Module .NET

UFR Sciences et Techniques

Université de Rouen

Master 2 GIL

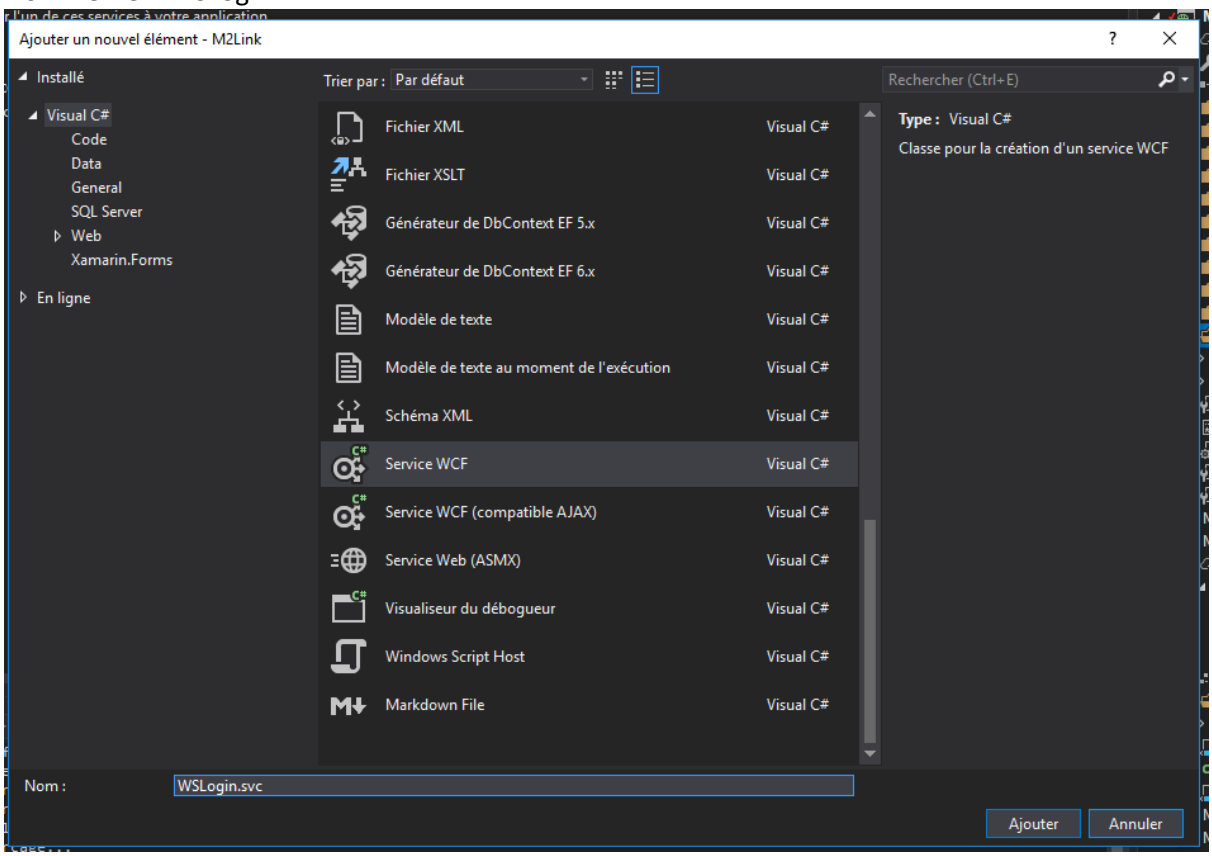
FTEL

TP3 & TP4

Durée 3h & 4h

Création de Service Web « WSLogin »

1. Créez un dossier « WebServices »
2. Clic droit, Ajouter, Nouvel élément
3. Trouvez « Service WCF »
4. Nommez le « WSLogin »



Testez votre Service Web via Navigateur

1. Testez la méthode « HelloWorld » auto-générée
    a. /WebServices/WSLogin.svc

Testez votre Service Web via WcfTestClient.exe

1. Lancez WcfTestClient.exe
    a. C:\Program Files (x86)\Microsoft Visual Studio\2017\Community\Common7\IDE\WcfTestClient.exe
2. Ajoutez un service en renseignant l'adresse vers votre Service Web
    a. /WebServices/WSLogin.svc
3. Testez la méthode « HelloWorld »

Création de votre méthode « Validate »

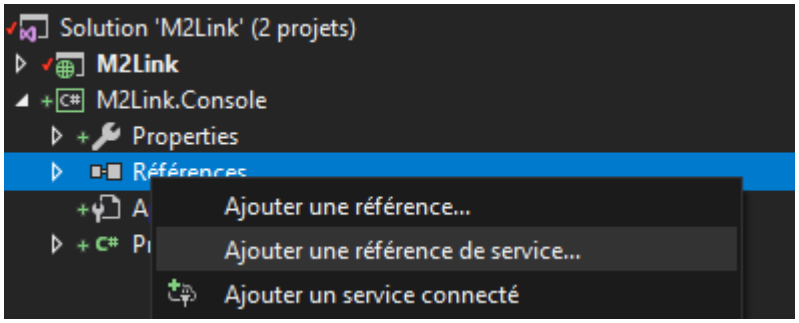
1. Créez dans votre WebService une méthode Validate qui prend 2 paramètres : login / mot de passe
2. Faîtes en sorte qu'elle retourne « null » si le login / mot de passe est invalide
3. Faîtes en sorte qu'elle retourne un model User si le login / mot de passe est valide
4. Vérifiez que votre méthode fonctionne correctement avec WcfTestClient.exe

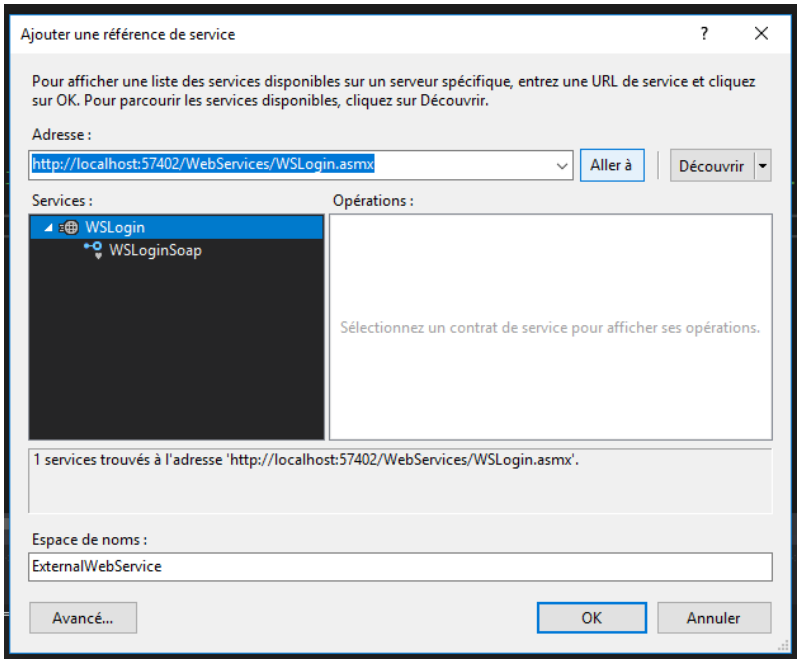
Création du Service Web « WSMessage »

1. Créez un Service Web « WSMessage »
2. Créez une méthode pour qu'un utilisateur récupère l'ensemble des messages des utilisateurs qu'il suit
3. Créez une méthode pour poster un nouveau message
4. Vérifiez que tout méthode fonctionne correctement avec WcfTestClient.exe

Utilisation des Services Web dans un projet Console

1. Avant de commencer, si vous utilisez IIS Express, il faut que votre application Web soit en train de tourner en arrière-plan.
    a. Pour cela, Fichier -> Afficher dans le navigateur
    b. Puis vérifiez que vos Services Web sont bien accessibles sans être en debug
2. Dans votre solution, ajoutez un nouveau projet « Application Console (.Net Framework) » nommé « M2Link.Console »
3. Ajoutez deux fois une référence de service vers vos deux Service Web

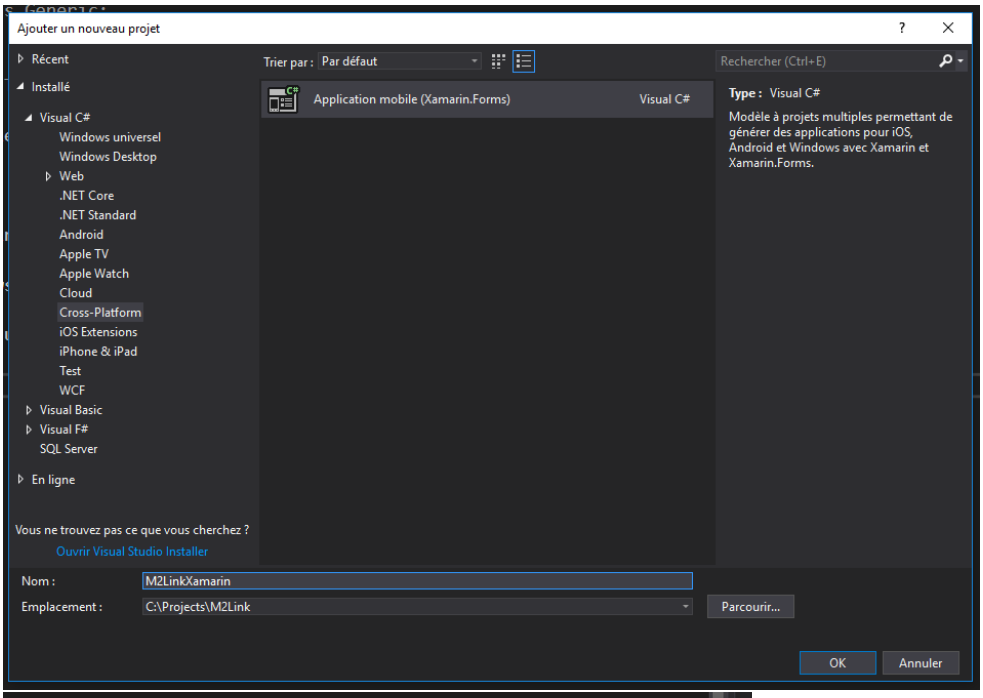


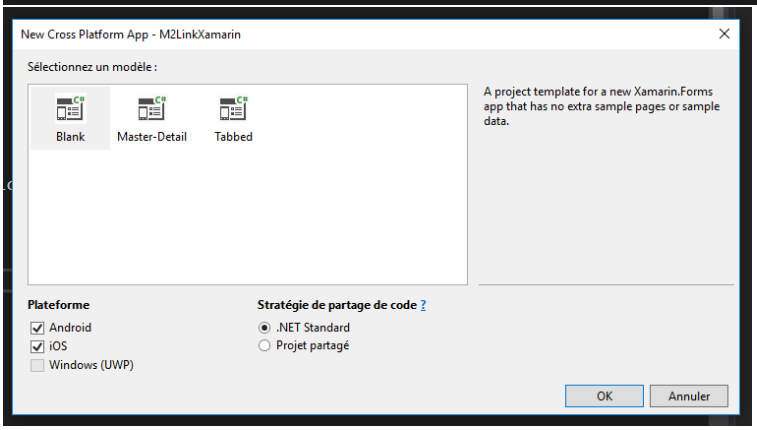


    a. Renseignez « ExternalWebService » dans l'espace de noms
4. Appelez vos méthodes de Service Web dans la fonction Main
5. Si vous avez la possibilité, mettez-vous en binôme afin de consommer le WebService de votre binôme en faisant la même manipulation

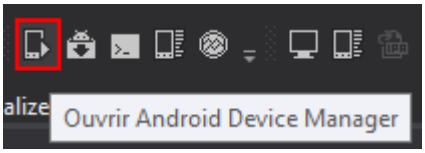
Création de vos projets Xamarin

5. Si ce n'est pas déjà fait, installez Xamarin avec l'installateur de Visual Studio
6. Créez un nouveau projet C# > Cross-Platform > Application mobile (Xamarin.Forms)
    a. Nommez le « M2LinkXamarin »

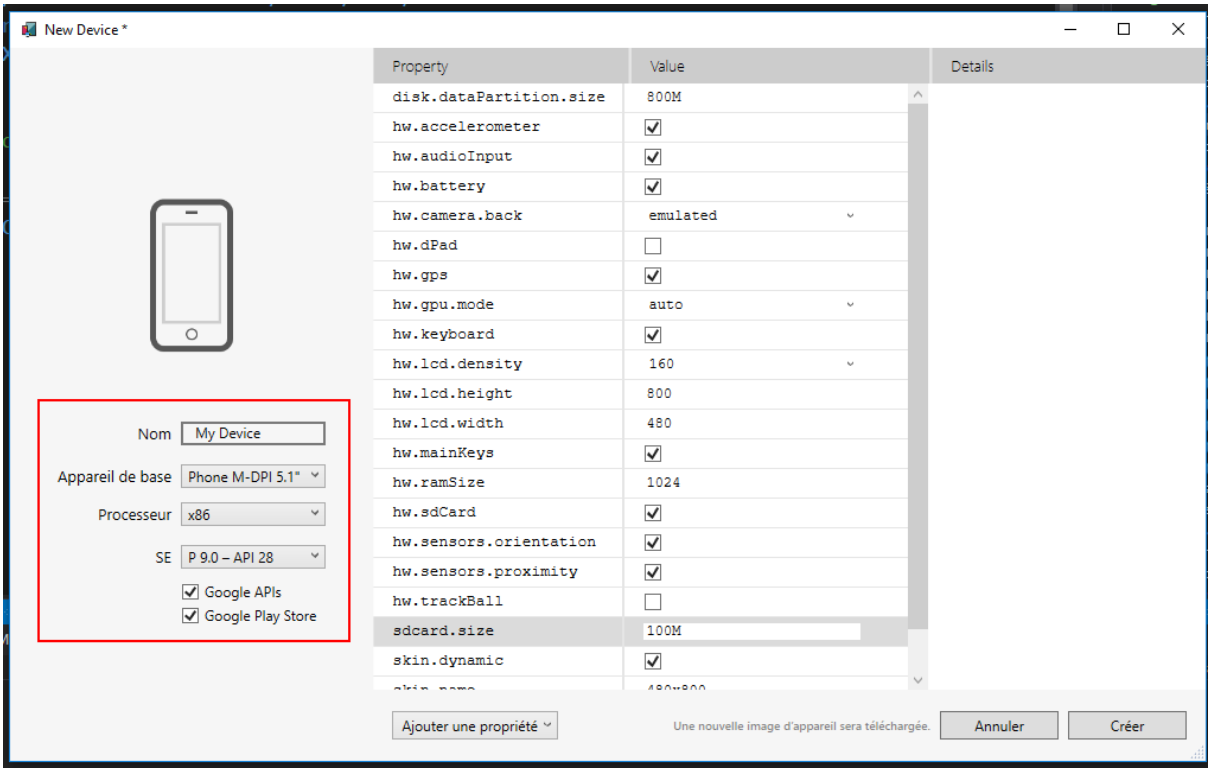




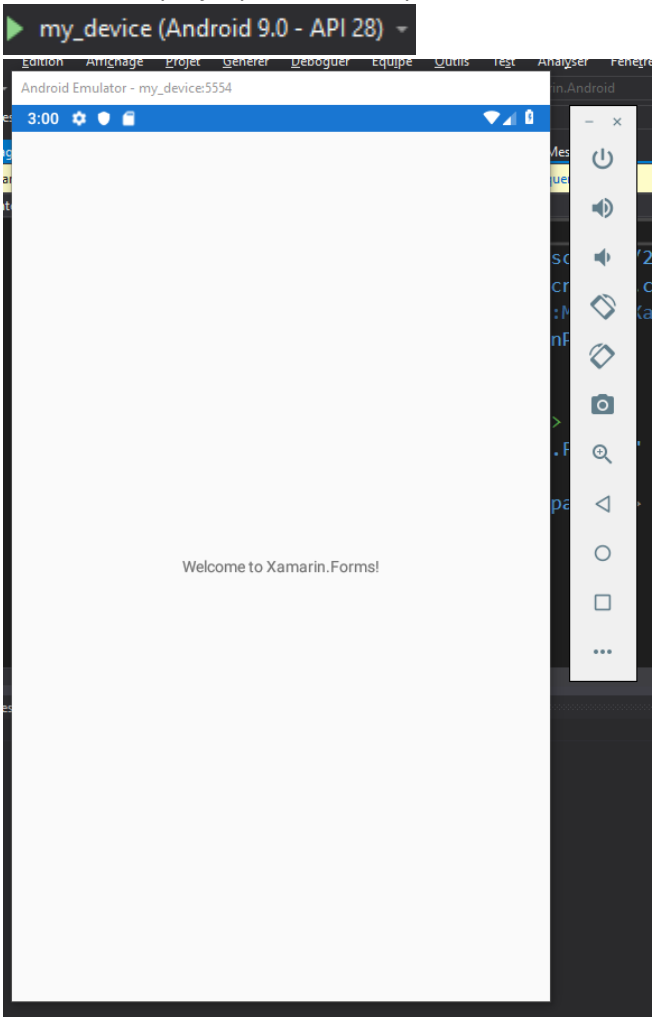
    b. Cela va vous créer 3 projets
        i. M2LinkXamarin : Votre projet partagé
        ii. M2LinkXamarin.Android
        iii. M2LinkXamarin.iOS
7. Définissez votre projet Android comme projet de démarrage
8. Vous pouvez décharger le projet iOS
9. Cliquez sur « Android Device Manager »



10. Créez un nouveau device



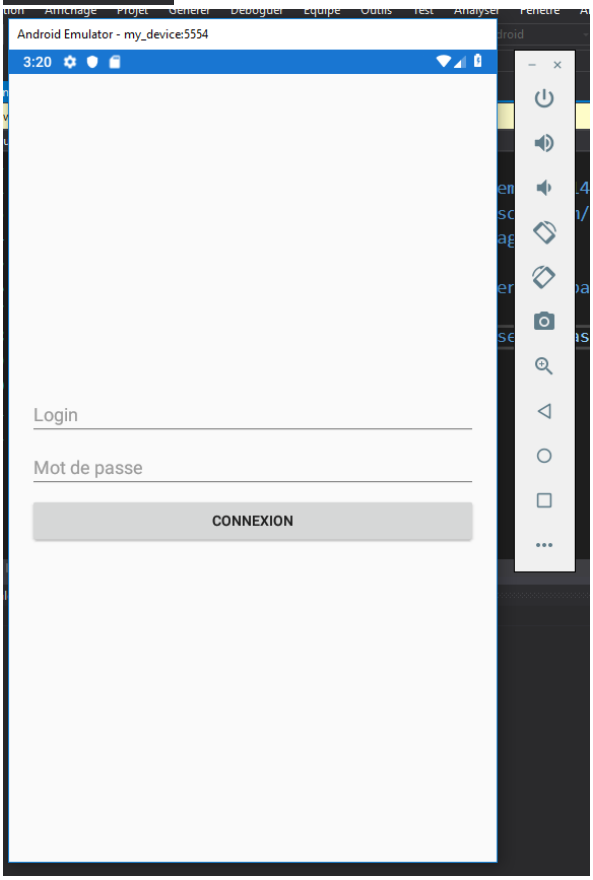
11. Lancez votre projet pour vérifier que tout fonctionne



12.

Dans M2LinkXamarin,

1. Supprimez la MainPage
2. Créez une nouvelle page de contenue XAML
    a. Nommez la « LoginPage.xaml »
    b. Utilisez le générateur d'aperçu et la boite à outil pour créer votre interface



3. Faite de votre page, la MainPage de l'application

Liaison avec vos Services Web

Avant de commencer, il faut que votre Application Web soit accessible depuis l'extérieur car votre device et votre Application Web ne seront pas sur la même machine.

1. Configurez votre IISExpress afin que tous les hosts puissent y accéder
    a. Dans un éditeur de texte, ouvrez le fichier [Votre solution]\.vs\config\applicationhost.config

b. Puis éditez la ligne suivante pour mettre « * » à la place de « localhost »

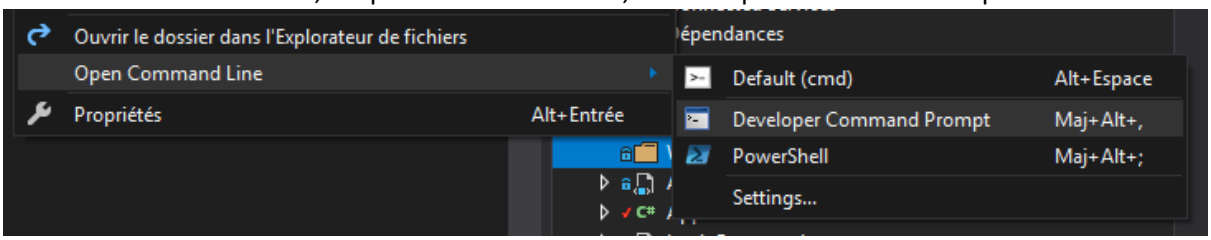
```
<site name="M2Link" id="2">
    <application path="/" applicationPool="Clr4IntegratedAppPool">
        <virtualDirectory path="/" physicalPath="C:\Projects\M2Link\M2Link" />
    </application>
    <bindings>
        <binding protocol="http" bindingInformation="*:57402:*" />
    </bindings>
</site>
```

Dans M2LinkXamarin,

1. Créez une classe « AppConstants » à a racine de votre projet qui contiendra les constantes de votre application
2. Définissez dans votre « AppConstants » l'adresse de votre serveur comme cela :

```
namespace M2LinkXamarin
{
    public class AppConstants
    {
        public static string WSServer = "http://192.168.0.84:57406";
    }
}
```

3. Créez un dossier « WebServiceClients »
4. Clic droit sur votre dossier, « Open Command Line », « Developer Command Prompt »



5. Pour chaque service web, exécutez la commande :

```
svcutil /namespace:*,M2LinkXamarin.WebServiceClients /async
http://192.168.0.84:57406/WebServices/WSLogin.svc
```

Cela va générer une classe proxy permettant d'appeler facilement votre service web dans le projet partagé Xamarin

6. Créez une classe « WSHelper »
7. Pour chaque web service, créez un attribut l'initialisant en utilisant « BasicHttpBinding » et votre constante

```
namespace M2LinkXamarin
{
    public class WSHelper
    {
        public static WSLoginClient WSLoginClient = new WSLoginClient(
            new BasicHttpBinding(),
            new EndpointAddress(AppConstants.WSServer + "/WebServices/WSLogin.svc")
        );
    }
}
```

8. Enfin, utilisez vos web services dans vos pages

```
private void Button_Clicked(object sender, EventArgs e)
{
    DisplayAlert("WSLoginHelper", WSHelper.WSLoginClient.HelloWorld(), "Ok");
}
```

Fonctionnalités sur Android

1. Développer les fonctionnalités suivantes :
    a. Connexion
    b. Inscription
    c. Affichage des utilisateurs
    d. Suivre un utilisateur
    e. Affichage des messages des utilisateurs que l'on suit
    f. Poster un message

Pour aller plus loin

1. Ajoutez des fonctionnalités à votre projet