TECHNOLOGIES C# .NET POUR DÉPLOYER DES IHM

Objet de ce document: Explorer les différentes technologies pour déployer des applications avec IHM sur les plateformes Windows, Linux et Androïd.

Terminologie
Les plateformes sont les O.S qui supporteront l'application. l'IHM est appelée U.I (Interface Utilisateur).

Les Plateformes

Nous distinguerons 2 types de plateforme. Les plateformes desktop et mobile. En pratique, les plateformes sont différentes selon le matériel utilisé. L'architecture et les composants matériels sont différents (CPUs, écrans, entrées, sorties, capteurs embarqués…). L'intégration et les services à rendre peuvent énormément différer.

Plateformes Desktop

- Windows 10
- Linux
- MacOS

Plateformes Mobiles

- Androïd
- IOS
- Universal Windows Platform
- Windows Mobile

C# Et .NET

Windows offre la technologie .NET permettant de développer des applications "Cross Platform" (Windows, Linux, Androïd, IOS), facilement portable, intégrant de nombreuses librairies et exécutant du code managé. Ce code peut-être écrit avec des langages implémentant la Common Language Infrastructure (CLI). C# fait parti de ces langages.

Implémentations .NET

Références: https://docs.microsoft.com/en-us/dotnet/standard/components

.NET Standart n'est pas une implémentation de .NET mais une spécification d'API disponibles sur toutes les implémentations .NET. Autrement dit, tout code utilisant les API de .NET Standart pourra être utilisé par n'importe quelle implémentation .NET.
Pour en savoir plus sur .NET Standard: https://docs.microsoft.com/fr-fr/dotnet/standard/net-standard

Une implémentation de .NET comprend l'API Standart auxquelles s'ajoutent d'autres API, ainsi que des frameworks (ASP.NET, Windows Forms, WPF…). Une implémentation comprend aussi un "runtime", c'est à dire l'environnement d'exécution qui va permettre d'éxécuter le code.

Les implémentations de .NET sont les suivantes:
- .Net Framework
- .NET Core avec pour runtime CoreCLR
- Mono avec plusieurs runtimes: Xamarin.iOS, Xamarin.Android, Xamarin.Mac et le framework de bureau Mono
- Universal Windows Platform avec pour Runtime .NET Native

.NET Framework

Références: https://docs.microsoft.com/fr-fr/dotnet/framework/index
C'est le .NET historique (2002) et sa dernière version est la 4.7. Il intègre de nombreuses API et se destine pour le développement d'application Desktop. Il intègre notamment les API suivantes: ASP.NET 4.x, Windows Forms et Windows Presentation Foundation (WPF).
Son runtime est CLR (Common Language Runtime)

.NET CORE

Références: https://docs.microsoft.com/fr-fr/dotnet/core/
Ce .NET est open source, Cross-Platform et s'exécute sur Windows, macOS et Linux. Il intègre les API ASP.NET Core, Windows Forms et WPF.
Le Runtime est CoreCLR qui peut aussi tourner sur Linux. Cela ne veut pas dire que toutes les API .NET Core peuvent tourner sous Linux. Par exemple Windows Presentation Foundation (WPF) est une API pour gérer **nativement** l'UI Windows et n'est donc pas portable sous Linux.

Mono

Références: https://www.mono-project.com/docs/
Mono est open source, supporte plusieurs plateformes (Windows, Linux, Androïd, etc…) et implémente les API de .Net Framework avec quelques restrictions:
Mono supports everything in .NET 4.7 except WPF, WWF, and with limited WCF and limited ASP.NET async stack.
Donc pas de WPF ici non plus. Par contre il supporte les Windows Forms et GTK#.
Concernant les runtimes, mono en a plusieurs pour les diverses plateformes:
- Linux, BSD, Mac OS, windows… runtime mono sur https://www.mono-project.com
- Androïd, IOS (les runtimes respectifs sont xamarin.android/, xamarin.ios

Le runtime xamarin.android expose le SDK Android aux appels des procédures .NET.

Universal Windows Platform (UWP)

UWP cible tout type de matériels (PC, tablette, mobiles, Xbox, objets connectés). Il est plutôt dédié à l'embarqué et permet de déployer les applications au travers d'un app store. Il propose une API universelle pour toutes les plateformes windows 10 (Windows 10 pour Desktop, Mobile, IoT, XBox, etc…).
Son runtime n'est pas une machine universelle mais une chaine de compilation produisant des binaires Windows.
Donc UWP est universel dans le monde Windows 10 uniquement.

Les U.I

Chaque implémentation de .Net intègre des API pour les Interfaces Utilisateurs. Commençons par les interfaces communes à toutes les implémentations, c'est à dire celles comprises dans la spécification .NET Standard. Il n'y a que l'API System.Drawing qui donne accés aux fonctionalités de base de GDI+, le système d'affichage Windows.
Autant dire qu'il n'y a pas d'UI digne de ce nom qui est commun aux 4 implémentations.

Quelques U.I

Windows Forms

Références: https://docs.microsoft.com/en-us/dotnet/framework/winforms/
Windows Forms est disponible dans les implémentations .NET Core, .NET Framework, MONO.
Par exemple, on peut le trouver dans la bibliothèque de base de .NET Framework 4.8:
https://docs.microsoft.com/fr-fr/dotnet/api/?view=netframework-4.8
Il peut s'exécuter dans les runtimes CoreCLR, CLR et Mono. les runtimes CoreCLR et Mono peuvent s'exécuter sur Windows, Linux et MacOS.
Conclusion:
Windows Forms peut être utilisé sur Linux avec le Runtime Mono ou bien CoreCLR.

Il n’est pas utilisable avec les runtimes Xamarin.android ou Xamarin.Ios

WPF

Références: https://docs.microsoft.com/en-us/dotnet/framework/wpf/getting-started/
WPF est disponible dans les implémentations .NET Core et .NET Framework et donc sur la plateforme Windows.
Même s’il est disponible sur Windows Core et donc potentiellement exécutable par CoreCLR sur Linux, il s’agit d’une API pour gérer nativement l’UI Windows. Mono (pas le runtime mais l’implémentation) n’implémente pas WPF pour Linux.
Conclusion:
WPF n’est disponible que pour la plateforme Windows avec les runtimes CLR et CoreCLR.

Xamarins Forms

Références:
- Pour une présentation de Xamarin: https://docs.microsoft.com/fr-fr/xamarin/get-started/what-is-xamarin
Xamarin Forms est une API d’Interface Utilisateur multiplateforme pour les plateformes suivantes: Androïd, IOS et UWP. Elle fait partie de l’API Xamarin supplémentaire à .NET Framework et son espace de nom racine est Xamarin. Cette API mappe des fonctions natives aux plateformes sur lesquelles elle s’exécutent.
Celle API complémente l’API Mono (qui est elle-même une implémentation de .NET Framework) et s’exécute dans un runtime Xamarin.Android ou Xamarin.IOS, ou encore peut être compilé par .Net Native pour UWP. Sous Androïd, Xamarin.Android wrappe des fonctions exposés par le SDK Java et les expose au travers de sa propre API Xamarin. Pour en savoir plus sur cette architecture: https://docs.microsoft.com/fr-fr/xamarin/android/internals/architecture

Conclusion:
Xamarin Forms n’est pas disponible sous Linux. Cette API, bien qu’elle s’exécute dans un environnement Mono (en complément de l’implémentation libre de .NET Framework) ne wrappe pas d’U.I Linux native. Toutefois, un projet (statut : en développement) de portage des Xamarins Forms sur GTK# existe ici: https://github.com/jsuarezruiz/forms-gtk-progress/blob/master/Status.md

GTK#

Références: https://www.mono-project.com/docs/gui/gtksharp/
GTK# est une API pour U.I reposant sur Mono et .NET. Il est utilisable sur Windows, Linux et MacOS

Conclusion

Il n’y a pas de plateforme “universelle” pour déployer des applications avec un code pour l’U.I commun aux plateformes Windows, Linux et Andoïd.

- Windows Forms: Windows et Linux
- WPF: Windows
- Xamarin Forms: Windows, Androïd, IOS.

Par contre, l’évolution de .NET Core 3 se fera directement vers .NET Core 5 en novembre 2020 et sera réellement multiplateforme: Il fera tourner les runtimes CoreCLR et Mono et intègrera de nombreuses APIs (Windows Forms, WPF, UWP, ASP.NET MVC, Entity Framework, LINQ...)