

Azure Mobile Apps & Xamarin.Forms ハンズオン

(2016/06/30)

目次

P.02... 0.はじめに

P.03... 1.Azure Mobile Appsの構築

P.11... 2.Windows (VisualStudio 2015)

 P.12... 2-1. 新規プロジェクトの作成

 P.14... 2-2. 新しい項目(ファイル/フォルダ)の追加

 P.16... 2-3. パッケージ管理

 P.18... 2-4. スタートアッププロジェクトの変更と実行方法

P.20... 3.Mac (Xamarin Studio)

 P.21... 3-1. 新規ソリューションの作成

 P.24... 3-2. 新しい項目(ファイル/フォルダ)の追加

 P.26... 3-3. パッケージ管理

 P.28... 3-4. スタートアッププロジェクトの設定と実行方法

P.30... 4.サンプルのソースコード

 P.31... 4-1. 構成と追加するパッケージ

 P.33... 4-2. ソースコード



0.はじめに

■はじめに

今回は、Azure Mobile AppsとXamarin.Forms を使った簡単なスマホアプリを作成します。

■作成するもの

お買い物リストアプリを作成します。

データをクラウド(Azure)に格納するため、複数人で同時に情報の参照や更新が可能です。



■ソースコード

GitHub : <https://github.com/t-miyake/XamarinForms/tree/master/ShoppingList>

■本資料に関するお問い合わせ先

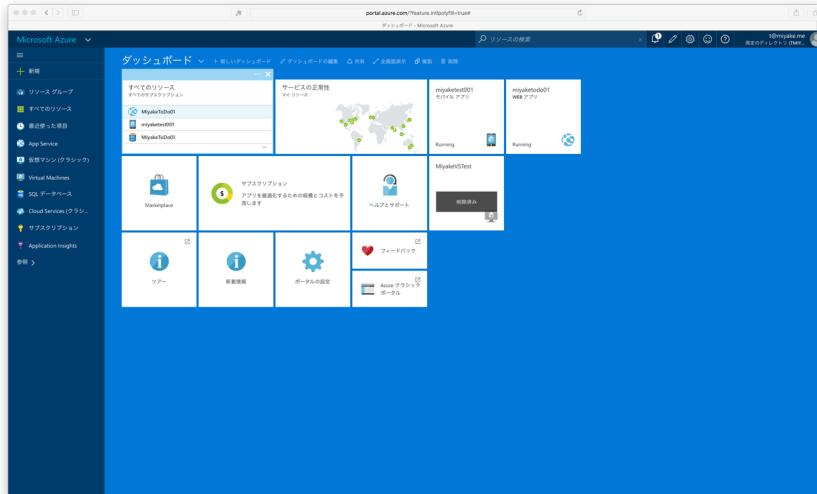
- ・株式会社のらねこ
三宅貴文

- メール miyake@noraneko.co.jp
- Twitter @T_Miyake

1.Azure Mobile Appsの構築

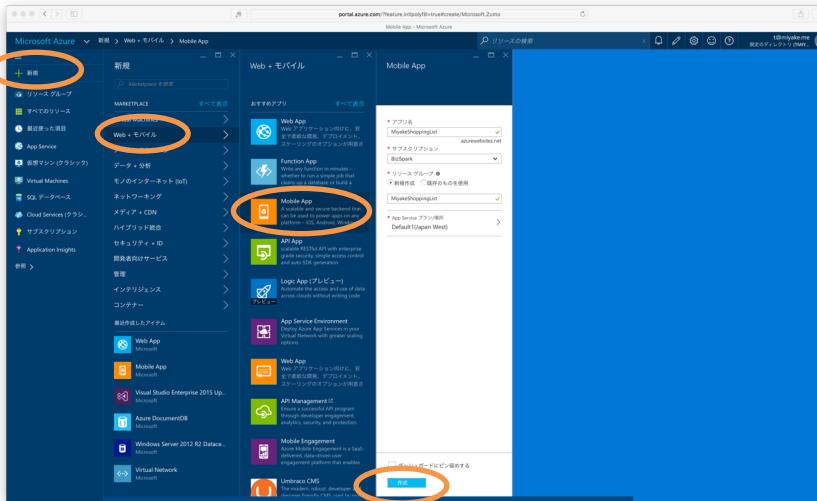
1. Azure Mobile Appsの構築

1



<https://portal.azure.com/> にアクセスします。

2



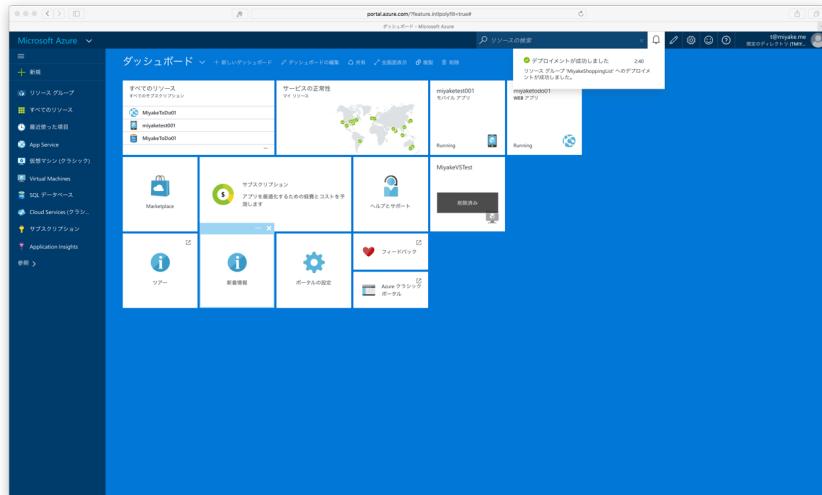
下記の順番でMobile Appsを新規作成します。
新規 → Web + モバイル → Mobile App

- ・アプリ名
お好きな前にしてください。(他の人と重複するものは使えません)
- ・リソースグループ
新規作成とし、アプリ名と同じにしてください。
- ・プラン
デフォルトで大丈夫です。(無い場合、呼んでください。)

OKを押して、少し待ちます。

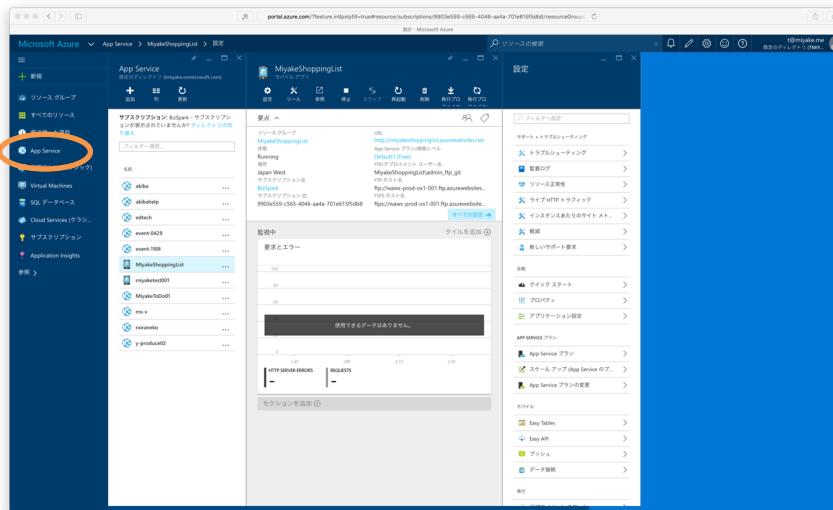
1. Azure Mobile Appsの構築

3



デプロイメントが完了しました。と表示されたら、準備完了です。

4

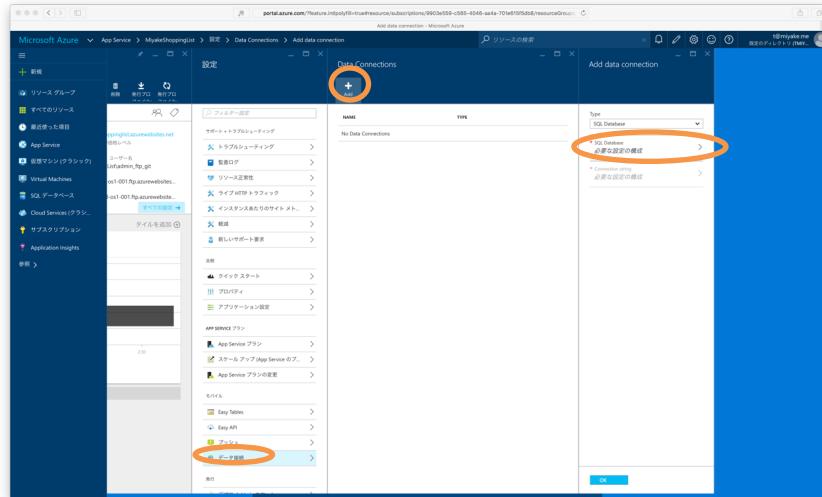


左の App Serviceから、先ほど作成したMobile Appsを選択します。
選択すると、設定画面が表示されます。

5

1. Azure Mobile Appsの構築

5

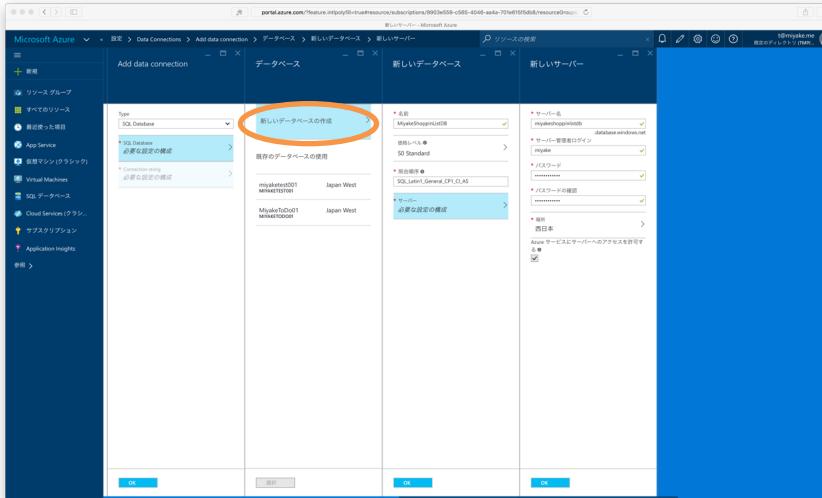


右のメニューから、データ接続を選択します。

データ接続(Data Connections)の設定が表示されるので、左上の Addボタンを押します。

Add data connectionの設定から、SQL Database欄の必要な設定の構成 を選びます。

6



データベースの設定から、新しいデータベースの作成を選択します。

新しいデータベースの設定

- 名前：好きなお名前を設定します。
- 価格レベル：Freeを選択します。
- 照合順序：デフォルトのままにします。

新しいサーバの設定

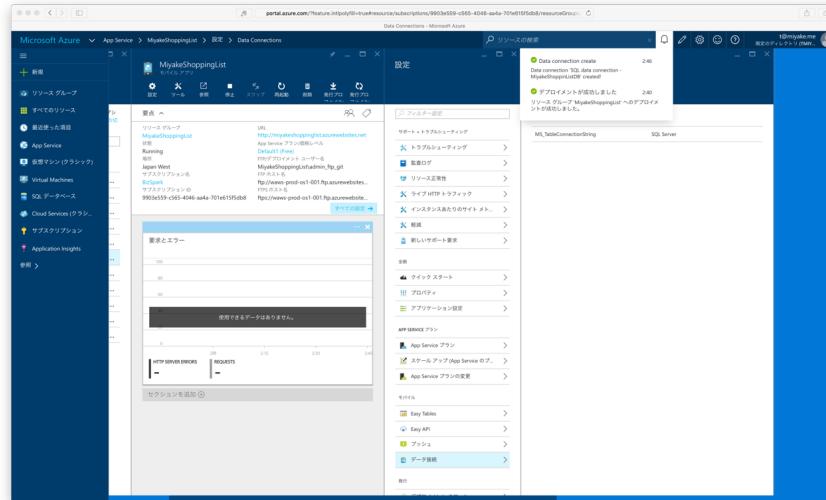
- サーバ名：好きなお名前を設定します。(他の人と重複するものは使えません)
- サーバ管理者ログイン：お好きな名前を設定します。
- パスワード：少し複雑なものにしてください。
- 場所：西日本を選択してください。

入力できたら、OKを押してください。

6

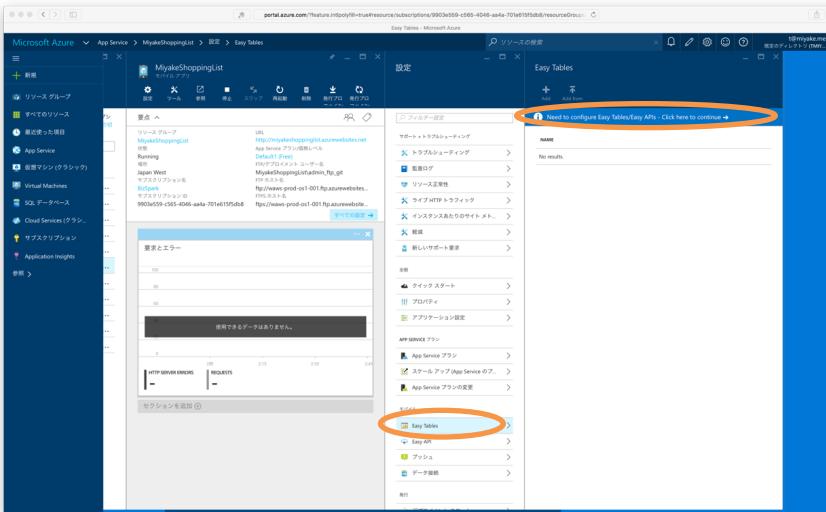
1. Azure Mobile Appsの構築

7



Data connection create と通知されれば準備完了です。

8



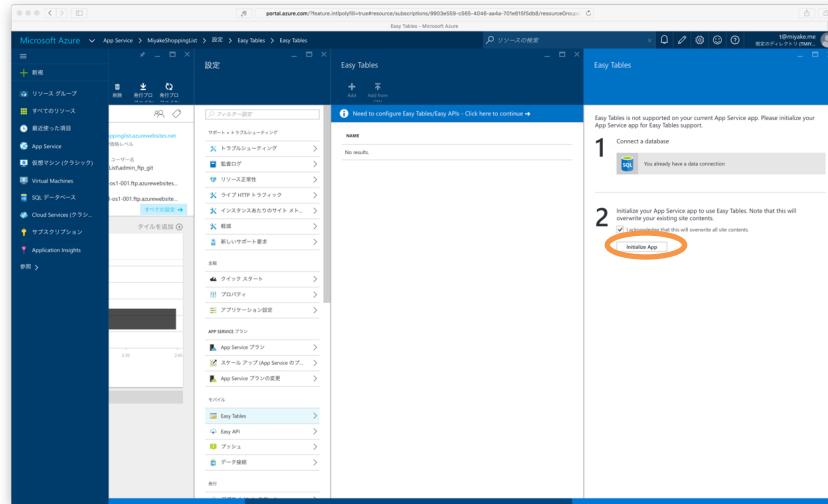
設定から、Easy Tablesを選択し、少し待つと、右側に下記が表示されます。

Need to configure Easy Tables/Easy APIs – Click here to continue →

この表示された部分を押してください。

1.Azure Mobile Appsの構築

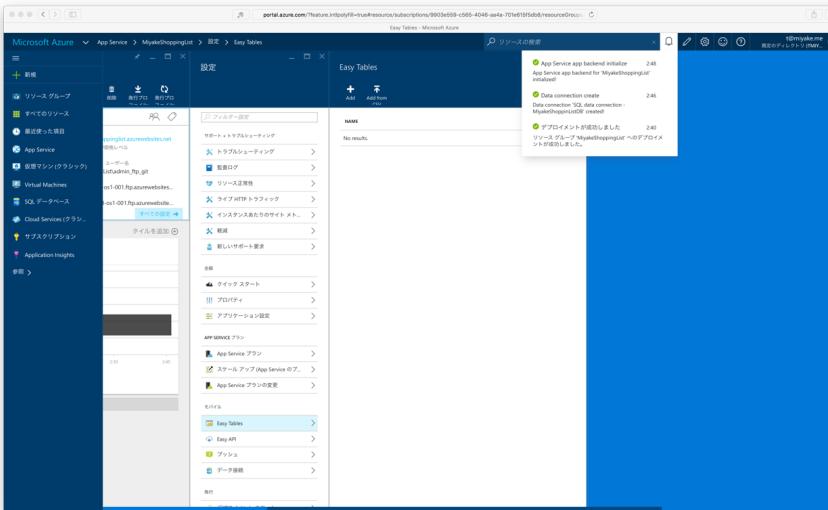
9



しばらく待つと、Easy Tablesの設定が開きます。

2の欄にある Initialize Appボタンを押します。

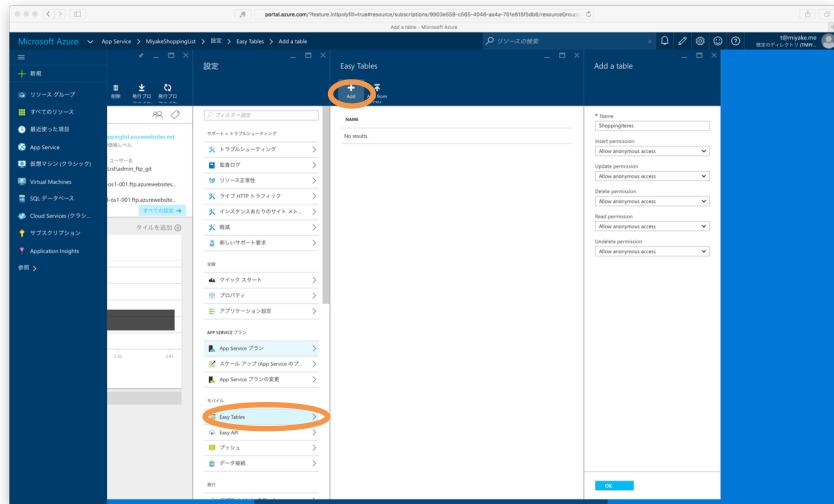
10



App Service app backend initialize と通知されれば準備完了です。

1. Azure Mobile Appsの構築

11



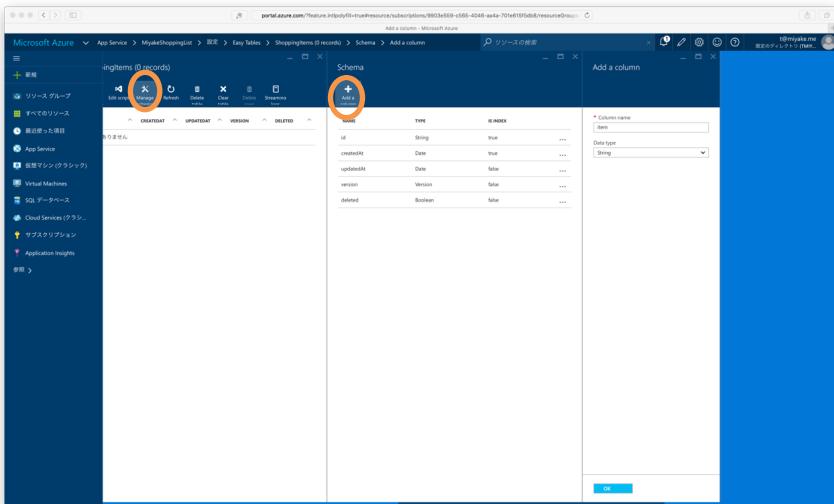
再度設定に戻り、Easy Tablesを選択します。

Easy Tablesの設定から、Addを選択します。

Add a tableからテーブルを追加します。

- Name
ShoppingItems
にしてください。(サンプルの動作に必要です)
- その他
デフォルトのままにしてください。

12



ShoppingItemsテーブルが作成されるため、その設定を行います。

メニューから Manage schema を選びます。

Schmanの設定から、Add a columnを選び下記を追加します。
(カラムを2つ追加します。)

- Column name
item
- Data type
String
- Column name
bought
- Data type
Boolean

1. Azure Mobile Appsの構築

13

The screenshot shows the Azure portal interface for managing a mobile app service. On the left, the navigation pane includes 'App Service', 'Easy Tables', and 'Schema'. The main area displays the schema for the 'ShoppingItems' table. The columns listed are:

NAME	TYPE	IS INDEX
id	String	true
createdAt	Date	true
updatedAt	Date	false
version	Version	false
name	String	false
item	String	false
bought	Boolean	false

適切にカラムを追加すると左図のようになります。

14

The screenshot shows the Azure portal interface for managing an app service. The left sidebar lists various resources, including 'App Service' and 'MyakeShoppingList'. The main area shows the configuration for 'MyakeShoppingList'. The 'SETTINGS' tab is selected, displaying the URL 'http://myakeshoppinglist.azurewebsites.net'. Other settings shown include 'WEBSITE_PRIMARIES' and 'APP SERVICE PLANS'.

以上で、Azure Mobile Appsの設定は完了です。

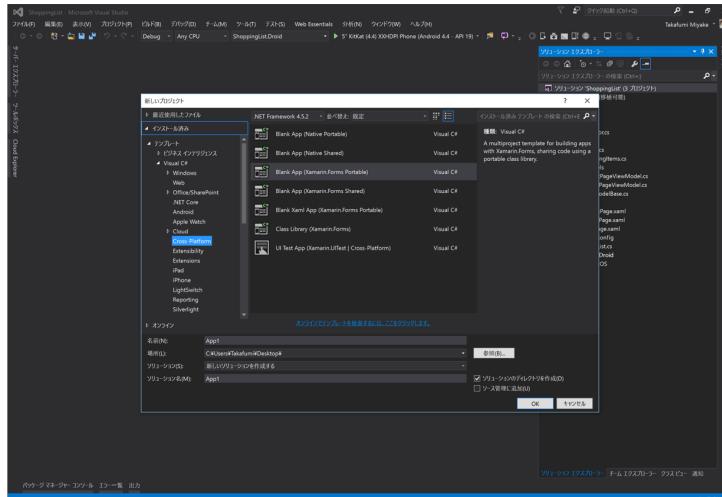
Mobile AppsのURLは後から使うので控えておいてください。

2.Windows (VisualStudio 2015)

2-1. 新規プロジェクトの作成

2-1. 新規プロジェクトの作成

1

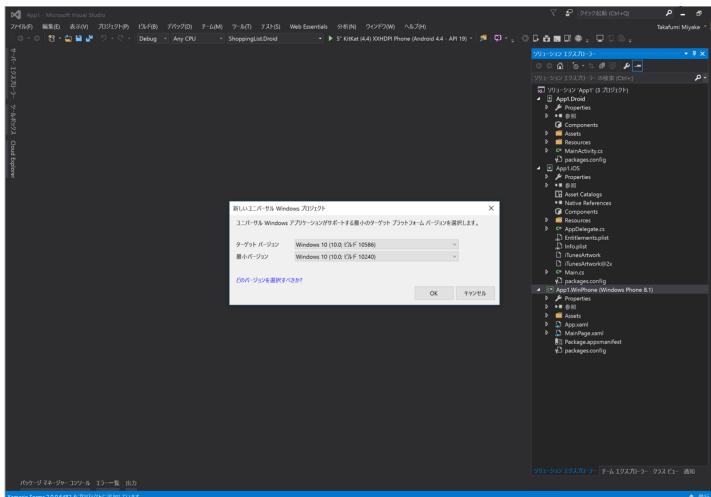


ファイル(F)の新規作成(N)から、プロジェクト(P)...を選びます。

新しいプロジェクトウィンドウが表示されたら、
今回は、Cross-Platform内の下記を選んでください。
Blank App (Xamarin.Forms Portable)

名前は下記にしてください。
ShoppingList

2



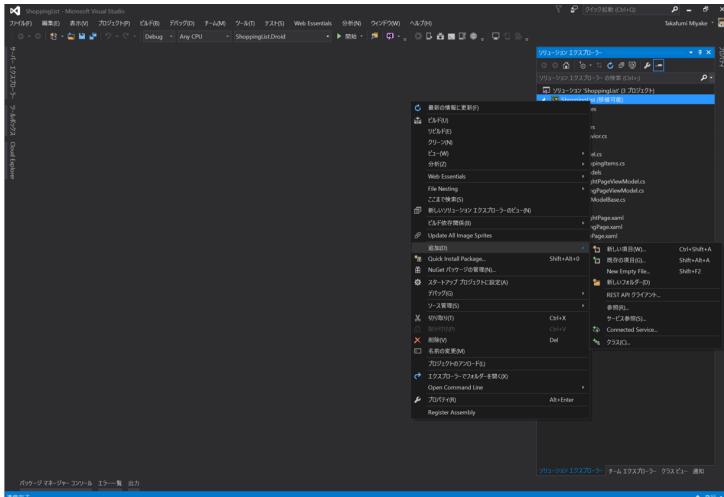
新しいユニバーサル Windows プロジェクトと表示されたら、
そのままOK押してください。

また、Macとの接続を促されるウィンドウが表示されるので、
こちらも気にせず閉じてください。

2-2. 新しい項目(ファイル/フォルダ)の追加

2-2. 新しい項目(ファイル/フォルダ)の追加

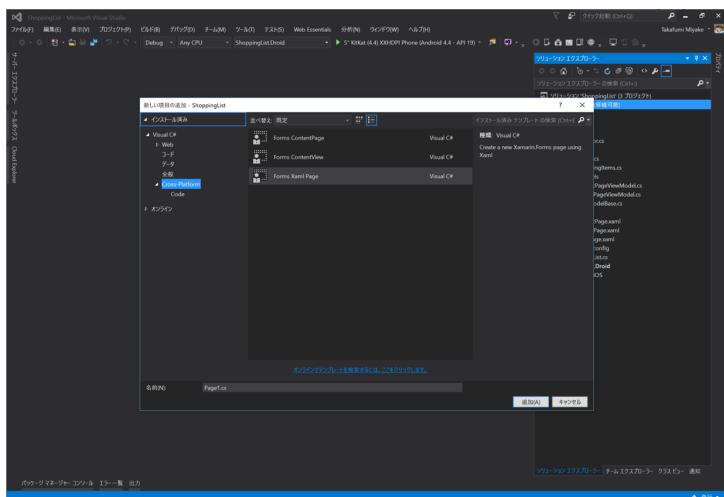
1



新しいファイルやフォルダは、プロジェクト名の右クリックメニューにある、追加(D)から追加できます。

フォルダ分けはビルド結果に影響を与えないで、無理にサンプル通りに分ける必要はありません。

2

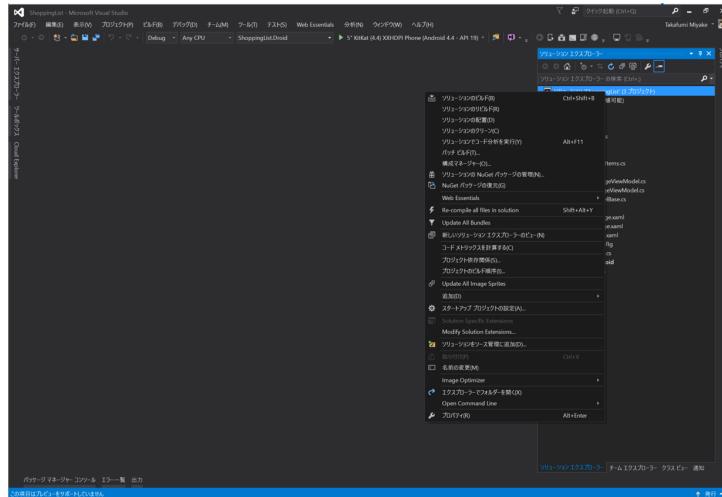


Xamlファイルを新規追加する際は、Cross-Platformの下記を選択してください。
Forms XAML Page

2-3. パッケージ管理

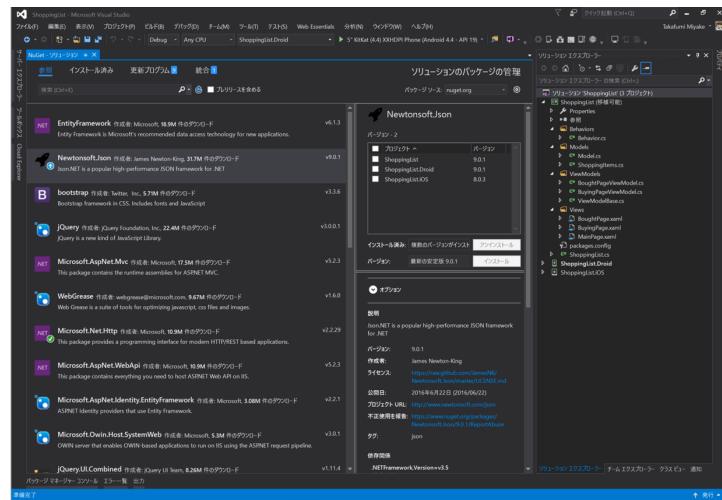
2-3. パッケージ管理

1



ソリューションを右クリックして、
ソリューションのNuGetパッケージの管理(N)...を開きます。

2



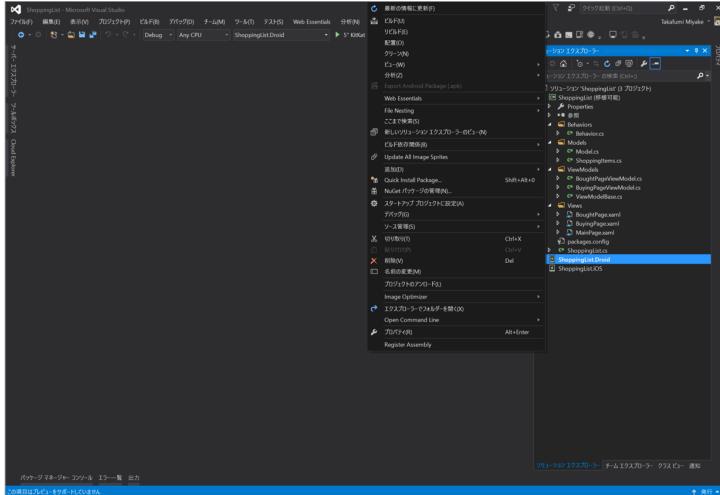
ソリューションのパッケージの管理画面から、
パッケージの追加や削除、更新が行えます。

今回は下記を全プロジェクトに追加してください。
Microsoft.Azure.Mobile.Client (Azure Mobile Apps SDK)

2-4. スタートアッププロジェクトの変更と実行方法

2-4. スタートアッププロジェクトの変更と実行方法

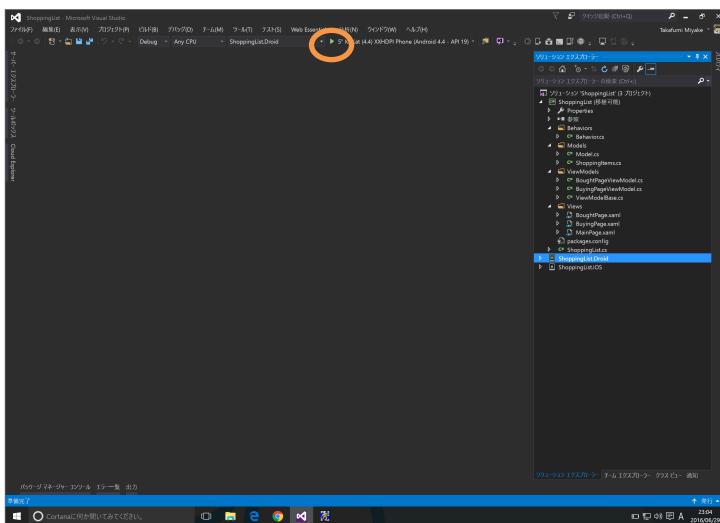
1



プロジェクトの右クリックメニューの、
スタートアッププロジェクトに設定を押すとスタートアッププロジェクトに設定できます。

今回は、.Droidと付く物を選択してください。(Androidのプロジェクトです)

2



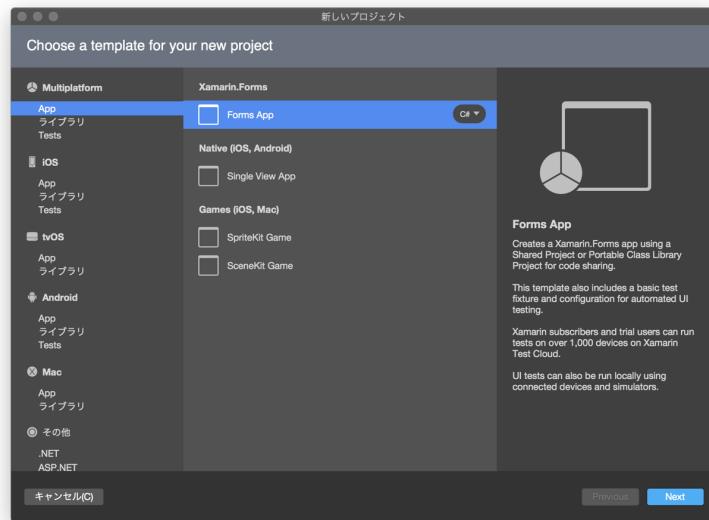
ビルドとエミュレータでの実行は、左図の再生ボタンです。

3.Mac (Xamarin Studio)

3-1. 新規プロジェクトの作成

3-1. 新規プロジェクトの作成

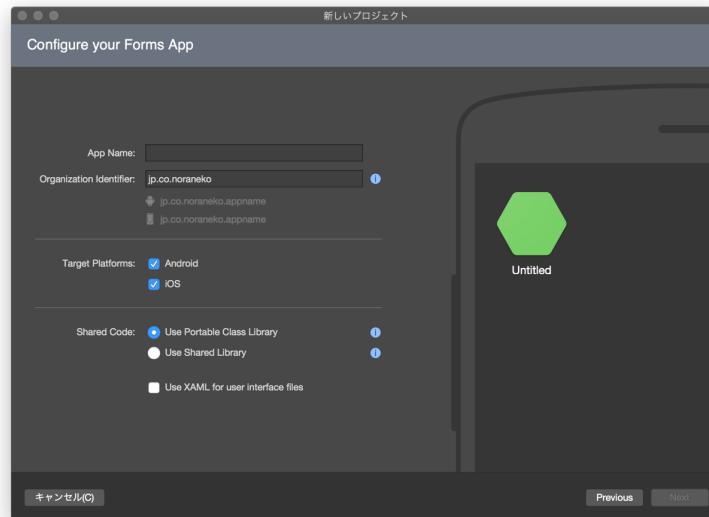
1



ファイル(F)→新規(N)→ソリューション(S)...を選択します。

新しいプロジェクトウィンドウが表示されたら、
Multiplatform内のAppから、
Forms Appを選択してください。

2

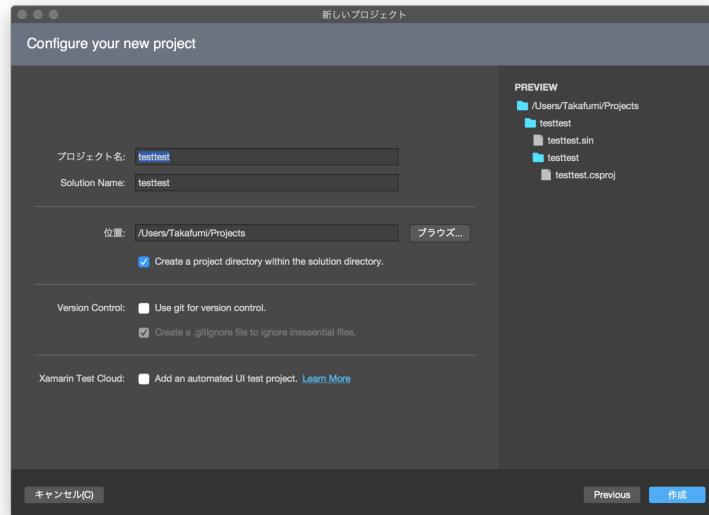


App Name欄に下記を入れてください。
ShoppingList

他はデフォルトのままで大丈夫です。

3-1. 新規プロジェクトの作成

3

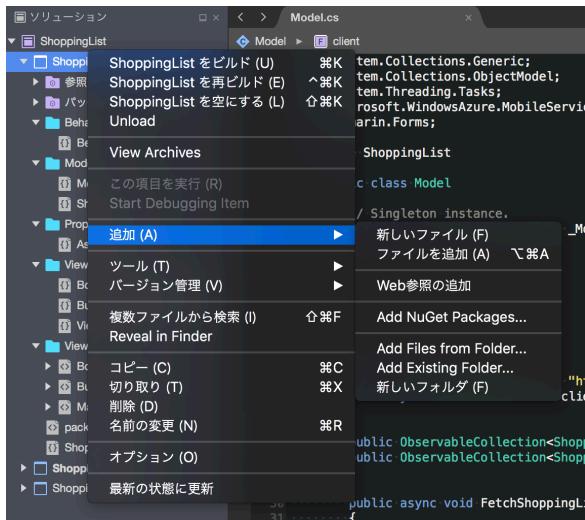


デフォルトのまま作成ボタンを押して大丈夫です。

3-2. 新しい項目(ファイル/フォルダ)の追加

3-2. 新しい項目(ファイル/フォルダ)の追加

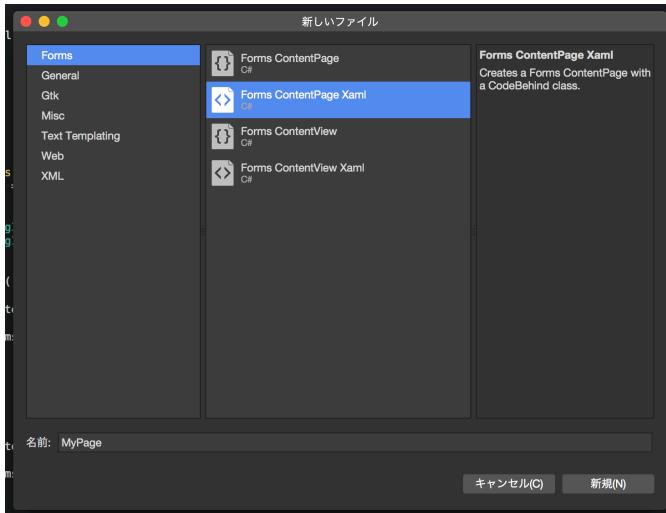
1



新しいファイルやフォルダは、プロジェクト名の右クリックメニューにある、追加(A)から追加できます。

フォルダ分けはビルド結果に影響を与えないでの、無理にサンプル通りに分ける必要はありません。

2

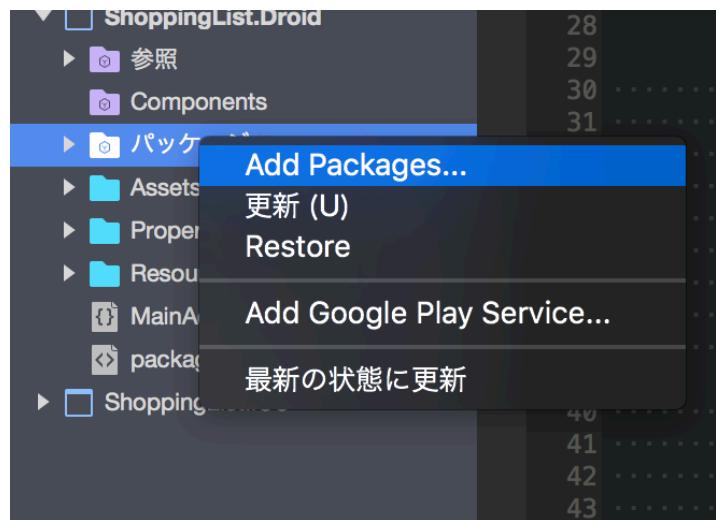


Xamlファイルを新規追加する際は、Formsの下記を選択してください。
Forms ContentPage Xaml

3-3. パッケージ管理

3-3. パッケージ管理

1

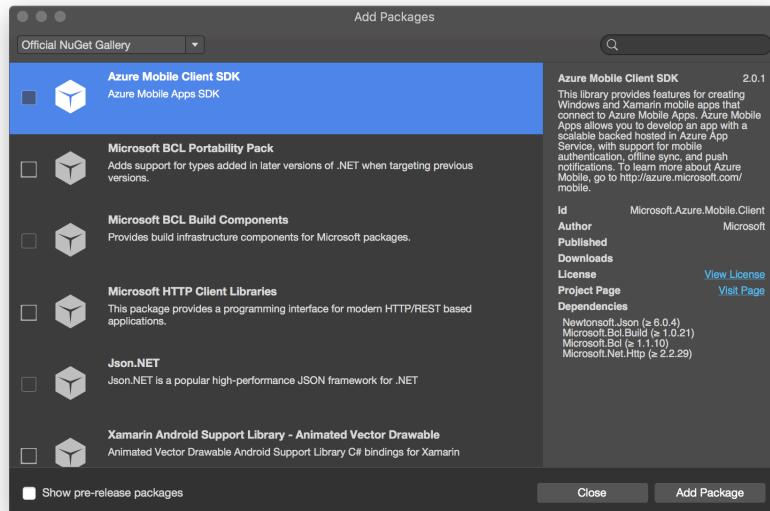


パッケージは各プロジェクト内にある、パッケージディレクトリを右クリックすると管理が可能です。

追加する際は、Add Packages...を選択してください。

※プロジェクトごとに管理されているため、.Droidや.iOSプロジェクトの中にも、それぞれパッケージディレクトリがあります。

2

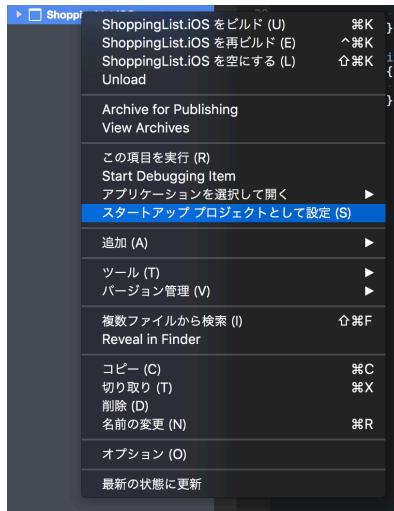


今回は、Azure Mobile Client SDK (Azure Mobile Apps SDK)を追加してください。
※ .Droidや.iOSプロジェクトの中にあるパッケージディレクトリにも追加してください。

3-4. スタートアッププロジェクトの変更と実行方法

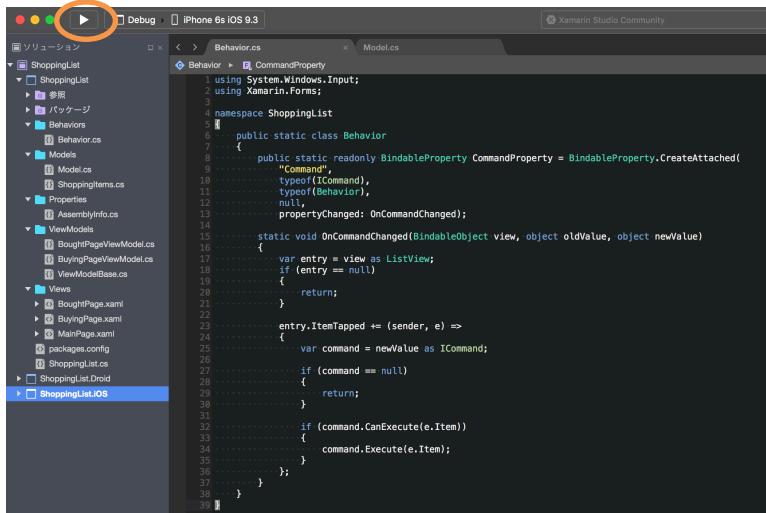
3-4. スタートアッププロジェクトの変更と実行方法

1



プロジェクトディレクトリを右クリックし、
スタートアッププロジェクトとして設定(S)を押すと、設定できます。

2



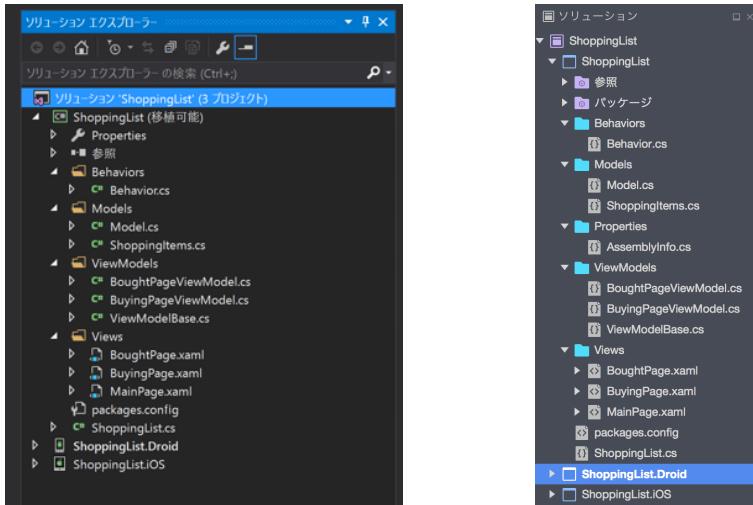
左上の再生ボタンを押すと、ビルドやエミュレータでの実行が可能です。

4. サンプルのソースコード

4-1. 構成と追加するパッケージ

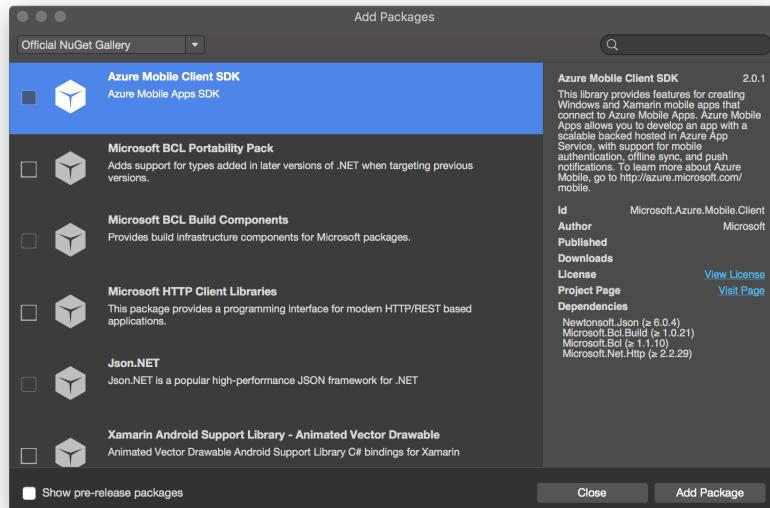
4-1. 構成と追加するパッケージ

1



全体の構成は左図のようになります。
※フォルダ分けは必須ではありません。

2



追加するパッケージは、

Azure Mobile Client SDK (Azure Mobile Apps SDK)

のみで大丈夫です。

Windows/Mac問わず、全プロジェクトに追加してください。

4-2. ソースコード

Azure Mobile SDK の初期化コード

ShoppingList.iOS/AppDelegate.cs

```
using Foundation;
using UIKit;

namespace ShoppingList.iOS
{
    [Register("AppDelegate")]
    public partial class AppDelegate : global::Xamarin.Forms.Platform.iOS.FormsApplicationDelegate
    {
        public override bool FinishedLaunching(UIApplication app, NSDictionary options)
        {
            global::Xamarin.Forms.Forms.Init();

            Microsoft.WindowsAzure.MobileServices.CurrentPlatform.Init();

            LoadApplication(new App());

            return base.FinishedLaunching(app, options);
        }
    }
}
```

ShoppingList.Droid/MainActivity.cs

```
using Android.App;
using Android.Content.PM;
using Android.OS;

namespace ShoppingList.Droid
{
    [Activity(Label = "ShoppingList.Droid", Icon = "@drawable/icon", Theme = "@style/MyTheme", MainLauncher = true, ConfigurationChanges = ConfigChanges.ScreenSize | ConfigChanges.Orientation)]
    public class MainActivity : global::Xamarin.Forms.Platform.Android.FormsAppCompatActivity
    {
        protected override void OnCreate(Bundle bundle)
        {
            TabLayoutResource = Resource.Layout.Tabbar;
            ToolbarResource = Resource.Layout.Toolbar;

            base.OnCreate(bundle);

            global::Xamarin.Forms.Forms.Init(this, bundle);

            Microsoft.WindowsAzure.MobileServices.CurrentPlatform.Init();

            LoadApplication(new App());
        }
    }
}
```

ShoppingList.cs

ShoppingList.cs

```
using Xamarin.Forms;

namespace ShoppingList
{
    public class App : Application
    {
        public App()
        {
            // The root page of your application
            MainPage = new NavigationPage(new MainPage());
        }

        protected override void OnStart()
        {
            // Handle when your app starts
        }

        protected override void OnSleep()
        {
            // Handle when your app sleeps
        }

        protected override void OnResume()
        {
            // Handle when your app resumes
        }
    }
}
```

MainPage.xaml と MainPage.xaml.cs

MainPage.xaml

```
<?xml version="1.0" encoding="UTF-8"?>
<TabbedPage xmlns="http://xamarin.com/schemas/2014/forms" xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    xmlns:local="clr-namespace:ShoppingList;assembly=ShoppingList" x:Class="ShoppingList.MainPage" Title="お買い物リスト">
    <TabbedPage.Children>
        <local:BuyingPage />
        <local:BoughtPage />
    </TabbedPage.Children>
</TabbedPage>
```

MainPage.xaml.cs

```
using Xamarin.Forms;

namespace ShoppingList
{
    public partial class MainPage : TabbedPage
    {
        public MainPage()
        {
            InitializeComponent();
        }
    }
}
```



BuyingPage.xaml と BuyingPage.xaml.cs

BuyingPage.xaml

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms" xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    xmlns:b="clr-namespace:ShoppingList;assembly=ShoppingList" x:Class="ShoppingList.BuyingPage" Title="買う物">
    <ContentPage.Content>
        <StackLayout Padding="10,5,10,0">
            <StackLayout Orientation="Horizontal" Padding="5,5,5,5">
                <Entry HorizontalOptions="FillAndExpand" Placeholder="商品の名前" Text="{Binding EntryText}" x:Name="NewItem" />
                <Button Text="+" WidthRequest="70" Command="{Binding AddButton}" CommandParameter="{x:Reference NewItem}" />
            </StackLayout>
            <ListView ItemsSource="{Binding ShoppingList}" b:Behavior.Command="{Binding ItemTapped}">
                <ListView.ItemTemplate>
                    <DataTemplate>
                        <TextCell Text="{Binding Item}" />
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
        </StackLayout>
    </ContentPage.Content>
</ContentPage>
```

BuyingPage.xaml.cs

```
using Xamarin.Forms;

namespace ShoppingList
{
    public partial class BuyingPage : ContentPage
    {
        public BuyingPage()
        {
            InitializeComponent();
            BindingContext = new BuyingPageViewModel();
        }

        protected override void OnAppearing()
        {
            base.OnAppearing();

            var ViewModel = BindingContext as BuyingPageViewModel;
            ViewModel.Initialize();
        }
    }
}
```

BoughtPage.xaml と BoughtPage.xaml.cs

BoughtPage.xaml

```
<?xml version="1.0" encoding="UTF-8"?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms" xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml" x:Class="ShoppingList.BoughtPage" Title="買った物">
    <ContentPage.Content>
        <StackLayout Padding="10,5,10,0">
            <ListView ItemsSource="{Binding BoughtList}">
                <ListView.ItemTemplate>
                    <DataTemplate>
                        <TextCell Text="{Binding Item}" />
                    </DataTemplate>
                </ListView.ItemTemplate>
            </ListView>
        </StackLayout>
    </ContentPage.Content>
</ContentPage>
```

BoughtPage.xaml.cs

```
using Xamarin.Forms;

namespace ShoppingList
{
    public partial class BoughtPage : ContentPage
    {
        public BoughtPage()
        {
            InitializeComponent();
            BindingContext = new BoughtPageViewModel();
        }

        protected override void OnAppearing()
        {
            base.OnAppearing();

            var ViewModel = BindingContext as BoughtPageViewModel;
            ViewModel.Initialize();
        }
    }
}
```

ViewModelBase.cs

ViewModelBase.cs

```
using System.ComponentModel;

namespace ShoppingList
{
    public class ViewModelBase : INotifyPropertyChanged
    {
        public event PropertyChangedEventHandler PropertyChanged;

        public void OnPropertyChanged(string info)
        {
            if (PropertyChanged != null)
            {
                PropertyChanged(this, new PropertyChangedEventArgs(info));
            }
        }
    }
}
```



BuyingPageViewModel.cs

BuyingPageViewModel.cs

```
using System.Collections.ObjectModel;
using System.Windows.Input;
using Xamarin.Forms;

namespace ShoppingList
{
    public class BuyingPageViewModel : ViewModelBase
    {
        public ICommand AddButton { get; set; }
        public ICommand ItemTapped { get; set; }

        Model Model = Model.Instance;

        public BuyingPageViewModel()
        {
            AddButton = new Command((EventArgs) =>
            {
                if (System.Net.NetworkInformation.NetworkInterface.GetIsNetworkAvailable())
                {
                    Model.ItemAdd((Entry)EventArgs);
                    EntryText = "";
                }
            });
            ItemTapped = new Command(async (EventArgs) =>
            {
                if (await Application.Current.MainPage.DisplayAlert("確認", "買いましたか?", "買った!", "まだ!"))
                {
                    if (System.Net.NetworkInformation.NetworkInterface.GetIsNetworkAvailable())
                    {
                        Model.BoughtItem(EventArgs as ShoppingItems);
                    }
                }
            });
        }

        public void Initialize()
        {
            if (System.Net.NetworkInformation.NetworkInterface.GetIsNetworkAvailable())
            {
                Model.FetchShoppingList();
            }
        }

        public ObservableCollection<ShoppingItems> ShoppingList
        {
            get
            {
                return Model.ShoppingList;
            }
            set
            {
                Model.ShoppingList = value;
            }
        }

        private string _EntryText;
        public string EntryText
        {
            get
            {
                return _EntryText;
            }
            set
            {
                _EntryText = value;
                OnPropertyChanged("EntryText");
            }
        }
    }
}
```

BoughtPageViewModel.cs

BoughtPageViewModel.cs

```
using System.Collections.ObjectModel;

namespace ShoppingList
{
    public class BoughtPageViewModel
    {
        Model Model = Model.Instance;

        public void Initialize()
        {
            if (System.Net.NetworkInformation.NetworkInterface.GetIsNetworkAvailable())
            {
                Model.FetchBoughtList();
            }
        }

        public ObservableCollection<ShoppingItems> BoughtList
        {
            get
            {
                return Model.BoughtList;
            }
            set
            {
                Model.BoughtList = value;
            }
        }
    }
}
```

ShoppingItems.cs

ShoppingItems.cs

```
using Newtonsoft.Json;

namespace ShoppingList
{
    public class ShoppingItems
    {
        public string Id { get; set; }

        [JsonProperty(PropertyName = "item")]
        public string Item { get; set; }

        [JsonProperty(PropertyName = "bought")]
        public bool Bought { get; set; }
    }
}
```



Model.cs

Model.cs

```
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.Threading.Tasks;
using Microsoft.WindowsAzure.MobileServices;
using Xamarin.Forms;

namespace Shoppinglist
{
    public class Model
    {
        // Singleton instance.
        private static readonly Model _Model = new Model();
        public static Model Instance
        {
            get
            {
                return _Model;
            }
        }
        private Model() { }

        const string ApplicationUrl = "ここにAzure Mobile AppsのURLを入れてください";
        readonly MobileServiceClient client = new MobileServiceClient(ApplicationUrl);

        public ObservableCollection<ShoppingItems> ShoppingList = new ObservableCollection<ShoppingItems>();
        public ObservableCollection<ShoppingItems> BoughtList = new ObservableCollection<ShoppingItems>();

        public async void FetchShoppingList()
        {
            var FetchedItems = await FetchItems(false);
            ShoppingList.Clear();
            foreach (var item in FetchedItems)
            {
                ShoppingList.Add(item);
            }
        }

        public async void FetchBoughtList()
        {
            var FetchedItems = await FetchItems(true);
            BoughtList.Clear();
            foreach (var item in FetchedItems)
            {
                BoughtList.Add(item);
            }
        }

        public async Task<List<ShoppingItems>> FetchItems(bool bought)
        {
            var FetchedItems = await client.GetTable<ShoppingItems>().Where(item => item.Bought == bought).ToListAsync();
            FetchedItems.Reverse();
            return FetchedItems;
        }

        public async void ItemAdd(Entry item)
        {
            if (!string.IsNullOrEmpty(item.Text))
            {
                var newItem = new ShoppingItems { Bought = false, Item = item.Text };
                await client.GetTable<ShoppingItems>().InsertAsync(newItem);
                ShoppingList.Insert(0, newItem);
            }
        }

        public async void BoughtItem(ShoppingItems item)
        {
            item.Bought = true;
            await client.GetTable<ShoppingItems>().UpdateAsync(item);
            ShoppingList.Remove(item);
        }
    }
}
```

Behavior.cs

Behavior.cs

```
using System.Windows.Input;
using Xamarin.Forms;

namespace ShoppingList
{
    public static class Behavior
    {
        public static readonly BindableProperty CommandProperty = BindableProperty.CreateAttached(
            "Command",
            typeof(ICommand),
            typeof(Behavior),
            null,
            propertyChanged: OnCommandChanged);

        static void OnCommandChanged(BindableObject view, object oldValue, object newValue)
        {
            var entry = view as ListView;
            if (entry == null)
            {
                return;
            }

            entry.ItemTapped += (sender, e) =>
            {
                var command = newValue as ICommand;

                if (command == null)
                {
                    return;
                }

                if (command.CanExecute(e.Item))
                {
                    command.Execute(e.Item);
                }
            };
        }
    }
}
```

