoft in its long-term road map?
- Which tools and platforms are available?
- What about the upcoming big release 2.0 of .NET Core and .NET Standard?

Rainer will start his session with a discussion of questions like this. As usual, Rainer will not just show slides but also demonstrate many samples live on stage. Rainer will close his session with performance- and diagnostics-related topics. How does the .NET Core perform? What about cross-platform profiling and debugging? Rainer assumes that you have basic .NET knowledge. You do not need in-depth knowledge or hands-on experience of .NET Core to benefit from this session.

Why .NET Core?

Refactor .NET Framework

Establish a Standard Library for the various incarnations of .NET
.NET Core is not 100% compatible with .NET 4.x (details)

Make it a real cross-platform solution

Windows, Mac OS, Linux (details in .NET Core Roadmap)

Make it open source

A .NET Foundation project
MIT License

Details: https://docs.microsoft.com/dotnet/

Status of .NET Core

.NET Core 1.1 is RTM

Download current version

2.0 is scheduled for Summer 2017 (roadmap, overview in docs)

.NET CLI 1.0 is RTM

Visual Studio 2017 is RTM

Get VS2017 Preview + .NET Core 2.0 Preview 1 to play with .NET Core 2.0

C# is RTM

VB and F# are coming

See also: https://github.com/dotnet/core/blob/master/roadmap.md

What can you build with Core?

Console applications

ASP.NET Core applications

UWP applications

Xamarin applications

No legacy frameworks like WinForms, WPF, etc.

See also: https://github.com/dotnet/core/blob/master/roadmap.md

Where to get .NET Core?

.NET Core landing page

With Visual Studio tools (Visual Studio prerequisites)
Command-line tools (with your own editor, e.g. VSCode, download)

.NET Install Script (details, download)

You have to care for the prerequisites

NuGet

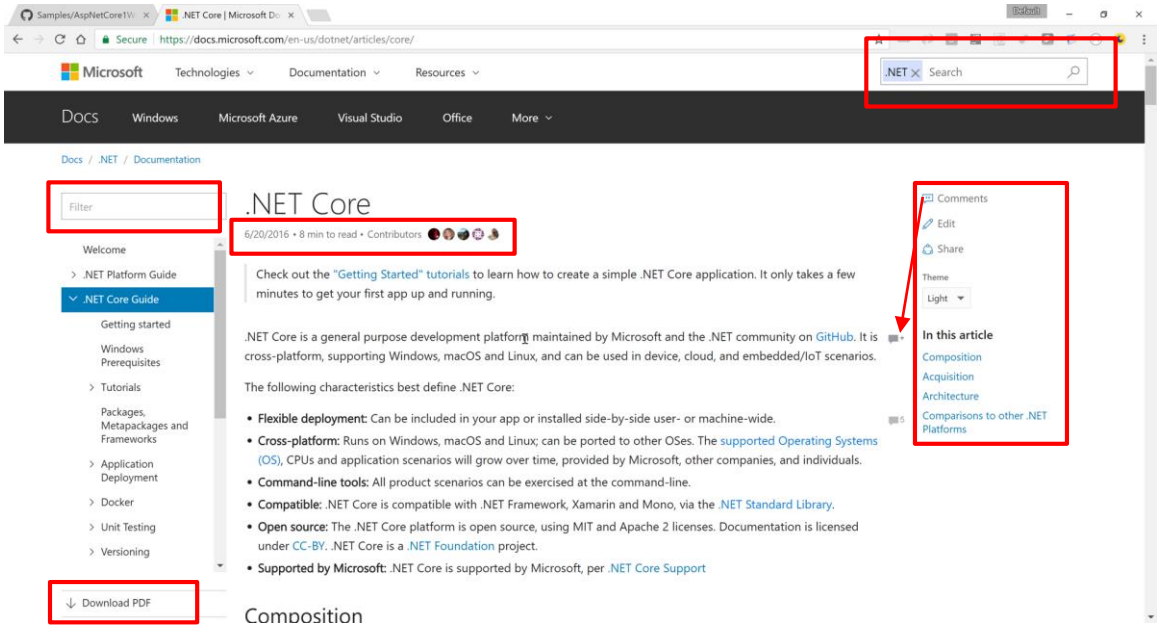
Packages and Metapackages

Docker: `microsoft/dotnet` image (details)

.NET Core Source Browser

See also: https://github.com/dotnet/core/blob/master/roadmap.md

Getting Help



New https://docs.microsoft.com

Demo

Intro Sample

Create console app
- CLI
- Visual Studio

Analyze *.csproj*
- Switch target frameworks

Run it
- Windows
- Linux

Demo

VS Docker Support

Create ASP.NET App

Add Docker Support

Show Debugging

- .NET Core in Linux Docker Container

Demo

Linux Debugging

Remote debugging
- ASP.NET Core
  - Client: Visual Studio
  - Server: Ubuntu

.NET Standard Library

Why a standard library?

CLR (CLI) has already been standardized (ECMA 334)

- No standardized BCL prior to .NET Core

Goal: Standard BCL API for all .NET platforms

- Easier to create portable libraries
- Reduce conditional compilation

What about PCLs?

- Well defined API instead of just intersection of platforms
- Better versioning
- Overlapping PCL profiles (details)

.NET Standard	1.0	1.1	1.2	1.3	1.4	1.5	1.6	2.0
.NET Core	1.0	1.0	1.0	1.0	1.0	1.0	1.0	2.0
.NET Framework (with tooling 1.0)	4.5	4.5	4.5.1	4.6	4.6.1	4.6.2		
.NET Framework (with tooling 2.0)	4.5	4.5	4.5.1	4.6	4.6.1	4.6.1	4.6.1	4.6.1
Mono	4.6	4.6	4.6	4.6	4.6	4.6	4.6	vNext
Xamarin.iOS	10.0	10.0	10.0	10.0	10.0	10.0	10.0	vNext
Xamarin.Android	7.0	7.0	7.0	7.0	7.0	7.0	7.0	vNext
Universal Windows Platform	10.0	10.0	10.0	10.0	10.0	vNext	vNext	vNext
Windows	8.0	8.0	8.1					
Windows Phone	8.1	8.1	8.1					
Windows Phone Silverlight	8.0							

Details: https://docs.microsoft.com/en-us/dotnet/articles/standard/library

Versioning

Framework version changes when APIs are added

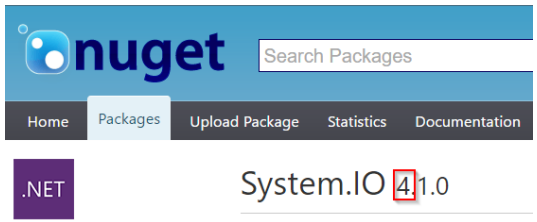
No implementation → no patch numbers

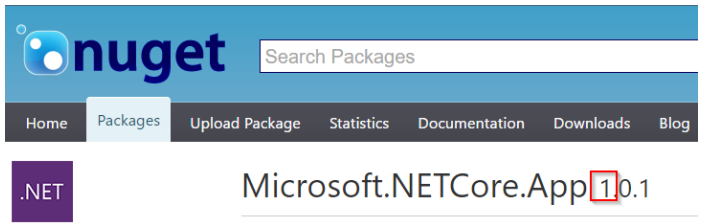
Example: **`netcoreapp1.0`**

Package versions

`System.*` packages (e.g. System.IO) use 4.x numbers (overlap with .NET Framework)

Packages without overlapping with .NET Framework → 1.x (e.g. Microsoft.NETCore.App)





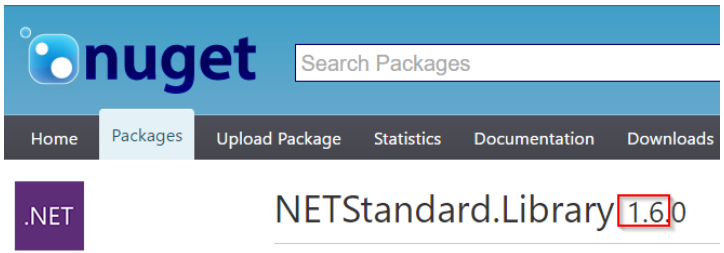
https://docs.microsoft.com/en-us/dotnet/articles/core/versions/index

Versioning

.NET Standard Library

Versioning independent of any .NET runtime, applicable to multiple runtimes

1.6 for .NET Core 1.0



Examples

https://docs.microsoft.com/en-us/dotnet/articles/core/versions/index

Demo

.NET Standard

Create .NET Standard library

Reference from .NET Core

Reference from Full FX

View assembly redirects

.NET Core 2.0

What's New?

.NET Standard 2.0 support

Supported on more platforms

~20k more APIs than .NET Standard 1.6 → easier to migrate 4.6.1 code

References to .NET Framework supported

Reuse existing libraries without recompile

Supported in VS2017 Preview 15.3

See also https://blogs.msdn.microsoft.com/dotnet/2017/05/10/announcing-net-core-2-0-preview-1/

Demo

.NET Core 2.0

FullFX reference in .NET Core 2.0
- Windows
- Linux

ApiPort tooling

Summary

State of the Union

C# and .NET are popular
- In the top 10 of stackoverflow's most-used and most-wanted 2017

.NET Core is the future of .NET

.NET Core is RTM → ready for production workload
- Migrating existing .NET Framework code is sometimes hard (tip: Use *ApiPort*)

With .NET Standard/Core 2.0, migration becomes easier
- Will raise adoption

Platform coverage is growing
- Windows, more and more Linux distros, MacOS, Docker, etc.

State of the Union

ASP.NET Core shows good performance

ASP.NET Core Benchmarks

TechEmpower Web Framework Benchmarks (Round 14)

Tooling has become great with VS2017

External tools like *dnSpy* or *PerfView* just work

Still rough on Linux in areas like Performance Tracing

.NET Core is ready for prime-time

devone

Q&A

Thank your for coming!

Rainer Stropek

software architects gmbh

Mail rainer@timecockpit.com
Web http://www.timecockpit.com
Twitter @rstropek

time cockpit
Saves the day.