Linkar Suite

Creando Apps móviles con Xamarin y MV usando Linkar 2.0

CON SERVICIO WEB o SIN ÉL.

Marzo 2020

by Kosday Solutions

kosday solutions

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

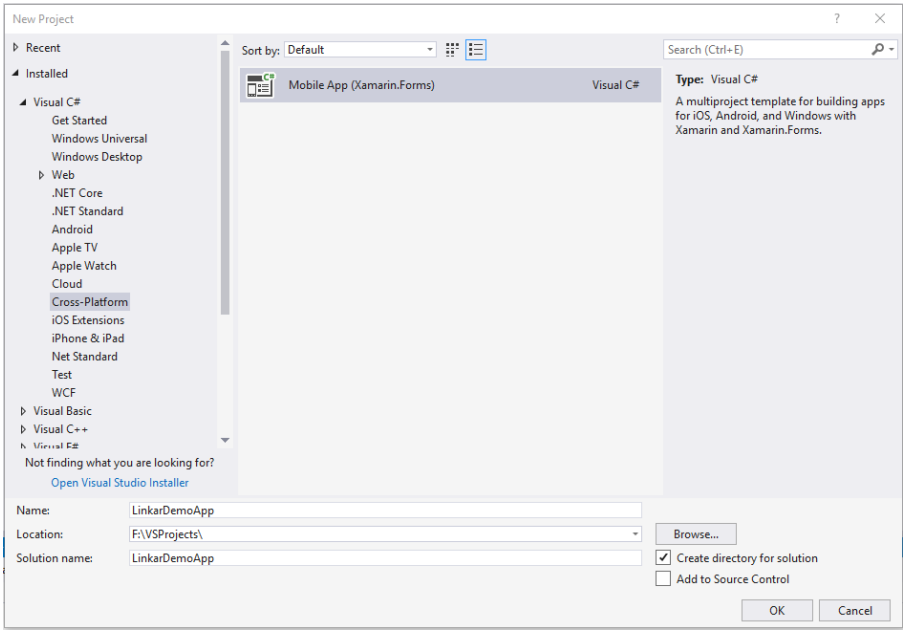
Vamos a desarrollar una app sencilla en Xamarin para leer los datos de un fichero y visualizarlos en el dispositivo. Para realizar las llamadas utilizaremos dos métodos de conexión diferentes:

- A través de un **SERVICIO WEB REST** publicado que reciba las peticiones del cliente y se conecte a Linkar para obtener las respuestas de la Base de Datos.
- Mediante **CONEXIÓN DIRECTA a LINKAR**, **sin pasar por ningún Servicio Web**. De esta manera será Linkar el responsable de recibir las peticiones del cliente y devolverle las respuestas directamente, eliminando así el elemento intermedio del servicio web.

Requisitos previos

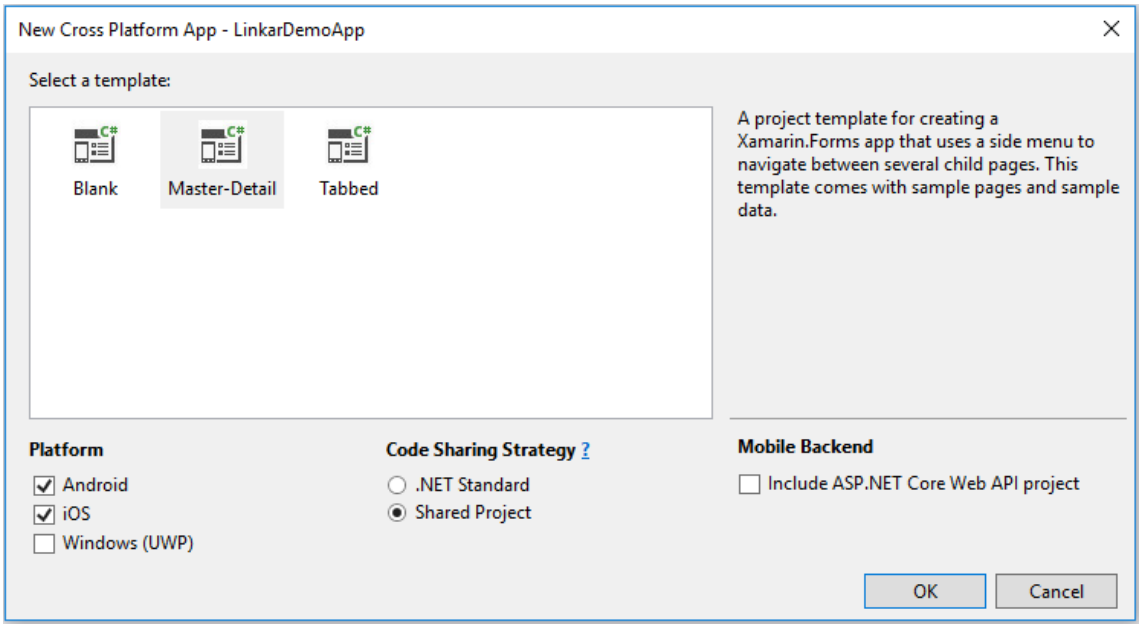
- Instalado Linkar Server
- Para la CONEXIÓN DIRECTA usaremos la librería de Linkar LinkarClientMono.dll
- Visual Studio 2017
    - Xamarin https://docs.microsoft.com/en-us/xamarin/cross-platform/get-started/installation/windows

Abrir Visual Studio. Crear un nuevo proyecto Mobile App, para ello vaya a File -> New... -> Project seleccione el menú Cross-Platform y escoja Mobile.App (Xamarin.Forms). En este caso llamaremos tanto al proyecto como a la solución LinkarDemoApp.



Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

Seleccionaremos la plantilla Master-Detail para trabajar sobre un código de ejemplo. Las Plataformas de destino en nuestro caso serán Android e IOS, el Código Compartido usará un proyecto compartido y no añadiremos la opción del Backend. Tienes un ejemplo completo de Backend en este enlace.

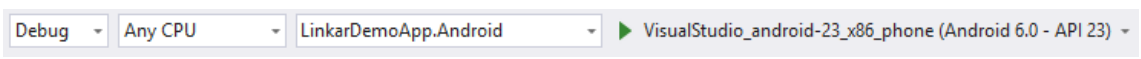


Esto dará como resultado una Solución con tres proyectos. Los proyectos son los siguientes:

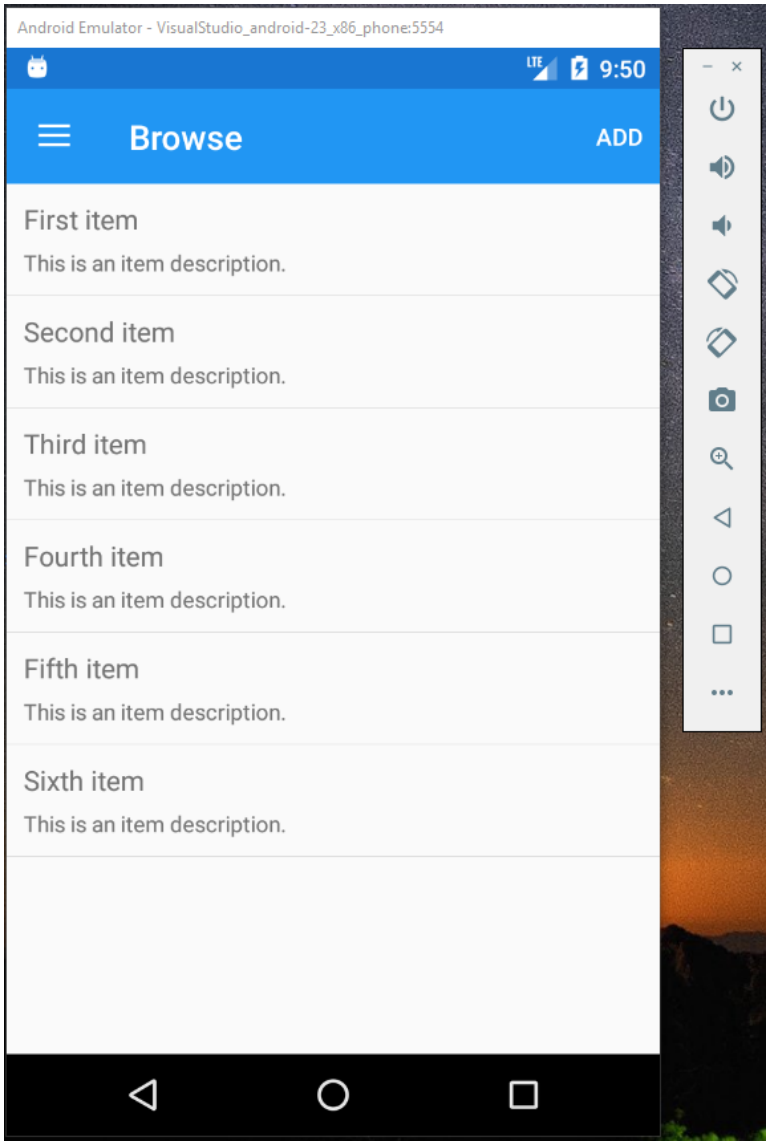
- LinkarDemoApp: proyecto compartido donde definiremos la mayor parte del código, asi como las pnatallas de la aplicación.
- LinkarDemoApp.Android: proyecto Android con los elementos puramente Android como el AndroidManifest, el MainActivity o los recursos (iconos, imágenes…).
- LinkarDemoApp.IOS: proyecto IOS con los elementos puramente IOS como el Info.plist o los recursos (iconos, imágenes…).

Este ejemplo esta desarrollado en un PC Windows y por comodidad usaremos el proyecto LinkarDemoApp.Android como proyecto de inicio. Esto nos permite usar un emulador de Android para probar la aplicación.

Gracias a la plantilla tenemos ya una aplicación básica que podemos ejecutar y probar.

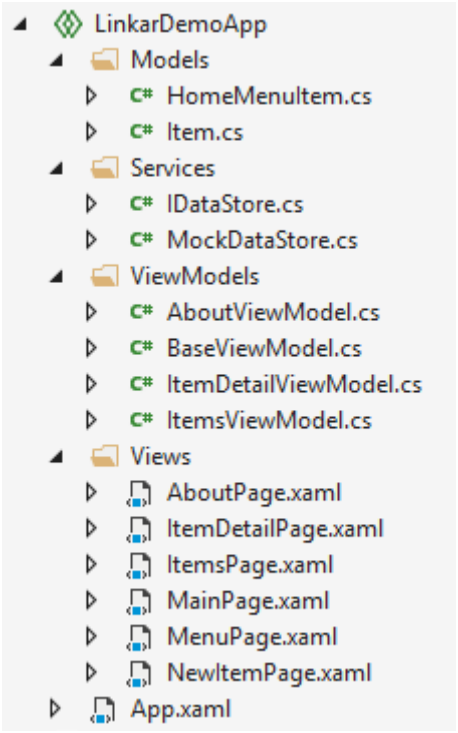


Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.



Antes de realizar cambios sobre la plantilla explicaremos la estructura del proyecto LinkarDemoApp (Compartido).

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.



La plantilla utiliza un modelo MVVM (Model-View-ViewModel), si desea saber más sobre este patrón de diseño visite el siguiente enlace:

https://docs.microsoft.com/en-us/xamarin/xamarin-forms/enterprise-application-patterns/mvvm

El documento App es el punto de entrada al código compartido, será llamado desde los proyectos de Android e IOS. En él se pueden definir algunos eventos que afectan directamente al dispositivo y estilos que serán utilizados en las diferentes vistas de la aplicación. Para más detalles visite:

https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.application?view=xamarin-forms

La carpeta Services muestra un ejemplo local de como programar las diferentes acciones que serán llamadas desde los diferentes ViewModels. Aquí explicaremos dos formas de llamar a las funciones de Linkar y trabajar así con su base de datos.

1. **SERVICIO WEB** (Backend)

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

Esta conexión implica la programación y publicación de unos servicios web que actuaran como intermediario entre la aplicación y Linkar Server. De esta forma cuando se quiera obtener por ejemplo una lista de Items llamaremos a un método GetItems (vía HTTP/HTTPS) declarado en el servicio que a su vez usará Linkar Client para realizar una Select de Items, tratará el resultado y lo devolverá a la aplicación.

Puedes leer nuestro How To sobre como craer servicios web con Linkar en nuestro blog https://www.kosday.com/blog/how-to-web-service-with-linkar.

Un ejemplo de cómo llamar a un servicio desde la aplicación es la siguiente clase:

```
using System.Collections.Generic;
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;
using Newtonsoft.Json;
using Sample.Models;

namespace Sample.Services
{
    public class ItemDataStore
    {
        HttpClient client;
        IEnumerable<Item> items;

        public ItemDataStore()
        {
            client = new HttpClient();
            client.BaseAddress = new Uri($"{App.servicesUrl}/");
            items = new List<Item>();
        }

        public async Task<IEnumerable<Item>> GetItemsAsync(bool forceRefresh = false)
        {
            if (forceRefresh)
            {
                var json = await client.GetStringAsync($"api/Sample/GetItems);
                items = await Task.Run(() => JsonConvert.DeserializeObject<IEnumerable<Item>>(json));
            }

            return items;
        }

        public async Task<Item> GetItemAsync(string id)
        {
            if (id != null)
            {
```

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
                var json = await client.GetStringAsync($"api/Sample/GetItem?id=" +
id);
                return              await              Task.Run(()              =>
JsonConvert.DeserializeObject<Item>(json));
            }

            return null;
        }
    }
}
```

2. CONEXIÓN DIRECTA a LINKAR

La conexión directa usará la librería MONO de LinkarClient para poder realizar operaciones directamente desde la aplicación, sin ningún otro intermediario.

Para usar esta conexión se debe añadir la referencia a la librería LinkarClientMono.dll en los proyectos LinkarDemoApp.Android y LinkarDemoApp.IOS. El siguiente paso es añadir las instrucciones “using” en aquellas clases donde queramos utilizar el cliente.

```
using LinkarClient;
using LinkarCommon;
```

Primero debemos declarar el cliente de Linkar, para poder usarlo desde diferentes clases lo haremos como una variable estática en el archivo App.xaml.cs

```
public static LinkarClt linkarClt = new LinkarClt();
```

En el mismo archivo vamos a implementar una función que nos ayudara a gestionar las excepciones que el cliente de Linkar pueda producir.

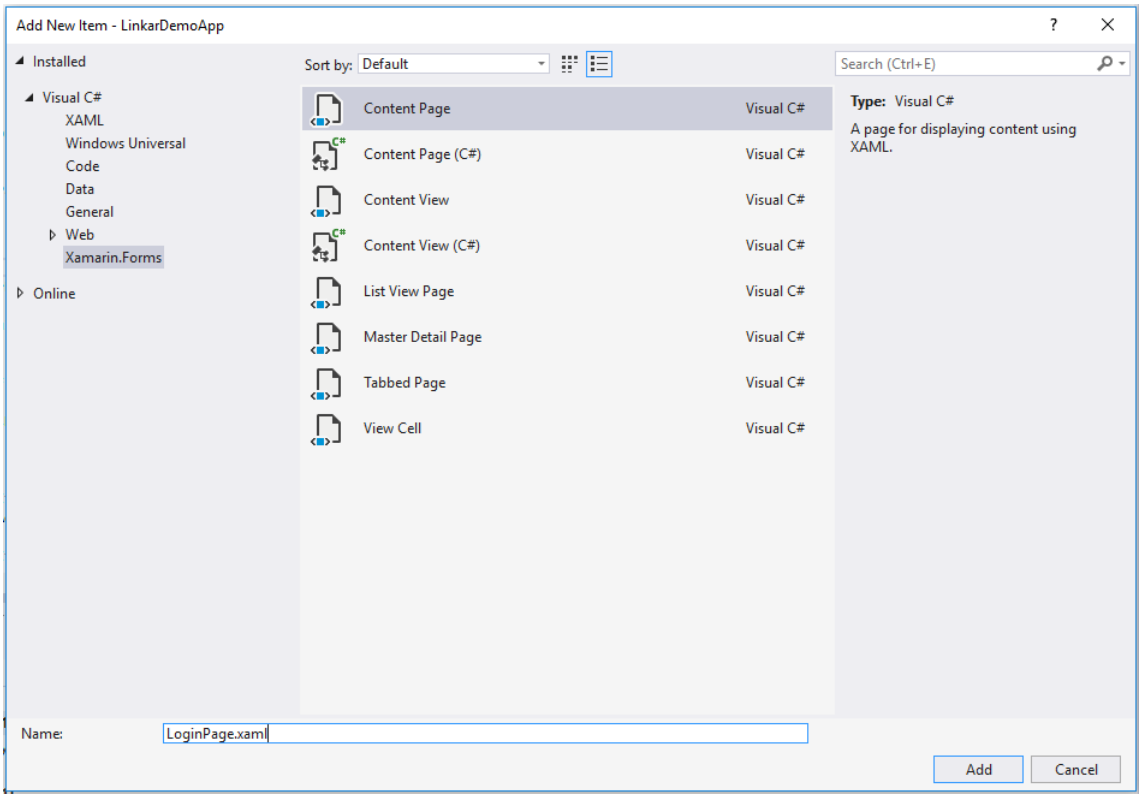
```
        public static string GetException(Exception ex)
        {
            string msg = "";
            if (ex.GetType() == typeof(LkException))
            {
                LkException lkex = (LkException)ex;
                msg = "LINKAR EXCEPTION ERROR";
                if (lkex.ErrorCode == LkException.ERRORCODE.C0003)
                    msg += "\r\nERROR CODE: " + lkex.ErrorCode +
                           "\r\nERROR MESSAGE: " + lkex.ErrorMessage +
```

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
                    "\r\nInternal ERROR CODE: " + lkex.InternalCode +
                    "\r\nInternal ERROR MESSAGE: " + lkex.InternalMessage;
            else
                msg += "\r\nERROR CODE: " + lkex.ErrorCode +
                    "\r\nERROR MESSAGE: " + lkex.ErrorMessage;
        }
        else if (ex.GetType() == typeof(System.Net.Sockets.SocketException))
        {
            System.Net.Sockets.SocketException soex =
(System.Net.Sockets.SocketException)ex;
            msg = "SOCKET EXCEPTION ERROR\r\n" + soex.Message;
        }
        else
        {
            msg = "EXCEPTION ERROR\r\n" + ex.Message;
        }

        return msg;
    }
}
```

Ahora crearemos una nueva página principal previa a la muestra de los datos. Debemos hacer click derecho sobre la carpeta Views Add -> New Item…



Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

Y añadimos un archivo Content Page y sustituimos su código con el siguiente:

LoginPage.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="LinkarDemoApp.Views.LoginPage">

    <ContentPage.Resources>
        <ResourceDictionary>
            <Color x:Key="Primary">#2196F3</Color>
            <Color x:Key="Accent">#cce8fe</Color>
            <Color x:Key="LightTextColor">#999999</Color>
        </ResourceDictionary>
    </ContentPage.Resources>

    <ContentPage.Content>
        <Grid AbsoluteLayout.LayoutBounds="0,0,1,1"
                AbsoluteLayout.LayoutFlags="All"
                VerticalOptions="FillAndExpand"
                HorizontalOptions="FillAndExpand">
                <Grid.RowDefinitions>
                    <RowDefinition Height="Auto" />
                    <RowDefinition Height="*" />
                </Grid.RowDefinitions>
            <StackLayout         BackgroundColor="{StaticResource         Accent}"
VerticalOptions="FillAndExpand" HorizontalOptions="Fill">
                <StackLayout  Orientation="Horizontal"  HorizontalOptions="Center"
VerticalOptions="Center">
                    <ContentView                                Padding="0,30,0,20"
VerticalOptions="FillAndExpand">
                        <Image  Source="linkar_logo.png"  VerticalOptions="Center"
HeightRequest="64" />
                    </ContentView>
                </StackLayout>
            </StackLayout>
                <ScrollView Grid.Row="1">
                    <StackLayout   Orientation="Vertical"   Padding="16,10,16,10"
Spacing="10" >
                    <Label >
                        <Label.FormattedText>
                            <FormattedString>
                                <FormattedString.Spans>
                                    <Span    Text="This    app    shows    Linkar
functionality and performance. Our focus is not to show you how to make pretty
interfaces or how to program." />
                                </FormattedString.Spans>
                            </FormattedString>
                        </Label.FormattedText>
                    </Label>
```

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
                    <Label>
                        <Label.FormattedText>
                            <FormattedString>
                                <FormattedString.Spans>
                                    <Span Text="This app uses Xamarin and a Linkar
Server web service." />
                                </FormattedString.Spans>
                            </FormattedString>
                        </Label.FormattedText>
                    </Label>
                    <Label>
                        <Label.FormattedText>
                            <FormattedString>
                                <FormattedString.Spans>
                                    <Span Text="You have all the source code in
our web in the Linkar Demos folder. Take a look at it and discover how easy it is."
/>
                                </FormattedString.Spans>
                            </FormattedString>
                        </Label.FormattedText>
                    </Label>
                    <Button         Text="Login"         Clicked="Button_Clicked"
TextColor="White" BackgroundColor="{StaticResource Primary}" HeightRequest="40"/>
                </StackLayout>
                </ScrollView>
            </Grid>
    </ContentPage.Content>
</ContentPage>

LoginPage.xaml.cs

using LinkarDemoApp.ViewModels;
using System;
using System.Threading.Tasks;

using Xamarin.Forms;
using Xamarin.Forms.Xaml;

namespace LinkarDemoApp.Views
{
       [XamlCompilation(XamlCompilationOptions.Compile)]
       public partial class LoginPage : ContentPage
       {
        LoginViewModel viewModel;

        public LoginPage()
        {
            InitializeComponent();
            NavigationPage.SetHasNavigationBar(this, false);  // Hide nav bar

            BindingContext = viewModel = new LoginViewModel();
        }
```

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
        async void Button_Clicked(object sender, EventArgs e)
        {
            await LoginAsync();
        }

        public async Task LoginAsync()
        {
                        CredentialsOptions      credentialsOptions      =      new
CredentialsOptions("127.0.0.1", "DEMOEP", 11300, "demo", "demo1234");
            try
            {
                await Task.Run(() => {
                    App.linkarClt.Login(credentialsOptions);
                    App.Current.MainPage = new MainPage();
                    });
            }
            catch (Exception ex)
            {
                await this.DisplayAlert("ERROR", App.GetException(ex), "OK");
            }
        }
    }
}
```

Al no realizar una captura de datos ni un binding de la información resultante prescindimos de los correspondientes ViewModel y Model y realizamos la operación directamente en el código del botón como puede verse en el ejemplo. Para un formulario de Login más completo se recomienda crear toda la estructura. Cuando el Login es correcto navegaremos a MainPage.

Volvemos un momento al archivo App.xaml.cs para cambiar en el constructor la página inicial de la aplicación por la nueva página de Login.

```
        public App()
        {
            InitializeComponent();
            MainPage = new LoginPage();
        }
```

Vamos a cambiar ahora el modelo Items.cs para que refleje los datos que tenemos en el fichero LK.ITEMS.

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
public class Item
{
    public string Id { get; set; }
    public string Description { get; set; }
    public double Stock { get; set; }
}
```

Este cambio nos obligara a hacer pequeños ajustes en los siguientes Views y ViewModels:

NewItemPage.xaml.cs: reemplazamos su constructor por el siguiente código.

```
 public NewItemPage()
{
    InitializeComponent();

    Item = new Item
    {
        Stock = 0,
        Description = "This is an item description."
    };

    BindingContext = this;
}
```

ItemDetailPage.xaml.cs: reemplazamos su constructor por el siguiente código.

```
public ItemDetailPage()
        {
            InitializeComponent();

            var item = new Item
            {
                Stock = 0,
                Description = "This is an item description."
            };

            viewModel = new ItemDetailViewModel(item);
            BindingContext = viewModel;
        }
```

ItemDetailViewModel.cs: reemplazamos su constructor por el siguiente código.

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
public ItemDetailViewModel(Item item = null)
        {
            Title = item?.Description;
            Item = item;
        }
```

En el código XAML de las Views (ItemsPage.xaml, ItemDetailPage.xaml y NewItemPage.xaml) simplemente cambiaremos las referencias que encontremos a la propiedad Text por la propiedad Stock. Por ejemplo:

```
<Label Text="Stock:" FontSize="Medium" />
<Label Text="{Binding Item.Stock}" FontSize="Small"/>
```

A continuación, vamos a modificar el archivo MockDataStore.cs de la carpeta Services para obtener la lista de ítems desde Linkar Server. Para ello primero modificaremos el constructor para quitar los datos fijos que aparecen en la plantilla y añadiremos también las instrucciones using de Linkar.

```
public MockDataStore()
{
    items = new List<Item>();
}
```

Sobre el mismo archivo modificamos el método GetItemsAsync para que obtenga los datos de Linkar Server.

```
public async Task<IEnumerable<Item>> GetItemsAsync(bool forceRefresh = false)
        {

            if (forceRefresh)
            {
                items = null;
                try
                {

                    string lkstring = App.linkarClt.Select_Text("LK.ITEMS", "", "BY
CODE", "", "", new SelectOptions(false, false, 10, 1, true), DATAFORMAT_TYPE.MV, "",
0);

                    if (!string.IsNullOrEmpty(lkstring))
                    {
                        char delimiter = ASCII_Chars.FS_chr;
```

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
                        char delimiterThisList = DBMV_Mark.AM;
                        String recordIds = "";
                        String records = "";
                        String recordCalculateds = "";
                        String[] parts = lkstring.Split(delimiter);
                        if (parts.Length >= 1)
                        {
                            String[] ThisList = parts[0].Split(delimiterThisList);
                            int numElements = ThisList.Length;
                            for (int i = 1; i < numElements; i++)
                            {
                                if (ThisList[i].Equals("RECORD_ID"))
                                {
                                    recordIds = parts[i];
                                }
                                if (ThisList[i].Equals("RECORD"))
                                {
                                    records = parts[i];
                                }
                                if (ThisList[i].Equals("CALCULATED"))
                                {
                                    recordCalculateds = parts[i];
                                }
                            }
                        }

                        //Fill all the records with response data
                        String[] lstids = recordIds.Split(ASCII_Chars.RS_chr);
                        String[] lstregs = records.Split(ASCII_Chars.RS_chr);
                        String[] lstcalcs =
recordCalculateds.Split(ASCII_Chars.RS_chr);

                        items = new List<Item>();

                        for (int i = 0; i < lstids.Length; i++)
                        {
                            Item record = new Item();
                            if (recordCalculateds != null && recordCalculateds !=
"")
                            {
                                record = LkItem.GetRecord(lstids[i], lstregs[i],
lstcalcs[i]);
                            }
                            else
                                record = LkItem.GetRecord(lstids[i], lstregs[i],
"");

                            items.Add(record);
                        }
                    }
                }
                catch (Exception ex)
                {
                    string error = App.GetException(ex);
```

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
                    await
Xamarin.Forms.Application.Current.MainPage.DisplayAlert("ERROR", error, "OK");
                }
            }

            return items;
        }
```

Esta es solo una de las formas de trabajar con el resultado de una función Linkar, si desea conocerlas todas para usar la que más se adecue a sus necesidades consulte nuestro manual https://www.kosday.com/Manuals/es_WEB_LINKAR/index.html

Debemos añadir por último el método GetRecord que usaremos para alimentar cada uno de los Items de nuestra lista, asignándole a cada propiedad de la clase el valor correspondiente obtenido por la función.

```
        private Item GetRecord(string recordID, string record, string recordCalculated)

            {

                Item item = new Item();

                //Create empty record

                string[] reg = new string[2];

                for (int j = 0; j < reg.Length; j++)

                    reg[j] = "";


                //Fill the record

                if (record != null && record != "")

                {

                    string[] aux = record.Split(DBMV_Mark.AM);

                    for (int j = 0; j < aux.Length; j++)

                        reg[j] = aux[j];

                }


                //Create empty calculated record

                string[] regI = new string[0];
```

Creando Apps móviles con Xamarin y MV usando Linkar 2.0 CON SERVICIO WEB o SIN ÉL.

```
            for (int k = 0; k < regI.Length; k++)
                regI[k] = "";

            //Fill the calculated record
            if (recordCalculated != null && recordCalculated != "")
            {
                string[] auxI = recordCalculated.Split(DBMV_Mark.AM);
                int k = 0;
                for (; k < auxI.Length; k++)
                    regI[k] = auxI[k];
            }

            //Fill the record ID property
            item.Id = recordID;
            if (record == null || record == "")
                return null;

            //Fill the class properties
            item.Description = LinkarDataTypes.GetAlpha(reg[0]);
            item.Stock = LinkarDataTypes.GetDecimal(reg[1]);
            return item;
        }
```

Esto es solo un pequeño ejemplo de cómo modificar la plantilla que nos proporciona Visual Studio para hacer un Login contra Linkar Server y posteriormente una Select para visualizar el resultado en pantalla.

Creando Apps móviles con Xamarin y MV usando Linkar 2.0
CON SERVICIO WEB o SIN ÉL.

Para un ejemplo mucho más completo puede descargarse la carpeta “Linkar Demos” en nuestra página https://www.kosday.com/Product/Download/7, y abre la carpeta LinkarViewFilesDirect dentro del archivo comprimido sources.zip.

GRACIAS

www.kosday.com

support@kosday.com