|  |  |  |
| --- | --- | --- |
| **WPF** | **UWP** | **WinUI** |
| **Підтримка платформ** | | |
| Desktop (win xp and higher) | Windows devices: desktop (win 10 and higher), mobile, xbox. | Desktop (win 10 and higher), з Uno Platform – кросплатформа. |
| **Які МП використовуються** | | |
| C#, VB | C++, C#, VB, JS + CSS + HTML | C#, C++ |
| **Працює поверх** | | |
| Win32 | WinRT | WinRT |
| **Безпечність** | | |
| Обмежень безпечності немає. | Т.я. реалізована підтримка різних windows-пристроїв через сендбокс – є обмеження в безпечності: в доступі до файлів, в використанні додатком реквізитів для входу користувача, девайсів вводу/виводу,. | Обмежень для десктопу немає. |
| **Які IDE підтримують розробку** | | |
| Visual Studio/Rider | Visual Studio | Visual Studio |
| **Графічні компоненти** | | |
| Дизайн графічних компонентів простий, писати сучасний дизайн треба власноруч або використати готові теми. Графічні компоненти прив'язані до фреймворка .Net та/або до дотакових бібліотек. | Дизайн графічних компонентів виглядає сучасно, нативна підтримка різних пристроїв дозволяє досягти адаптивності. З коробки є підтримка світлої/темної теми.  Для підтримки нових графічних компонентів та дизайну необхідно оновлювати Windows. | Сучасний дизайн графічних компонентів. Графічні компоненти не прив'язані до Windows – підтримка оберненої сумісності. |
| **Підтримка пристроїв введення** | | |
| Клавіатура\миша | Клавіатура\миша, тачпад, геймпад, ручка | Клавіатура\миша, тачпад, геймпад, ручка |
| **Розгортування** | | |
| Зазвичай використовують Visual Studio Installer (особисто я тільки його використовував), можна використати MSIX. Можна постачати через інсталер, або через MS store.  Оновляти можна вручну (найпоширеніше) або автоматично через MS store. | Завжди пакуються в APPX або MSIX. Постачають через MS store. Пакувати в .msi або .exe не можна.  Оновлюються автоматично через MS store. | Можна пакувати в .msix, .exe, .msi  Оновлюються або вручну, або автоматично через MS store. |
| **Особливості в роботі з технологією.** | | |
| Є CommandManger, є біндинг до контексту батьківського контролу (Ancestor), Є можливість відслідкувати закриття певного вікна (Window.OnClos[ing, ed]). | Немає біндингу до контексту батьківського контролу (але можна реалізувати). Немає CommandManager. Відсутність мультибіндінгів. | Не працював з нею. |
| **Висновки (імхо)** | | |
| Орієнтована виключно на десктоп і для нього підходить найкраще. Просто працювати з графікою. Підходить якщо треба написати GUI для старіших версій windows. На мою думку на WPF розробляють здебільшого програми орієнтовані на ентерпрайз або на відносно невелику групу користувачів. Але це не відміняє факту що можна розробити і програму «загального призначення» але через старість платформи потрібно буде використовувати новіші стилі/теми. Питання лише в тому, чи буде це програмою загального призначення – оскільки підтримуватися вона буде лише на ПК з Windows. | Новіші стилі та теми – через це і те, що розробляти можна не тільки під Windows, і те, що додаток просто розгортати і оновляти, підходить для написання додатків загального призначення краще ніж WPF. Але, якщо постає завдання розробити якись застосунок для фірми – через урізаність UWP можливо буде важко втілити всі вимоги до додатку. | Найркаще підходить для написання додатків загального призначення (завдяки оберненій сумісності для десктопних додатків і завдяки Uno platform для кросплатформенних додатків). І якщо постає завдання розробити з нуля сучасного вигляду ентерпрайз додаток під Windows (10+) – підійде краще ніж WPF, оскільки так само як на WPF WinUI надає повний доступ до системи (без використання сендбоксу).  Але якщо розробляти додаток треба не з нуля (наприклад взяти код із іншого .Net framework додатку), або з використанням написаних на .Net framework бібліотек – тоді WPF або UWP в залежності від типу і складності додатку. |

Матеріали що використав:

По темі:

[A deep-dive into WinUI 3 in desktop apps - Windows Developer Blog](https://blogs.windows.com/windowsdeveloper/2020/07/07/a-deep-dive-into-winui-3-in-desktop-apps/)

[WinUI vs WPF, WinForms, UWP, and MFC | by MESCIUS inc. | MESCIUS inc. | Medium](https://medium.com/mesciusinc/winui-vs-wpf-winforms-uwp-and-mfc-f65bab0cc618)

[Choosing the Right UI Framework for Native Windows Applications -- Visual Studio Magazine](https://visualstudiomagazine.com/articles/2024/02/13/desktop-dev.aspx)

[Building Modern Desktop Apps—Is WinUI 3.0 the Way to Go? (telerik.com)](https://www.telerik.com/blogs/building-modern-performant-desktop-apps-winui-30-the-way-to-go)

По суміжним темам:

[Windows Runtime - Wikipedia](https://en.wikipedia.org/wiki/Windows_Runtime)

[What is Win32? (computerhope.com)](https://www.computerhope.com/jargon/w/win32.htm)

[What is MSIX? - MSIX | Microsoft Learn](https://learn.microsoft.com/en-us/windows/msix/overview)