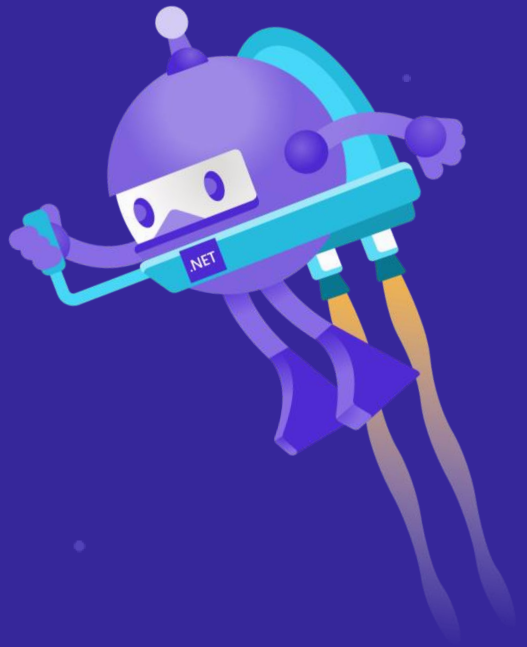
.NET 5.0 in salsa desktop: le novità per gli sviluppatori Windows



Matteo Pagani

Windows App Consult Engineer @ Microsoft

matteo.pagani@microsoft.com

Twitter: @qmatteoq | GitHub: qmatteoq

SPONSOR

Microsoft

avanade

ENGINEERING

blexin.
CON NOI È SEMPLICE

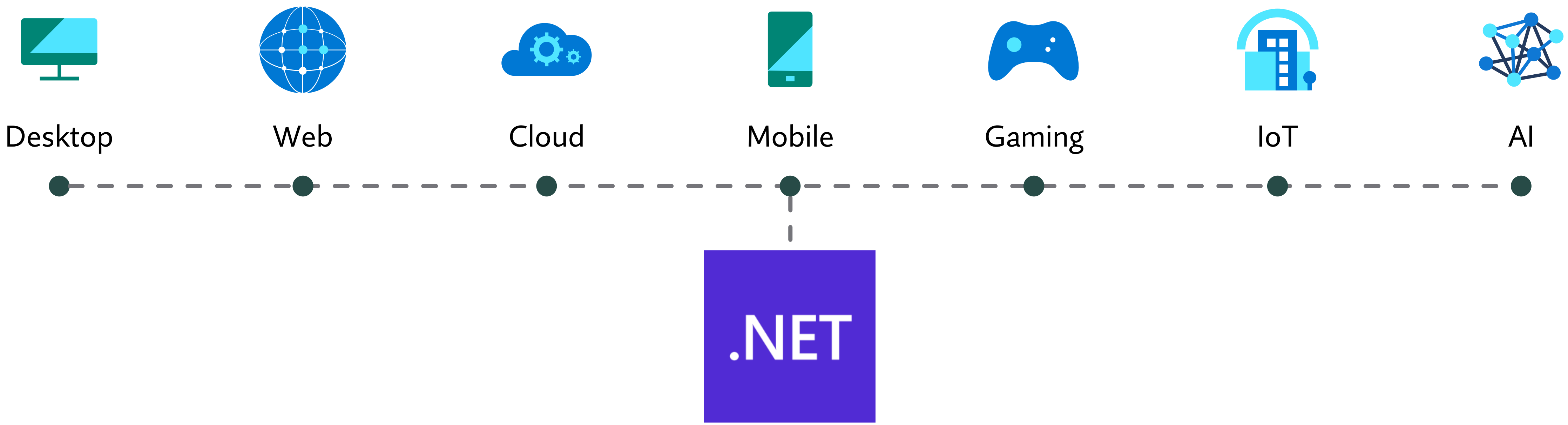
Packt›

PLURALSIGHT

JET BRAINS

.NET

Your platform for building anything



Desktop

Web

Cloud

Mobile

Gaming

IoT

AI

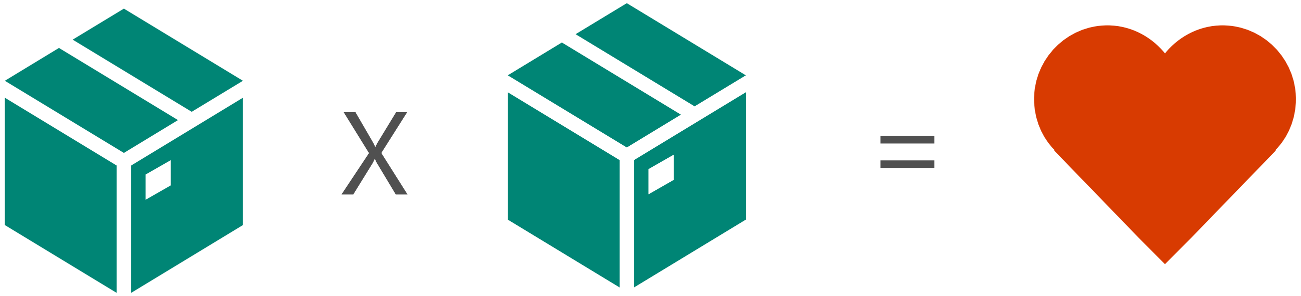
.NET

The future of .NET Framework

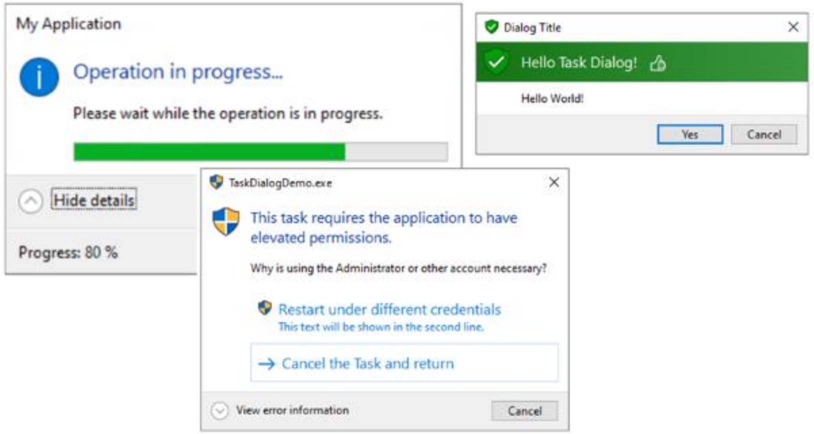
- .NET Framework 4.8 is the last major version of .NET Framework on Windows
- Support policy remains the same:
  - Will always be in Windows
  - Will be patched with Windows
  - Will be supported with Windows
- Keep existing mature applications on .NET Framework
- Recommend .NET Core for new applications or existing applications which will continue to be evolved

Why Windows desktop on .NET 5.0?

- Deployment flexibility
  - Side-by-side support
  - Machine global or app local framework
  - Self-contained EXEs
- Core runtime and API improvements
- Performance



New features in Windows Forms



My Application

Operation in progress...

Please wait while the operation is in progress.

Progress: 80 %

Dialog Title

Hello Task Dialog!

Hello World!

TaskDialogDemo.exe

This task requires the application to have elevated permissions.

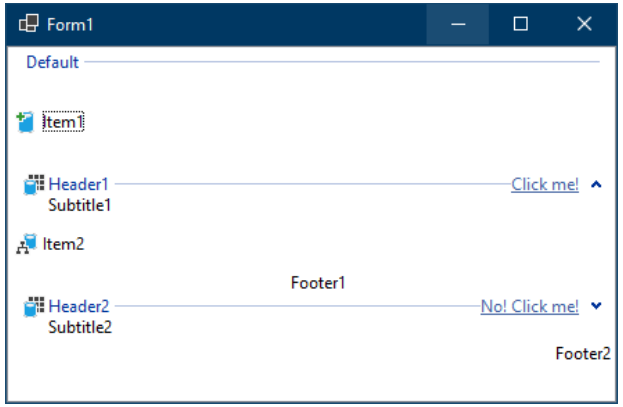
Why is using the Administrator or other account necessary?

Restart under different credentials

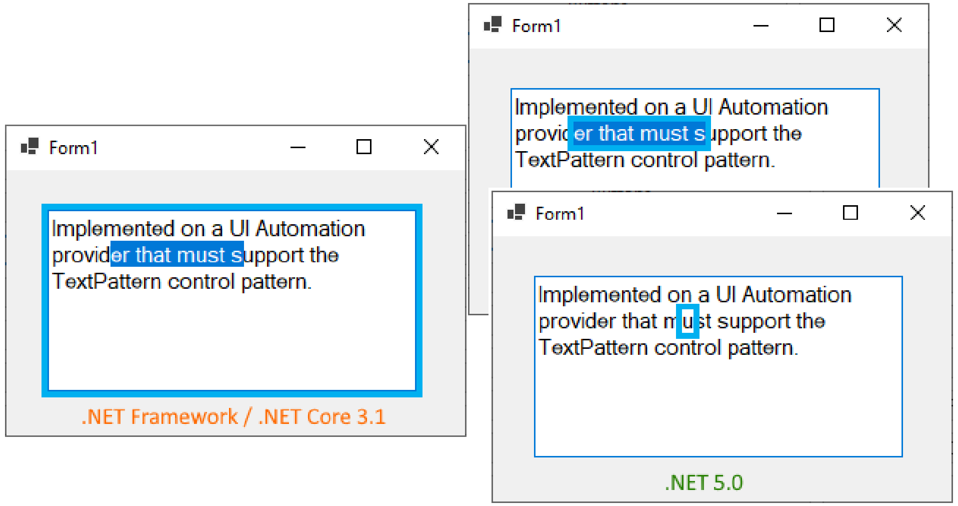
This text will be shown in the second line.

Cancel the Task and return

New TaskDialog control



ListView enhancements



Form1

Implemented on a UI Automation provider that must support the TextPattern control pattern.

.NET Framework / .NET Core 3.1

Form1

Implemented on a UI Automation provider that must support the TextPattern control pattern.

Form1

Implemented on a UI Automation provider that must support the TextPattern control pattern.

.NET 5.0

Accessibility improvements

Microsoft Update

- The problem for IT Administrators: keeping .NET runtime up-to-date with security fixes
- The solution: Microsoft Update will distribute runtime updates
- Easily deployed with SSCM, Intune, etc.
- It will push only minor revisions
- If you have .NET 5.0:
  - You will get 5.0.x
  - You won’t get 5.1 or 6.0

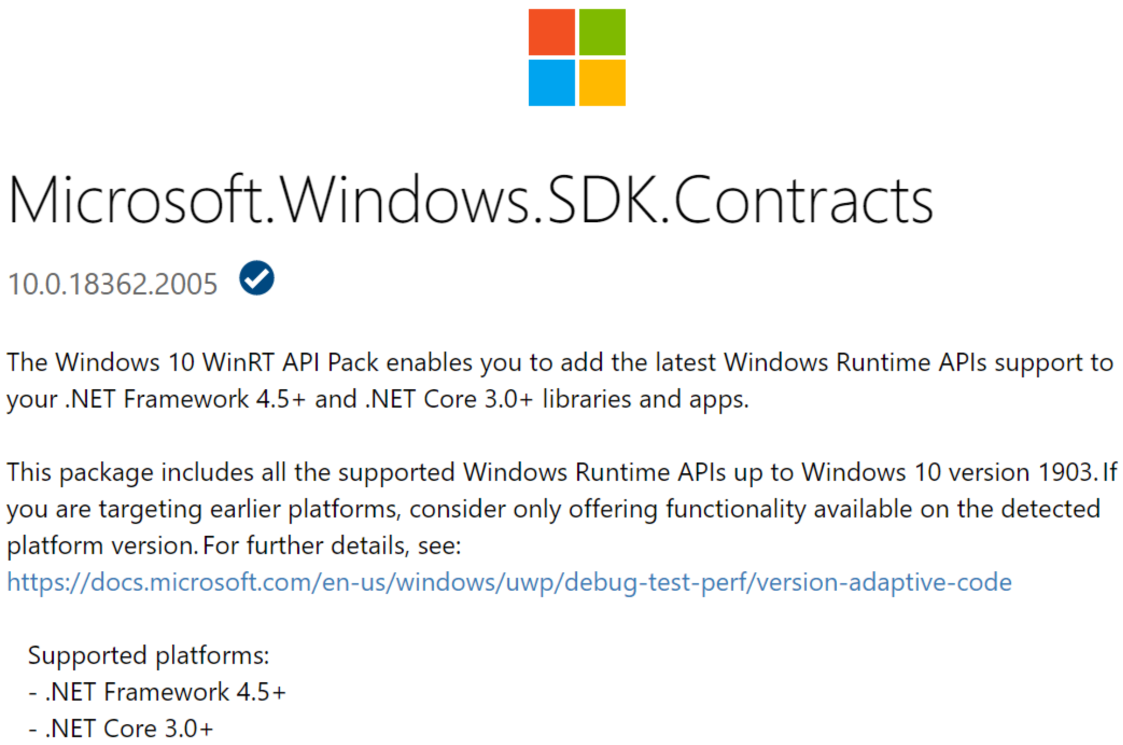
Target framework for Windows apps

```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>WinExe</OutputType>
    <TargetFramework>net5.0-windows</TargetFramework>
    <UseWPF>true</UseWPF>
  </PropertyGroup>

</Project>
```

- Support to Windows 7, 8 and 10
- Access to specific Windows APIs like:
  - Registry
  - Event Viewer
  - WCF services
  - Windows Services

.NET Framework and .NET Core 3.x

- The WinRT projection are included in the compiler
- Package available on NuGet to reference the correct metadata
- Get access to all the UWP APIs from a .NET application



Microsoft.Windows.SDK.Contracts

10.0.18362.2005 ✓

The Windows 10 WinRT API Pack enables you to add the latest Windows Runtime APIs support to your .NET Framework 4.5+ and .NET Core 3.0+ libraries and apps.

This package includes all the supported Windows Runtime APIs up to Windows 10 version 1903. If you are targeting earlier platforms, consider only offering functionality available on the detected platform version. For further details, see: https://docs.microsoft.com/en-us/windows/uwp/debug-test-perf/version-adaptive-code

Supported platforms:
- .NET Framework 4.5+
- .NET Core 3.0+

https://www.nuget.org/packages/Microsoft.Windows.SDK.Contracts

C# / WinRT

- NuGet-packaged toolkit that provides Windows Runtime (WinRT) projection support for the C# language
- It enables .NET 5.0 application to access to Windows Runtime APIs
  - Geolocation
  - Bluetooth
  - Biometric authentication
  - Notifications
  - And much more…
- It enables to consume Windows Runtime components in a .NET 5.0 application

New approach in .NET 5

- It leverages the C# / WinRT project
- The goal is to make .NET completely agnostic, by lifting the WinRT projection support out of the compiler
- No more NuGet packages to reference
- Just target the specific TFM

```xml
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <OutputType>WinExe</OutputType>
    <TargetFramework>net5.0-windows10.0.19041.0</TargetFramework>
    <UseWPF>true</UseWPF>
  </PropertyGroup>

</Project>
```

Supported versions

- 10.0.17763.0
- 10.0.18362.0
- 10.0.19041.0

Using Windows Runtime APIs

Demo

Consuming Windows Runtime components

- You can use C# / WinRT to create a projection library for a Windows Runtime Component
- You can pack this library into a NuGet component or just distribute the DLL
- Use the library from a .NET 5.0 application

```
<PropertyGroup>
  <CsWinRTIncludes>GeolocationComponent</CsWinRTIncludes>
  <CsWinRTGeneratedFilesDir>$(OutDir)</CsWinRTGeneratedFilesDir>
</PropertyGroup>
```

Consuming a Windows Runtime component

Demo

Making apps *great* for the people who use them

- Support new hardware
- Modern user experience
- App deployment and management
- Reliability, security, and privacy
- System performance and battery life

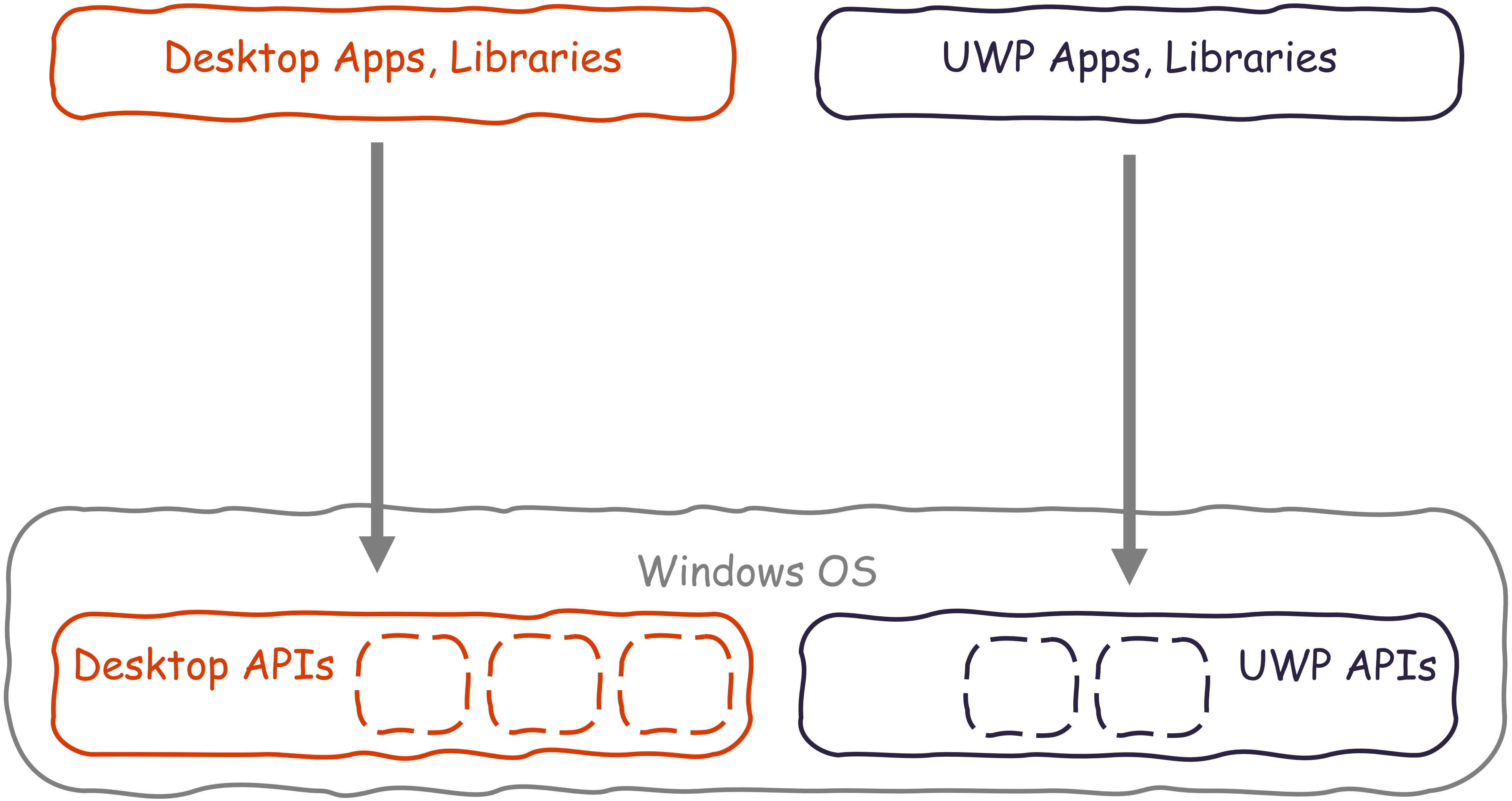
Starting where you are

- Compatibility with all your code – WinForms, WPF, MFC…
- Support for your existing packaging and deployment
- Features that work across users’ versions of Windows 10

101010
010101
101010

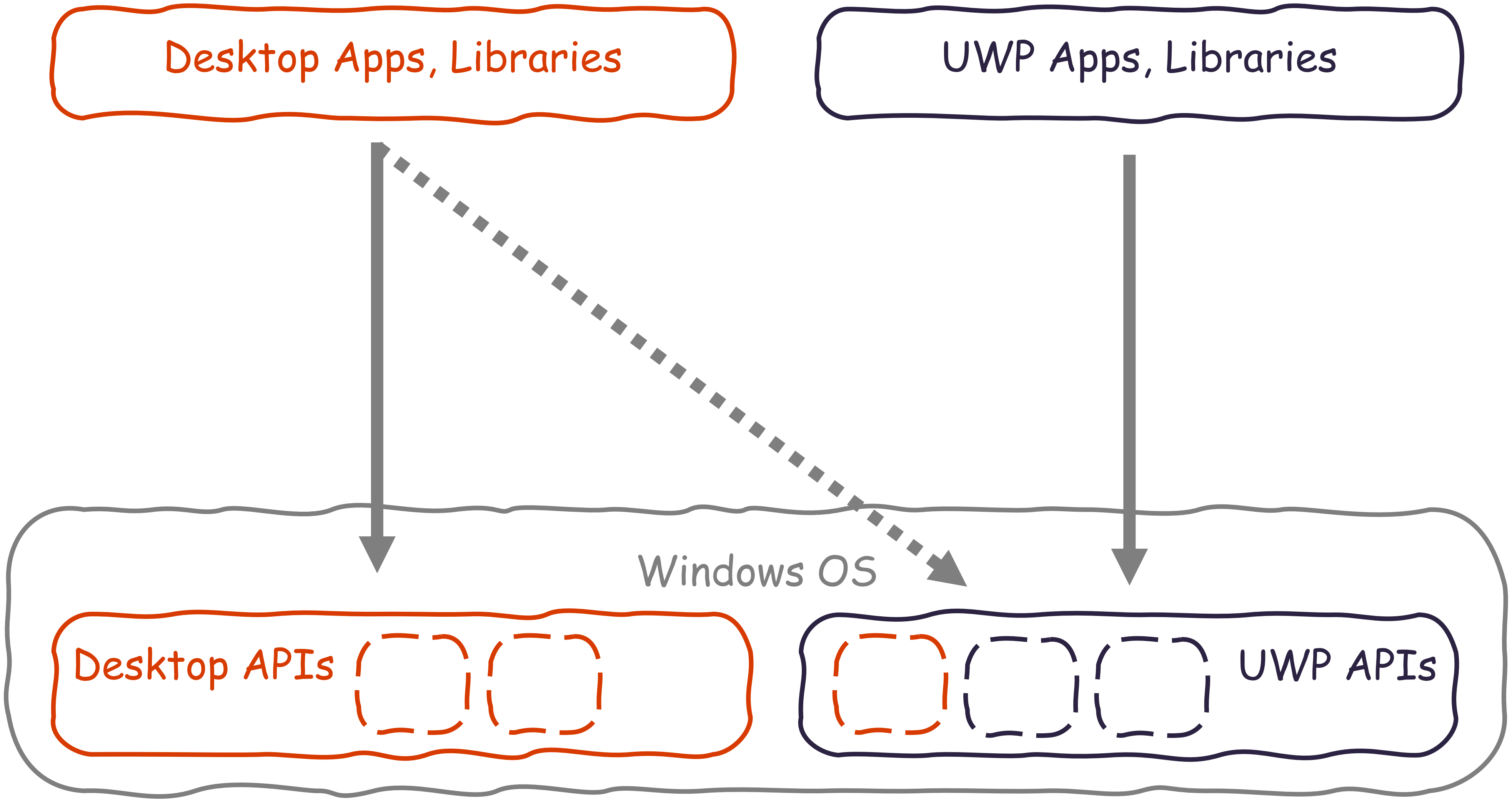
Project Reunion

Today



Desktop Apps, Libraries
UWP Apps, Libraries
Windows OS
Desktop APIs
UWP APIs

Today



Desktop Apps, Libraries

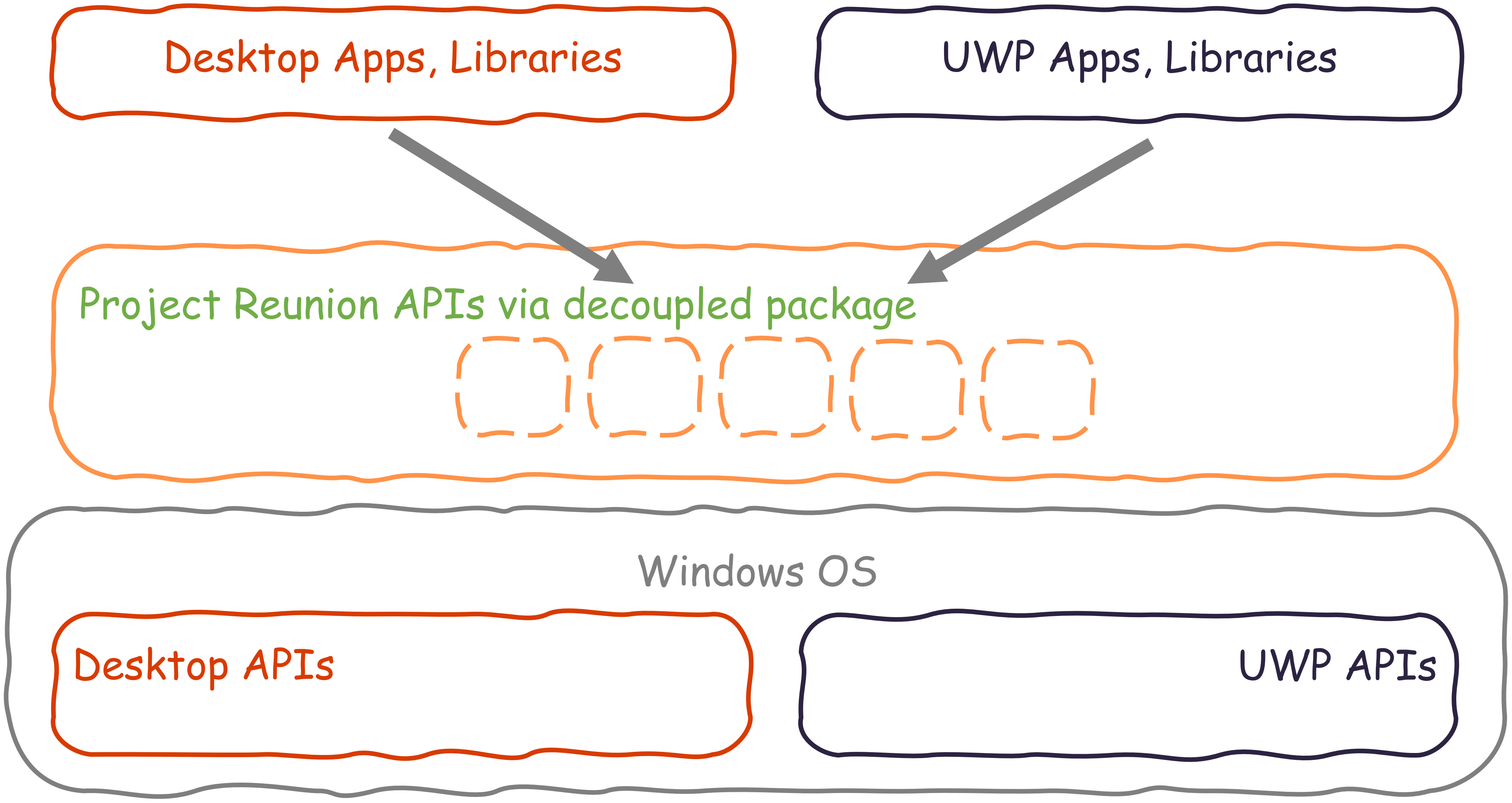
UWP Apps, Libraries

Windows OS

Desktop APIs

UWP APIs

Project Reunion



Desktop Apps, Libraries

UWP Apps, Libraries

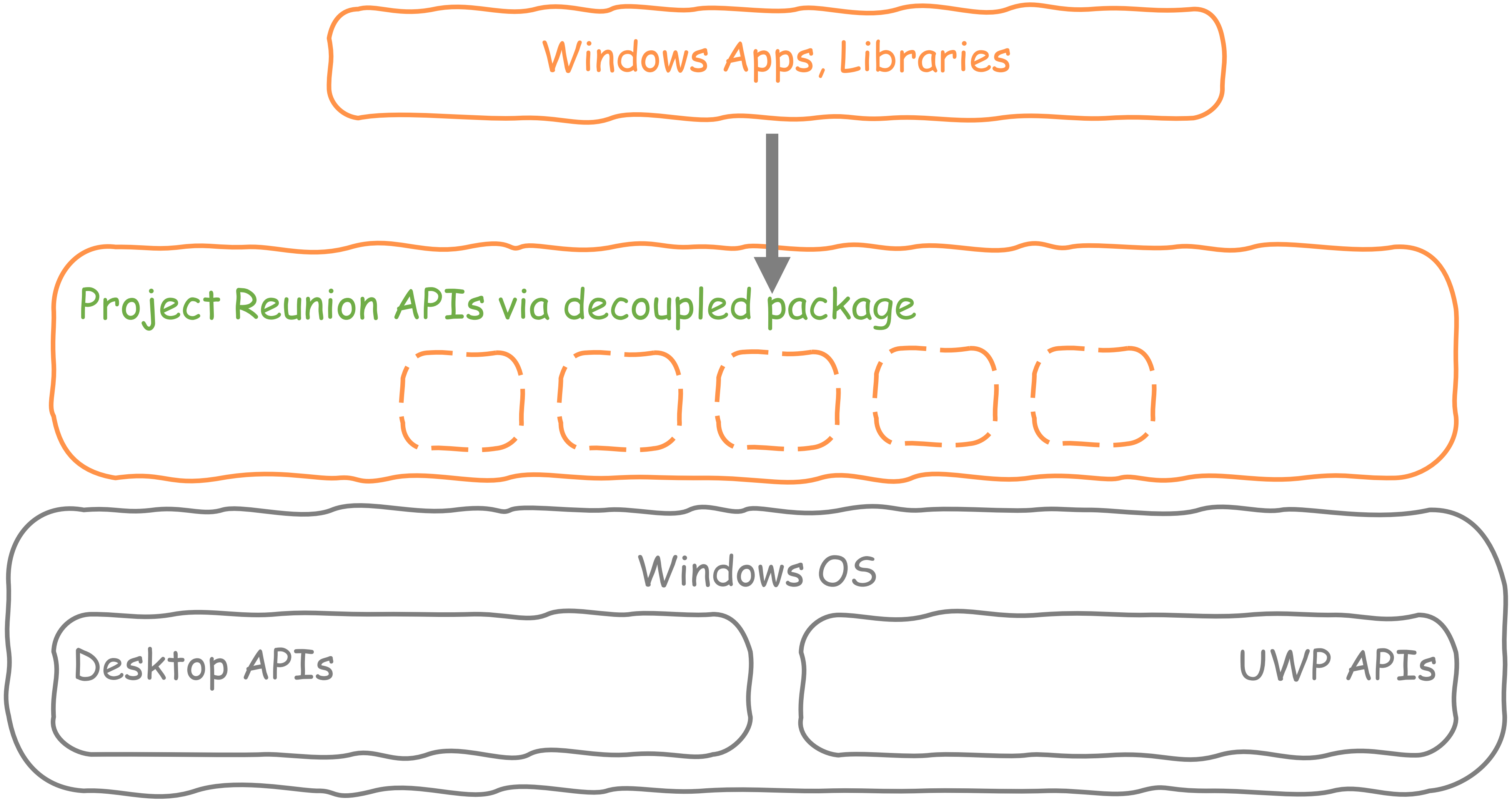
Project Reunion APIs via decoupled package

Windows OS

Desktop APIs

UWP APIs

Project Reunion



Agility through backwards compatibility

New APIs and platform improvements can become instantly available, without waiting for user os updates to catch up.

Web experiences

Supporting web experiences

WebBrowser control in WPF and Windows Forms

Welcome new WebView2 control!

WebView2

- Based on Edge Chromium engine
- Supports WPF, Windows Forms, C++ and UWP
- It’s independent from the Edge browser, but it needs the WebView2 runtime
- Two deployment approaches:
  - Evergreen: the installer takes care of keeping the runtime up-to-date
  - Fixed version: you must manually upgrade to newer versions
- Development and testing phase: use a non-stable Edge version
  - Beta
  - Dev
  - Canary

Demo

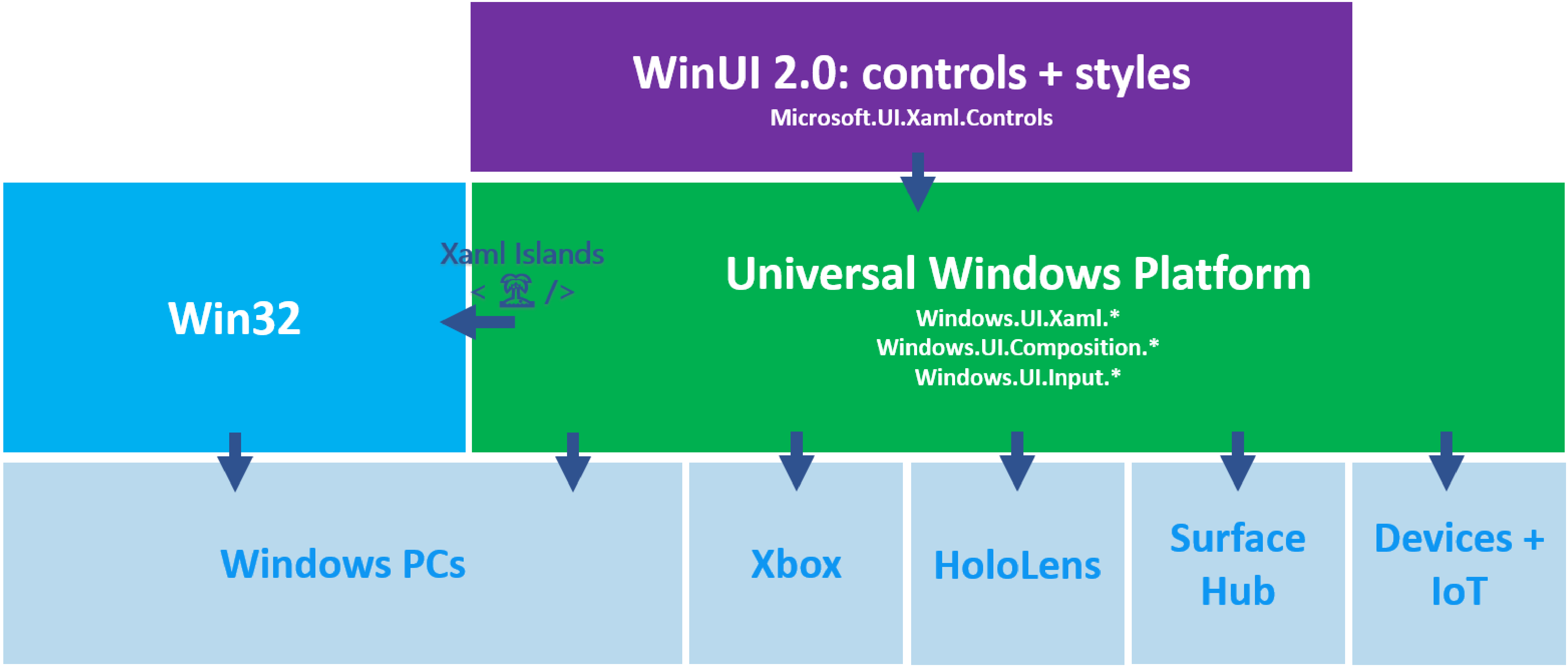
WebView 2

Modern and accessible UI and input

WinUI

- Native UI platform for Windows 10
- It moves controls and resources from the OS to a NuGet package
- It makes easier to leverage UI controls independently from the OS version
- More agility in fixing bugs and adding new features
- Open source on GitHub: https://github.com/microsoft/microsoft-ui-xaml

WinUI 2.x


WinUI 2.0: controls + styles
Microsoft.UI.Xaml.Controls
Xaml Islands
< 🏝 />
Win32
Universal Windows Platform
Windows.UI.Xaml.*
Windows.UI.Composition.*
Windows.UI.Input.*
Windows PCs
Xbox
HoloLens
Surface Hub
Devices + IoT

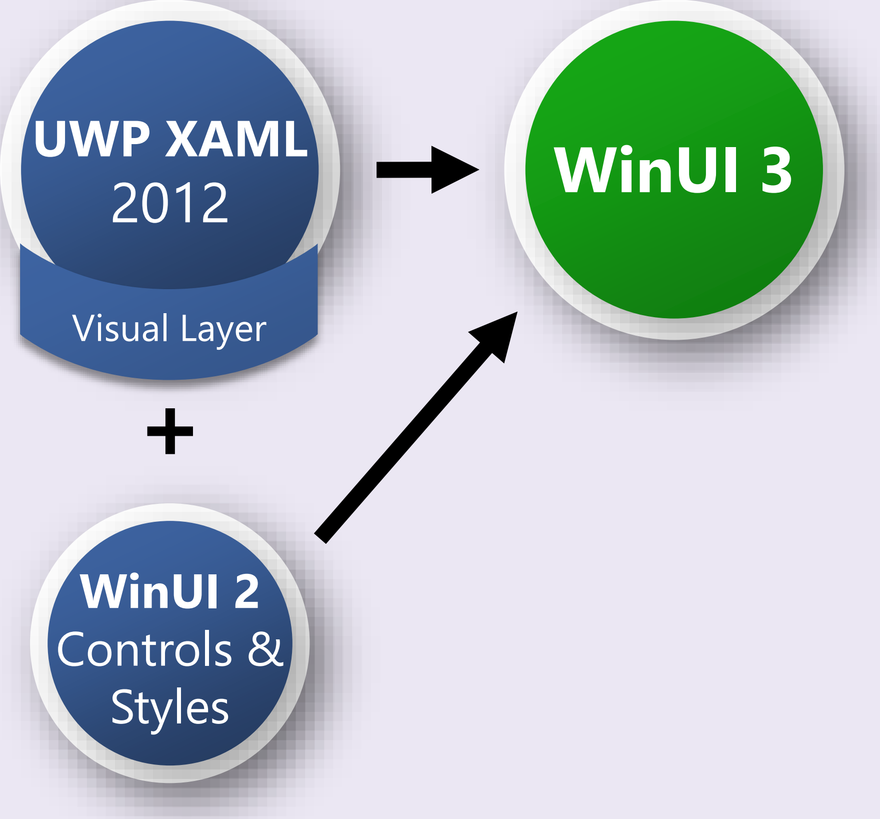
Desktop (Win32) Apps

MFC
1992

WinForms
2002

WPF
2006

UWP Apps



UWP XAML
2012
Visual Layer

+

WinUI 2
Controls &
Styles

WinUI 3

Desktop (Win32) Apps

UWP Apps

WinUI 3 will provide a state-of-the-art UX framework for **every** Windows developer.

Always contains latest Fluent controls & styles

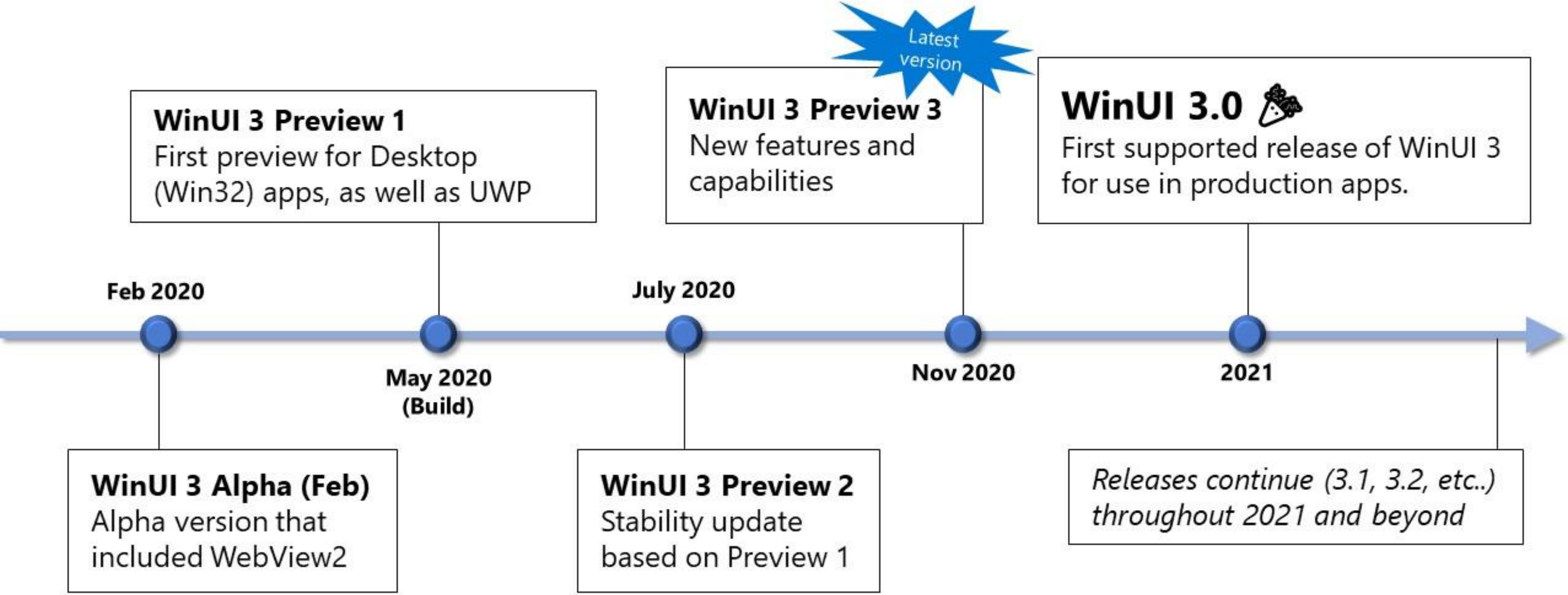
Native UI performance in any app (C++, or .NET)

You choose when to upgrade!

Demo

WinUI 3.0 Preview 3

WinUI 3 Release Roadmap



Feb 2020

WinUI 3 Alpha (Feb)
Alpha version that included WebView2

May 2020 (Build)

WinUI 3 Preview 1
First preview for Desktop (Win32) apps, as well as UWP

July 2020

WinUI 3 Preview 2
Stability update based on Preview 1

Nov 2020

WinUI 3 Preview 3
New features and capabilities

Latest version

2021

WinUI 3.0
First supported release of WinUI 3 for use in production apps.

Releases continue (3.1, 3.2, etc..) throughout 2021 and beyond

ClickOnce is back!

- But I wouldn’t use it ☺
- MSIX is the way forward to deploy Windows applications:
  - Clean install and uninstall
  - Automatic updates
  - Differential updates and storage optimizations
  - Easy integration with CI/CD pipelines
- Add a Windows Application Packaging Project to your solution to package your .NET 5.0 app as MSIX

https://aka.ms/msix

Questions?

https://github.com/qmatteoq/DotNetConf2020-Rome

Ask me about the Windows Developer Enterprise TAP!

Matteo Pagani
Windows App Consult Engineer @ Microsoft
matteo.pagani@microsoft.com
Twitter: @qmatteoq | GitHub: qmatteoq