TP .Net Maui n° 1 : Utilisation de l'API système

Objectif : Utiliser les fonctionnalités accessibles des APIs Système à travers .Net Maui

Partie 1 : Création du projet et tests

Lancez « Visual Studio » et créez un nouveau projet de type « Application .NET MAUI ».

Lancez l'exécution en sélectionnant « Windows Machine ». Testez l'application.

Allez sous « Windows Machine » et faites « Gestionnaire d'appareils Android », faites « Nouveauté », sélectionnez « Pixel 2 (+Store), x86, Android 9 - API28 » et laissez les autres paramètres par défaut, faites ensuite « Créer ».
Démarrez votre émulateur.
Revenez à Visual Studio et lancez votre application sur votre émulateur.

Ouvrez le fichier « App.xaml et App.xaml.cs » de quel code s'agit-il ? :

Démarrage de l'application, chargement des couleurs et des styles, renvoie à AppShel

Ouvrez le fichier « AppShell.xaml et AppShell.xaml.cs » de quel code s'agit-il ? :

Du code de l'organisation visuelle de l'application (des fois appelé routeur graphique)

Ouvrez le fichier « MainPage.xaml et MainPage.xaml.cs » de quel code s'agit-il ? :

Page principale actuelle de l'application

Allez dans « Ressources/Styles » et ouvrez le fichier « Colors.xaml », que contient-il ? :

Les couleurs xaml de l'application

Modifiez la couleur Primary à #005067 et la couleur Secondary à #E84D0D. Ceux sont les couleurs officielles de la charte de 3iL.

Allez dans « Ressources/Styles » et ouvrez le fichier « Styles.xaml », que contient-il ? :

Les styles par défaut des contrôles de l'application

Démarrez votre application en mode « Windows Machine ». Lorsqu'elle est affichée, revenez à Visual Studio et modifiez le texte du premier Label (fichier MainPage.xaml). Que pouvez constater ? :

Le texte se modifie directement sur l'application, c'est de rechargement à chaud

Partie 2 : Création des premières pages

Que fait d'après vous le « VerticalStackLayout » ? :

contrôle de disposition qui organise ses enfants verticalement