Move your Windows apps forward with .NET Core 3.0, MSIX, XAML Islands and WinUI

Matteo Pagani
matteo.pagani@microsoft.com | @qmatteoq
Windows AppConsult Engineer

CommunityDays.it

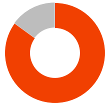
Most PCs are or soon will be on Windows 10

96% of enterprise customers piloting Windows 10

+700M
active devices around the world running Windows 10

~2/3
of organizations will complete Windows migration in less than a year**

14 months
Payback period for Windows 10 deployment costs*



85%
of Enterprises will have started Windows 10 deployments by the end of the year**

33%
reduction in security issues and resolution time*

49%
of Enterprises named security improvements as a driver for planned Windows 10 deployment**

---

* "The Total Economic Impact of Windows 10"; Forrester,
** "User Survey Analysis: Windows 10 looks heathy;"

+700M
active devices around the world running Windows 10

~2/3
of organizations will complete Windows migration in less than a year**

14 months
Payback period for Windows 10 deployment costs*

85%
of Enterprises will have started Windows 10 deployments by the end of the year**

33%
reduction in security issues and resolution time*

49%
of Enterprises named security improvements as a driver for planned Windows 10 deployment**

Most customers want to move to Windows 10 as soon as possible for security and app deployment advantages, well **before Windows 7 End Of Support January 2020**

* "The Total Economic Impact of Windows 10"; Forrester
** "User Survey Analysis: Windows 10 looks heathy;" Gartner

The challenge of modernizing the desktop ecosystem

Rationalizing two distinct scenarios

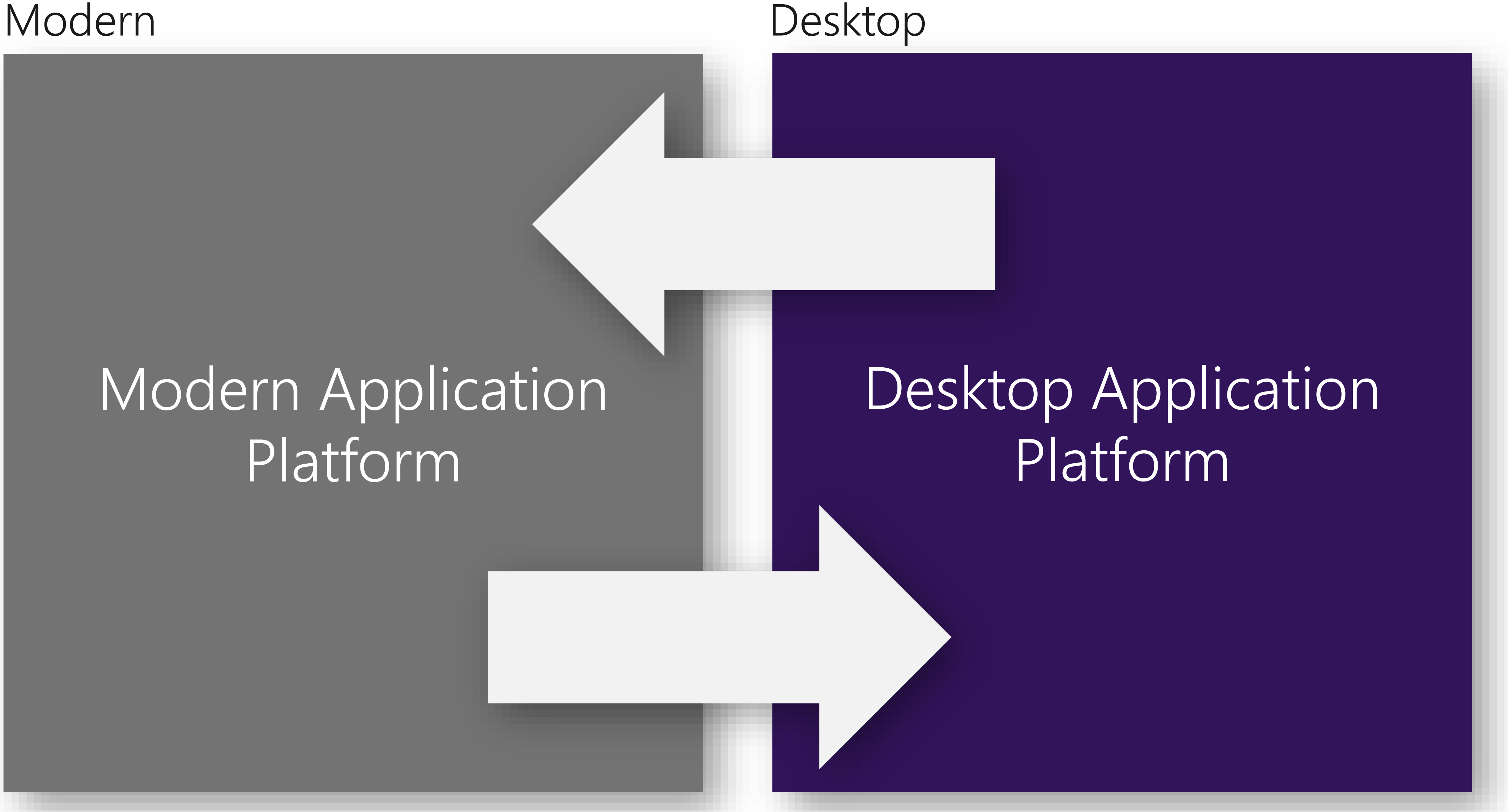
Modern Application Attributes

- Contained Execution & Great Fundamentals
- Platform Managed Install & Update
- Trusted Distribution & Signed Packages
- Modern UI
- Cloud First Data with Insights
- Designed for Reuse

Desktop Application Attributes

- User & Admin Level Security
- MSI, Custom Installers & Updaters
- Web, SCCM & Custom Distribution
- Windows Forms, WPF, MFC, etc.
- Local Data
- Designed for Windows Desktop Only

Bring both platforms closer together



Modern

Desktop

Modern Application Platform

Desktop Application Platform

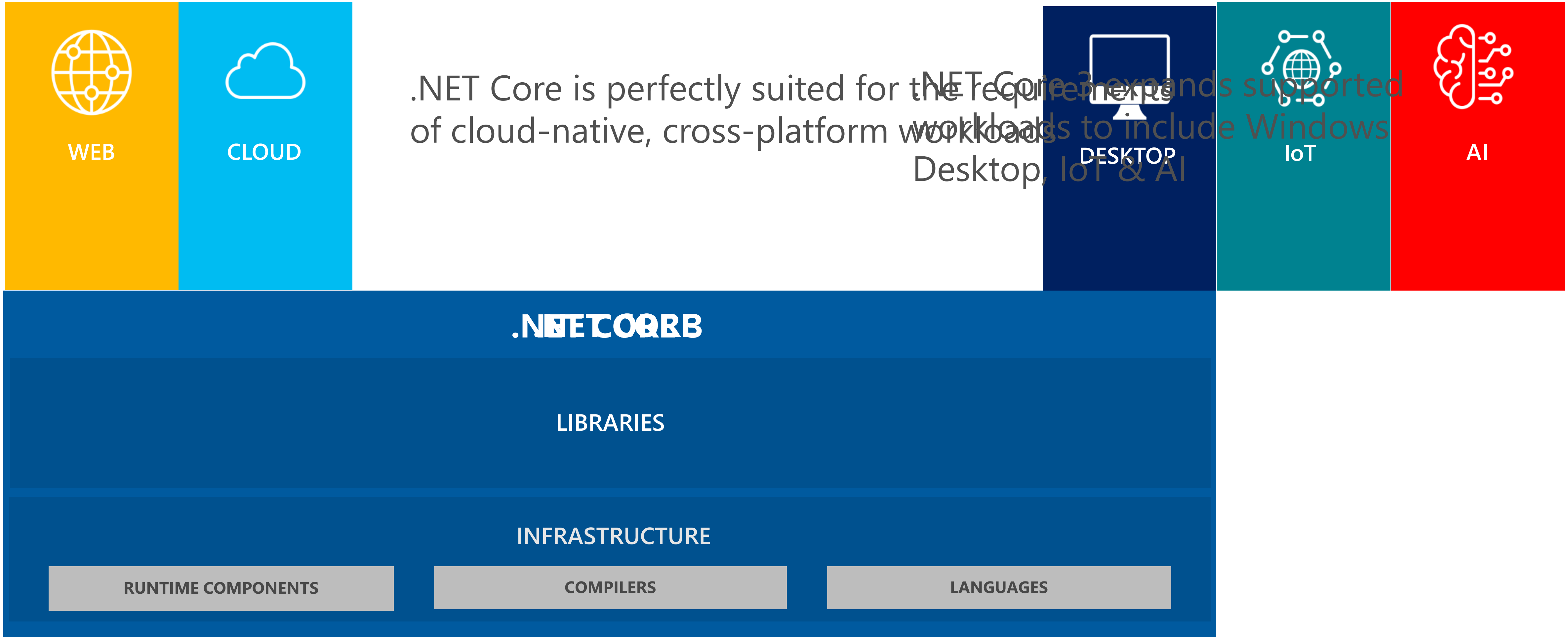
Enhancing your Desktop assets - .NET Core

.NET Core 2.x

.NET Core 3.0

.NET Core is perfectly suited for the requirements of cloud-native, cross-platform workloads

.NET Core 3 expands supported workloads to include Windows Desktop, IoT & AI



.NET Core 3. Why?

Faster

Side-by-Side installation

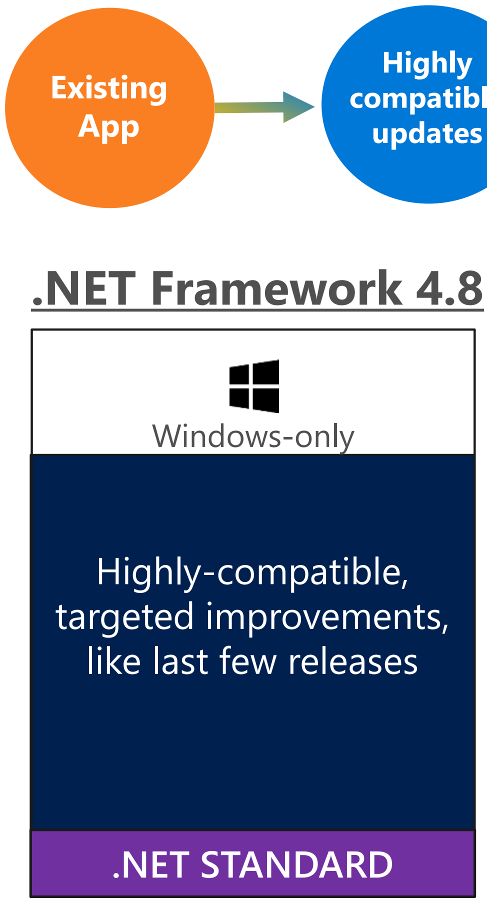
Better packaging options (MSIX)

Open Source

.NET Standard 2.1

New Features

Update .NET Framework Apps



Existing App → Highly compatible updates

.NET Framework 4.8

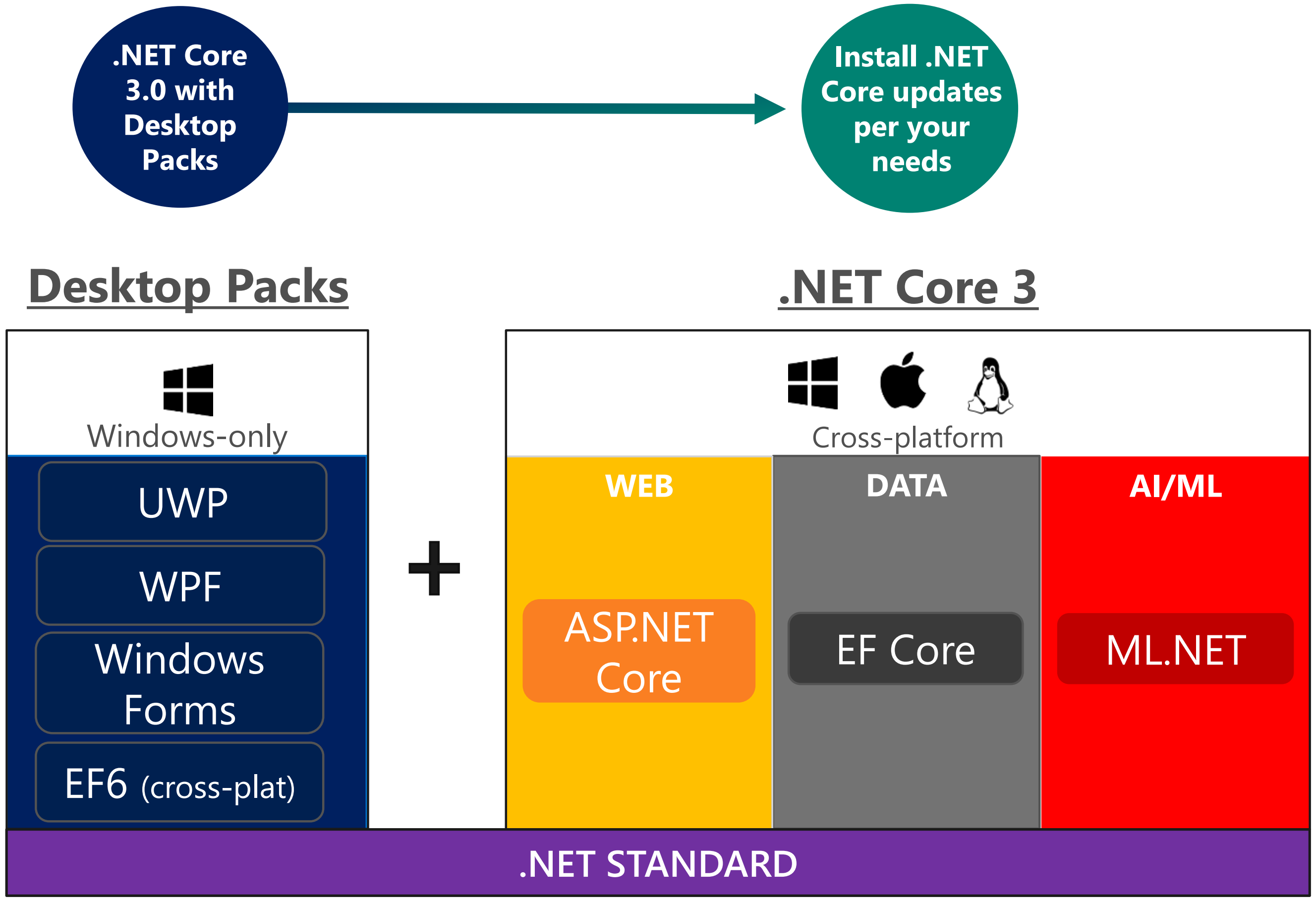
Windows-only

Highly-compatible, targeted improvements, like last few releases

.NET STANDARD

- .NET Framework support unchanged (supported for life of Windows)

Modernize Desktop Apps with .NET Core 3



.NET Core 3.0 with Desktop Packs → Install .NET Core updates per your needs

Desktop Packs

Windows-only

UWP

WPF

Windows Forms

EF6 (cross-plat)

+

.NET Core 3

Cross-platform

WEB	DATA	AI/ML
ASP.NET Core	EF Core	ML.NET

.NET STANDARD

Get started moving to .NET Core

- Use Portability Analyzer to check how much code you can port https://aka.ms/portabilityAnalyzer
- Move to the new project's file format and to PackageReference for NuGet
- Leverage **Microsoft.Windows.Compatibility** to add support to specific Windows APIs
- Leverage **Microsoft.DotNet.Analyzers.Compatibility** to find out which APIs aren't cross-platform

Demo

WPF & .NET Core 3.0

Windows

Inkpowered

Enhanced

Modernize your

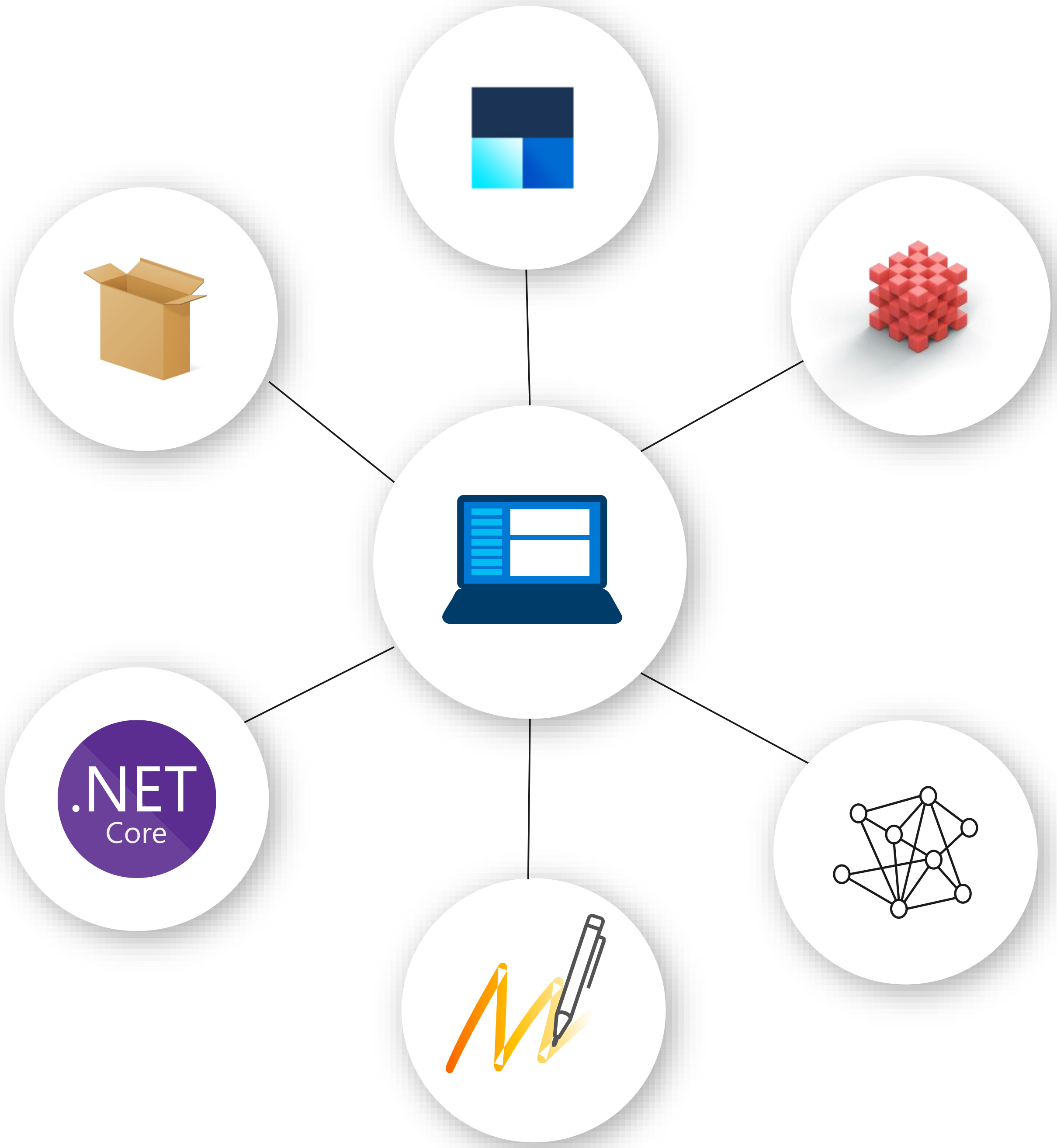
Apps

Containers

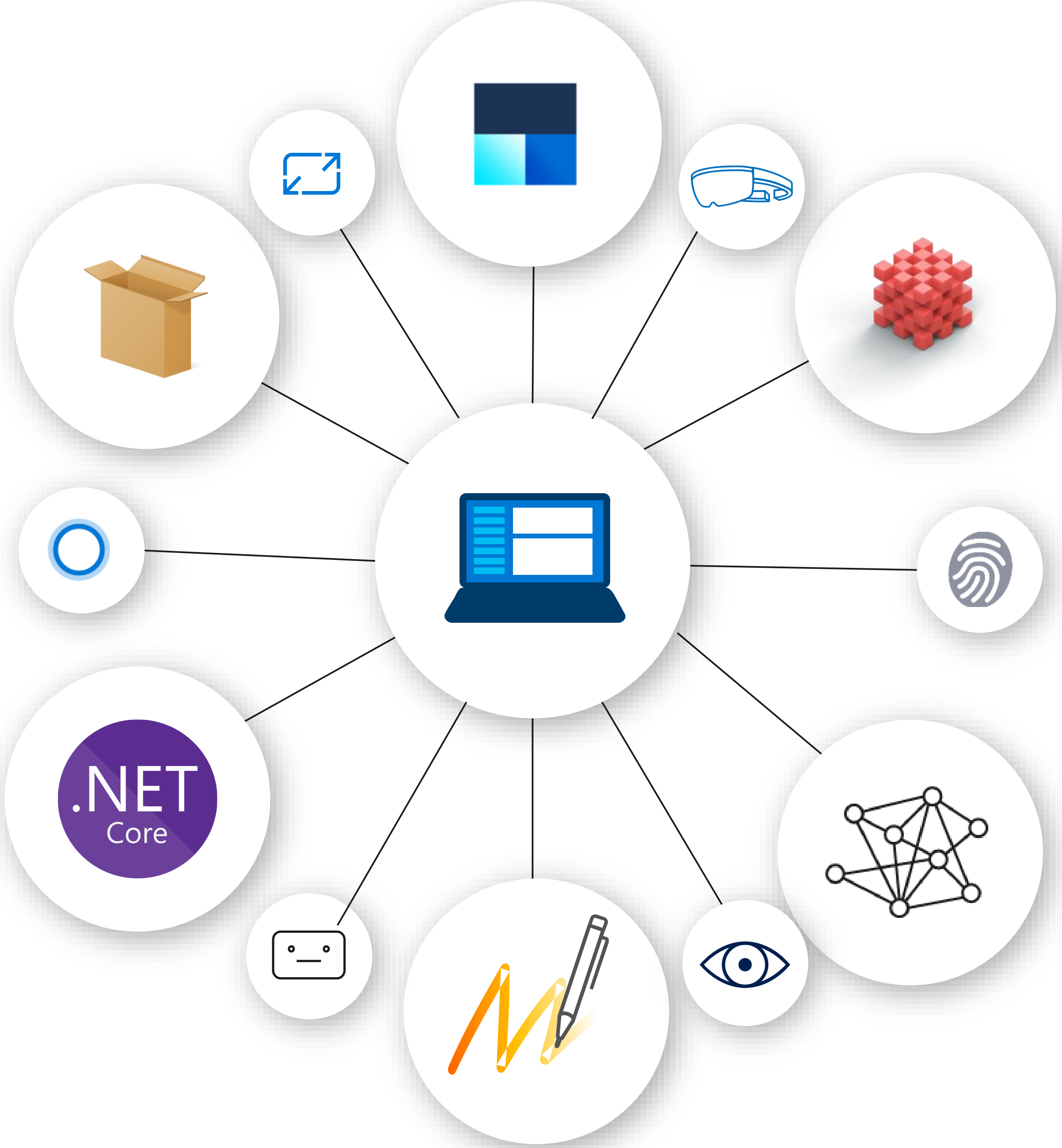
Experiences

Desktop Apps

Learning



And more...



Enhancing your Desktop assets – UWP APIs

Current status

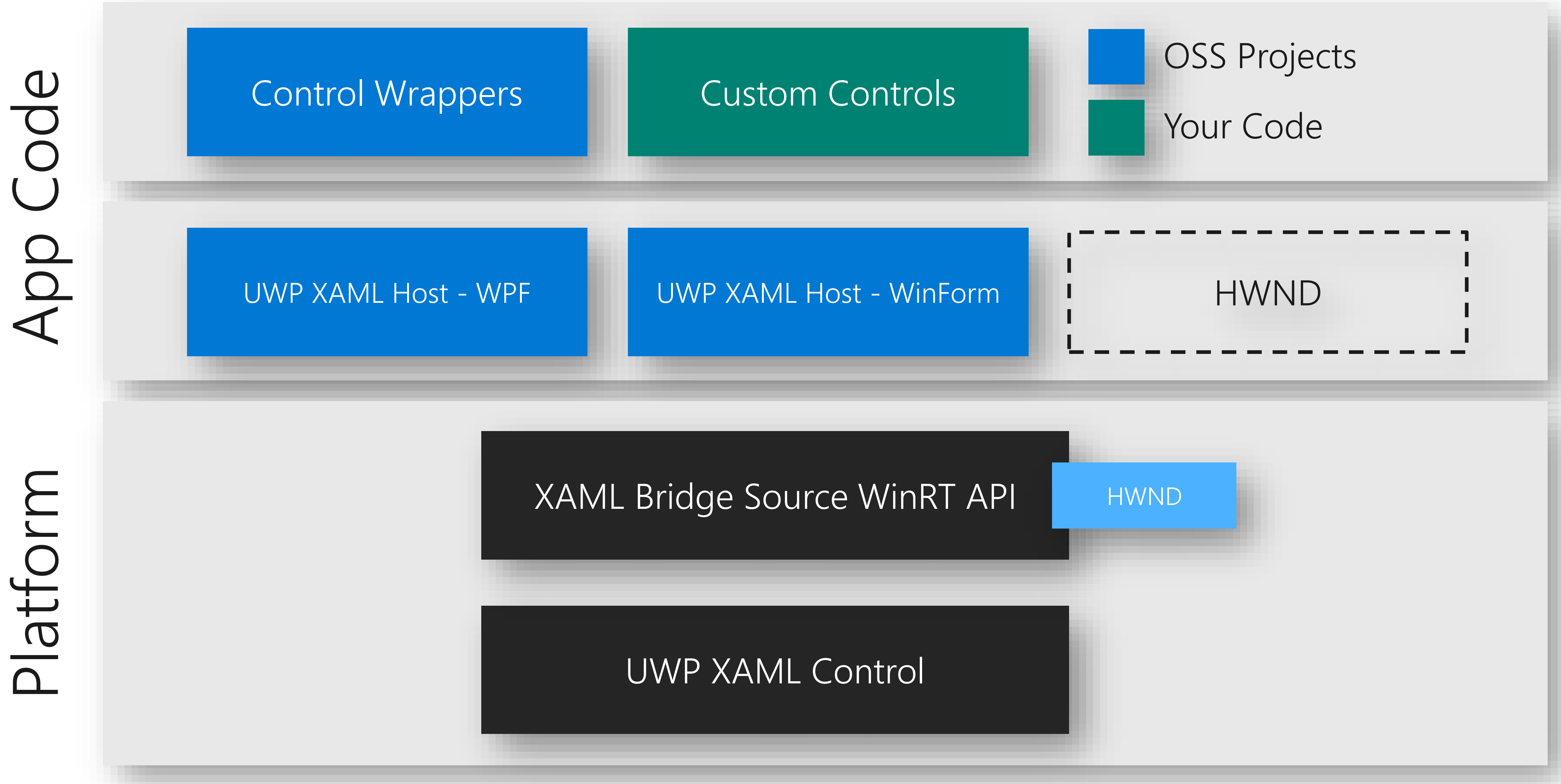
- ➢ Windows features exposed thru UWP APIs
  - Devices & Sensors (ie: BluetoothLE)
  - XAML, Ink, Media, Speech/Cortana
  - Mixed Reality/3D, Machine Learning
  - New Input modes - Eye tracking
- ➢ Most APIs available to Win32/.NET apps today!
  - DualApiPartition attribute – can call today
- ➢ Some APIs require UWP Concepts
  - Identity – Requires MSIX package
  - CoreWindow – APIs show UI and XAML Controls requires XAML Islands

Benefits

Add modern features to your existing Win32 WPF or WinForms applications with minimum code changes

Just add a reference to the **Microsoft.Windows.SDK.Contracts** package on NuGet

The architecture



App Code

Control Wrappers

Custom Controls

🟦 OSS Projects

🟩 Your Code

UWP XAML Host - WPF

UWP XAML Host - WinForm

HWND

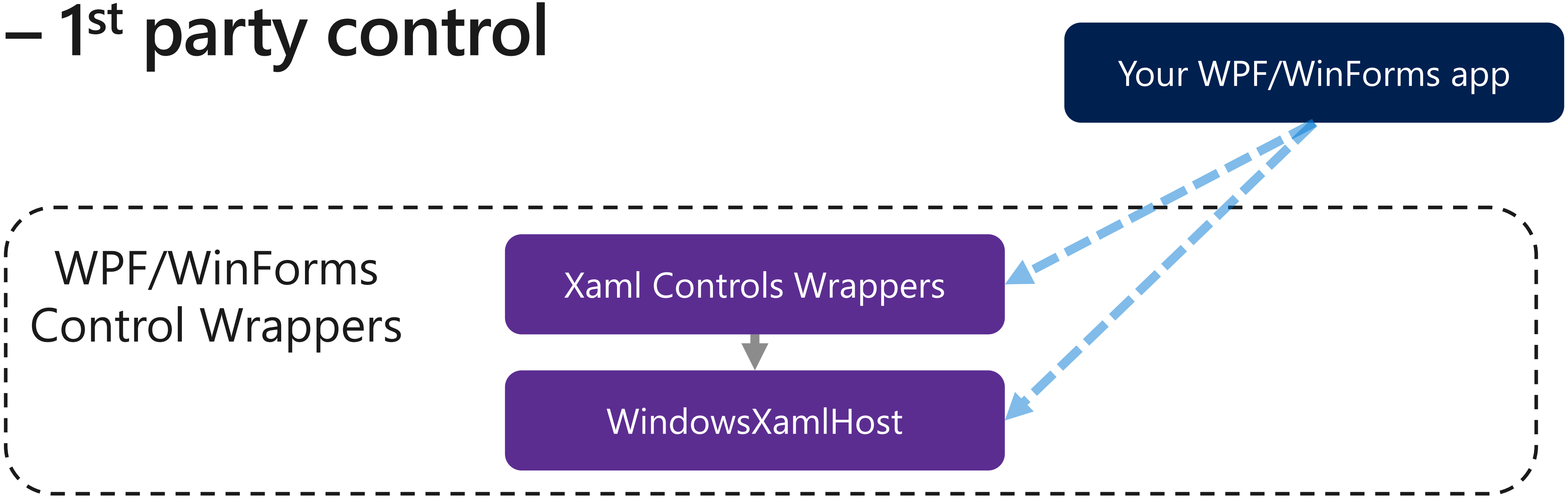
Platform

XAML Bridge Source WinRT API

HWND

UWP XAML Control

Windows Community Toolkit wrappers – 1st party control


Your WPF/WinForms app

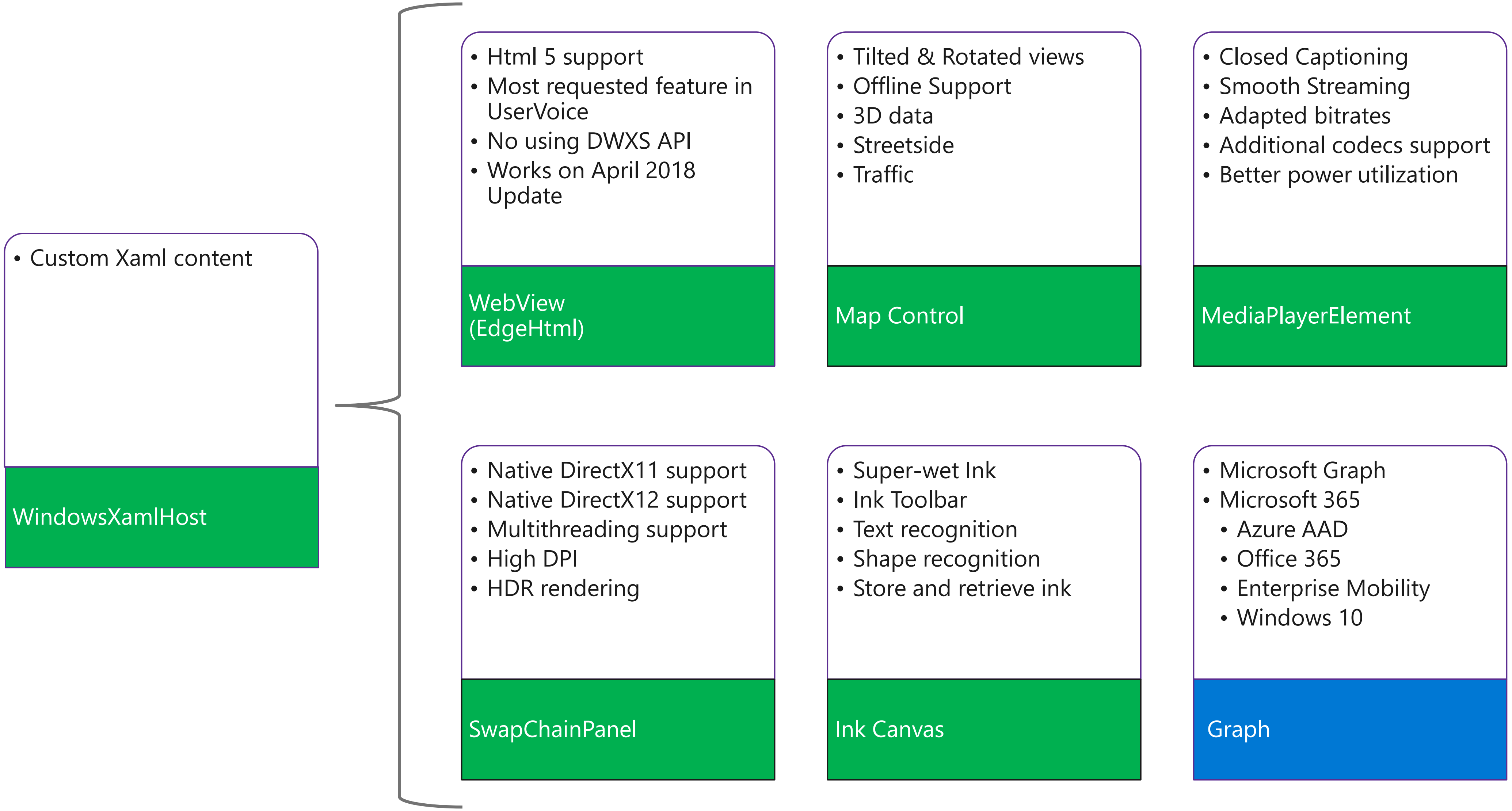
WPF/WinForms Control Wrappers

Xaml Controls Wrappers

WindowsXamlHost

- No WinRT concepts
- NuGet Packages for Windows Forms and WPF
- Designer Support

XAML Control Wrappers



- Custom Xaml content

WindowsXamlHost

- Html 5 support
- Most requested feature in UserVoice
- No using DWXS API
- Works on April 2018 Update

WebView (EdgeHtml)

- Tilted & Rotated views
- Offline Support
- 3D data
- Streetside
- Traffic

Map Control

- Closed Captioning
- Smooth Streaming
- Adapted bitrates
- Additional codecs support
- Better power utilization

MediaPlayerElement

- Native DirectX11 support
- Native DirectX12 support
- Multithreading support
- High DPI
- HDR rendering

SwapChainPanel

- Super-wet Ink
- Ink Toolbar
- Text recognition
- Shape recognition
- Store and retrieve ink

Ink Canvas

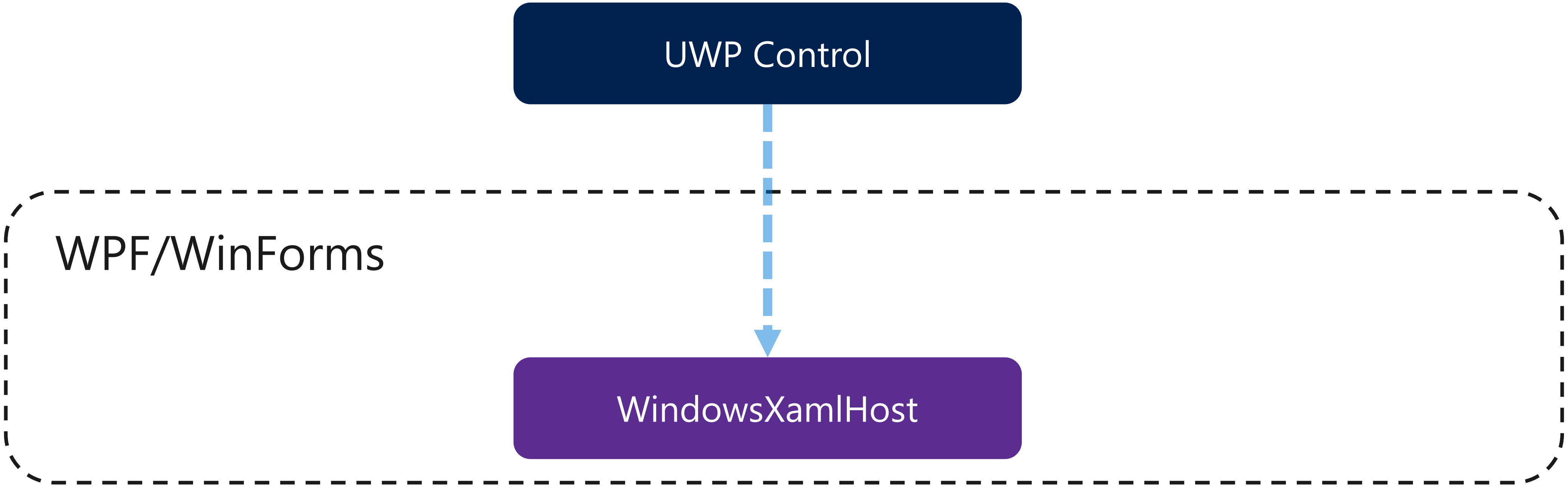
- Microsoft Graph
- Microsoft 365
  - Azure AAD
  - Office 365
  - Enterprise Mobility
  - Windows 10

Graph

🟩 Shipped in Preview

🟦 In development

Windows Community Toolkit – WindowsXamlHost

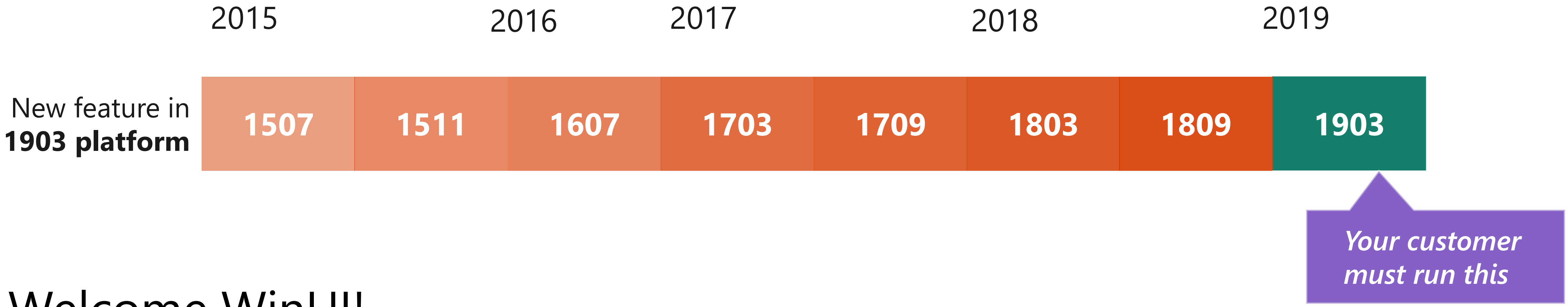


- Host for any UWP Control
- Dedicated NuGet package:
  - WPF: https://www.nuget.org/packages/Microsoft.Toolkit.Wpf.UI.XamlHost/
  - Windows Forms: https://www.nuget.org/packages/Microsoft.Toolkit.Forms.UI.XamlHost/
- If you are using 1st party wrapped controls, the package is already installed since it's a dependency

Demo

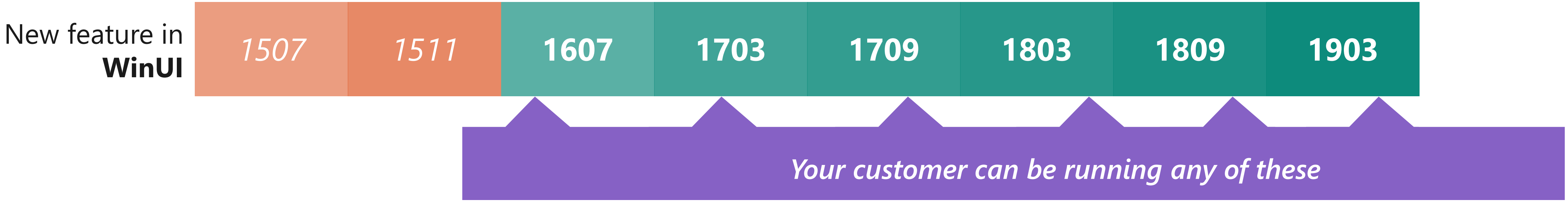
Adding UWP features and UI

The challenge in adopting UWP controls

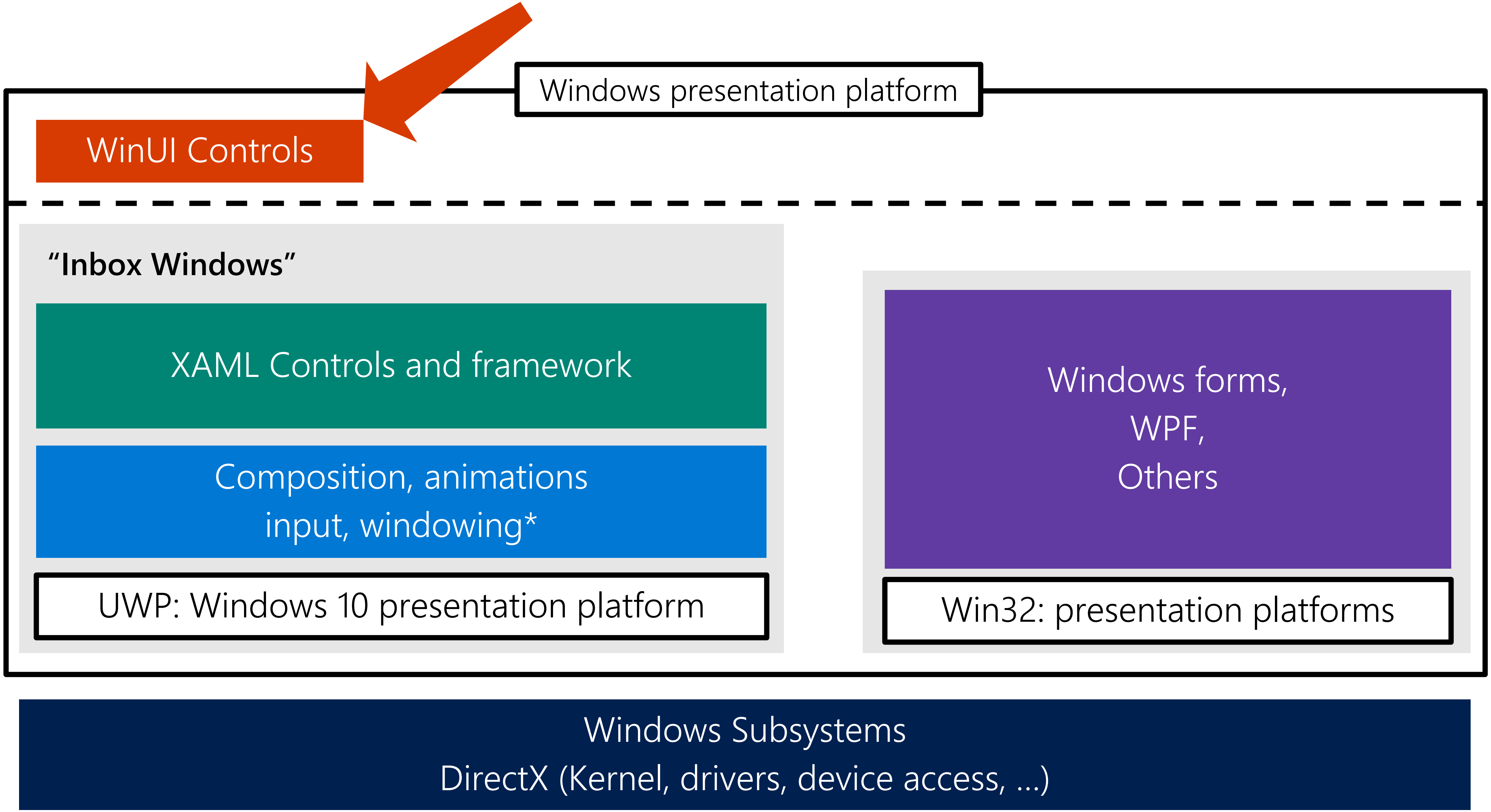


Your customer must run this

Welcome WinUI!



Your customer can be running any of these



Windows presentation platform

WinUI Controls

“Inbox Windows”

XAML Controls and framework

Composition, animations
input, windowing*

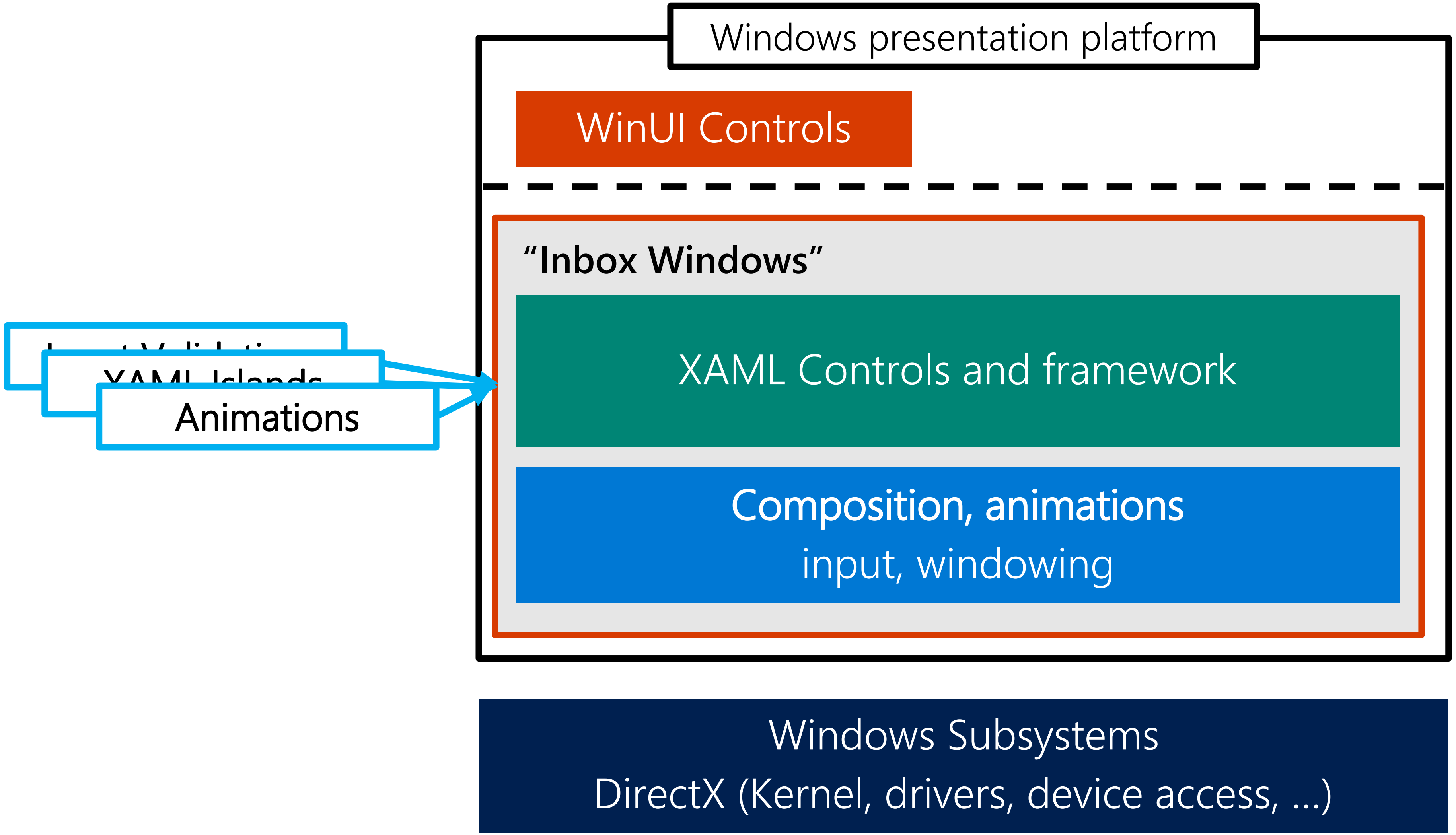
UWP: Windows 10 presentation platform

Windows forms,
WPF,
Others

Win32: presentation platforms

Windows Subsystems
DirectX (Kernel, drivers, device access, …)

Windows 10 presentation platform: Inbox challenge



XAML Islands

Animations

Windows presentation platform

WinUI Controls

“Inbox Windows”

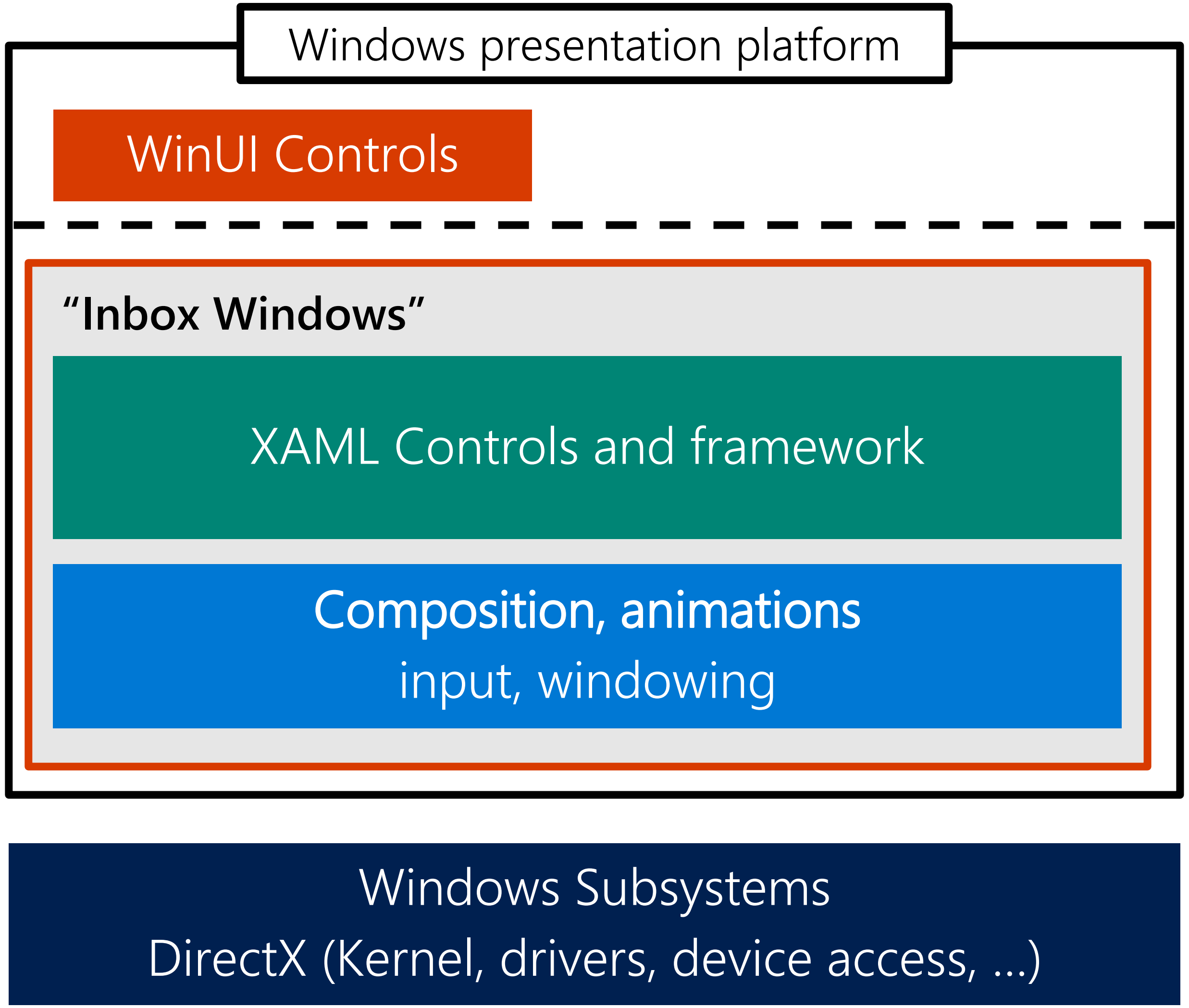
XAML Controls and framework

Composition, animations
input, windowing

Windows Subsystems
DirectX (Kernel, drivers, device access, …)

Announcing WinUI 3.0

Windows 10 presentation platform



Windows presentation platform

WinUI Controls

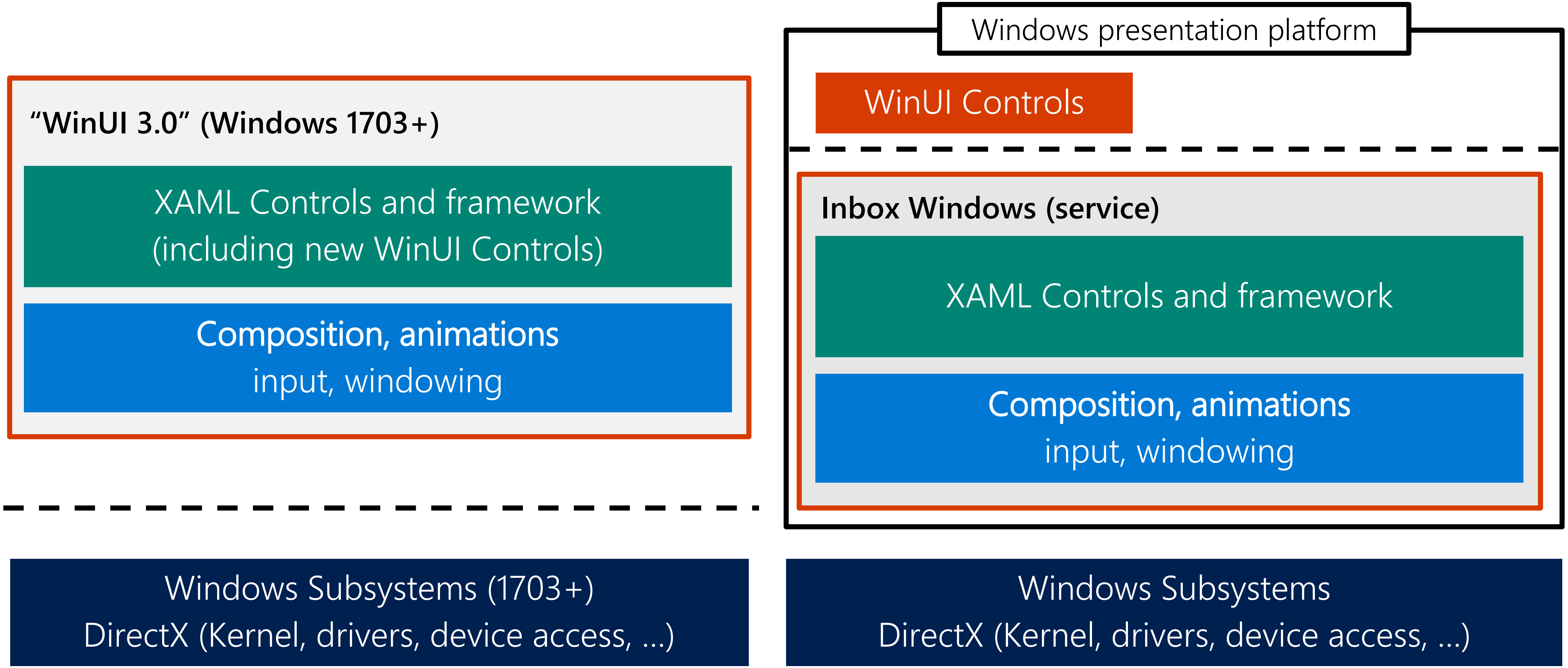
“Inbox Windows”

XAML Controls and framework

Composition, animations
input, windowing

Windows Subsystems
DirectX (Kernel, drivers, device access, …)

Windows 10 presentation platform



“WinUI 3.0” (Windows 1703+)

XAML Controls and framework
(including new WinUI Controls)

Composition, animations
input, windowing

Windows Subsystems (1703+)
DirectX (Kernel, drivers, device access, …)

Windows presentation platform

WinUI Controls

Inbox Windows (service)

XAML Controls and framework

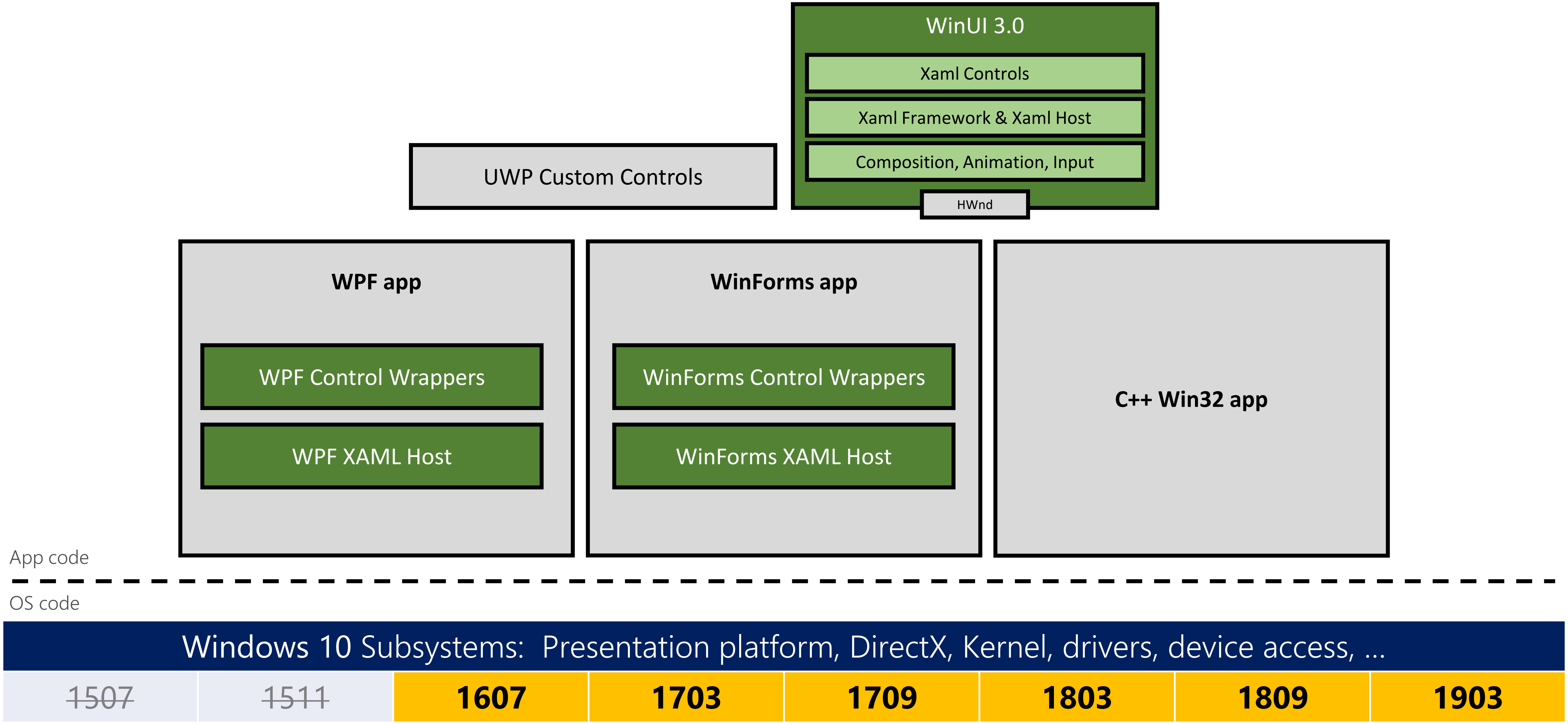
Composition, animations
input, windowing

Windows Subsystems
DirectX (Kernel, drivers, device access, …)

XAML Islands Components Future

🟩 Windows shipped

⬜ App developer code



WinUI 3.0

Xaml Controls

Xaml Framework & Xaml Host

Composition, Animation, Input

HWnd

UWP Custom Controls

WPF app

WPF Control Wrappers

WPF XAML Host

WinForms app

WinForms Control Wrappers

WinForms XAML Host

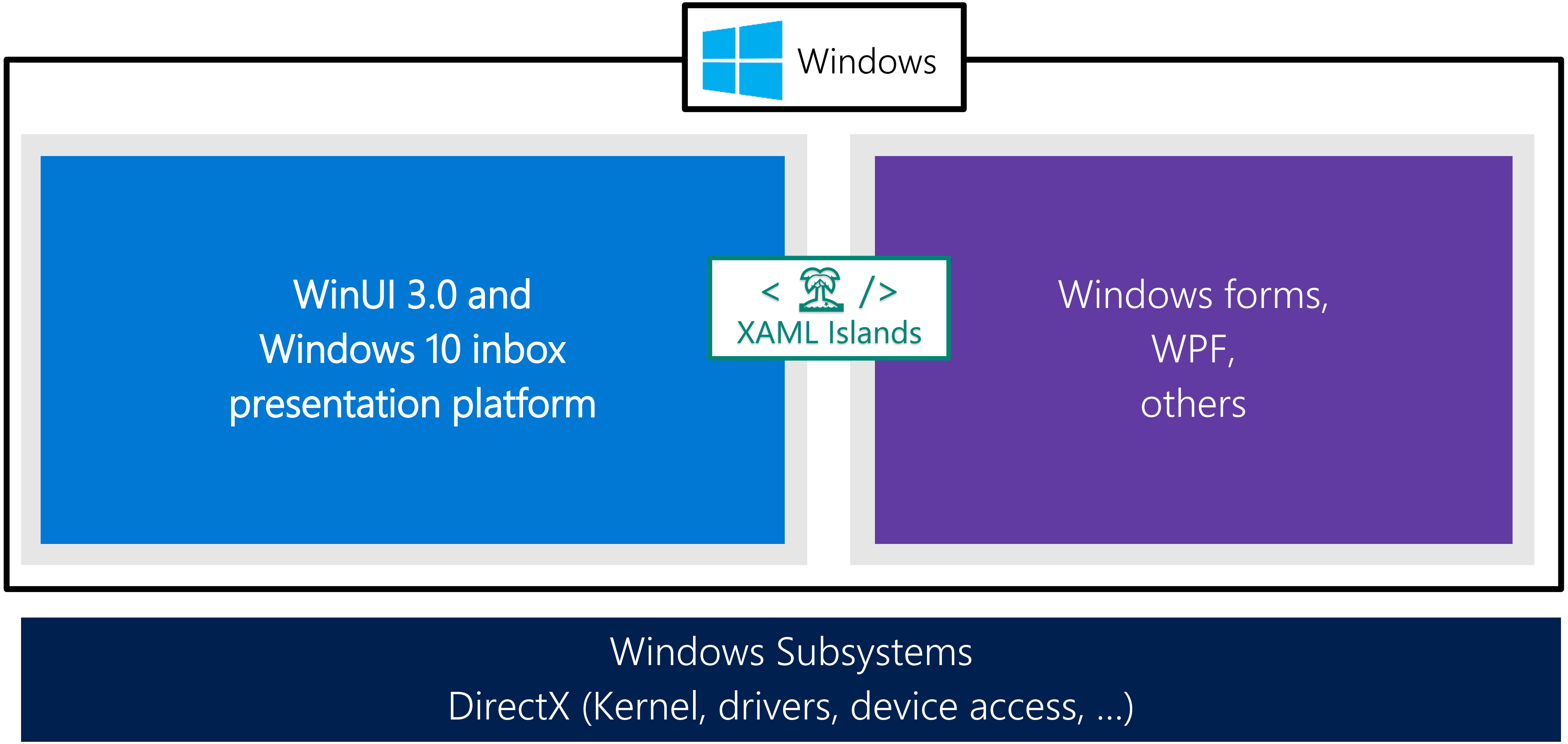
C++ Win32 app

App code

OS code

Windows 10 Subsystems: Presentation platform, DirectX, Kernel, drivers, device access, …

~~1507~~ ~~1511~~ **1607** **1703** **1709** **1803** **1809** **1903**



Windows

WinUI 3.0 and
Windows 10 inbox
presentation platform

< /> XAML Islands

Windows forms,
WPF,
others

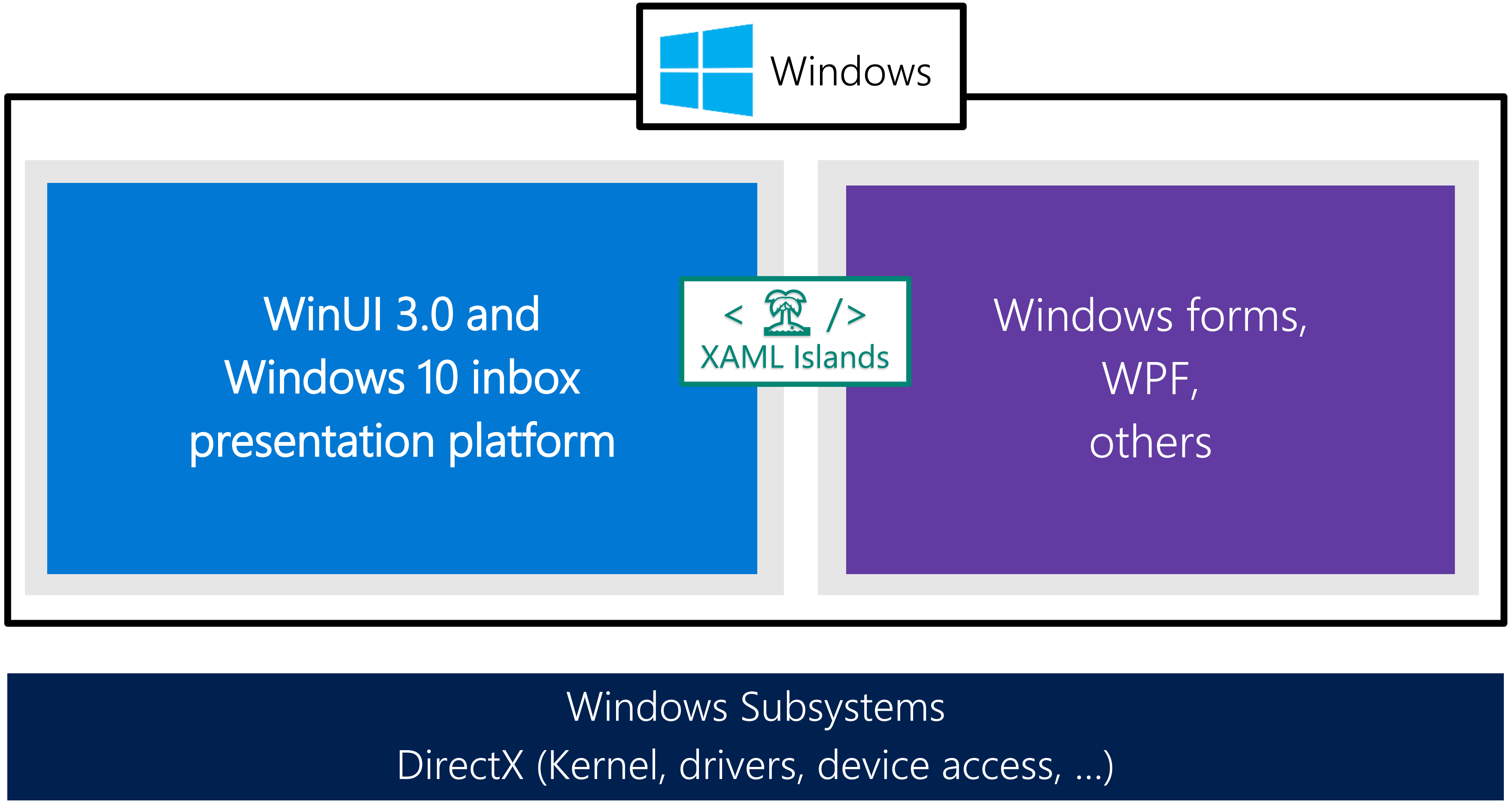
Windows Subsystems
DirectX (Kernel, drivers, device access, …)

Xamarin.Forms

(.NET)

React Native

(Web Developer)



Windows

WinUI 3.0 and
Windows 10 inbox
presentation platform

< /> XAML Islands

Windows forms,
WPF,
others

Windows Subsystems
DirectX (Kernel, drivers, device access, ...)

JS CSS JSX

Evolve to a modern deployment technology

Setup.exe

MSI

MSIX

Microsoft
Application
Virtualization

script.vbs

Key Benefits of MSIX

Modern IT Virtuous Cycle

- Enable OS, Apps & IT to update independently

Never Regret Installing an app

- Predictable, safe, and reliable deployment
- Trusted, clean install & uninstall – no Win ROT

Simpler packaging and deployment

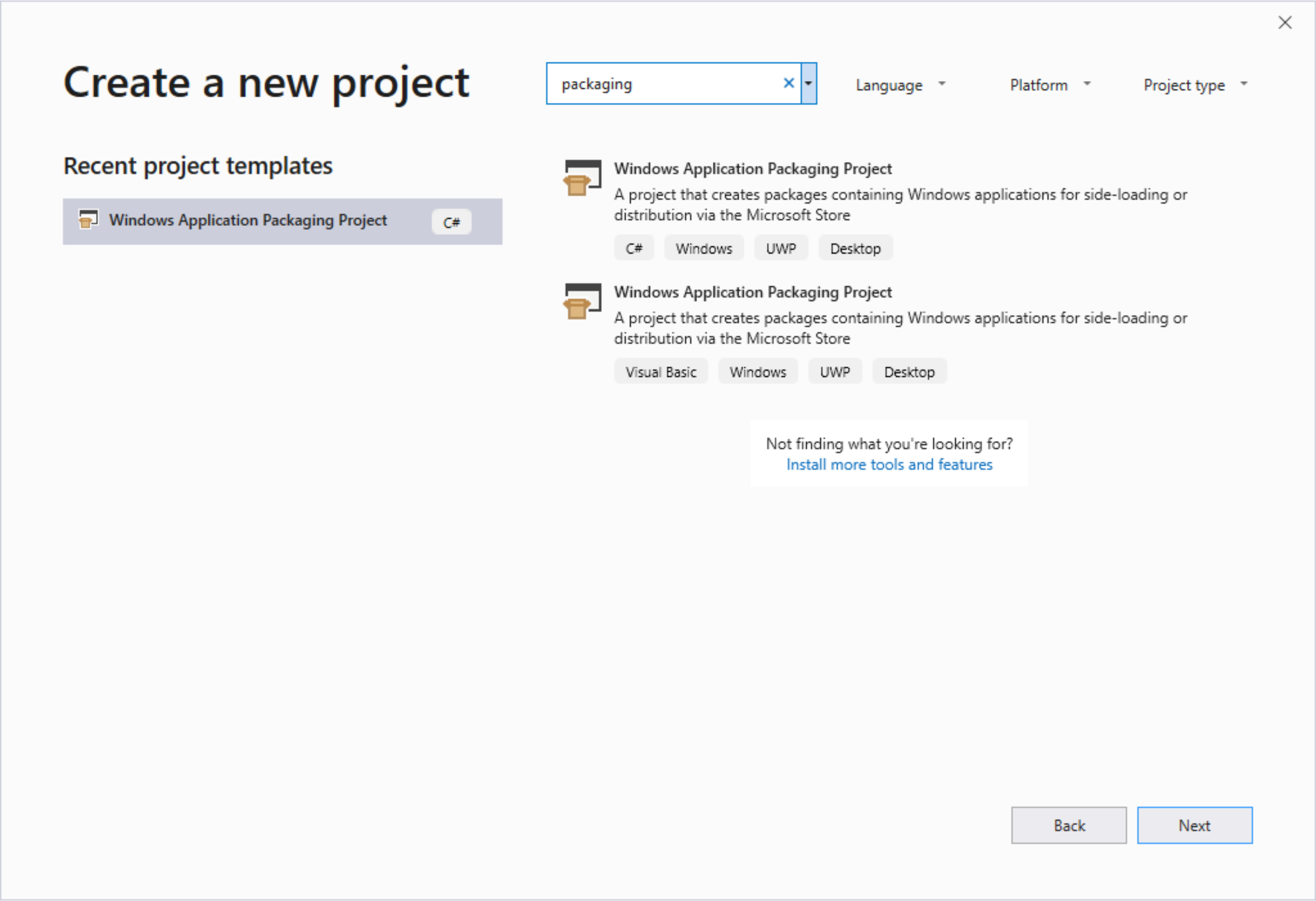
- Declarative install via Manifest file
- Identity, formal versioning and device targeting

Optimized

- Disk space
- Network optimization using differential updates

Windows Application Packaging Project

- Projects included in VS 2017 and VS 2019
- Easily debug packaged WPF, Windows Forms and Visual C++ from the source code
- **New!** Now you can use it to embed any kind of application, even if you don’t own the source
- **New!** Support to .NET Core 3.0 desktop apps



Create a new project

packaging

Recent project templates

Windows Application Packaging Project C#

Windows Application Packaging Project

A project that creates packages containing Windows applications for side-loading or distribution via the Microsoft Store

Windows Application Packaging Project

A project that creates packages containing Windows applications for side-loading or distribution via the Microsoft Store

Not finding what you're looking for?
Install more tools and features

Distribution

Store

- Great for consumer apps
- Signed with MS Cert
- Flights
- Auto Updates

Sideload

- Great for reduced distribution
- Custom signing
- Custom auto-updates
- Great for enterprises

Visual Studio App Center

- Great for testing
- Custom signing
- Manual updates

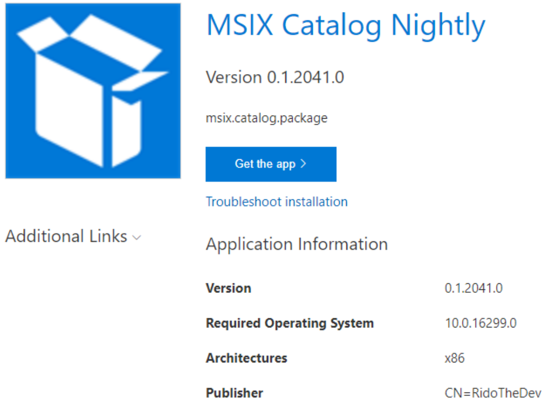
Enterprise

- SSCM
- Intune
- Other MDM tools

Automatic updates with the App Installer File

```xml
<?xml version="1.0" encoding="utf-8"?>
<AppInstaller Uri="https://msix-catalog.azurewebsites.net/AppxPackages/msix.catalog.package.net.appinstaller"
              Version="0.1.1874.0"
              xmlns="http://schemas.microsoft.com/appx/appinstaller/2017/2">
  <MainBundle Name="18656RidoMin.MSIXCatalogNightly"
              Version="0.1.1874.0"
              Publisher="CN=RidoTheDev"
              Uri="https://msix-catalog.azurewebsites.net/[..]/msix.catalog.package.net_0.1.1874.0_x86.appxbundle" />
  <UpdateSettings>
    <OnLaunch HoursBetweenUpdateChecks="0" />
  </UpdateSettings>
</AppInstaller>
```

ms-appinstaller:?source=https://../../app.appinstaller



MSIX Catalog Nightly

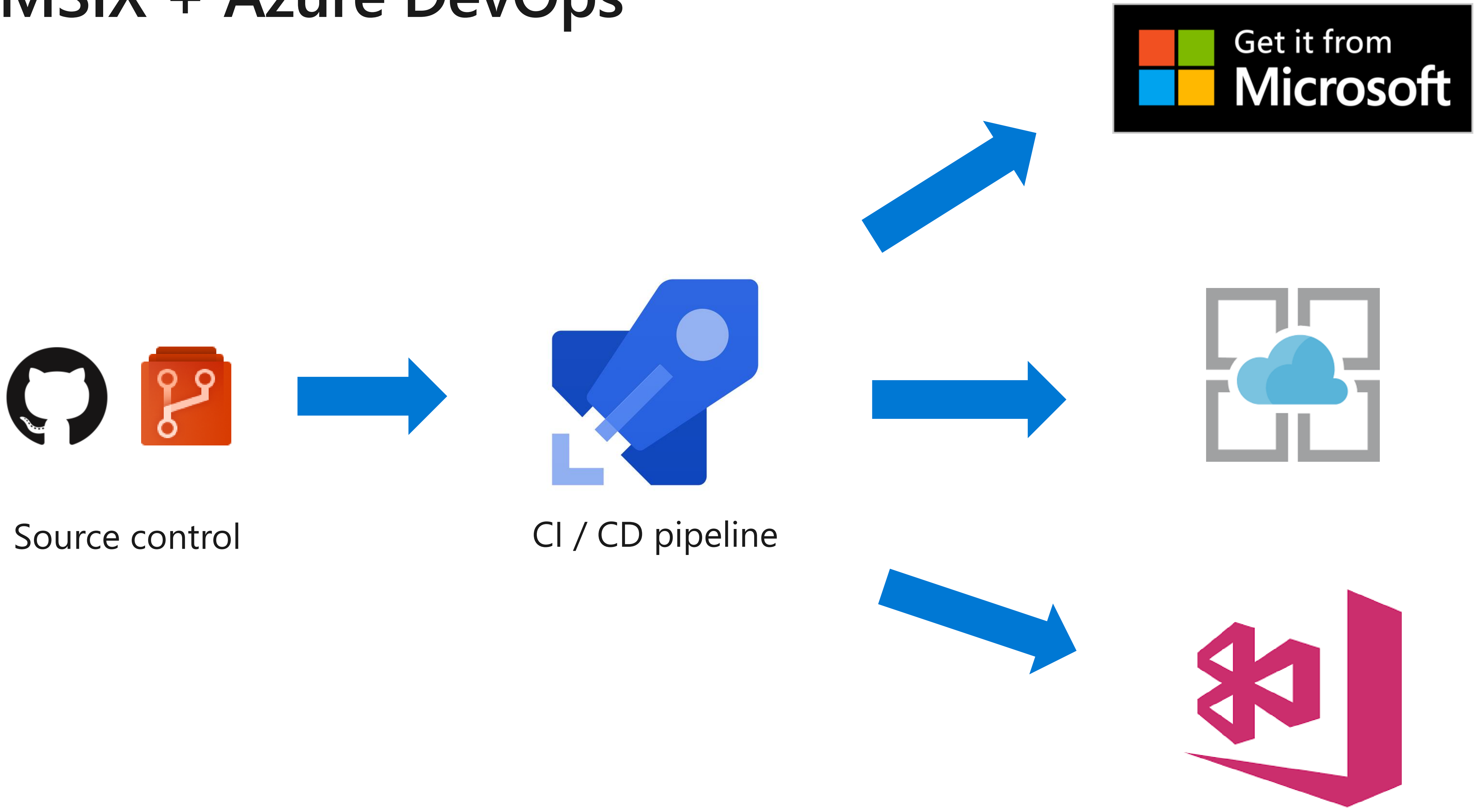
Version 0.1.2041.0

msix.catalog.package

Application Information

Version	0.1.2041.0
Required Operating System	10.0.16299.0
Architectures	x86
Publisher	CN=RidoTheDev

MSIX + Azure DevOps



Source control

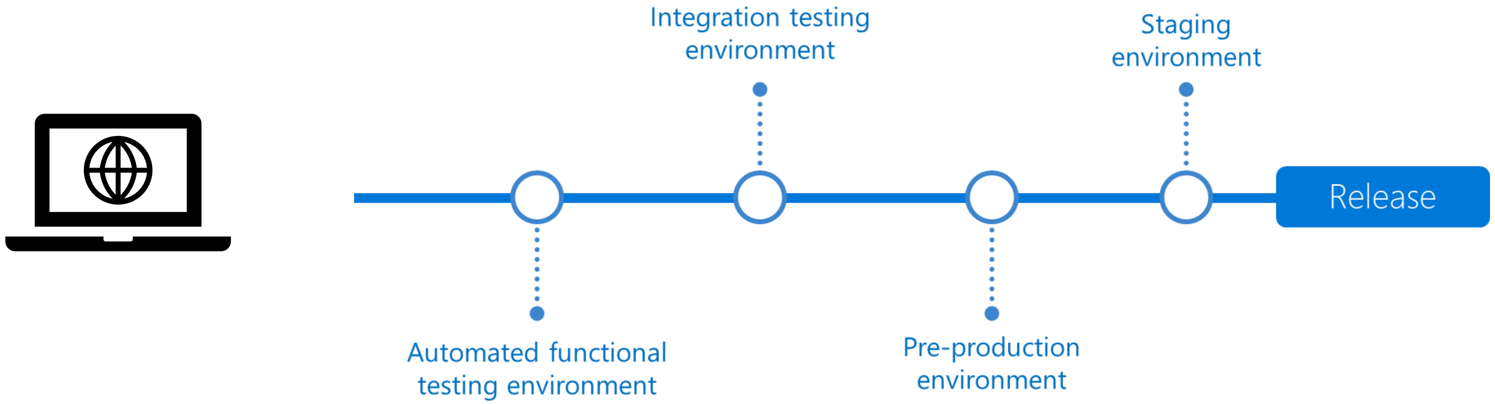
CI / CD pipeline

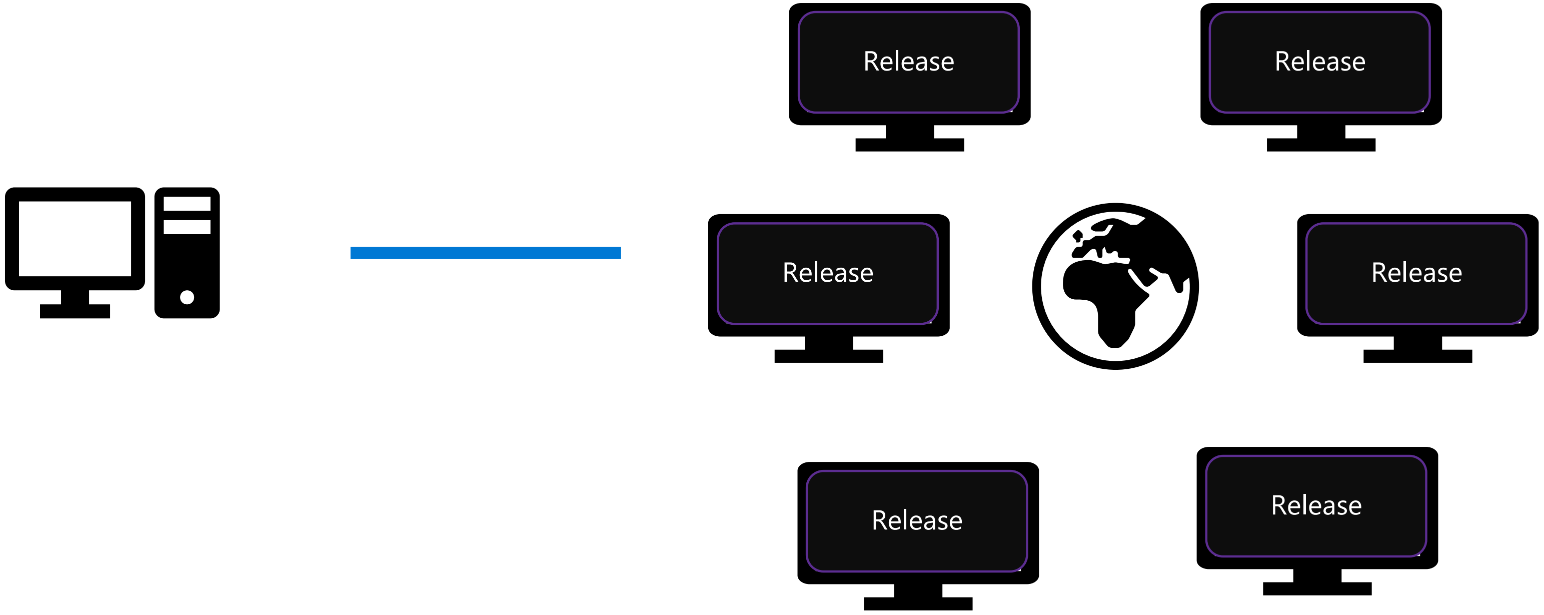
The challenge for desktop apps

101010
010101
101010

Continuous Deployment (CD)

- By combining continuous integration and infrastructure as code (IaC), you'll achieve identical deployments and the confidence to deploy to production at any time.
- With continuous deployment, you can automate the entire process from code commit to production if your CI/CD tests are successful.


Integration testing environment
Staging environment
Release
Automated functional testing environment
Pre-production environment



Demo

MSIX packaging and Azure DevOps

Riferimenti

- Migrating to .NET Core 3.0: https://aka.ms/porting-to-netcore3
- WinUI: https://docs.microsoft.com/en-us/uwp/toolkits/winui/
- MSIX: https://aka.ms/msix
- Windows App Modernization workshop: https://aka.ms/WinAppsWorkshop-Code
- Demos: https://github.com/qmatteoq/CommunityDays2019

Grazie!

- Il materiale sarà online nei prossimi giorni su http://www.communitydays.it