

Mobile Development Updates From #MSBuild 2017 & .NET Standard (& What It Means For You)

Andrew Birch



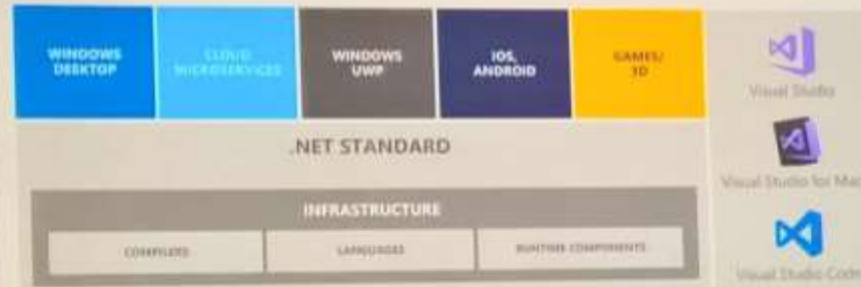
Day 1 Keynote – About To Begin!



Day 1 Keynote – Cognitive Services

THOSE GET THE EXACT SAME DOT
NET

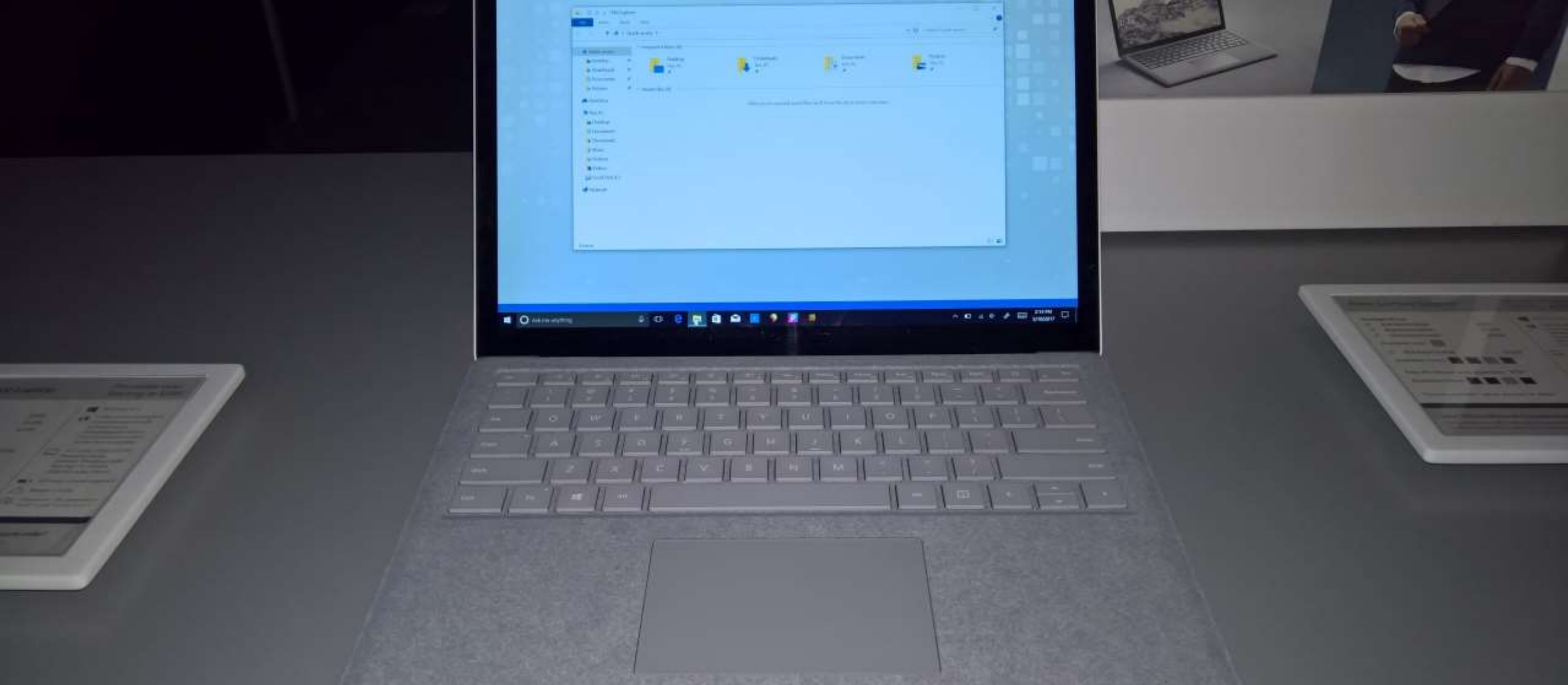
.NET Standard



.NET Standard allows sharing code, binaries, and skills between .NET client, server, and all flavors

.NET Standard provides a specification for any platform to implement
All .NET runtimes provided by Microsoft implement the standard

.NET Standard Breakout Session



Surface Laptop



Surface Laptop

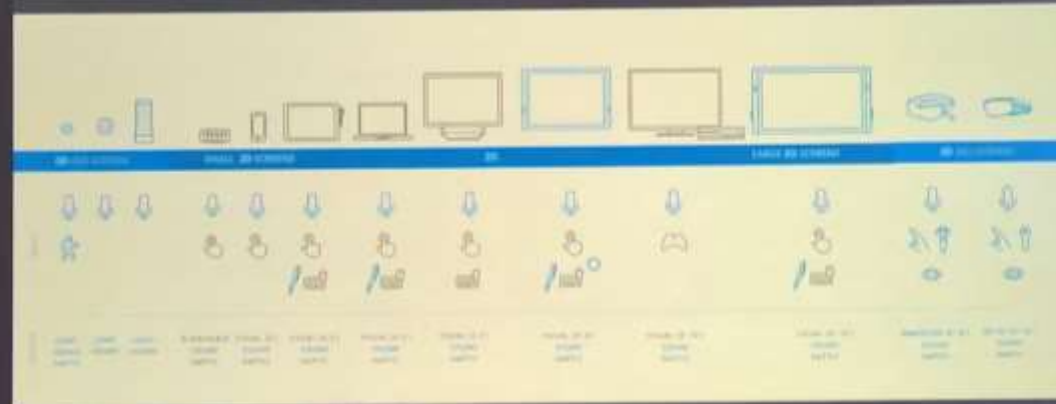


The Hub

THEMSELVES.
SO HERE YOU HAVE IT, THIS IS

Era of Ubiquitous Computing

The Playground Is Expanding



Fluent Design Breakout Session



Fluent Design!

Mobile Related Announcements From #MSBuild Keynotes

- Visual Studio For Mac RTM – Available now
- .NET Standard 2.0 - Coming later this year
- XAML Standard 1.0 – Coming later this year
- Timelines coming to Windows In the 'Fall Creators Update' (Spring for us)
- Fluent Design Language Announced!
- Device and Activity Roaming APIs (Project Rome) added to the Microsoft Graph
- Project Rome SDK (Now available for iOS) – Preview available now
- UWP support in Visual Studio Mobile Centre, Including Windows devices to the Test Cloud – Coming later this year
- Xamarin Live Player – Preview available now

Xamarin Platform & Tooling Announcements

- CocoaPods importer
- SwiftNetifier – Coming later this year
- Xamarin Linker now better optimised for Xamarin.iOS Projects with extensions
- Provisioning Profile configuration support in Visual Studio for Mac (Thanks to Fastlane)
- New integrated Android SDK Manager inside Visual Studio for Mac
- Material Design Designer (Generates Android theme code inside VS for Mac)
- Ahead of Time compilation for Android & macOS - Preview available now
- Hybrid Mode compilation (AOT & JIT) for Android & macOS - Preview available now
- Embedinator-4000: Turns .NET Code into Java, Swift, Objective-C Or C/C++

Xamarin.Forms Announcements

- Xamarin Forms 2.3.5 – Preview available now
 - .NET Standard compatible (1.1 & higher recommended)
 - Fast Renderers for Android (For Image, Label & Button)
 - macOS Preview
 - Accessibility support
 - XAMLC type improvements & bug fixes
- Xamarin Forms 3.x – Coming later this year (*Features rolling out throughout 3.x lifecycle, not all in 3.0.0*)
 - Performance improvements
 - Flexlayout (CSS Flexbox inspired layout system)
 - ListView improvements – Faster loading, direct item insertion, remove requirement to use cells
 - Standardised Renderer API – Bring renderer API in line on each platform (Where possible)
 - Other Improvements: Onetime binding, XAML improvements, CSS-like styling, Visual State Manager, better adaptive layout (mirror UWP relative layout)
 - Xamarin.Forms embedding (ContentPage: As fragment in Android, UIViewController in iOS, Page in UWP)
 - Xamarin Forms support for macOS, Linux (GTK#), WPF & Tizen)
 - **MAYBE** – Looking for feedback: Xamarin.Forms embedding for native iOS (Objective C/Swift) & Android (Java)

Xamarin Live Player

- This is an app you install on iOS & Android, which pairs with VS & allows near instant feedback cycles during development
- Has Two Modes:
 - Regular debug/run mode (works just like you normally would with emulators or devices)
 - Live Run Current View mode (displays only the currently open view in the player)
- Works with Visual Studio for Windows & Visual Studio for Mac
- Works with Xamarin.Forms & traditional Xamarin.iOS/Xamarin.Android projects

.NET Standard

- .NET Standard is the next generation of Portable Class Libraries (PCLs)
- The issue with PCLs
 - PCLs were lowest common denominator of selected platforms (called a profile)
 - Libraries could be written to support PCLs by declaring profiles they supported
 - If new platforms were added, new profiles had to be created
 - If libraries you referenced in your PCL didn't support a new profile, you couldn't change your PCL to that new profile, therefore you couldn't support new platforms that the new profile included
 - This was a problem for when moving from Windows Phone 8.0 to 8.1 or 8.1 to UWP. Was also a problem for any .NET libraries that never targeted the profiles that included Xamarin.

.NET Standard

- .NET Standard fixes the issues with PCLs by:
 - Replaces PCL profiles with .NET Standard versions
 - Each .NET Standard version introduces new APIs that are available in that version (and all future versions – only additive)
 - Any platform which wants to be compatible with a .NET Standard version must implement all APIs that are available in that version
 - Library writers pick the minimum .NET Standard version they wish to support and target that.
 - Their libraries are now automatically compatible with all future .NET Standard versions and any new platforms that implement those versions.
 - Need more APIs for your library? Pick a newer .NET Standard version for future library updates

.NET Standard 1.x

.NET platforms support

The following table lists all versions of .NET Standard and the platforms supported:

.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	vNext	4.6.1
.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							

.NET Standard 2.0

- Unifies .NET Framework (Windows), Xamarin (Mono) & .NET Core
- Will be the intersection of the .NET Framework and Xamarin APIs
- Once a library targets .NET Standard 2.0 it will be able to be used on Windows (Desktop, Mobile, IoT, HoloLens), iOS, tvOS, watchOS, macOS, Android, Linux, Tizen and any other .NET Standard 2.0 platforms
- Also means .NET Core 2.0 (will be based on .NET Standard 2.0) will have an enormous number of APIs added as compared to .NET Core 1.x

.NET Standard 2.0

- .NET Standard 2.0 will allow compatibility with all .NET Framework 4.6.1 libraries via a compatibility shim
- This means that .NET Standard 2.0 library projects can reference many .NET Framework 4.6.1 libraries, even if it's never been targeted to .NET Standard 2.0
- However, this will only work if the .NET Framework library only references APIs that are included in .NET Standard. So if it references any APIs that aren't included in .NET Standard (normally Windows only APIs that don't make sense to be shared cross-platform anyway), then the compatibility shim won't work.

XAML Standard 1.0

- UWP XAML & Xamarin.Forms XAML have similarities, but have quite different syntax
- They could be allowed to keep diverging, limiting code sharing and duplicating implementation effort, or a process to unify them could be started.
- Starting to unify them was the option chosen and XAML Standard is the method
- Like .NET Standard, once a version of XAML Standard is created, any platforms can implement that version so developers of that platform can use it
- For now it hasn't been decided if UWP XAML becomes more like Xamarin.Forms XAML or visa versa. Keeping backwards compatibility for all users is the goal for now
- Microsoft are looking for feedback on the draft of v1 right now on GitHub

Links

- All Things .NET: <https://dot.net>
- Visual Studio for Mac: <https://www.visualstudio.com/vs/visual-studio-mac>
- XAML Standard: <https://aka.ms/xamlstandard>
- .NET Standard: <https://docs.microsoft.com/en-us/dotnet/articles/standard/library>
- Embeddinator 4000: <https://mono.github.io/Embeddinator-4000>
- Project Rome – <https://aka.ms/projectrome>
- Xamarin Live Player – <https://xamarin.com/live>

Thanks!

@andrewtechhelp

Andrewtechhelp.com