

# CURSO DE XAMARIN

## LAB.01

23/09/2017

### INFORMACIÓN GENERAL

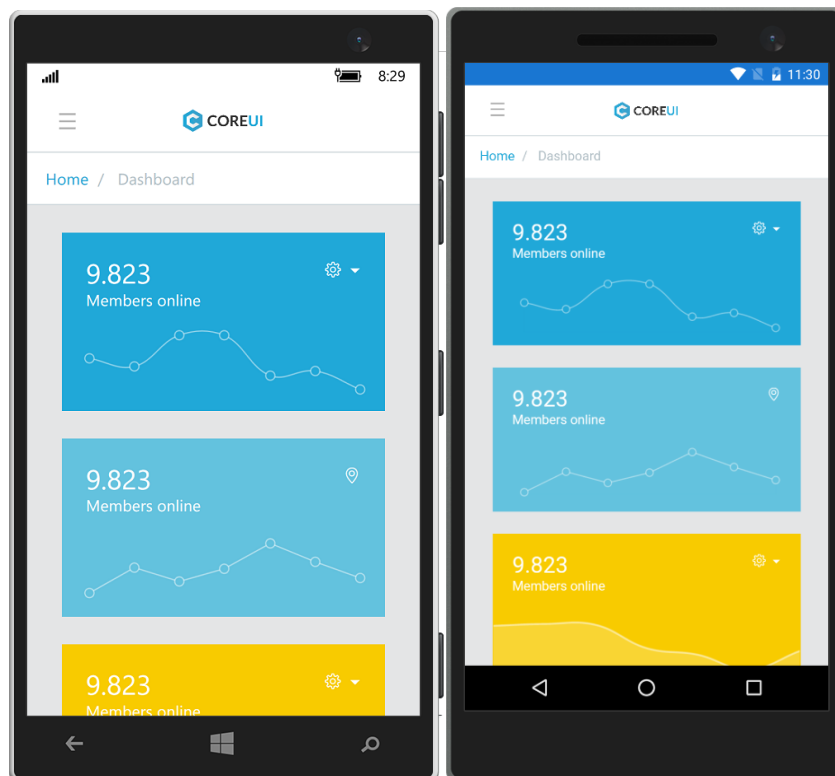
#### 1. Descripción del proyecto

Se creará una aplicación Xamarin Forms que permitirá embeber un sitio web dentro de una aplicación nativa por medio de un WebView.

Este primer laboratorio contiene variados comentarios sobre conceptos básicos con el objetivo de nivelar los conocimientos.

#### 2. Requisitos

- Visual Studio 2017 Community Edition o superior.

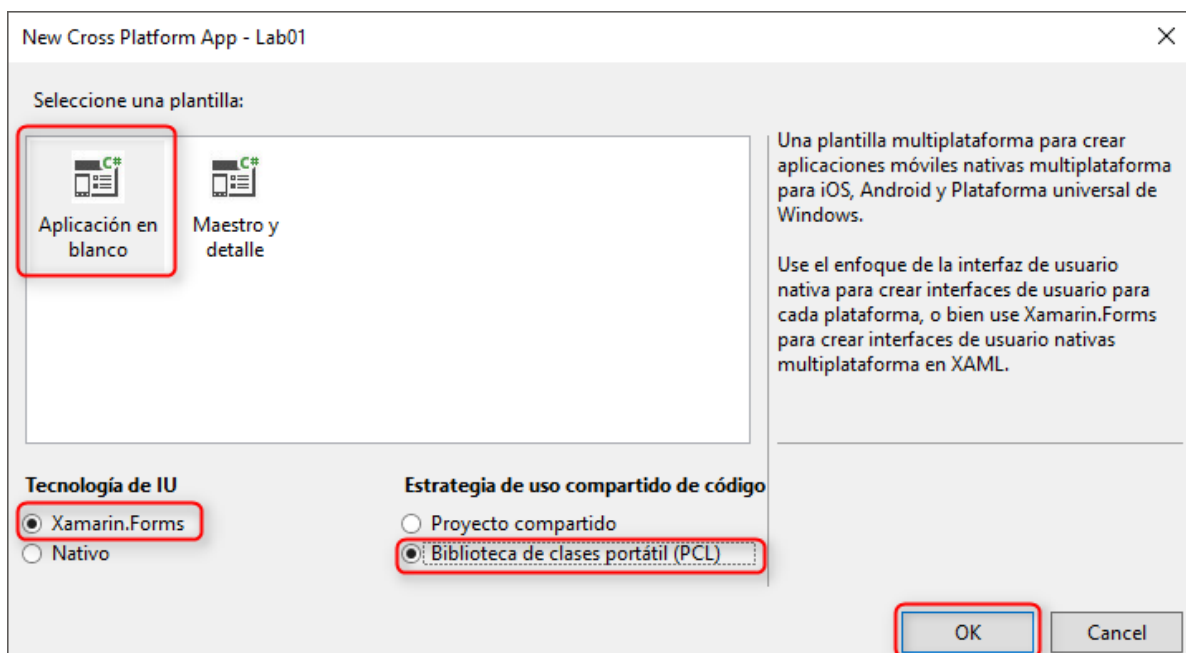
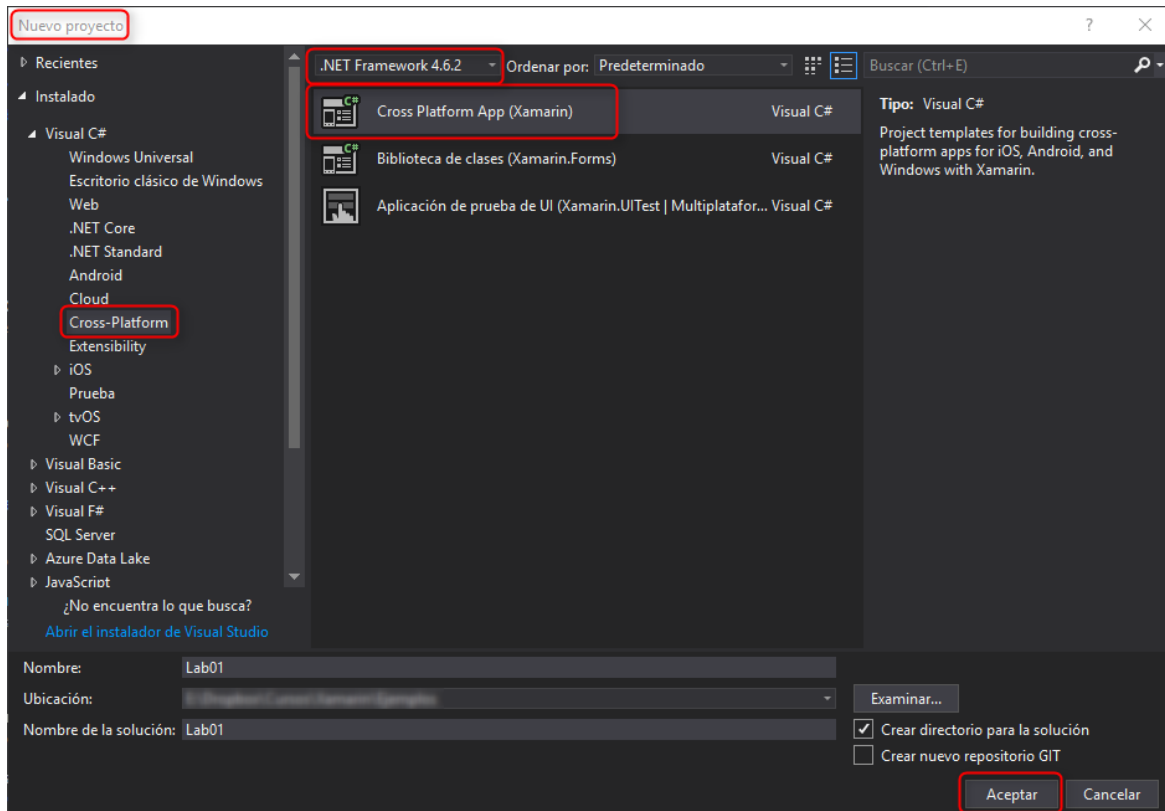


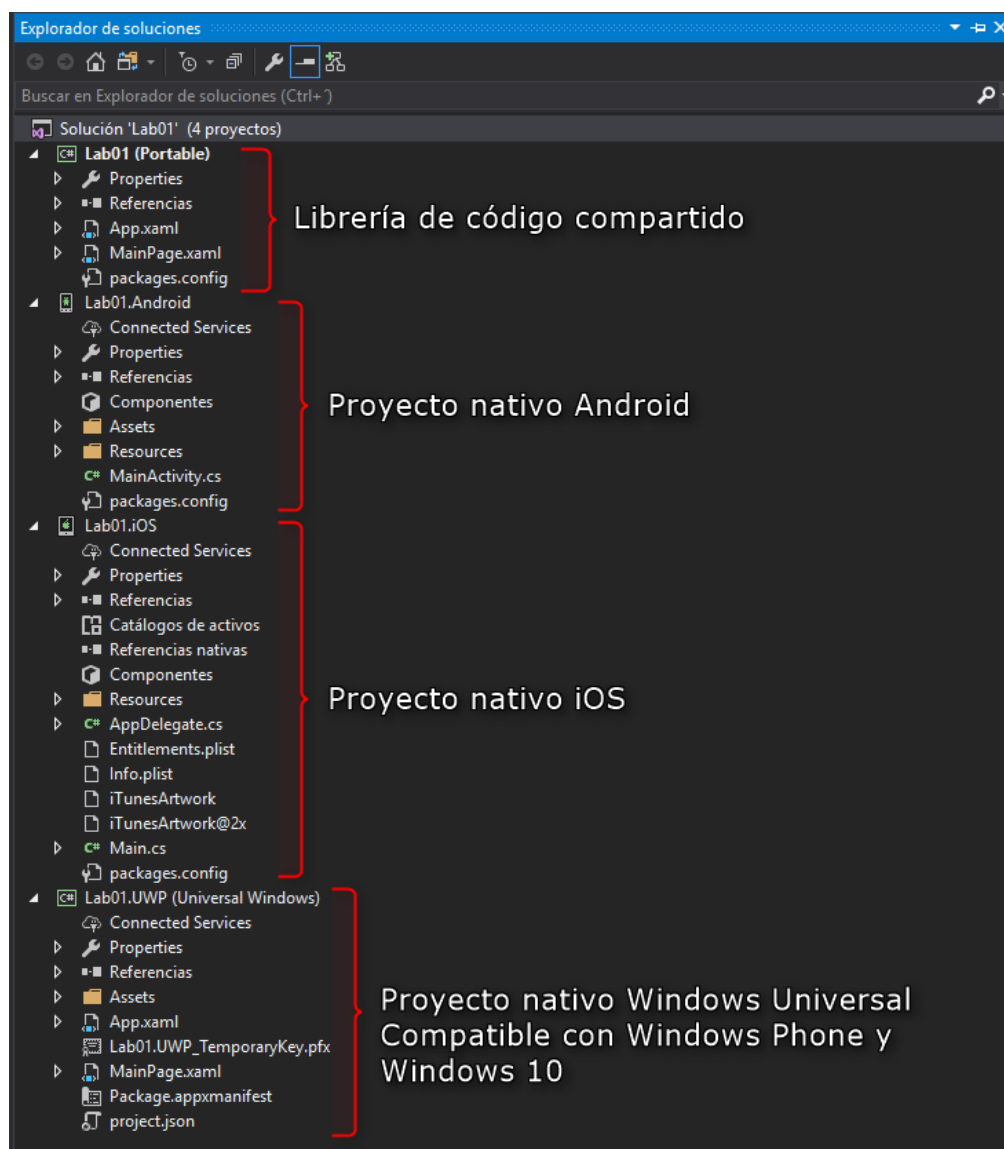
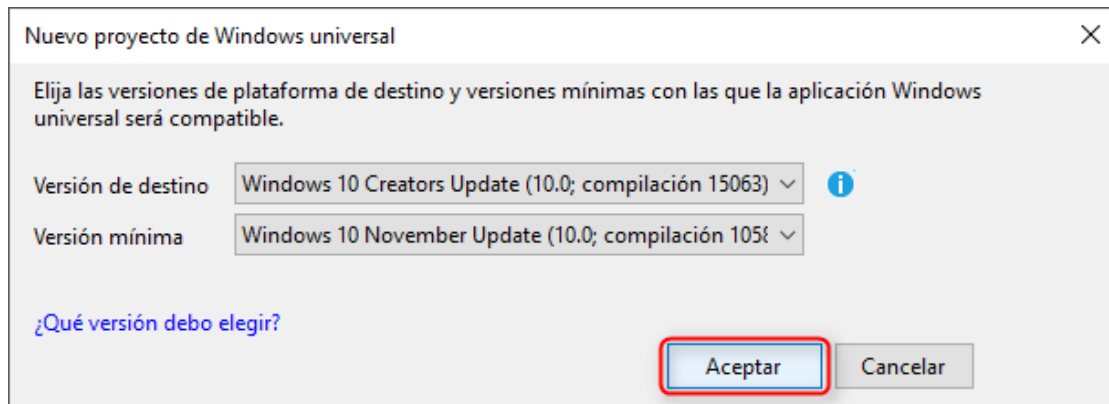
Resultado del laboratorio

### 3. Manos a la obra

#### Crear el proyecto

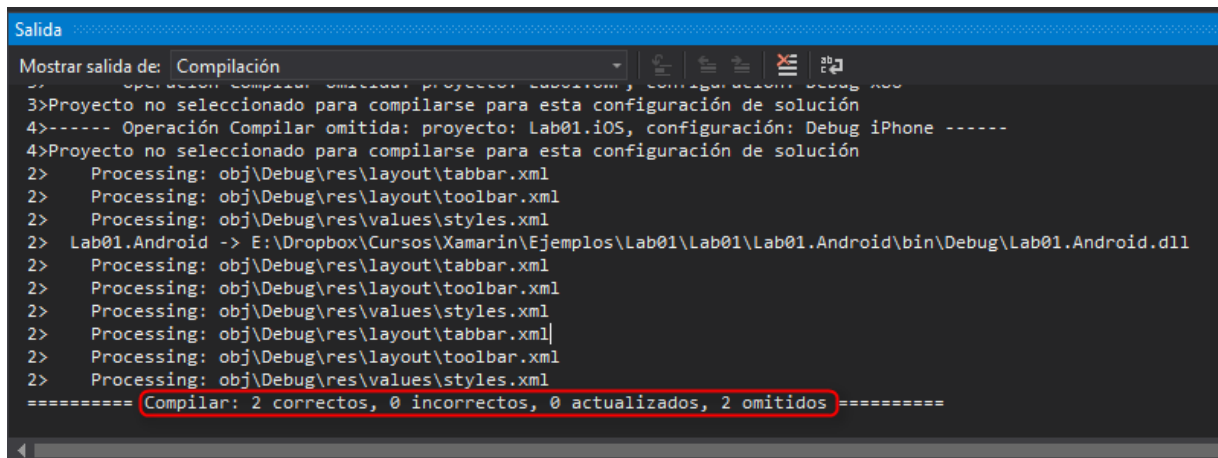
Abre VisualStudio como **Administrador** y crea un proyecto con nombre **Lab01**, siga la secuencia de imágenes para la creación.





## Compilar

Inmediatamente después de la creación del proyecto es recomendable realizar la primera compilación, antes de comenzar a codificar, con esto podrás darte cuenta si existe algún tipo de problema desde el inicio. Cuando no tienes el ambiente correctamente configurado es muy común ver casos en los cuales los proyectos no compilan con la plantilla inicial, los errores son muy variados y suelen solucionarse reinstalando el paquete con problemas.



```
Salida
Mostrar salida de: Compilación
3>Proyecto no seleccionado para compilarse para esta configuración de solución
4>----- Operación Compilar omitida: proyecto: Lab01.iOS, configuración: Debug iPhone -----
4>Proyecto no seleccionado para compilarse para esta configuración de solución
2> Processing: obj\Debug\res\layout\tabbar.xml
2> Processing: obj\Debug\res\layout\toolbar.xml
2> Processing: obj\Debug\res\values\styles.xml
2> Lab01.Android -> E:\Dropbox\Cursos\Xamarin\Ejemplos\Lab01\Lab01\Lab01.Android\bin\Debug\Lab01.Android.dll
2> Processing: obj\Debug\res\layout\tabbar.xml
2> Processing: obj\Debug\res\layout\toolbar.xml
2> Processing: obj\Debug\res\values\styles.xml
2> Processing: obj\Debug\res\layout\tabbar.xml
2> Processing: obj\Debug\res\layout\toolbar.xml
2> Processing: obj\Debug\res\values\styles.xml
=====[Compilar: 2 correctos, 0 incorrectos, 0 actualizados, 2 omitidos]=====
```

Es común que la primera compilación tarde varios minutos (1 a 4 minutos).

## Depuración

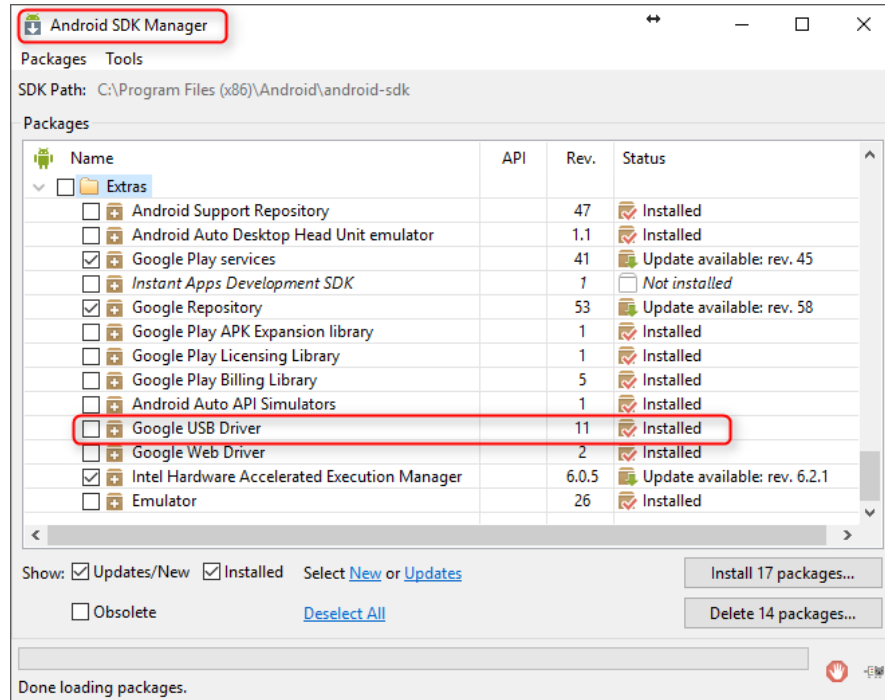
Tienes 2 posibilidades: utilizar un Emulador o un Dispositivo Físico.

Indistintamente de la elección que tomes, primero que todo debes seleccionar la plataforma que quieres usar, pero debes tener presente que existe algunos temas que debes tener claro antes de comenzar.

### Usando Visual Studio desde Windows:

- **Universal Windows:** Los emuladores corren sobre Hyper-V (incluida en Windows 10).
- **iOS:** Puedes compilar y depurar el proyecto desde Visual Studio corriendo en Windows, pero necesitas como prerequisite una computadora con Mac OS conectada en la misma Red y sincronizarlas. Una Mac vieja sirve, pero debes actualizar a la última versión del sistema operativo, lo mismo con Xcode y luego instalar el Simulador.

- **Android:** La depuración en dispositivos físicos (celulares o tabletas) solo funciona en Windows, debido a que el Driver no está disponible en Mac OS, pero en Mac puedes usar un emulador y funcionará sin problemas.



### Usando Visual Studio desde Mac:

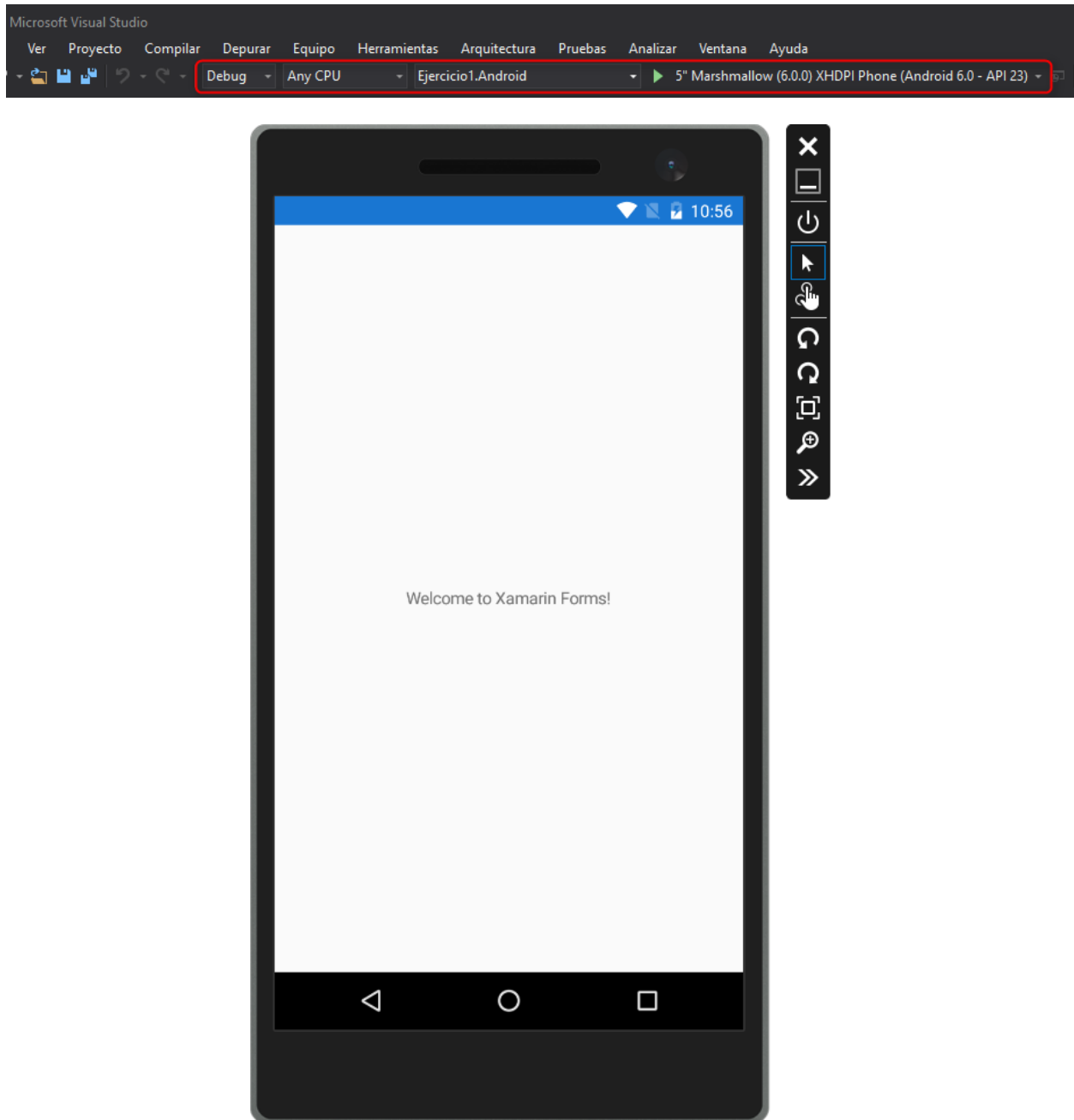
- **Universal Windows:** Este tipo de proyecto no es compatible en MAC OS.
- **iOS:** Debes actualizar a la última versión del sistema operativo, lo mismo con Xcode y luego instalar el Simulador.
- **Android:** La depuración solo funciona sobre el emulador.

Visual Studio trae incluidos varios emuladores para Android, pero en el caso de que estos no den el ancho, puedes instalar otros emuladores, de los cuales te recomiendo estos:

<https://freddylara.wordpress.com/2017/08/26/visual-studio-emulator-para-android/>

<https://freddylara.wordpress.com/2017/08/26/emuladores-para-android-xamarin-android-player/>

Como actualmente no hemos modificado el código, deberíamos ver algo similar a la siguiente imagen como resultado de la depuración sobre el emulador Android.

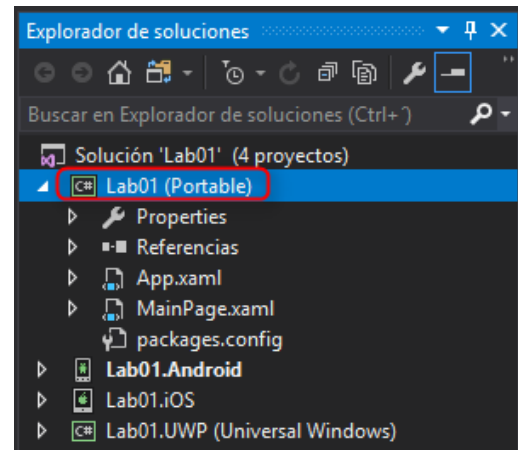


La imagen corresponde a VisualStudio Emulator for Android

<https://freddylara.wordpress.com/2017/08/26/emuladores-para-android-xamarin-android-player/>

## Implementar nuestro código

Ahora llegó el momento de incluir nuestras primeras líneas de código y en este caso vamos a hacer uso de la biblioteca de código compartido.



El archivo MainPage.xaml contiene el mensaje que vemos al momento de ejecutar la aplicación.

```
MainPage.xaml
ContentPage
1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3      xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4      xmlns:local="clr-namespace:Lab01"
5      x:Class="Lab01.MainPage">
6
7      <Label Text="Welcome to Xamarin Forms!"
8          VerticalOptions="Center"
9          HorizontalOptions="Center" />
10
11  </ContentPage>
```

Y ahora lo reemplazaremos por un control WebView.

```
MainPage.xaml
ContentPage
1  <?xml version="1.0" encoding="utf-8" ?>
2  <ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
3      xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
4      xmlns:local="clr-namespace:Lab01"
5      x:Class="Lab01.MainPage">
6
7      <WebView Source="http://coreui.io/demo/Angular2_Demo"
8          HorizontalOptions="FillAndExpand"
9          VerticalOptions="FillAndExpand">
10
11  </WebView>
12  </ContentPage>
```

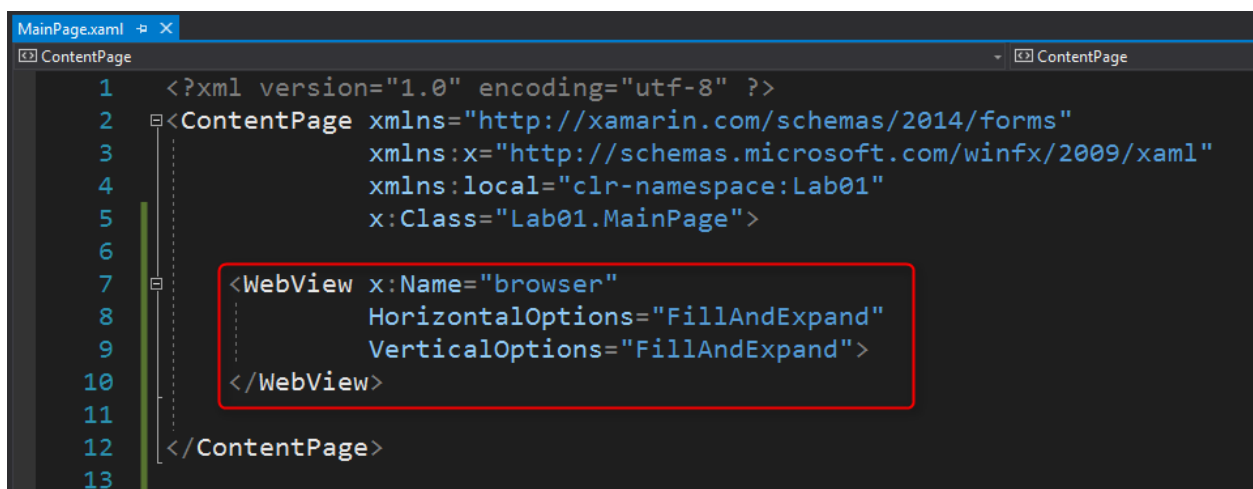
```
<WebView Source="http://coreui.io/demo/Angular2_Demo"
          HorizontalOptions="FillAndExpand"
          VerticalOptions="FillAndExpand">
</WebView>
```

## Implementar nuestro código sin acceso a internet

Existen variadas razones por las cuales renderizar HTML local, ya sea desde código o desde archivos locales.

Para más información: <https://developer.xamarin.com/guides/xamarin-forms/user-interface/webview/>

Se asigna un nombre al WebView para poder llamarlo desde el código.



```
<WebView x:Name="browser"
          HorizontalOptions="FillAndExpand"
          VerticalOptions="FillAndExpand">
</WebView>
```



```
MainPage.xaml.cs*  X
Lab01 Lab01.MainPage

8 namespace Lab01
9 {
10     4 referencias
11     public partial class MainPage : ContentPage
12     {
13         1 referencia
14         public MainPage()
15         {
16             InitializeComponent();
17
18             var htmlSource = new HtmlWebViewSource();
19             htmlSource.Html = @"<html><body>
20             <h1>Xamarin.Forms</h1>
21             <p>Welcome to WebView.</p>
22             </body></html>";
23             browser.Source = htmlSource;
24         }
25     }
```

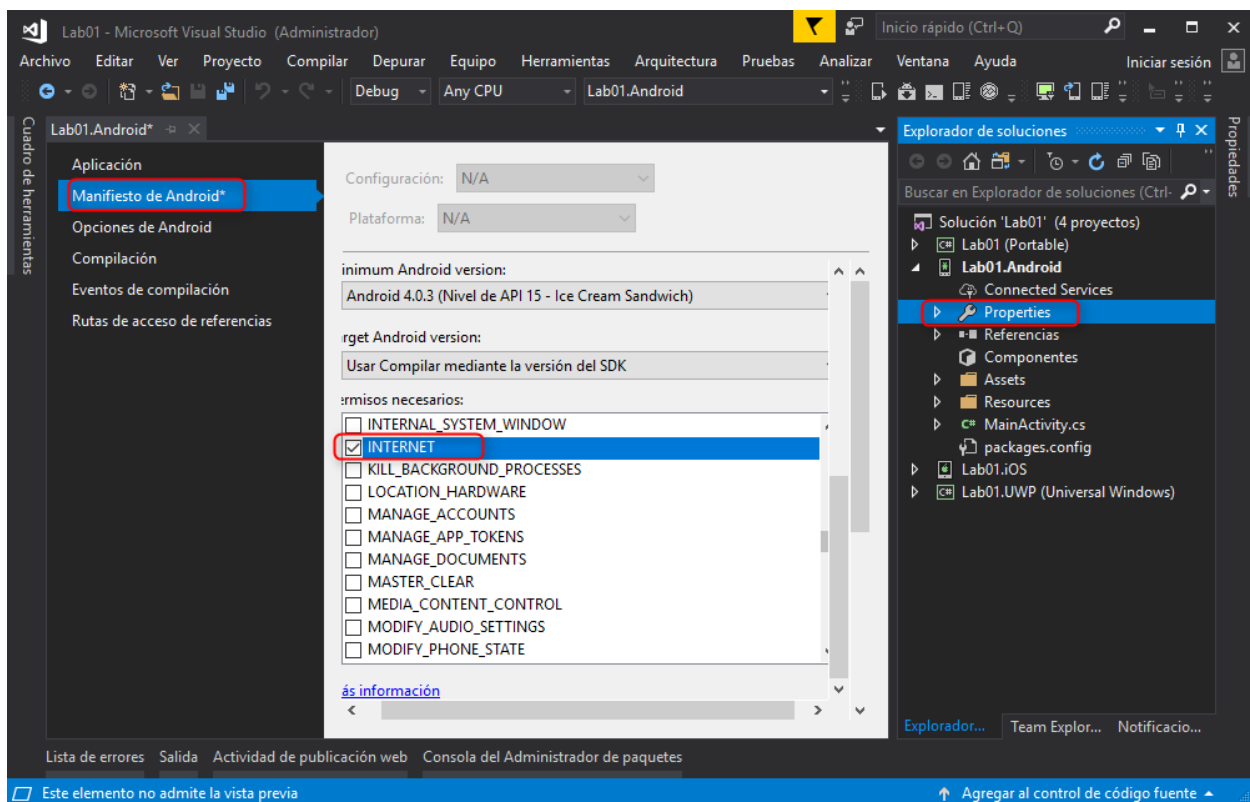
## Android

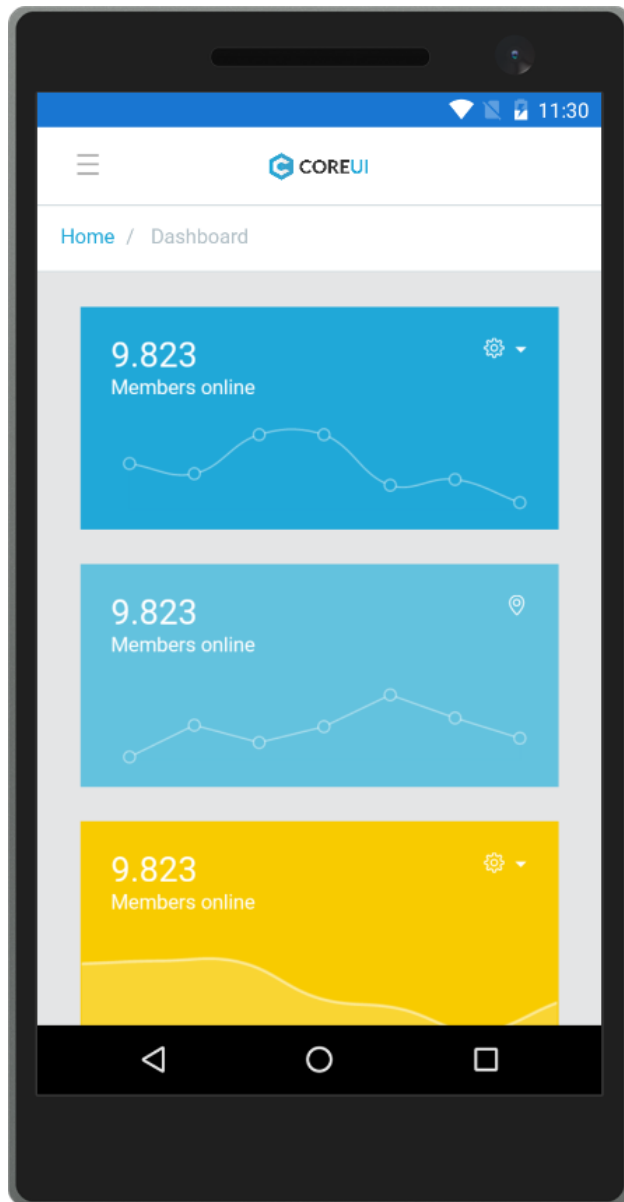
Ahora al ejecutar nuevamente la depuración, verás una aplicación web construida con Bootstrap + Angular embebida dentro de una aplicación nativa.

IMPORTANTE:

Si lo estas corriendo en tu emulador y no te funciona, muy posiblemente no tengas acceso a internet, revisa el siguiente link para configurar correctamente la Wifi de tu emulador: <https://freddylara.wordpress.com/2017/08/26/emulador-android-en-hyper-v-sin-conexion-a-internet/>

Debes incluir los permisos de internet en el manifiesto, mientras estés en modo depuración funcionará sin problemas, aunque no tengas asignado el permiso de internet, pero al momento de instalarlo en el dispositivo móvil será necesario tener correctamente asignados los permisos.

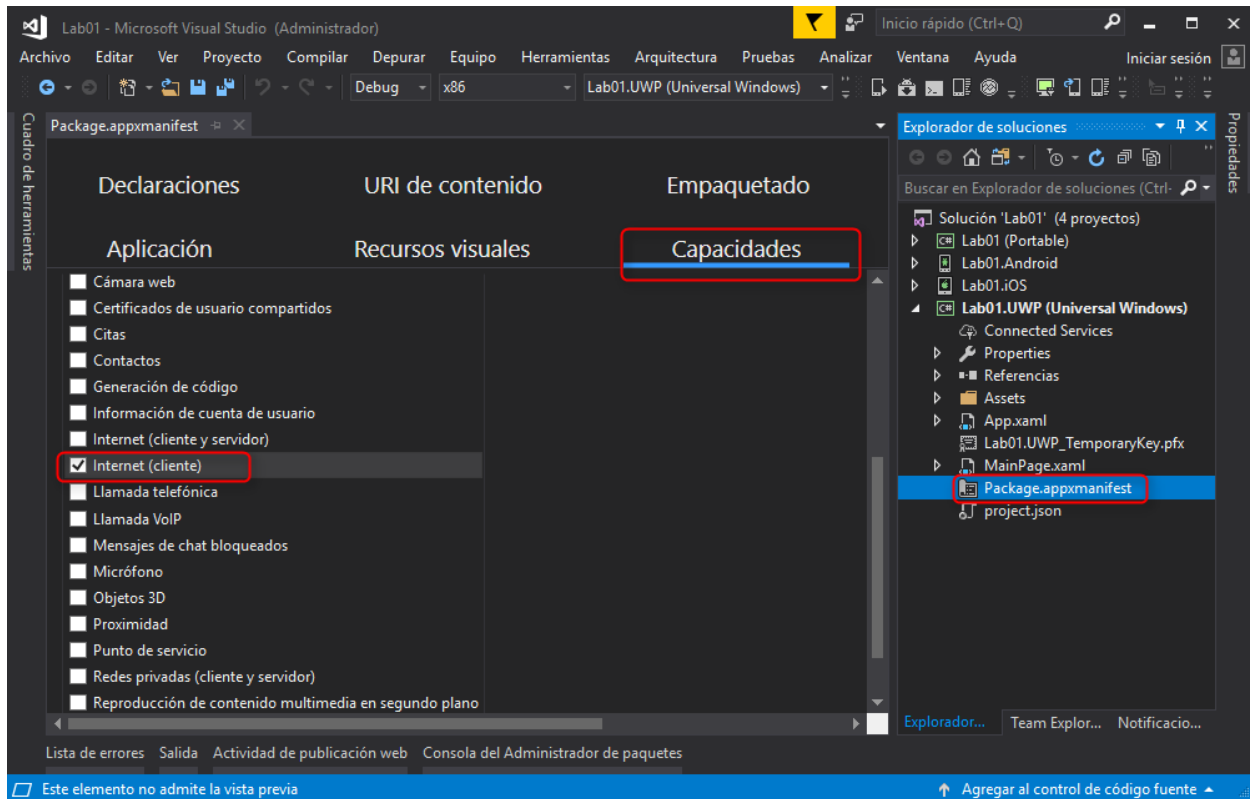




Resultado de la ejecución en Android

## Universal Windows

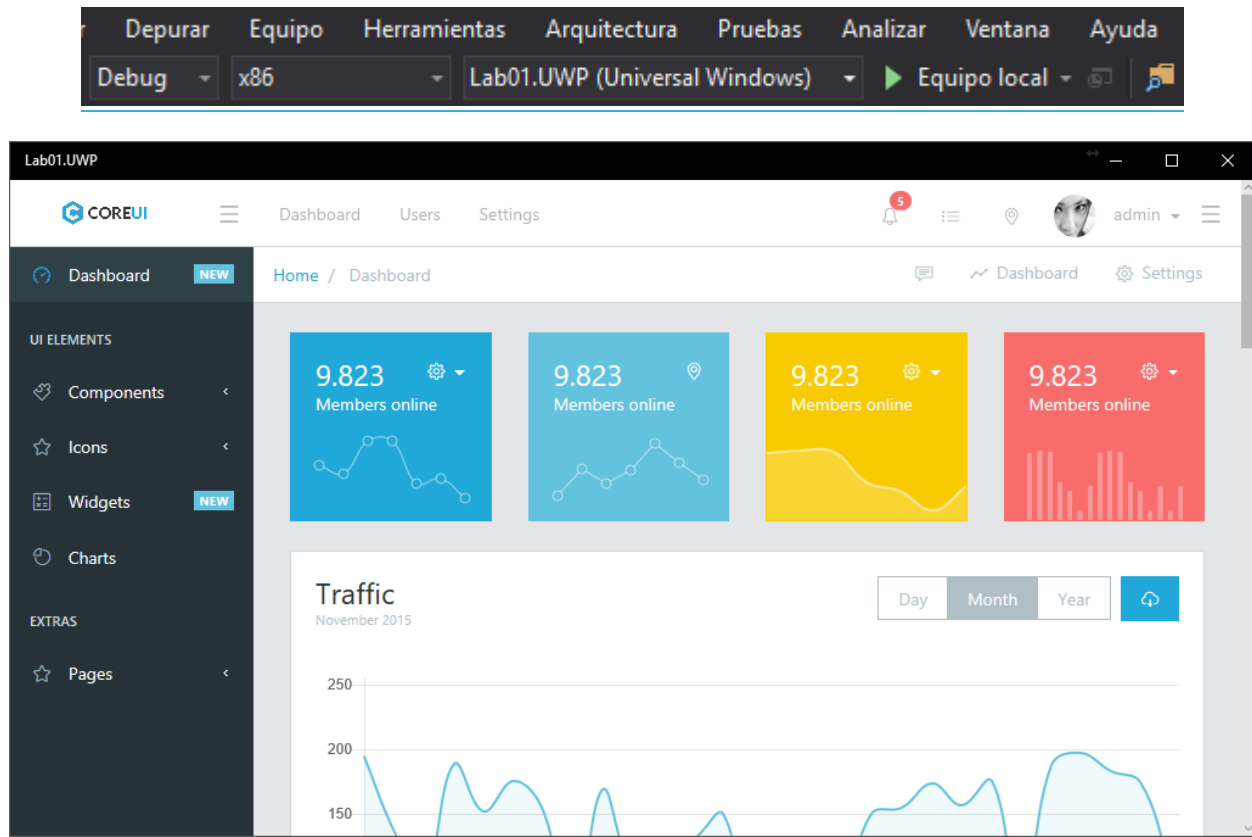
Debes incluir los permisos de internet en el manifiesto, aunque para Universal Windows estos ya vienen incluidos por defecto.



La configuración para Windows presenta 2 modos de ejecución:

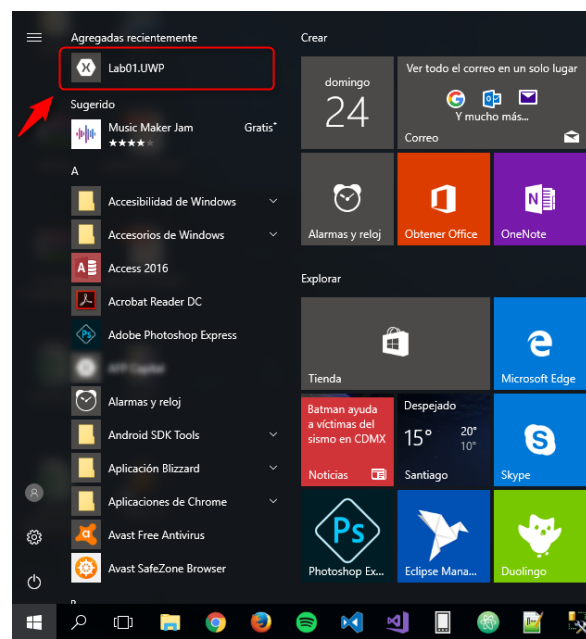
- **Equipo local:** La aplicación se ejecutará directamente en Windows.
- **Emulador:** La aplicación se ejecutará en el emulador de WindowsPhone.

## Equipo Local (Windows)

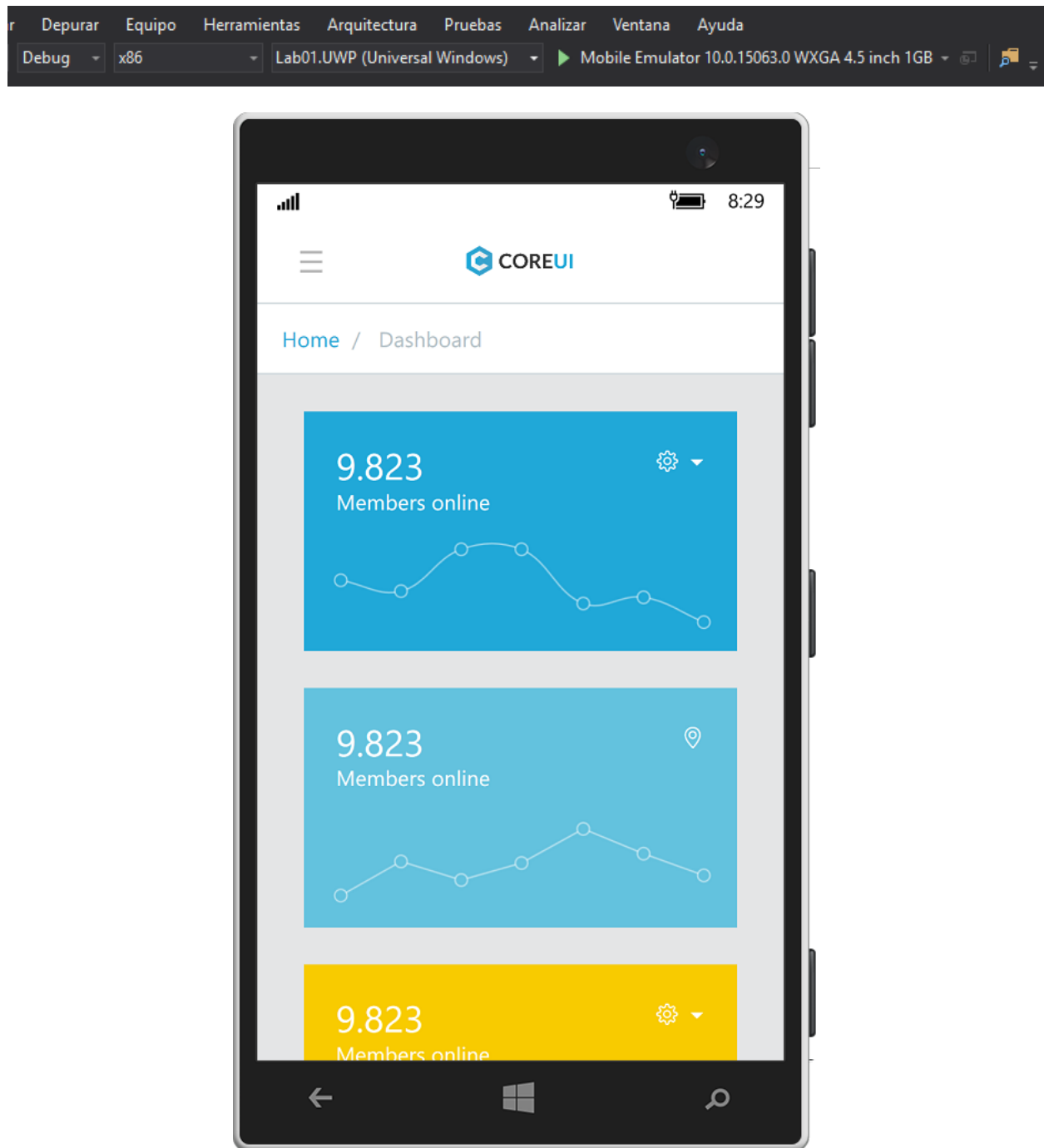


Resultado de la ejecución en Windows 10

La aplicación queda automáticamente anclada en el menú de inicio y puede ser ejecutada sin la necesidad de abrir VisualStudio.



## Emulador (Windows)

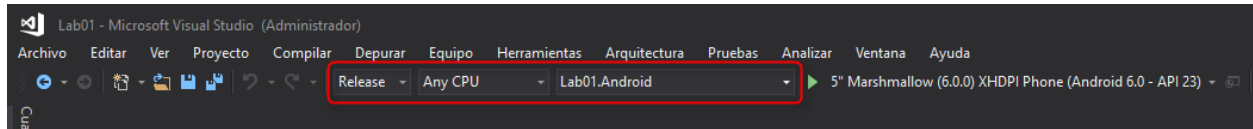


Resultado de la ejecución en Windows Phone

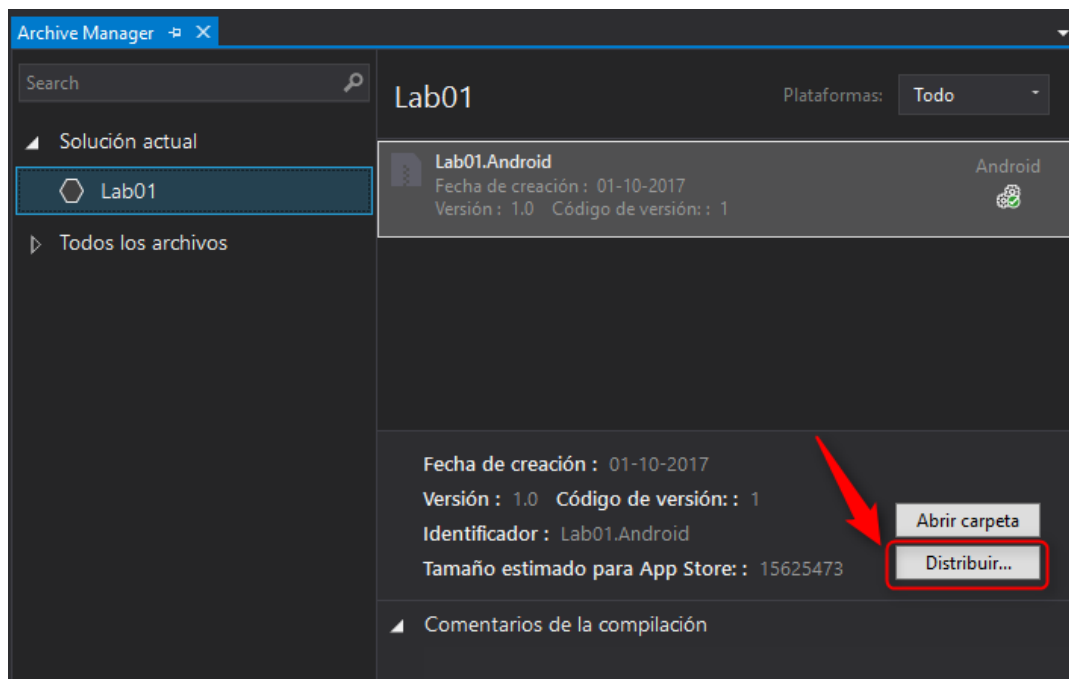
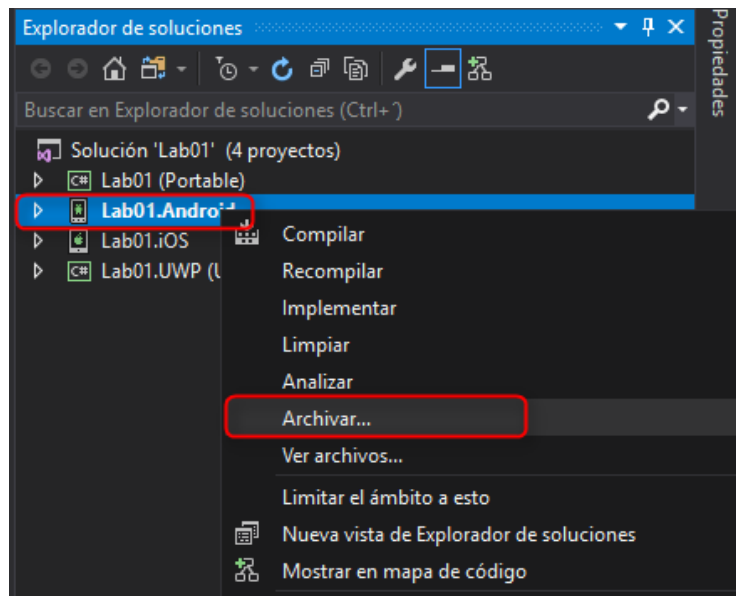
## 4. Publicación

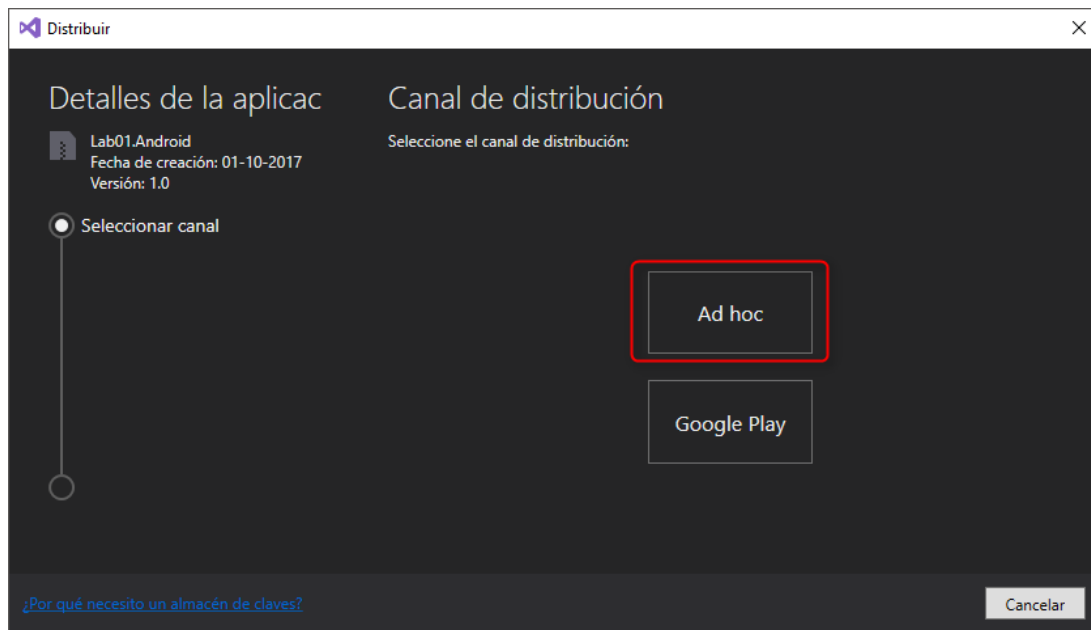
### Android

Primero que todo es necesario cambiar a modo **Release** y compilar nuevamente.

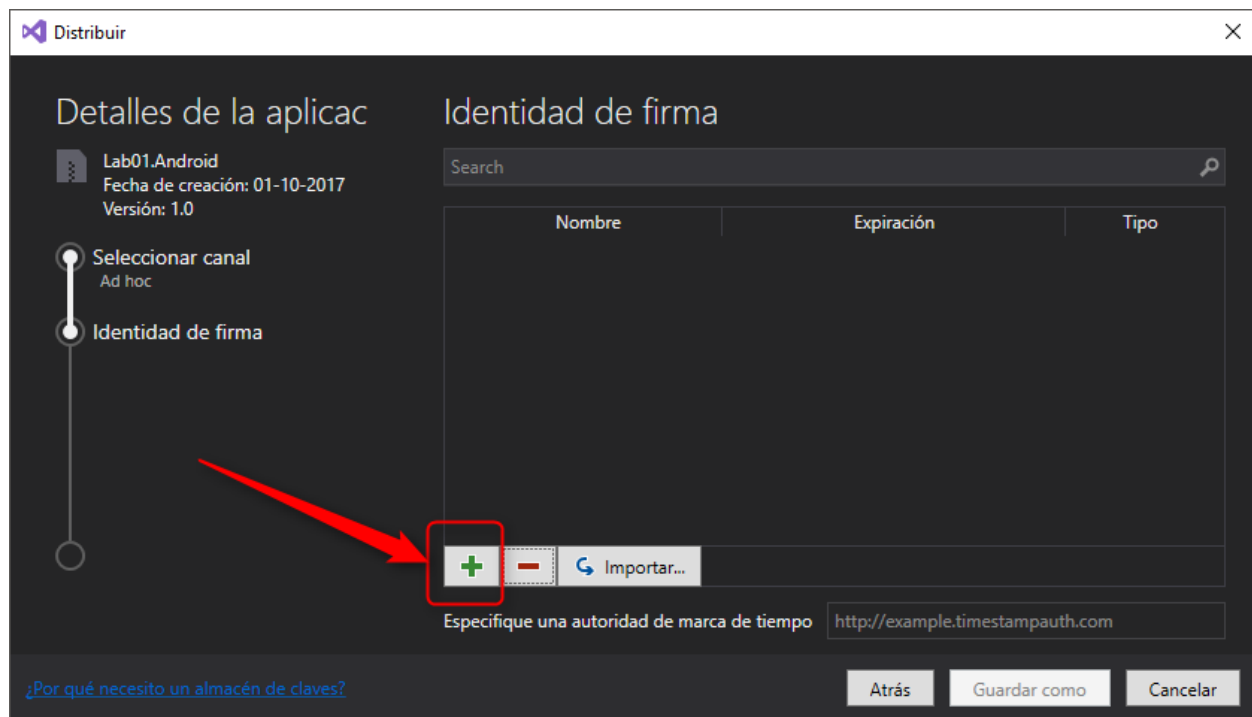


Luego, sobre el proyecto Android, seleccionar la opción **Archivar**.





Es necesario indicar una identidad de firma.





Almacén de claves Android

### Crear almacén de claves Android

Alias:

Contraseña:  Confirmar:

Validez:  (Años)

<b>Especifique al menos uno de los siguientes valores</b>

Nombre completo:

Unidad organizativa:

Organización:

Ciudad o localidad:

Estado o provincia:

Código de país:  (2 dígitos)

[¿Qué es un almacén de claves?](#)

Distribuir

### Detalles de la aplicac

Lab01.Android  
Fecha de creación: 01-10-2017  
Versión: 1.0

Seleccionar canal  
Ad hoc

Identidad de firma

### Identidad de firma

Search

Nombre	Expiración	Tipo
mirepositorio	Tue Sep 24 21:49:10 CLST 2047	

+ - Importar...

Especifique una autoridad de marca de tiempo

[¿Por qué necesito un almacén de claves?](#)

Luego de eso te solicitará una carpeta en la cual dejar el APK y debes poner la clave del repositorio cuando lo pida.

## Más plantillas Bootstrap que pueden utilizadas

Te recomiendo estas 2 plantillas:

<https://adminlte.io/themes/AdminLTE/index2.html>

Bootstrap

[http://coreui.io/demo/Angular2\\_Demo](http://coreui.io/demo/Angular2_Demo)

Bootstrap + Angular

En los siguientes links podrás encontrar una infinidad de plantillas:

<https://cssauthor.com/responsive-free-angularjs-admin-themes/>

<https://cssauthor.com/bootstrap-admin-templates/>

## Para más detalles sobre WebView:

<https://developer.xamarin.com/guides/xamarin-forms/user-interface/webview/>