Visual Studio LIVE!
EXPERT SOLUTIONS FOR .NET DEVELOPERS

WASHINGTON, DC
ROCK YOUR CODE TOUR · 2017

Getting to the Core of .NET Core

Adam Tuliper
Principal Software Engineer
Microsoft

Level: Introduction

About this guy

Adam Tuliper
Principal Software Engineer @ Microsoft

20+ year software architect

- Cloud, gaming & mixed reality, web, bots
- Orange County Unity Meetup
- Security background
  - ASM should be taught to everyone ☺

adamt@microsoft.com | @AdamTuliper
adamtuliper.com

What is .NET Core?

- A general purpose, modular, cross-platform and open source implementation of .NET
- Comprised of CoreFx & CoreCLR as major components

Visual Studio LIVE!

What is .NET (Full) Framework?

- Placeholder
- System-wide install
- JIT/etc

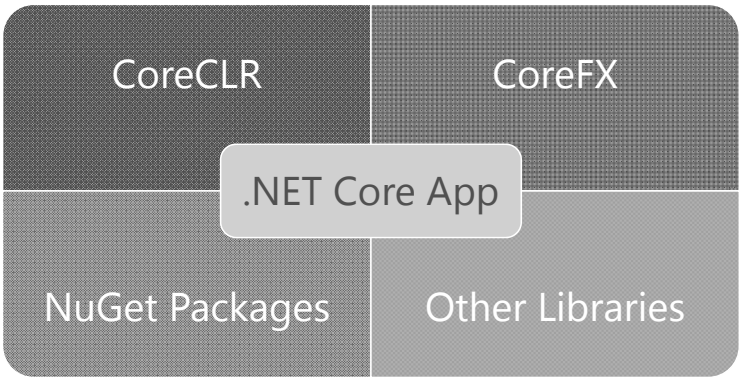
Visual Studio LIVE!

Why yet another?

- Supporting more platforms on .NET is challenging
- Speed
  - https://www.techempower.com/benchmarks/
  - 8x+ faster than Node, 3x faster than Go
- Side by side (SxS Deployment)
- Cross platform
- CLI First
  - Ensure Visual Studio uses command line tools same x-plat

Visual Studio LIVE!

The Makeup



Visual Studio LIVE!

What's in CoreCLR?

- Runtime
- Base Library (base classes & data types)
- Garbage Collector
- JIT Compiler

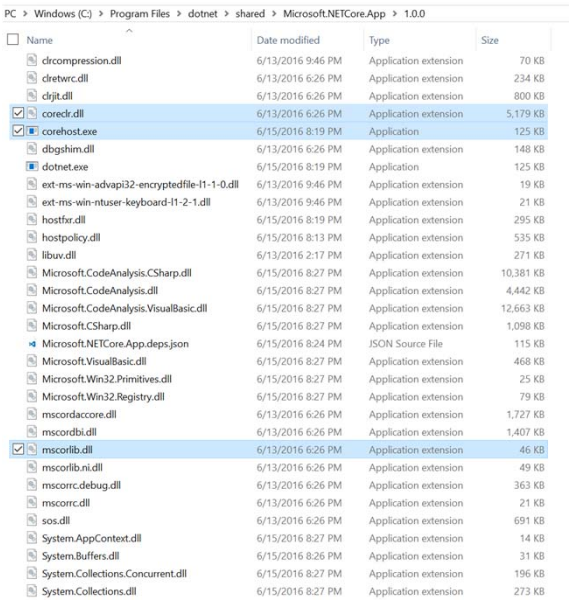
Visual Studio LIVE!

Specifically – CoreCLR on Windows

- dotnet.exe
  - The command line host.
  - Loads & starts the CoreCLR runtime
  - Passes the managed program you want to run to it.
- coreclr.dll
  - The CoreCLR runtime
- mscorlib.dll
  - The core managed library for CoreCLR, which contains all of the fundamental data types and functionality

Visual Studio LIVE!

CoreCLR on Windows



PC > Windows (C:) > Program Files > dotnet > shared > Microsoft.NETCore.App > 1.0.0

Name	Date modified	Type	Size
clrcompression.dll	6/13/2016 9:46 PM	Application extension	70 KB
clretwrc.dll	6/13/2016 6:26 PM	Application extension	234 KB
clrjit.dll	6/13/2016 6:26 PM	Application extension	800 KB
coreclr.dll	6/13/2016 6:26 PM	Application extension	5,179 KB
corehost.exe	6/15/2016 8:19 PM	Application	125 KB
dbgshim.dll	6/13/2016 6:26 PM	Application extension	148 KB
dotnet.exe	6/15/2016 8:19 PM	Application	125 KB
ext-ms-win-advapi32-encryptedfile-l1-1-0.dll	6/13/2016 9:46 PM	Application extension	19 KB
ext-ms-win-ntuser-keyboard-l1-2-1.dll	6/13/2016 9:46 PM	Application extension	21 KB
hostfxr.dll	6/15/2016 8:19 PM	Application extension	295 KB
hostpolicy.dll	6/15/2016 8:13 PM	Application extension	535 KB
libuv.dll	6/13/2016 2:17 PM	Application extension	271 KB
Microsoft.CodeAnalysis.CSharp.dll	6/15/2016 8:27 PM	Application extension	10,381 KB
Microsoft.CodeAnalysis.dll	6/15/2016 8:27 PM	Application extension	4,442 KB
Microsoft.CodeAnalysis.VisualBasic.dll	6/15/2016 8:27 PM	Application extension	12,663 KB
Microsoft.CSharp.dll	6/15/2016 8:27 PM	Application extension	1,098 KB
Microsoft.NETCore.App.deps.json	6/15/2016 8:24 PM	JSON Source File	115 KB
Microsoft.VisualBasic.dll	6/15/2016 8:27 PM	Application extension	468 KB
Microsoft.Win32.Primitives.dll	6/15/2016 8:27 PM	Application extension	25 KB
Microsoft.Win32.Registry.dll	6/15/2016 8:27 PM	Application extension	79 KB
mscordaccore.dll	6/13/2016 6:26 PM	Application extension	1,727 KB
mscordbi.dll	6/13/2016 6:26 PM	Application extension	1,407 KB
mscorlib.dll	6/13/2016 6:26 PM	Application extension	46 KB
mscorlib.ni.dll	6/13/2016 6:26 PM	Application extension	49 KB
mscorrc.debug.dll	6/13/2016 6:26 PM	Application extension	363 KB
mscorrc.dll	6/13/2016 6:26 PM	Application extension	21 KB
sos.dll	6/13/2016 6:26 PM	Application extension	691 KB
System.AppContext.dll	6/15/2016 8:27 PM	Application extension	14 KB
System.Buffers.dll	6/15/2016 8:26 PM	Application extension	31 KB
System.Collections.Concurrent.dll	6/15/2016 8:27 PM	Application extension	196 KB
System.Collections.dll	6/15/2016 8:27 PM	Application extension	273 KB

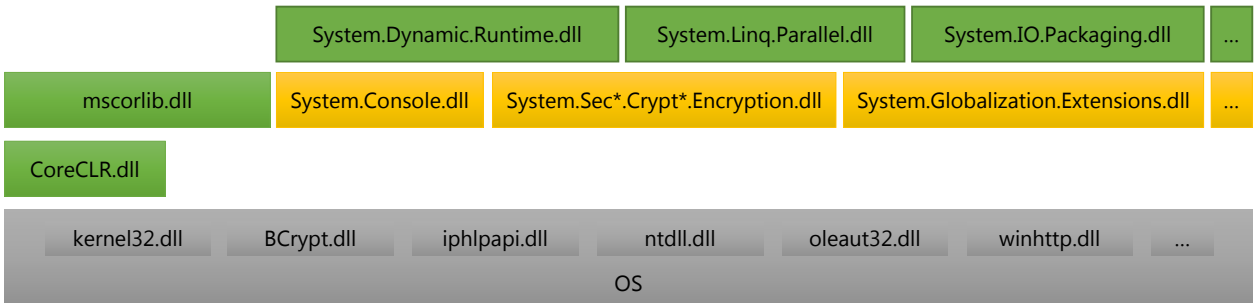
Visual Studio LIVE! EXPERT SOLUTIONS FOR .NET DEVELOPERS

Native Library Resolution

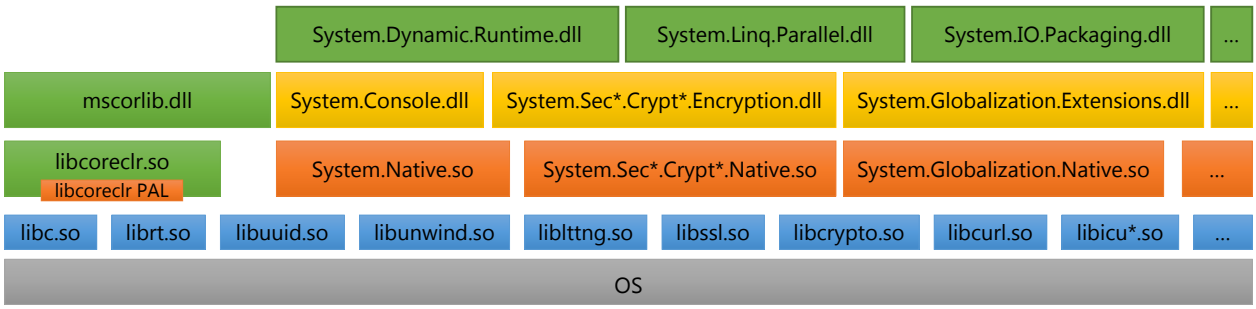
- Some features require native code
  - Ex. Security
- Windows
  - Pre-determined binaries installed with Windows
  - Allows knowledge of what a 'distribution' is
- Linux
  - No guarantee on existing libraries
  - May only fail when feature is accessed
- OSX
  - OpenSSL -> Apple Crypto Libs

Visual Studio LIVE! EXPERT SOLUTIONS FOR .NET DEVELOPERS

30K Foot View – .NET Core on Windows



30K Foot View – .NET Core on Unix



Three things to notice:

1. Different dependencies mean (partially) different managed implementations
2. Lots of dependencies in 3rd-party libraries rather than in OS/kernel
3. Native "shims" sitting between managed code & those libraries

libcoreclr PAL for libcoreclr/mscorlib
System.*Native for corefx

CoreFX Assemblies

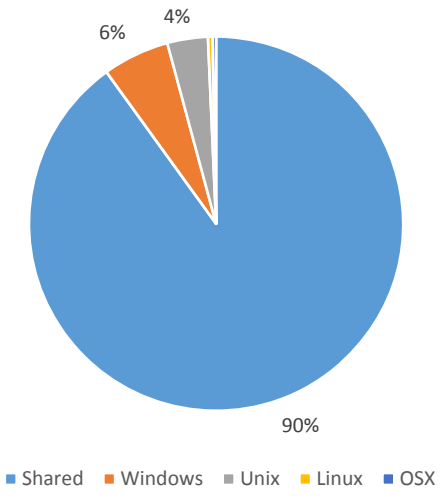
- ~135 assemblies
- 30 Full facades over mscorlib
  - System.Runtime.dll, System.Reflection.dll
- 60 Platform-Agnostic
  - No interaction with OS
    - System.Linq.dll, System..IO.Compression.dll
- 15 Windows Only
  - Microsoft.Win32.Registry.dll, System.Threading.Overlapped.dll, COM
- 30 Platform Specific
  - Separate Windows & Unix builds
    - System.Console.dll, System.IO.FileSystem.dll, System.Net.Http.dll

Visual Studio LIVE!

CoreFX – most shared code

~ Lines of C# Code in /src

90% Shared Cross Platform



Shared code, even in platform specific builds

~ .cs line count examples

System.Console. Shared: 1021, Win: 491, Unix: 1080
System.IO.FileSystem. Shared: 1854, Win: 2534, Unix: 1427
System.IO.Pipes. Shared: 943, Win: 661, Unix: 581
System.IO.MemoryMappedFiles. Shared: 453, Win: 411, Unix: 325
System.Net.Http. Shared: 8352, Windows: 2181, Unix: 2018
System.Net.Sockets. Shared: 6328, Win: 1953, Unix: 2005
System.Text.Encoding.CodePages. Shared: 9970, Win: 27, Unix: 7

Windows Install & Tools

Windows 7+, Server 2012+

Windows Nano Server

Visual Studio 2015 or command line (cli) tools

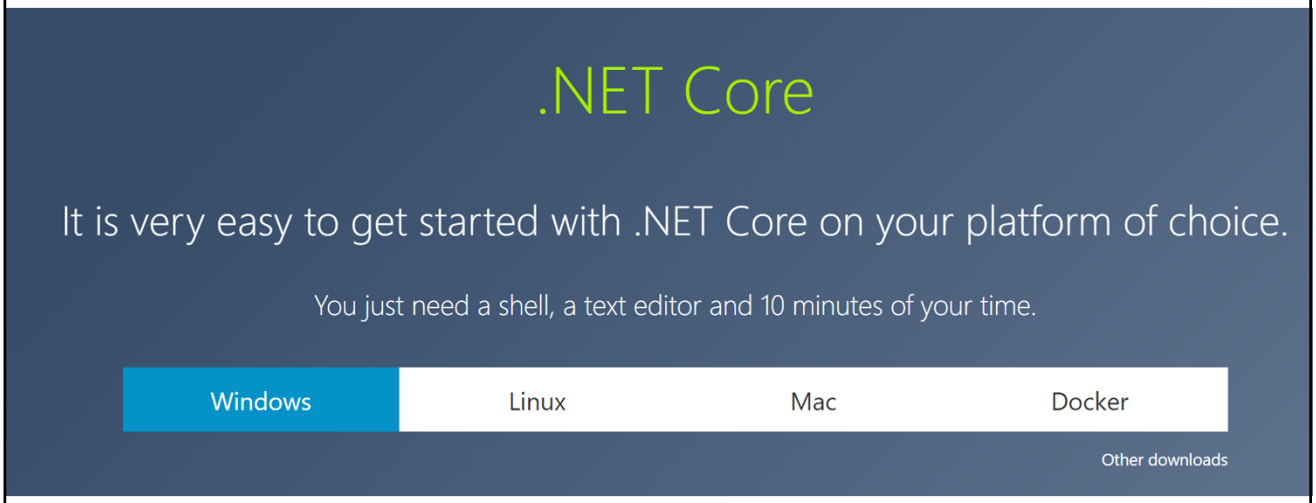
- Update 3 or higher

.NET Core requires VC++ Redistributable

- Installed with the installer
- If doing manually via powershell, manually install VC++

NuGet 3.5 or higher

Getting started is easy (SDK/CLI)

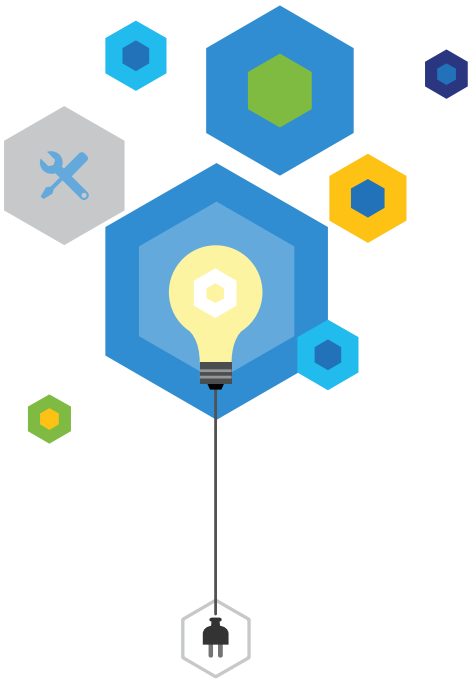


Visual Studio LIVE!

Creating New (or use VS<->CLI)

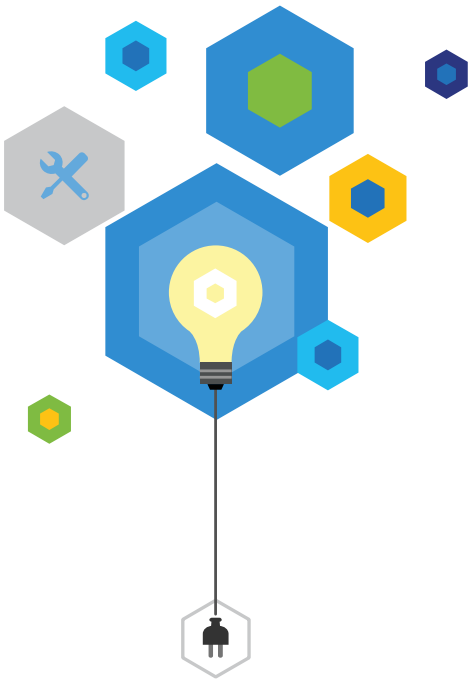
```
dotnet new
dotnet restore
dotnet run
OR
dotnet new
dotnet restore
dotnet build
dotnet <assembly>
```

Demo



Getting started with
.NET Core CLI

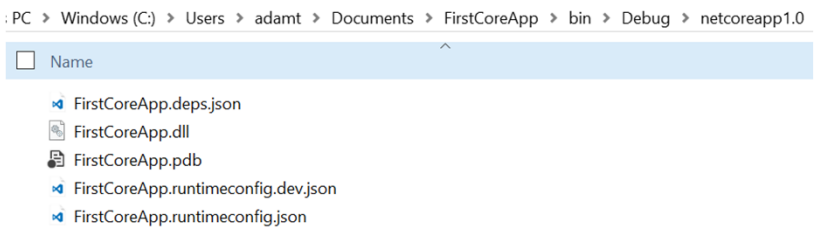
Demo



Linux – Same commands, same output

Results of dotnet build

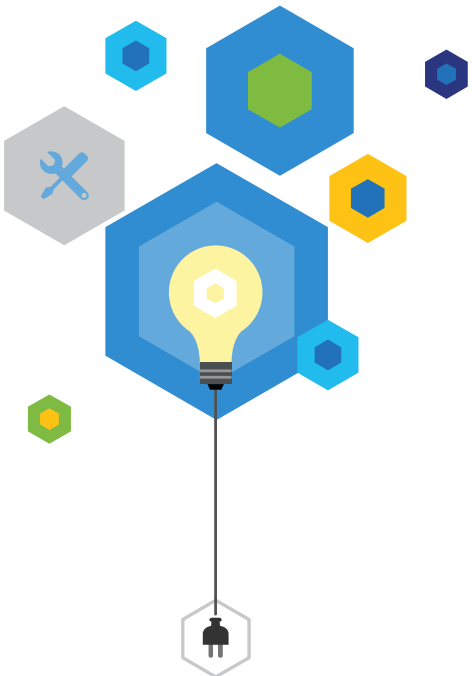
- deps.json
  - Tells the driver where to look for dependencies, package cache, etc. Don't edit.
- runtimeconfig.json
  - Additional runtime config options, gc specifics, don't roll forward version, etc.
  - Can be edited
  - ***Optional***



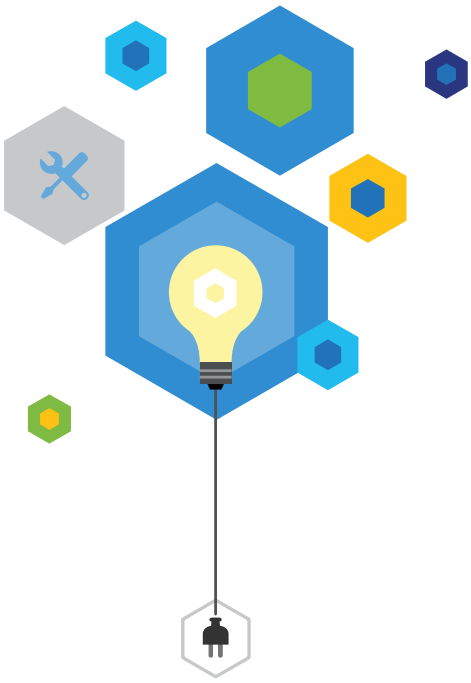
Visual Studio LIVE!

Demo

Peeking under the hood



Demo



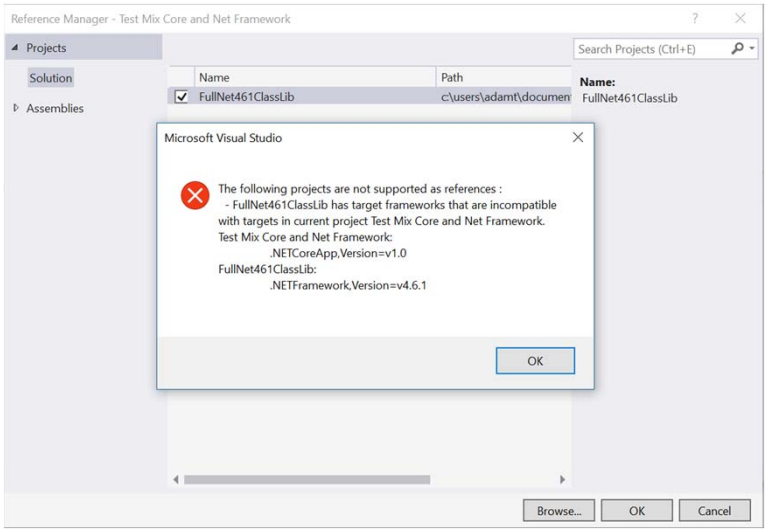
Really – what's the difference in my web app?

Versioning

LIBRARY COMPATIBILITY

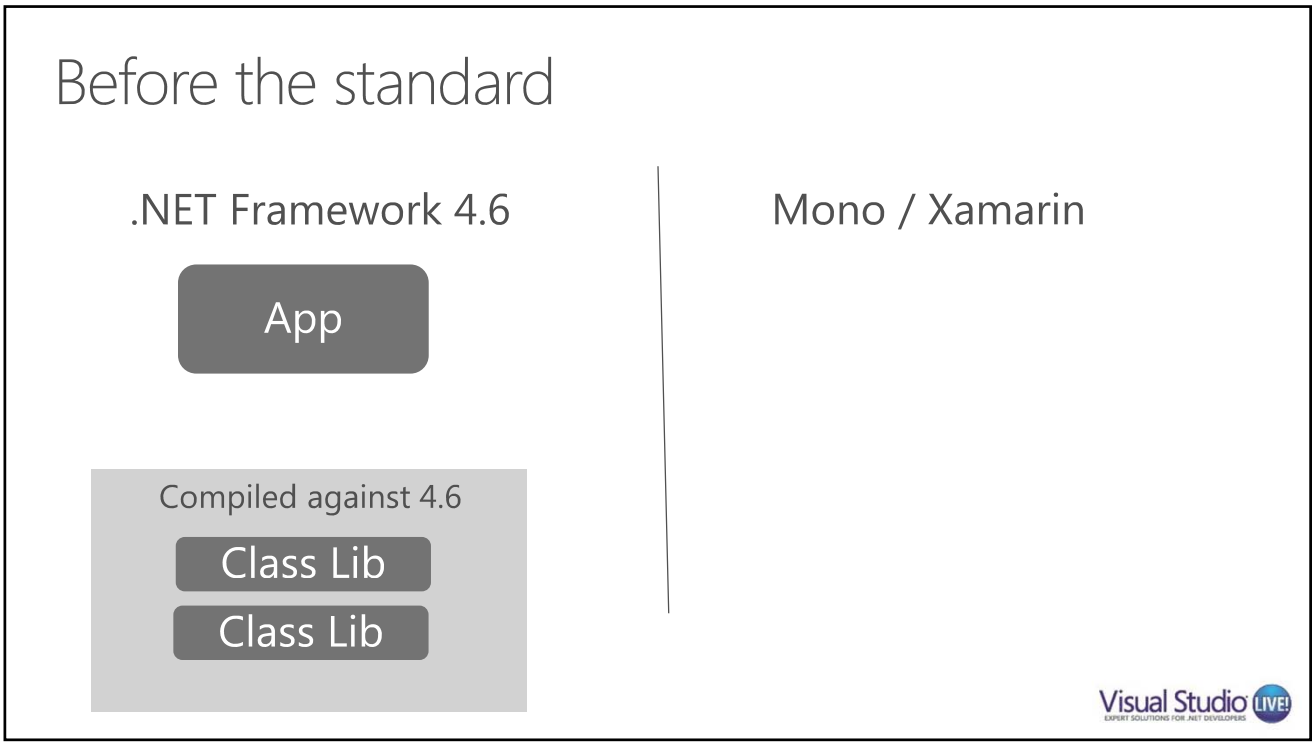
Compatibility

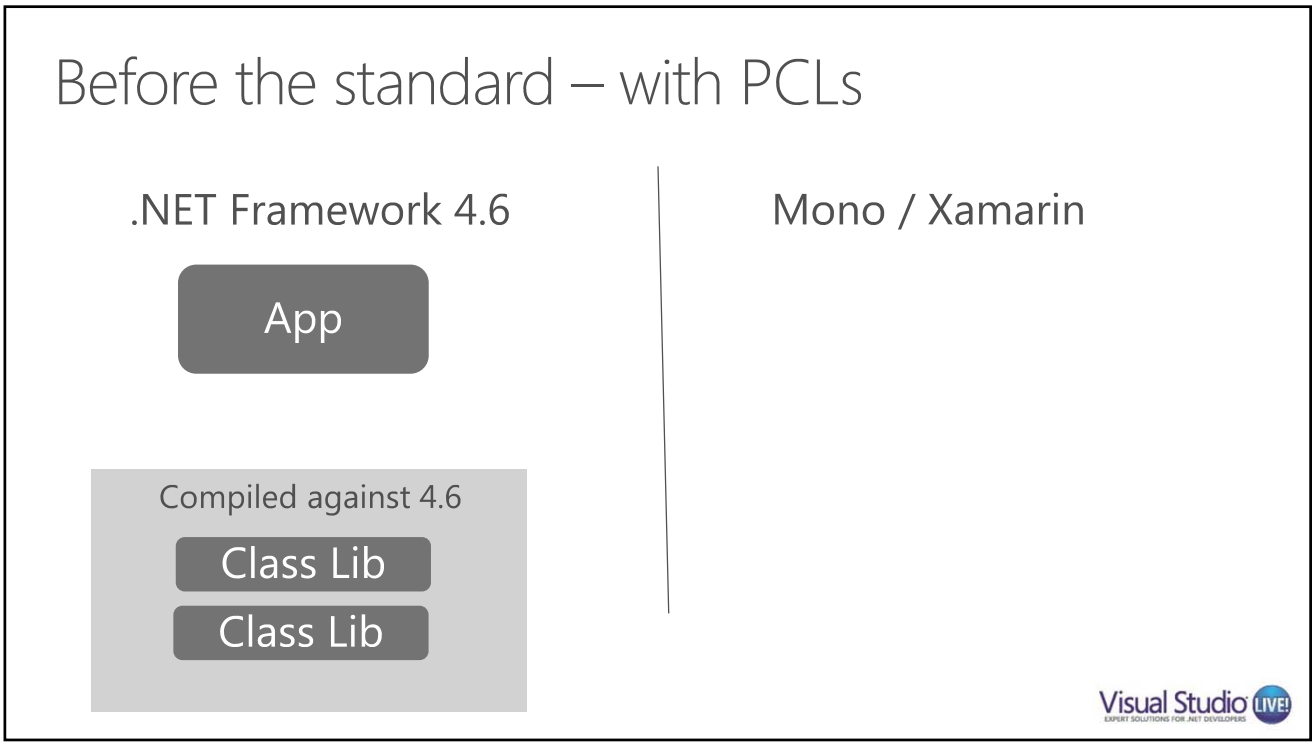
Don't mix?

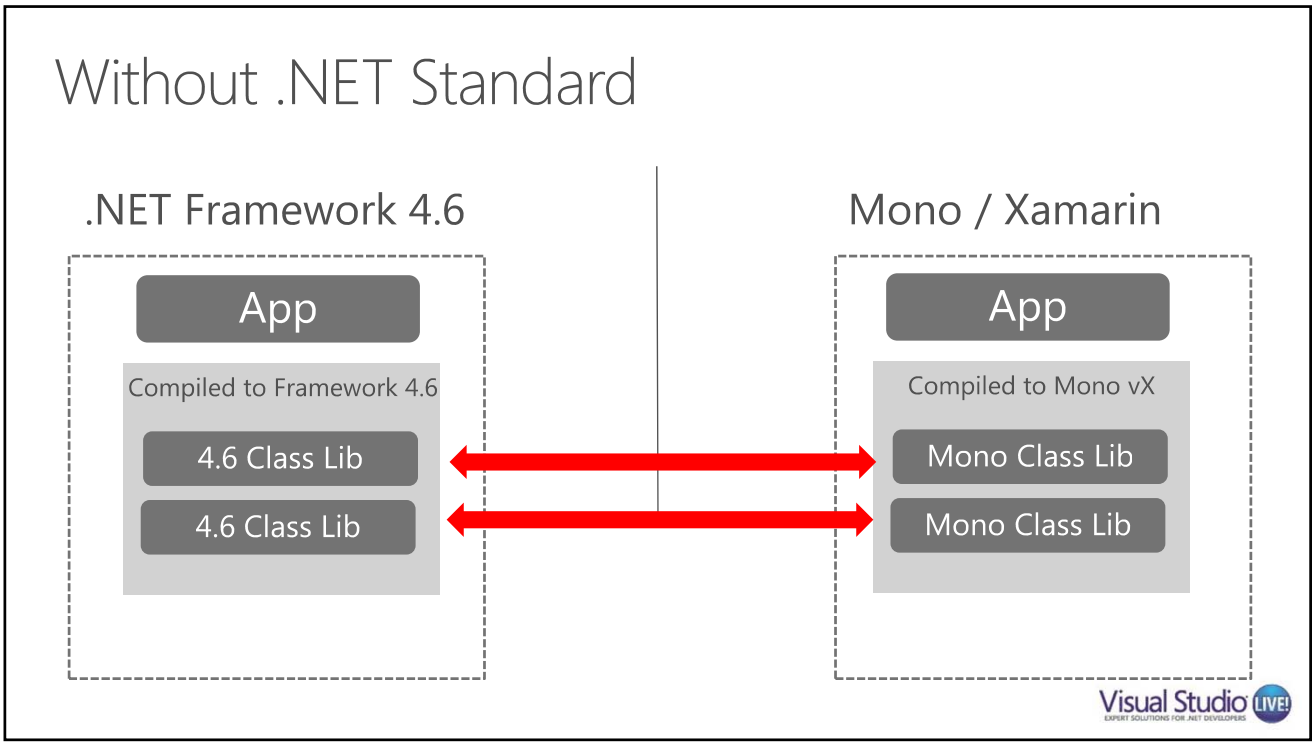


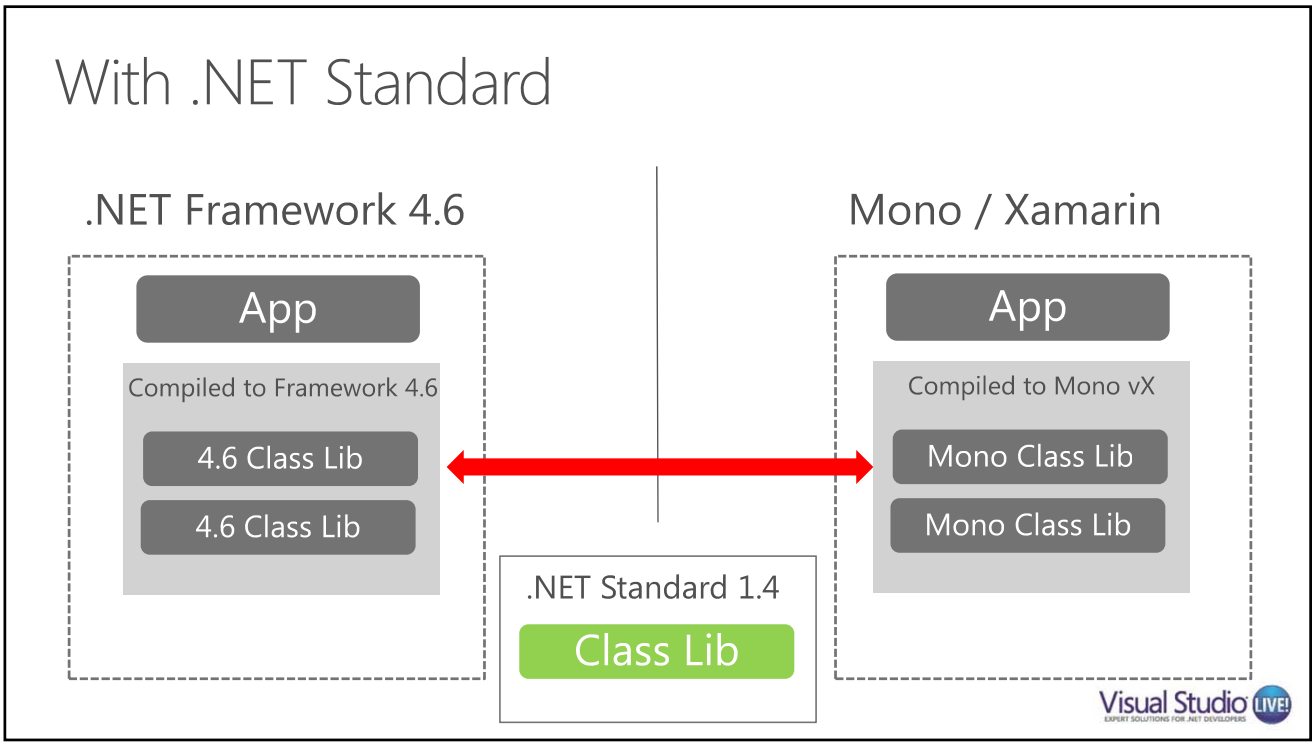
Visual Studio LIVE!

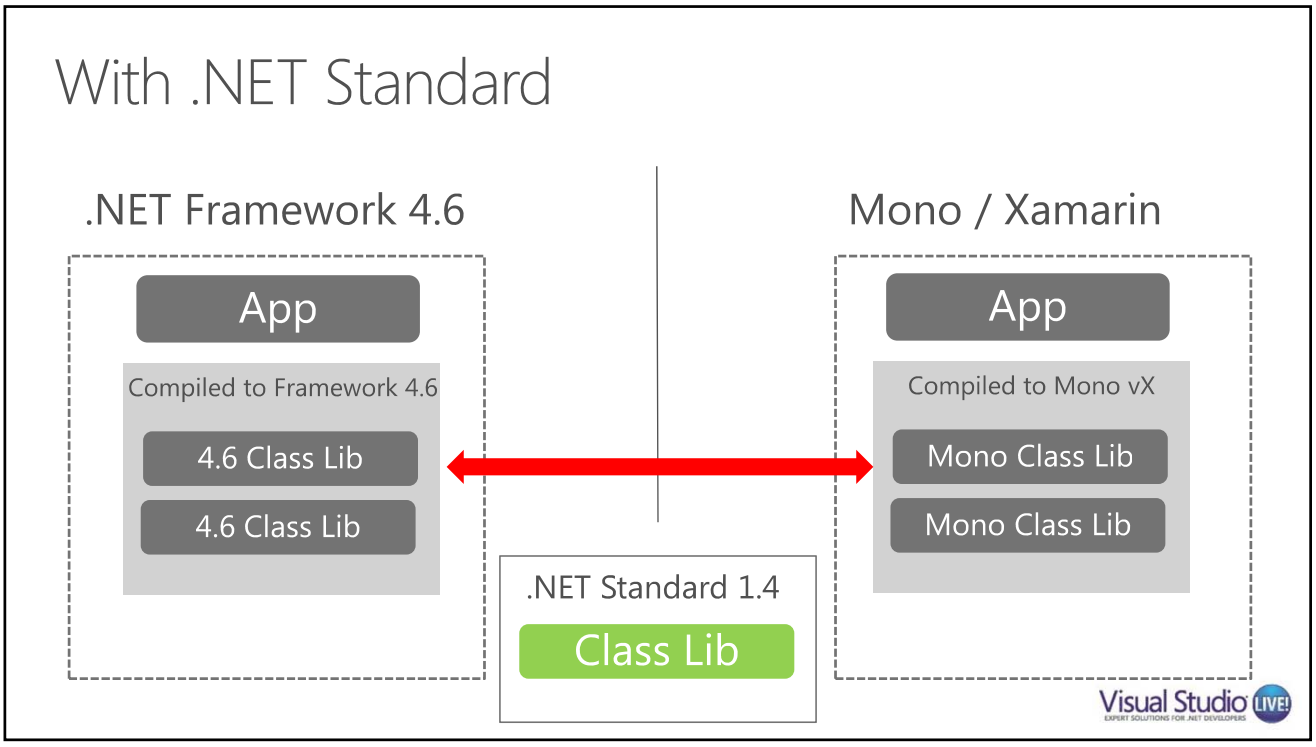
.NET Standard is a set of APIs that all .NET platforms have to implement.

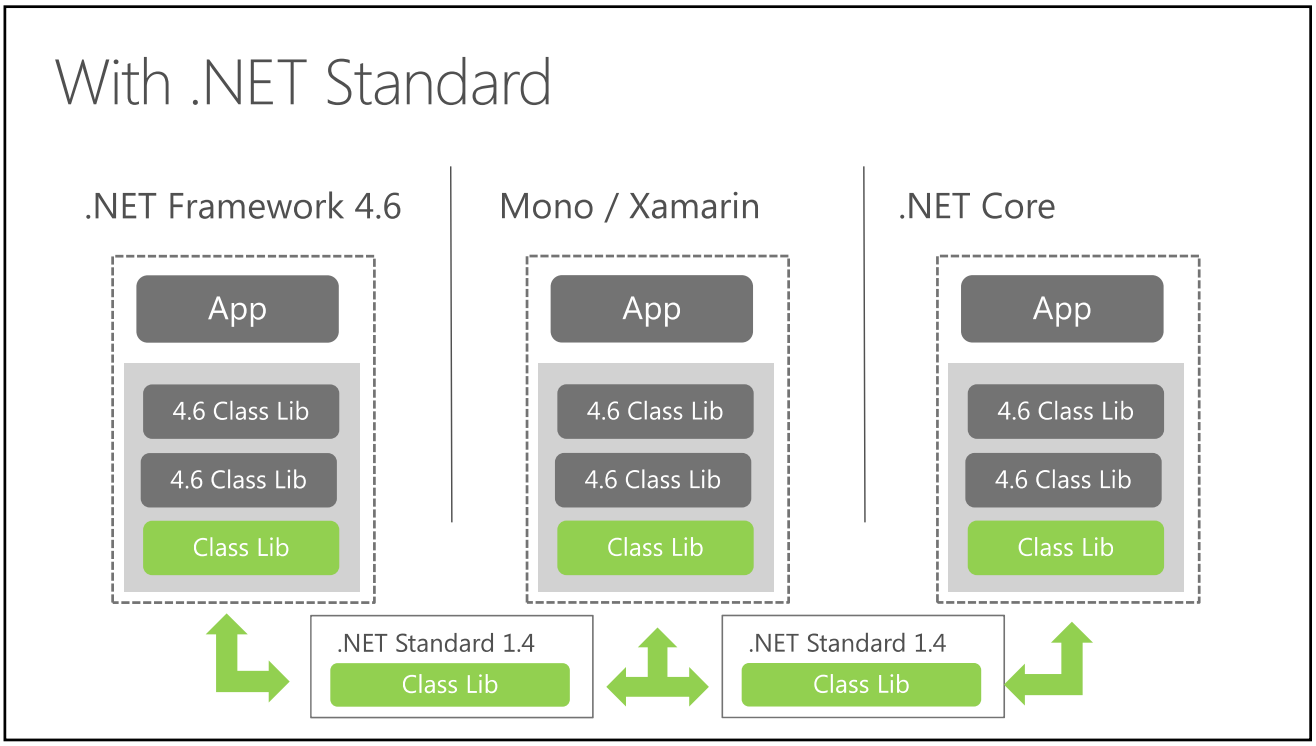

Before the standard
.NET Framework 4.6
Mono / Xamarin
App
Compiled against 4.6
Class Lib
Class Lib
Visual Studio LIVE!


Before the standard – with PCLs
.NET Framework 4.6
Mono / Xamarin
App
Compiled against 4.6
Class Lib
Class Lib
Visual Studio LIVE!


Without .NET Standard
.NET Framework 4.6
Mono / Xamarin
App
Compiled to Framework 4.6
4.6 Class Lib
4.6 Class Lib
App
Compiled to Mono vX
Mono Class Lib
Mono Class Lib
Visual Studio LIVE!


With .NET Standard
.NET Framework 4.6
Mono / Xamarin
App
Compiled to Framework 4.6
4.6 Class Lib
4.6 Class Lib
App
Compiled to Mono vX
Mono Class Lib
Mono Class Lib
.NET Standard 1.4
Class Lib
Visual Studio LIVE!


With .NET Standard
.NET Framework 4.6
Mono / Xamarin
App
Compiled to Framework 4.6
4.6 Class Lib
4.6 Class Lib
App
Compiled to Mono vX
Mono Class Lib
Mono Class Lib
.NET Standard 1.4
Class Lib
Visual Studio LIVE!


With .NET Standard
.NET Framework 4.6
Mono / Xamarin
.NET Core
App
4.6 Class Lib
4.6 Class Lib
Class Lib
App
4.6 Class Lib
4.6 Class Lib
Class Lib
App
4.6 Class Lib
4.6 Class Lib
Class Lib
.NET Standard 1.4
Class Lib
.NET Standard 1.4
Class Lib

The .NET Standard

.NET Platform	.NET Standard							
	1.0	**1.1**	**1.2**	**1.3**	**1.4**	**1.5**	**1.6**	**2.0**
.NET Core	→	→	→	→	→	→	1.0	vNext
.NET Framework	→	4.5	4.5.1	4.6	4.6.1	4.6.2	vNext	4.6.1
Xamarin.iOS	→	→	→	→	→	→	→	vNext
Xamarin.Android	→	→	→	→	→	→	→	vNext
Universal Windows Platform	→	→	→	→	10.0	→	→	vNext
Windows	→	8.0	8.1					
Windows Phone	→	→	8.1					
Windows Phone Silverlight	8.0							

Visual Studio LIVE!

Metapackage

There are two primary meta packages

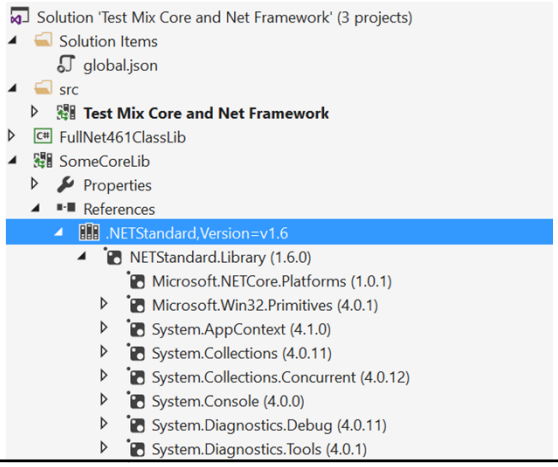
Simply a collection of other nuget packages

Visual Studio LIVE!

.NET Standard Library

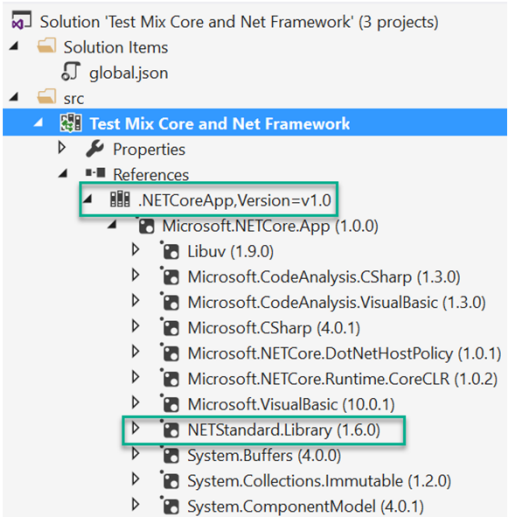
Contains a handful of packages (note icon)

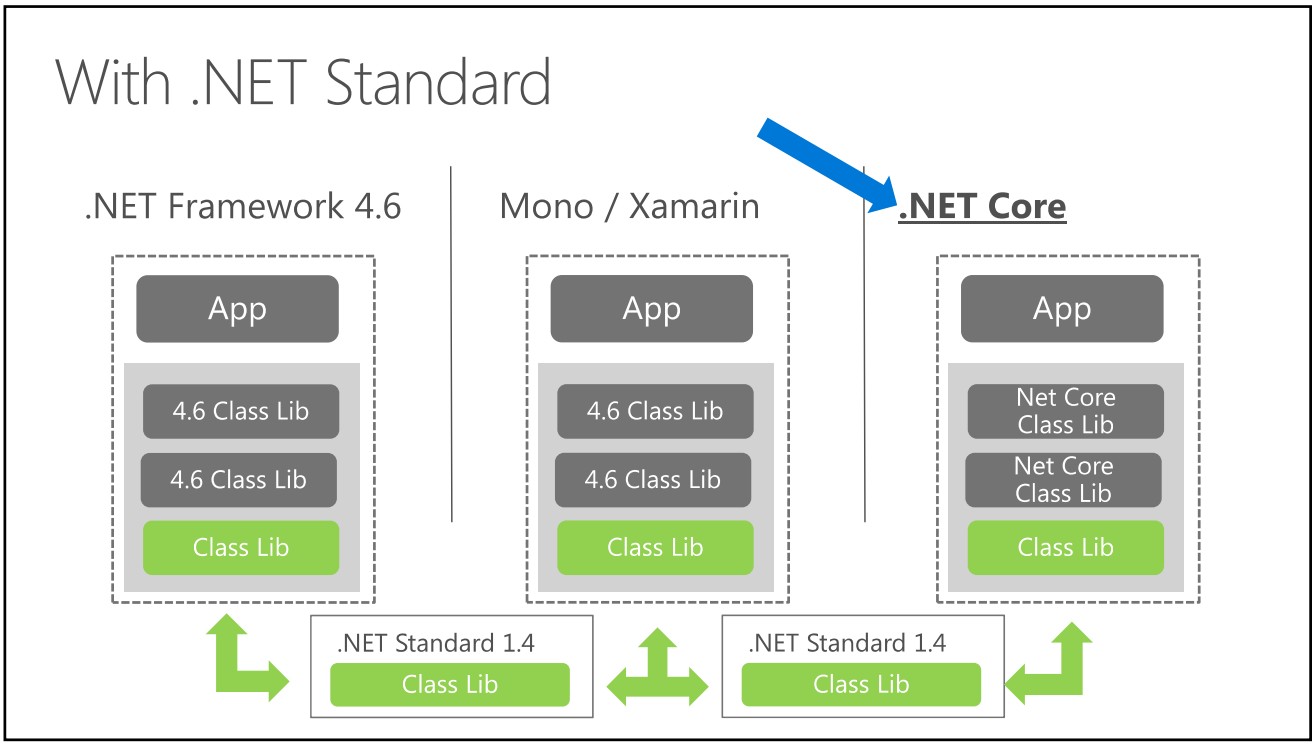
Typically used for libraries, not apps

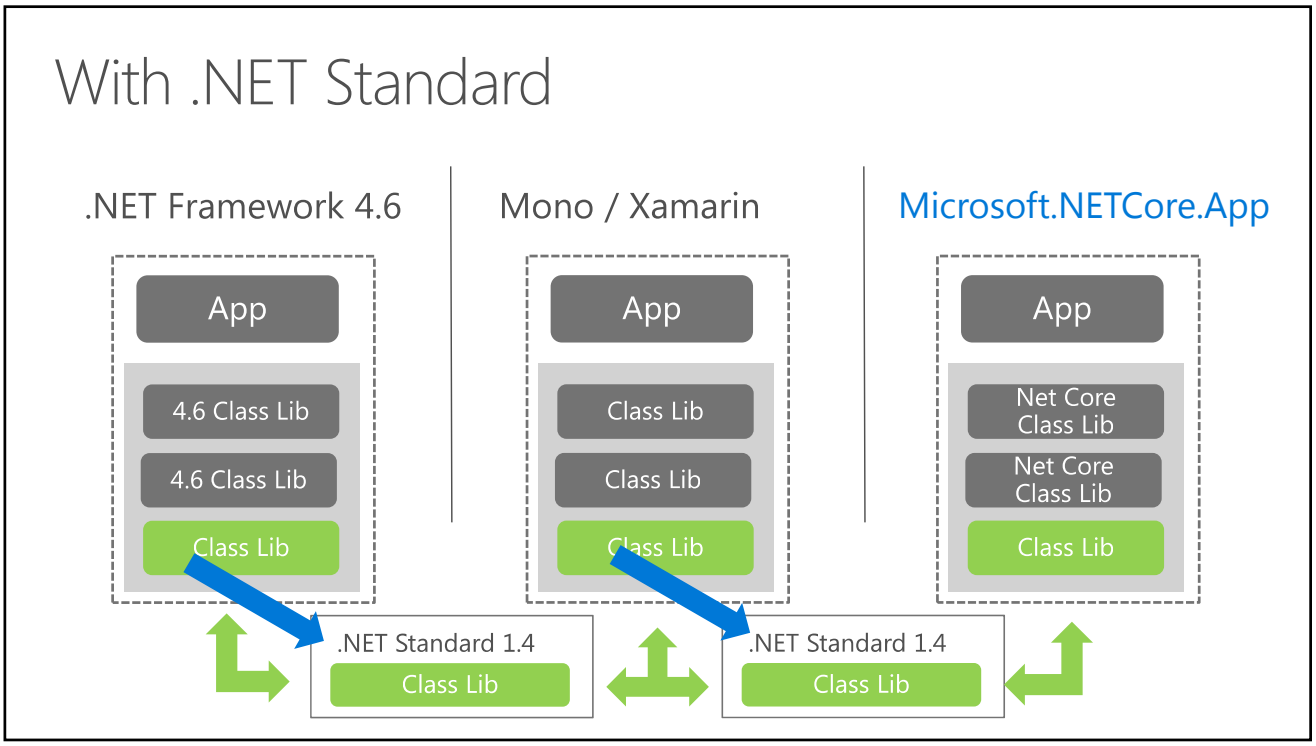


Microsoft.NETCore.App

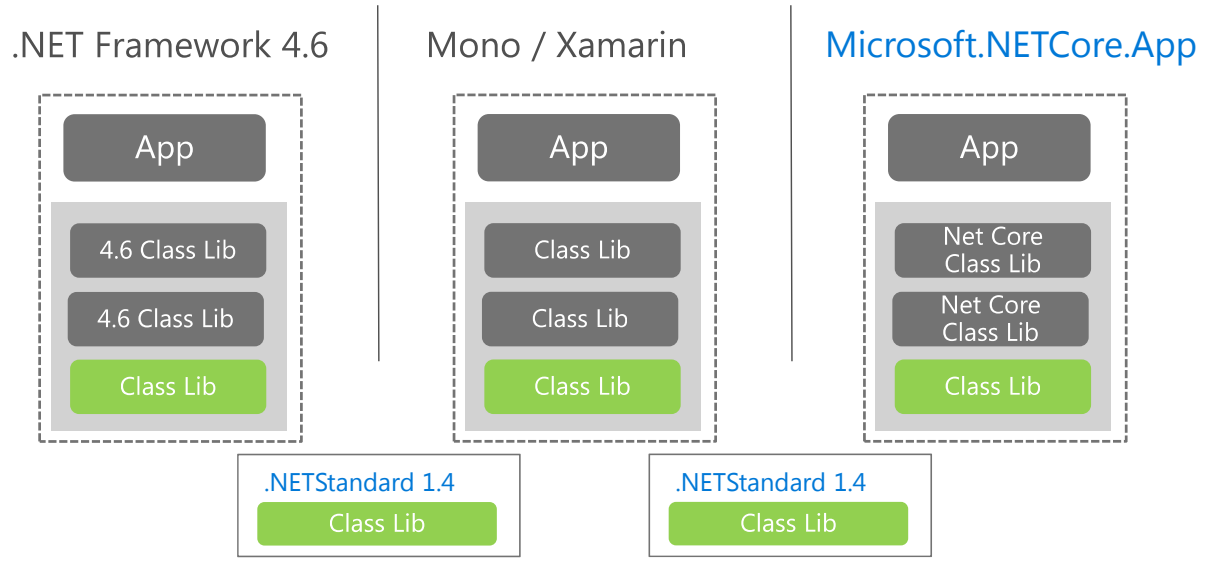
Contain .NET Standard 1.6 + more



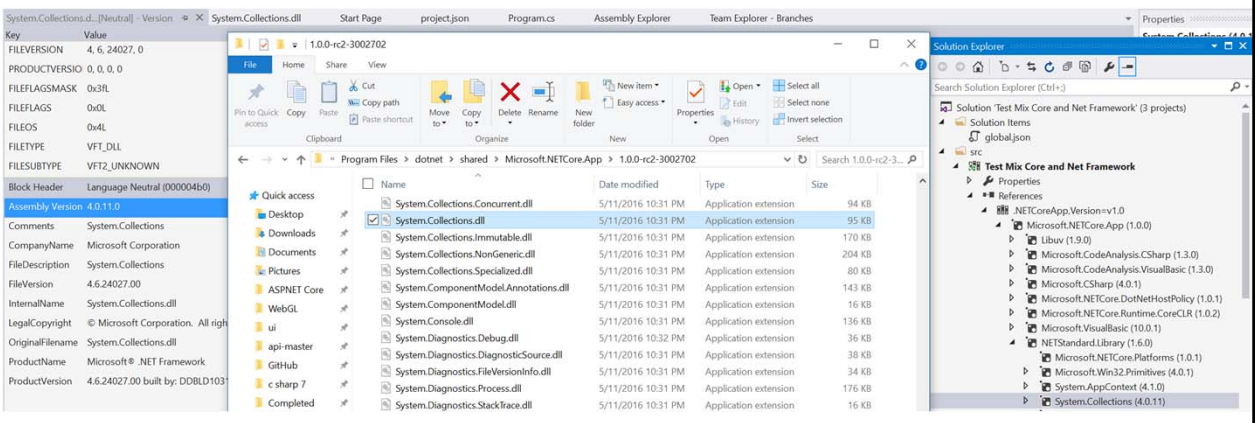

With .NET Standard
.NET Framework 4.6
Mono / Xamarin
.NET Core
App
4.6 Class Lib
4.6 Class Lib
Class Lib
App
4.6 Class Lib
4.6 Class Lib
Class Lib
App
Net Core Class Lib
Net Core Class Lib
Class Lib
.NET Standard 1.4
Class Lib
.NET Standard 1.4
Class Lib


With .NET Standard
.NET Framework 4.6
Mono / Xamarin
Microsoft.NETCore.App
App
4.6 Class Lib
4.6 Class Lib
Class Lib
App
Class Lib
Class Lib
Class Lib
App
Net Core Class Lib
Net Core Class Lib
Class Lib
.NET Standard 1.4
Class Lib
.NET Standard 1.4
Class Lib

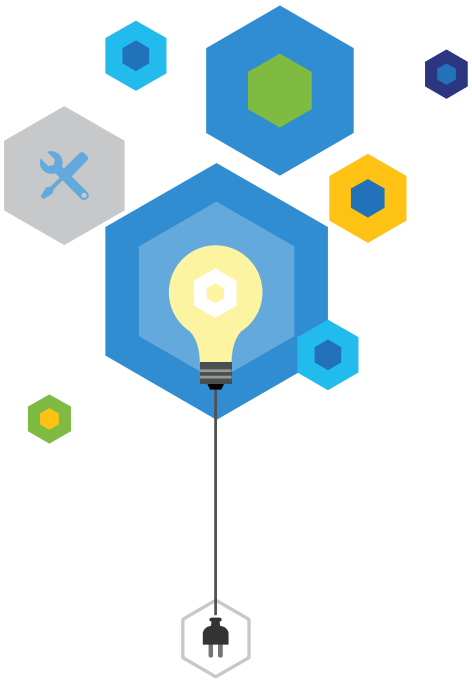
With .NET Standard



Microsoft.NETCoreApp has NETStandard libs

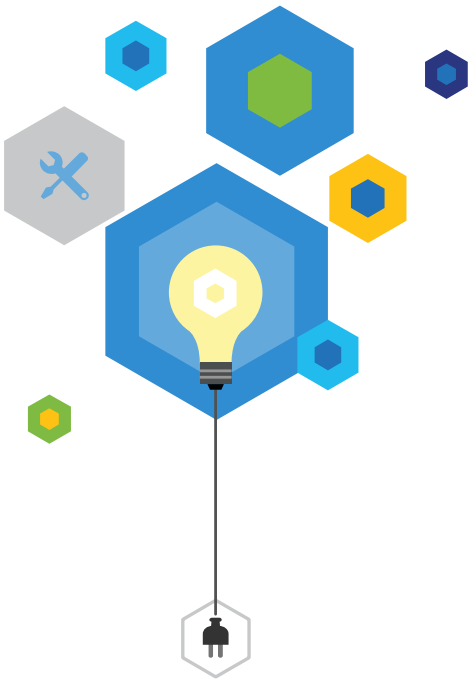


Demo



David Fowler's Example

Demo



.NET Standard and
Microsoft.NetCore.App

CLI & SDK

CLI commands: basic architecture

```
dotnet   build   --output [path]
```

the driver — verb (command) — verb arguments

CLI - Principles of design

Several principles when it comes to designing the CLI:

.NET Core is CLI first

The driver dotnet.exe knows enough to run the command(s) and no more

All core commands are consumable by humans **and** machines

Ex You @ command line and Visual Studio in the IDE

Dotnet.exe commands

Every command is a verb ("compile", "run", "restore" etc.)

Ex dotnet.exe run

The verb is a **command** that is implemented as:

A NuGet package – exdotnet bundle

A binary in the $PATH – ex dotnet custombuildscript

Driver invokes the command passing the **arguments** to it

- The command is responsible for the arguments

What is in the box: Core Commands

dotnet new
Create minimal console app, library or a web project in a directory

dotnet restore
Restore nuget references from project.json, update project.lock.json

dotnet run
Run the application from source

dotnet build
Compile the application, generating artifacts

dotnet publish
Copy app/library + all dependencies to a directory for distribution

dotnet pack
Pack up a NuGet package of your code

dotnet test
Run tests using the configured test runner

Visual Studio LIVE!

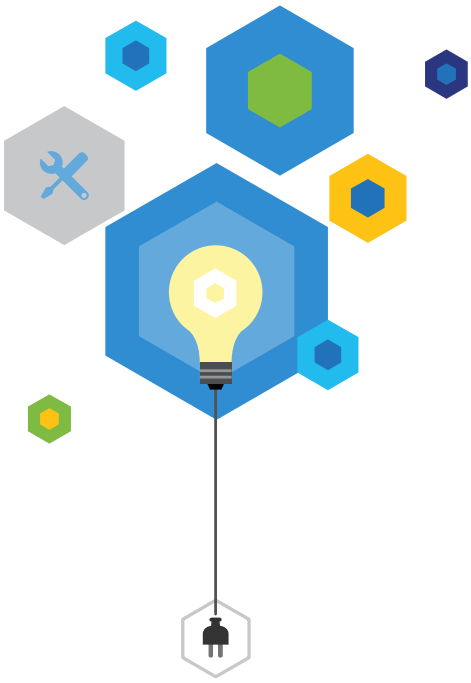
Layout of the SDK installation

- Installs SxS by default
  - So does the shared runtime as well as the host policies (logic)
- The layout is convention-driven
  - Each versioned, SxS component gets its own directory
  - Each version gets its own sub-directory
- "dotnet" binary is an exception
  - The only one that is updated in-place

Visual Studio LIVE!

Demo

Where are the sdk bits?



DEPLOYMENT OPTIONS

.NET Core deployment options

- Multiple types of deployment for .NET Core
- **Framework-dependent deployment (FDD)** - require shared component (sdk) on target
  - **FDD w/ native dependencies** - require shared component but has native dependencies
- **Self-contained deployment (SCD)** - package all with them (what we currently have)

Visual Studio LIVE!

Framework-dependent deployment

- If you want an application that is truly cross-platform, you need a shared component
- A shared API set that:
  - Can be targeted
  - Can be installed on its own
- Includes a host for running applications
- Your application is OS-agnostic
  - Can run wherever the shared runtime is installed
- Default deployment mode

Visual Studio LIVE!

FDD w/ native dependencies

- Application has a native dependency
  - Example: Kestrel
- The published application contains native assets
  - For each OS and architecture
- Native dependencies limit cross-platform compat
  - The application will run on those platforms where the native dependency runs

Visual Studio LIVE!

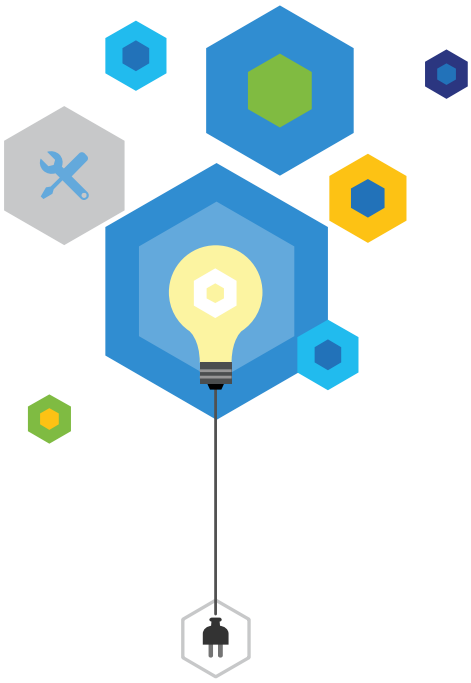
Self-contained deployment

- Packages the runtime & framework libraries within the application
  - Together with any dependencies of the application
  - Fully self contained
- Must specify the targeted platform(s)
  - Using Runtime Identifiers (RID) in project.json
  - Need target platforms for *platform specific* libraries like System.IO

Visual Studio LIVE!

Demo

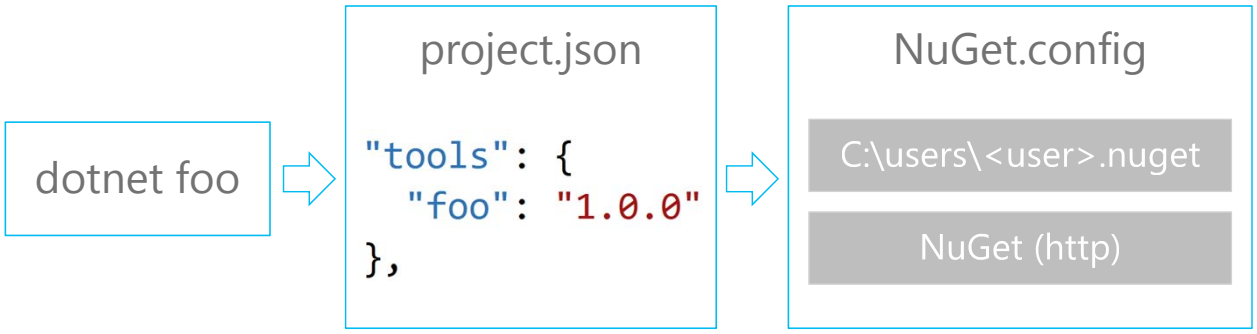
Self-contained Deployment



Using dotnet to extend the pipeline

TOOLS EXTENSIBILITY

Tools Extensibility: via NuGet

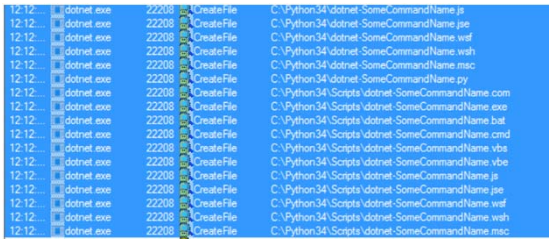


Tools Extensibility: via $PATH

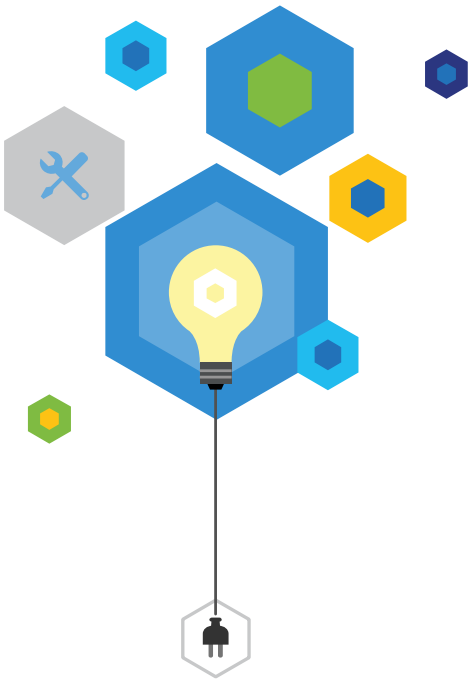
dotnet foo

Process Name	Path
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.com
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.exe
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.bat
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.cmd
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.vbs
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.vbe
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.js
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.jse
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.wsf
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.wsh
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.msc
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.com
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.exe
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.bat
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.cmd
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.vbs
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.vbe
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.js
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.jse
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.wsf
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.wsh
dotnet.exe	C:\Program Files\dotnet\sdk\1.0.0-preview2-003121\dotnet-foo.msc

Demo



Extending the toolset



Adam Tuliper | @AdamTuliper

Thank you!!!

adamt@microsoft.com