



Laboratorio

Manejo de Recursos en Android

Versión: 1.0.0
Mayo de 2017



[Miguel Muñoz Serafín](#)
@msmdotnet





CONTENIDO

INTRODUCCIÓN

EJERCICIO 1: MANEJANDO RECURSOS EN ANDROID

Tarea 1. Crear una aplicación Android.

Tarea 2. Crear recursos.

Tarea 3. Acceder a los recursos programáticamente.

Tarea 4. Acceder a los recursos desde XML.

EJERCICIO 2: VALIDANDO TU ACTIVIDAD

Tarea 1. Agregar los componentes de la Capa de acceso a Servicio.

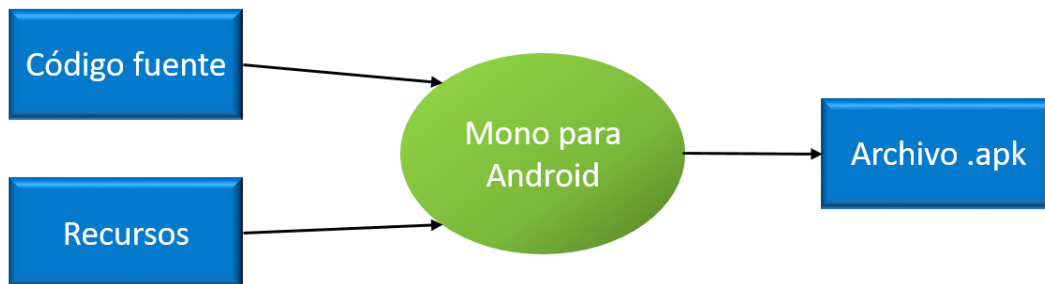
Tarea 2. Agregar la funcionalidad para validar la actividad.

RESUMEN



Introducción

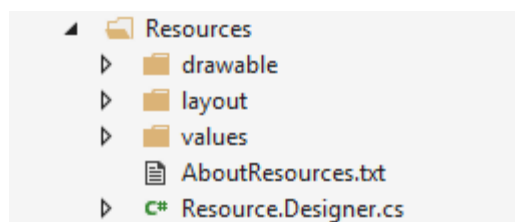
Una aplicación Android rara vez está compuesta de solo código. Frecuentemente podemos encontrar otros archivos que componen una aplicación: videos, imágenes, archivos de audio, etc. Colectivamente, esos archivos que no contienen código fuente son llamados **Recursos** y son compilados junto con el código fuente durante el proceso de compilación y se empaquetan como un archivo APK para su distribución e instalación en los dispositivos.



Los recursos ofrecen varias ventajas en una aplicación Android:

- **Separación de código.** Los recursos permiten separar del código fuente a las imágenes, cadenas, menús, animaciones, colores, etc. De esta forma, trabajar con recursos puede ayudar considerablemente al proceso de globalización y localización de una aplicación.
- **Desarrollo para múltiples dispositivos.** Los recursos proporcionan soporte para desarrollar aplicaciones para diferentes configuraciones de dispositivos sin necesidad de realizar cambios en el código.
- **Verificación en tiempo de compilación.** Los recursos son estáticos y compilados dentro de la aplicación. Esto permite que el uso de los recursos pueda ser verificado en tiempo de compilación cuando es más fácil detectar y corregir errores en lugar de hacerlo en tiempo de ejecución cuando es más difícil de localizarlos y más costosos de corregir.

Cuando un nuevo proyecto Xamarin.Android es creado, un directorio especial llamado **Resources** es creado junto con algunos subdirectorios.



Los recursos de la aplicación se organizan de acuerdo a su tipo, por ejemplo, las imágenes van en el directorio **drawable** mientras que las vistas van en el subdirectorio **layout**.



Estos recursos son llamados **Recursos Predeterminados (Default Resources)** y son utilizados por todos los dispositivos a menos que se especifique una coincidencia más específica. Adicionalmente, cada tipo de recurso puede tener opcionalmente **Recursos Alternativos (Alternate Resources)** que Android puede utilizar al ejecutarse en dispositivos específicos. Por ejemplo, se pueden proporcionar recursos para adecuarse a la configuración regional del usuario, al tamaño de la pantalla o para el caso en que el dispositivo sea girado 90 grados de vertical a horizontal, etc. En cada uno de estos casos, Android cargará los recursos para que la aplicación los utilice sin ningún esfuerzo adicional de codificación por parte del desarrollador.

Hay dos formas de acceder a los recursos en una aplicación Xamarin.Android: **programáticamente** en código y **declarativamente** en XML usando una sintaxis especial de XML.

En este laboratorio tendrás la oportunidad de explorar los conceptos y tareas básicas del manejo de recursos en Android.

Objetivos

Al finalizar este laboratorio, los participantes serán capaces de:

- Utilizar los recursos predeterminados de Android programática y declarativamente.
- Agregar recursos tales como imágenes o texto a una aplicación Android.

Requisitos

Para la realización de este laboratorio es necesario contar con lo siguiente:

- Un equipo de desarrollo con Visual Studio. Los pasos descritos en este laboratorio fueron realizados con Visual Studio 2017 y Windows 10 Professional, sin embargo, los participantes pueden utilizar la versión de Visual Studio 2015 que ya tengan instalada.
- Xamarin para Visual Studio.

Tiempo estimado para completar este laboratorio: **60 minutos**.



Ejercicio 1: Manejando recursos en Android

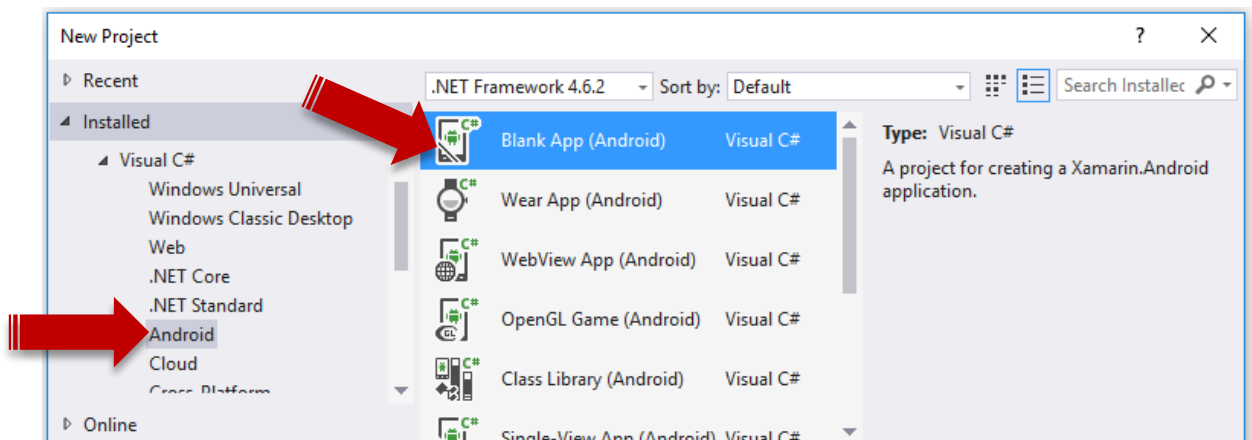
Casi todas las aplicaciones de Android tienen algún tipo de recurso en ellas, por lo menos, frecuentemente tienen los diseños de interfaz de usuario en forma de archivos XML.

En este ejercicio explorarás la forma en que los recursos predeterminados son estructurados en una aplicación Android. Aprenderás también a crear y acceder a los recursos de forma programática y de forma declarativa.

Tarea 1. Crear una aplicación Android.

Realiza los siguientes pasos para crear una aplicación Android desde Visual Studio.

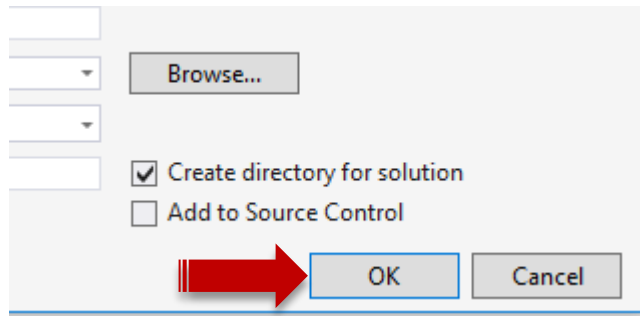
1. Abre Visual Studio en el contexto del Administrador.
2. Selecciona la opción **File > New > Project**.
3. En la ventana **New Project** selecciona la plantilla **Blank App (Android)**.



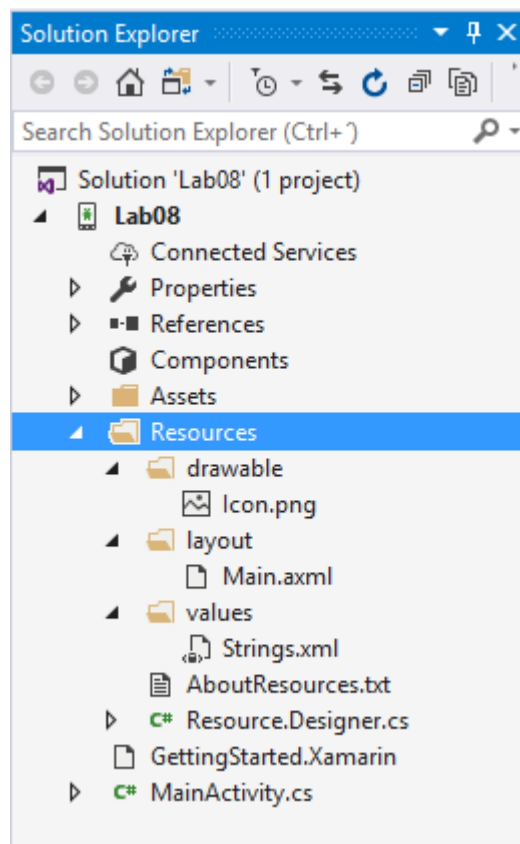
4. Proporciona el nombre y ubicación del proyecto.

Name:	Lab08
Location:	C:\demos\
Solution:	Create new solution
Solution name:	Lab08

5. Haz clic en **OK** para crear el proyecto.



Cuando se crea una aplicación Xamarin.Android, los *Recursos Predeterminados* son configurados por la plantilla de proyecto Xamarin.Android.



5 de los archivos que forman parte de los *Recursos Predeterminados* son creados en el directorio **Resources** y son los siguientes:

- **Icon.png**. El icono predeterminado para la aplicación.
- **Main.xml**. Es el archivo predeterminado de diseño de interfaz de usuario de una aplicación. Notemos que mientras que Android utiliza la extensión de archivo **.xml**, Xamarin.Android utiliza la extensión de archivo **.axml**.

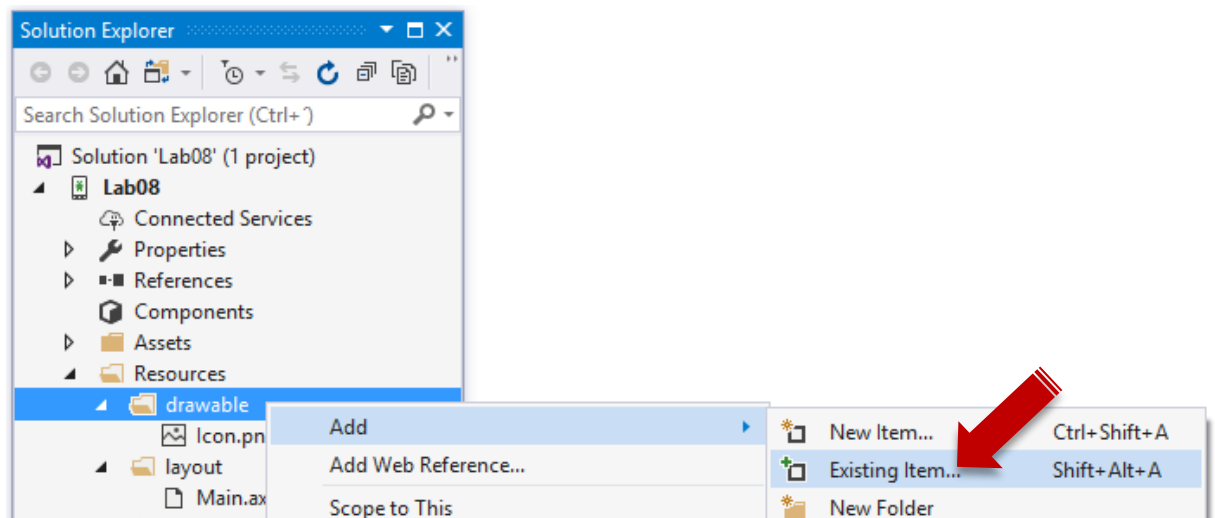


- **Strings.xml**. Un grupo de cadenas para ayudar con la globalización y localización de la aplicación.
- **AboutResources.txt**. Este archivo no es necesario y puede eliminarse sin problema. Sólo proporciona una descripción general del folder **Resources** y los archivos que contiene.
- **Resource.designer.cs**. Este archivo es generado y administrado automáticamente por Xamarin.Android, contiene los identificadores únicos asignados a cada recurso. Esto es muy similar e idéntico en propósito al archivo *R.java* que tendría una aplicación Android escrita en Java. Es creado automáticamente por las herramientas de Xamarin.Android y es regenerado cuando realizamos cambios en los recursos.

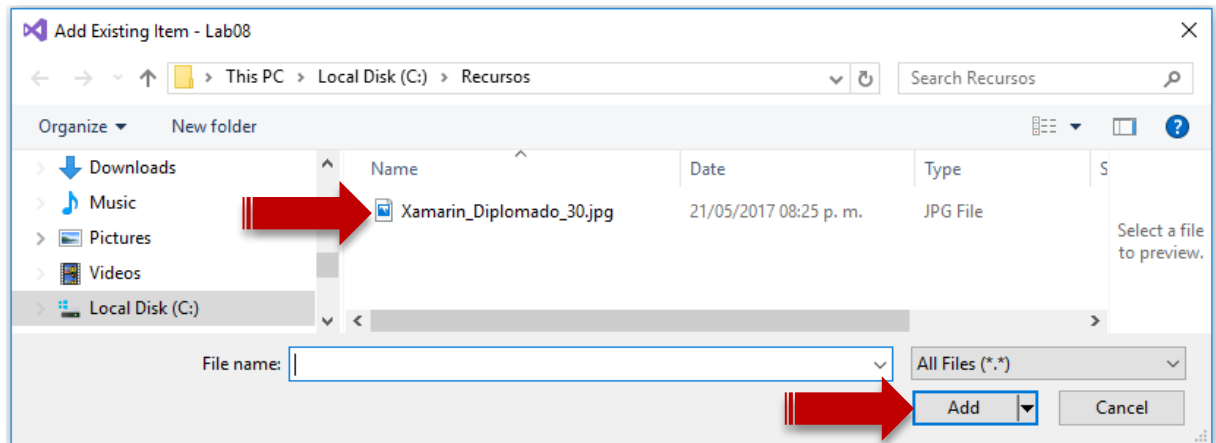
Tarea 2. Crear recursos.

La creación de recursos es tan simple como agregar archivos al directorio para el tipo de recurso en cuestión. Por ejemplo, para agregar una imagen como recurso, podemos agregar el archivo de imagen al directorio *drawable*.

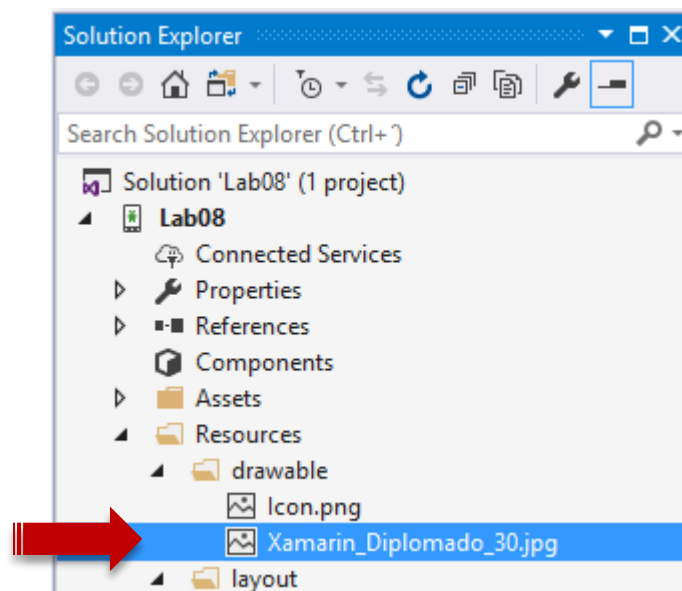
1. Selecciona la opción **Add > Existing Item...** del menú contextual del folder **Resources\drawable**. Esto te permitirá agregar un nuevo archivo de imagen como recurso.



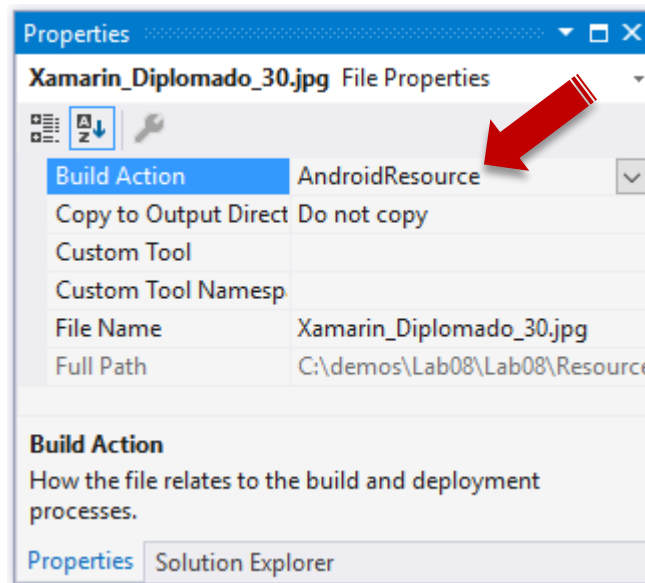
2. En la ventana **Add Existing Item** selecciona el archivo *Xamarin_Diplomado_30.jpg* adjunto a este documento y haz clic en **Add** para agregarlo.



El explorador de soluciones de Visual Studio mostrará el recurso imagen agregado.

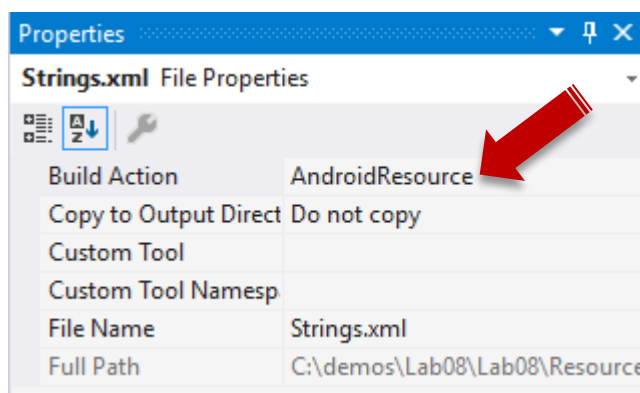


3. Haz clic en el archivo agregado y presiona **F4** para mostrar la ventana de propiedades. Podrás notar que cuando el archivo fue agregado, las herramientas de Xamarin.Android establecieron automáticamente la acción de compilación con el valor **AndroidResource**. Esto permite que las herramientas Xamarin.Android compilen e integren correctamente los recursos en el archivo APK. Si por alguna razón la acción de compilación no está establecida en **AndroidResource**, el archivo no será incluido en el APK y cualquier intento de cargar o acceder al recurso resultará en un error en tiempo de ejecución y la aplicación fallará.



Además, es importante tener en cuenta que mientras que Android sólo admite nombres de archivos en minúsculas para elementos de recursos, Xamarin.Android es un poco más indulgente ya que soporta nombres de archivos en mayúsculas y minúsculas como el archivo que acabas de agregar. La convención para nombres de imágenes es usar minúsculas con guion bajo como separadores (por ejemplo, xamarin_diplomado_30.jpg). Toma en cuenta que los nombres de recursos no se pueden procesar si se utilizan guiones o espacios como separadores.

4. Selecciona el archivo *Resources\values\Strings.xml* y presiona **F4** para mostrar la ventana de propiedades. Puedes notar que la acción de compilación de este archivo es **AndroidResource**.



Como mencionamos anteriormente, el archivo *Strings.xml* es uno de los archivos de recursos predeterminados y contiene cadenas de texto como recursos.

5. Abre el archivo *Strings.xml* y realiza los siguientes cambios:
 - a. Elimina el recurso **Hello**.



- b. Modifica el valor del recurso **ApplicationName** por el valor “Manejo de recursos en Android”
- c. Agrega el recurso **UserName** con el valor “Nombre del usuario”

El contenido del archivo *Strings.xml* será similar al siguiente.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="ApplicationName">Manejo de recursos en Android</string>
    <string name="UserName">Nombre del usuario</string>
</resources>
```

6. Guarda los cambios y compila la aplicación.

Una vez que se han añadido recursos a un proyecto, hay dos formas de utilizarlos en una aplicación, programáticamente (dentro del código) o desde archivos XML.

Tarea 3. Acceder a los recursos programáticamente.

Para permitir el acceso a un recurso mediante programación, las herramientas de Xamarin.Android asignan a cada recurso un ID de recurso único. Este ID de recurso es un número entero definido en una clase especial llamada **Resource** que se encuentra en el archivo *Resource.designer.cs*.

1. Abre el archivo *Resource.designer.cs*.

Cada ID de recurso está contenido dentro de una clase anidada que corresponde al tipo de recurso. Por ejemplo, cuando el archivo *Icon.png* fue agregado al proyecto, Xamarin.Android actualizó la clase *Resource*, creando una clase anidada llamada *Drawable* conteniendo una constante llamada *Icon*. Esto permite que el archivo *Icon.png* pueda ser referenciado en el código como *Resource.Drawable.Icon*. La clase *Resource* no debe ser editada manualmente ya que cualquier cambio que se realice en ella será sobrescrito por Xamarin.Android.

2. Localiza el identificador del recurso imagen que agregaste en la tarea anterior. El código debe ser similar al siguiente.

```
// aapt resource value: 0x7f020001
public const int Xamarin_Diplomado_30 = 2130837505;
```

Puedes notar que la herramienta Android Asset Packaging Tool (aapt) ha asignado el valor del identificador ID al recurso imagen.

3. Localiza el identificador del recurso *UserName*. El código debe ser similar al siguiente.

```
// aapt resource value: 0x7f040002
public const int UserName = 2130968577;
```

Cuando se hace referencia a recursos mediante programación (en código), se puede acceder a ellos mediante la jerarquía de la clase *Resource* que utiliza la siguiente sintaxis:



`@[<PackageName>.]Resource.<ResourceType>.<ResourceName>`

- **PackageName.** El paquete que proporciona el recurso y sólo se requiere cuando se utilizan recursos de otros paquetes.
- **ResourceType.** Es el tipo de recurso anidado que está dentro de la clase *Resource*, por ejemplo *Drawable*, *Layout* o *String*.
- **ResourceName.** Este es el nombre de archivo del recurso (sin la extensión) o el valor del atributo *android:name* para recursos que están en un elemento XML.

Cuando se hace referencia a recursos dentro de cadenas de texto, podemos utilizar la siguiente sintaxis (la sintaxis que se utiliza para referenciar recursos desde XML):

`@[<PackageName>:]<ResourceType>/<ResourceName>`

4. Abre el archivo **MainActivity.cs**.
5. Observa que el valor para el Icono de la aplicación hace referencia al recurso *Resources\drawable\icon.png* con la sintaxis **@drawable/icon**.

```
Icon = "@drawable/icon"
```

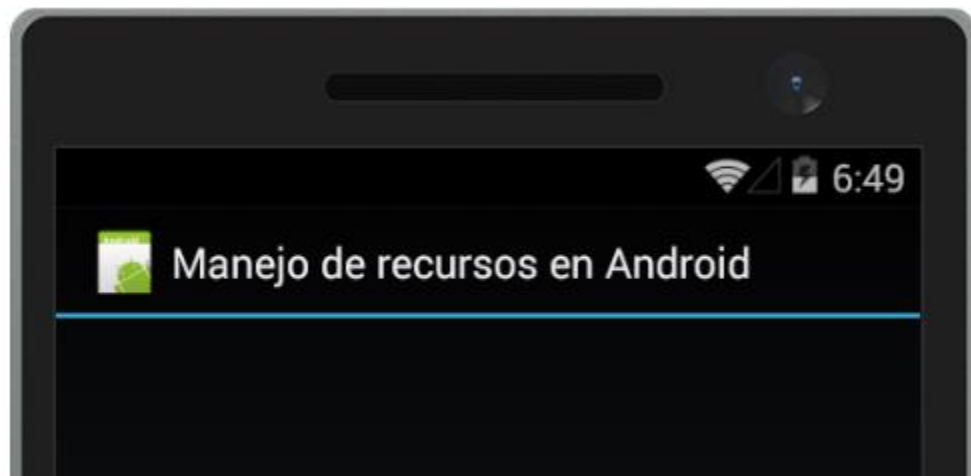
6. Modifica el valor del parámetro *Label* para que el recurso cadena *ApplicationName* sea utilizado.

```
Label = "@string/ApplicationName"
```

El código se verá similar a lo siguiente.

```
[Activity(Label = "@string/ApplicationName", MainLauncher = true, Icon = "@drawable/icon")]
```

7. Ejecuta la aplicación. Se mostrará una pantalla similar a la siguiente. Puedes notar que los recursos son mostrados.



8. Quita el comentario a la línea dentro del método **OnCreate** que establece el diseño de interfaz de usuario de la Activity.



```
protected override void OnCreate(Bundle bundle)
{
    base.OnCreate(bundle);

    // Set our view from the "main" layout resource
    SetContentView (Resource.Layout.Main);
}
```

Puedes notar la sintaxis utilizada para hacer referencia al recurso `Resources\layout\Main.xml` desde código: **`Resource.Layout.Main`**.

9. Agrega el siguiente código para obtener la vista raíz de la actividad actual.

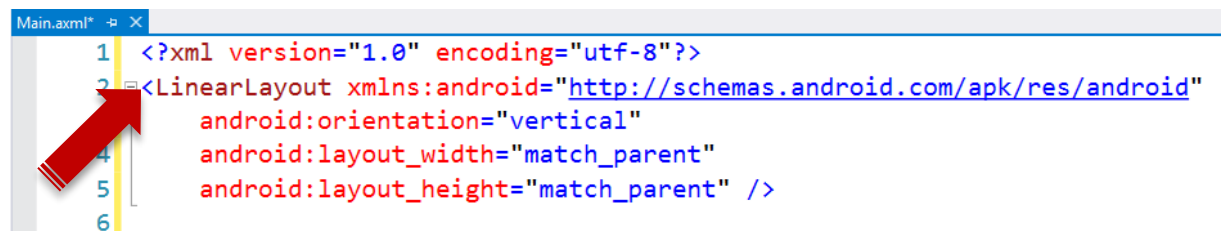
```
var ViewGroup = (Android.Views.ViewGroup)
    Window.DecorView.FindViewById(Android.Resource.Id.Content);
```

La constante **`Android.Resource.Id.Content`** hace referencia a la vista raíz de la actividad actual.

10. Agrega el siguiente código para obtener la vista que agregaste a la actividad mediante el método `SetContentView`.

```
var MainLayout = ViewGroup.GetChildAt(0) as LinearLayout;
```

Esta instrucción te permitirá obtener una referencia del diseño de interfaz de usuario *Main* para que le puedas agregar más elementos en tiempo de ejecución.



```
Main.xml*
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     android:orientation="vertical"
4     android:layout_width="match_parent"
5     android:layout_height="match_parent" />
6
```

11. Agrega el siguiente código para crear un elemento de tipo *ImageView*. Este elemento nos permite representar imágenes en una aplicación Android.

```
var HeaderImage = new ImageView(this);
```

12. Agrega el siguiente código para establecer el recurso imagen `Resources\drawable\Xamarin_Diplomado_30.jpg` como origen para el elemento *ImageView*.
Nota el uso de la sintaxis para hacer referencia al recurso mediante código:
`Resource.Drawable.Xamarin_Diplomado_30`

```
HeaderImage.SetImageResource(Resource.Drawable.Xamarin_Diplomado_30);
```

13. Agrega el siguiente código para incorporar el elemento *ImageView* a la colección de elementos que componen el diseño de la interfaz de usuario.



```
MainLayout.AddView(HeaderImage);
```

14. Agrega el siguiente código para crear un elemento de tipo *TextView*. Este elemento nos permite representar texto en una aplicación Android.

```
var UserNameTextView = new TextView(this);
```

15. Agrega el siguiente código para establecer el valor del recurso cadena *UserName* en la propiedad *Text* del elemento *TextView*. Nota el uso de la sintaxis para hacer referencia al recurso cadena mediante código: ***Resource.String.UserName***.

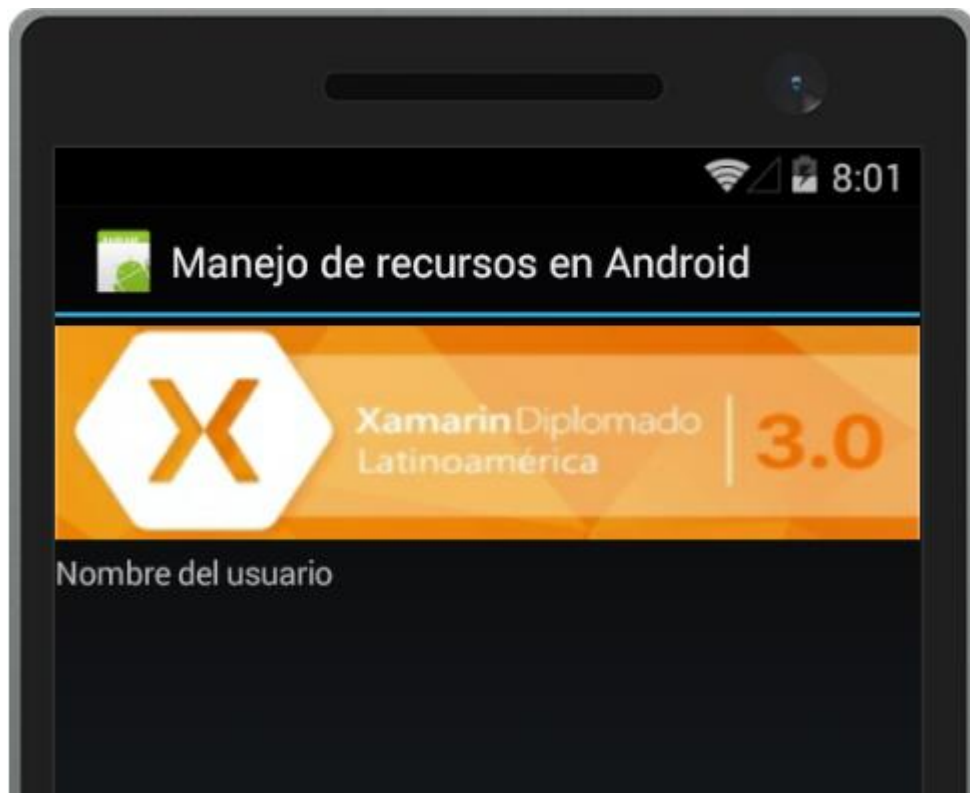
```
UserNameTextView.Text = GetString(Resource.String.UserName);
```

16. Agrega el siguiente código para incorporar el elemento *TextView* a la colección de elementos que componen el diseño de la interfaz de usuario.

```
MainLayout.AddView(UserNameTextView);
```

17. Guarda los cambios.

18. Ejecuta la aplicación. Se mostrará una pantalla similar a la siguiente.



19. Regresa a Visual Studio y detén la aplicación.



Tarea 4. Acceder a los recursos desde XML.

Los recursos en un archivo XML son accedidos mediante una sintaxis especial:

@[<PackageName>:]<ResourceType>/<ResourceName>

- **PackageName.** El paquete que proporciona el recurso y sólo se requiere cuando se utilizan recursos de otros paquetes.
- **ResourceType.** Este es el tipo de recurso anidado que está dentro de la clase *Resource*.
- **ResourceName.** Este es el nombre de archivo del recurso (sin la extensión de tipo de archivo) o el valor del atributo *android:name* para recursos que están en un elemento XML como en el caso de *Strings.xml*.

En la tarea anterior agregamos elementos a la interfaz de usuario *Main* mediante código. Ahora lo haremos mediante el diseñador de Android.

1. Abre el archivo **Main.axml**.
2. Agrega un elemento **ImageView** al diseño de la interfaz de usuario.
3. Establece el valor **@+id/HeaderImageView** a la propiedad **id** del elemento *ImageView*.
4. Establece el valor del recurso *Resources\drawable\Xamarin_Diplomado_30.jpg* a la propiedad **src** de elemento *ImageView*. Recuerda que la sintaxis es **@[<PackageName>:]<ResourceType>/<ResourceName>** y por lo tanto el valor para la propiedad es: **@drawable/xamarin_diplomado_30**
5. Observa que el código XML del elemento *ImageView* es similar al siguiente.

```
<ImageView
    android:src="@drawable/xamarin_diplomado_30"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/HeaderImageView" />
```

Este ejemplo tiene un *ImageView* que requiere un recurso *drawable* llamado *xamarin_diplomado_30*. El *ImageView* tiene su atributo **src** establecido en **@drawable/xamarin_diplomado_30**. Cuando se inicie la Activity, Android buscará dentro del directorio *Resource/drawable* un archivo llamado *xamarin_diplomado_30.jpg* (la extensión de archivo podría ser de otro formato de imagen, como *xamarin_diplomado_30.png*) y cargará ese archivo para mostrarlo en el *ImageView*.

6. Agrega un elemento **TextView** al diseño de la interfaz de usuario.



7. Establece el valor **@+id/UserNameTextView** a la propiedad **id** del elemento *TextView*.
8. Establece el valor del recurso cadena *UserName* a la propiedad **text** de elemento *TextView*.
El valor para la propiedad es: **@string/UserName**

Observa que el código XML del elemento *TextView* es similar al siguiente.

```
<TextView
    android:text="@string/UserName"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/UserNameTextView" />
```

9. Guarda los cambios realizados.
10. Abre el archivo *MainActivity.cs*.
11. Elimina o comenta las líneas de código que agregaste al método *OnCreate* en la tarea anterior. El código del método *OnCreate* se verá similar al siguiente.

```
protected override void OnCreate(Bundle bundle)
{
    base.OnCreate(bundle);

    // Set our view from the "main" layout resource
    SetContentView (Resource.Layout.Main);
}
```

12. Guarda los cambios y ejecuta la aplicación. La pantalla se mostrará igual que en la tarea anterior.





Los Recursos predeterminados son elementos que no son específicos de ningún dispositivo o factor de forma en particular y, por lo tanto, son la opción predeterminada del sistema operativo Android si no se pueden encontrar recursos más específicos. Por tal situación, los Recursos Predeterminados son el tipo de recurso más común de crear. Están organizados en subdirectorios dentro del directorio *Resources* según su tipo de recurso.

La siguiente es una lista de los recursos que podemos encontrar en una aplicación Android:

- **animator.** Los recursos *animator* son archivos XML que describen las Animaciones de Propiedades (Property Animations). Las animaciones de propiedades se introdujeron en el nivel de API 11 (Android 3.0) y proporcionan las animaciones de propiedades de un objeto. Las animaciones de propiedades son una forma más flexible y poderosa de describir animaciones sobre cualquier tipo de objeto.
- **anim.** Son archivos XML que describen animaciones *Tween*. Las animaciones *Tween* son una serie de instrucciones de animación para realizar transformaciones en el contenido de un objeto View, por ejemplo, rotar una imagen o aumentar el tamaño del texto. Las animaciones *Tween* se limitan a utilizarse en objetos View.
- **color.** Son archivos XML que describen una lista de colores de estado (*Color State List*). Para entender las listas de colores de estado, considera un widget de interfaz de usuario, como un botón. Puede ser que el botón tenga diferentes estados como *Presionado* o *Deshabilitado* y puede cambiar de color con cada cambio de estado. La lista se expresa en una lista de estados.
- **drawable.** Los recursos *drawable* son un concepto general para los gráficos que pueden ser compilados en la aplicación y luego ser accedidos mediante llamadas a APIs o referenciados por otros recursos XML. Algunos ejemplos de *drawables* son archivos de mapas de bits (.png, .gif, .jpg), bitmaps especiales redimensionables conocidos como *Nine-Patches*, lista de estado, formas genéricas definidas en XML, etc.
- **layout.** Son los archivos XML que describen un diseño de interfaz de usuario, como una Activity o una fila en una lista.
- **menu.** Son archivos XML que describen menús de aplicaciones como los *Menús de opciones*, *Menús contextuales* y *submenús*.
- **raw.** Son archivos arbitrarios que se guardan en su forma nativa binaria. Estos archivos se compilan en un formato binario dentro de una aplicación Android.
- **values.** Son archivos XML que contienen valores simples. Un archivo XML en el directorio **values** no define un solo recurso, sino que puede definir varios recursos. Por ejemplo, un



archivo XML puede contener una lista de valores Cadena, mientras que otro archivo XML puede contener una lista de valores Color.

- **xml.** Son archivos XML que son similares en función a los archivos de configuración de .NET. Estos son archivos XML arbitrarios que pueden ser leídos por la aplicación en tiempo de ejecución.



Ejercicio 2: Validando tu actividad

En este ejercicio agregarás funcionalidad a tu laboratorio con el único propósito de enviar una evidencia de la realización del mismo.

La funcionalidad que agregarás consumirá un ensamblado que representa una Capa de acceso a servicio (SAL) que será consumida por tu aplicación Android.

Es importante que realices cada laboratorio del diplomado ya que esto te dará derecho a obtener el diploma final del mismo.

Tarea 1. Agregar los componentes de la Capa de acceso a Servicio.

En esta tarea agregarás una referencia al ensamblado **SALLab08.dll** que implementa la capa de acceso a servicio. El archivo **SALLab08.dll** se encuentra disponible junto con este documento.

1. En el proyecto Xamarin.Android, agrega una referencia del ensamblado **SALLab08.dll**.

Este componente realiza una conexión a un servicio de Azure Mobile, por lo tanto, será necesario agregar el paquete NuGet **Microsoft.Azure.Mobile.Client**.

2. En el proyecto Xamarin.Android, instala el paquete NuGet **Microsoft.Azure.Mobile.Client**.

Tarea 2. Agregar la funcionalidad para validar la actividad.

El componente DLL que agregaste te permite registrar tu actividad en la plataforma de TI Capacitación y Microsoft. El componente se comunica con la plataforma de TI Capacitación para autenticarte y posteriormente envía un registro a la plataforma Microsoft.

1. Agrega las siguientes cadenas de recursos en el archivo *Strings.xml*:

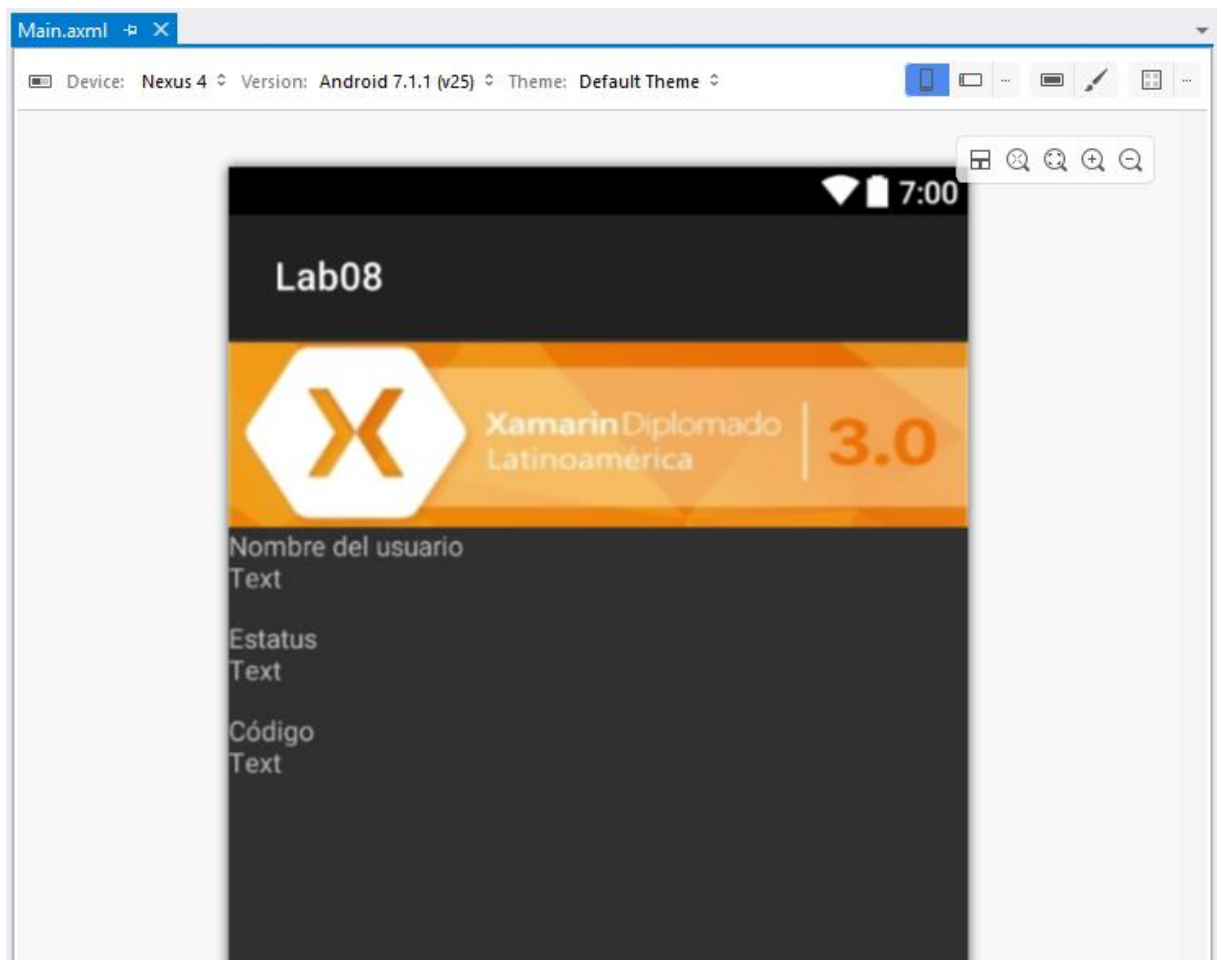
Nombre	Valor
Status	Estatus
Token	Código

2. Utiliza el *Diseñador de Android* para agregar los siguientes elementos al archivo *Main.axml*.
 - Un elemento que permita mostrar tu nombre completo una vez que tu aplicación haya sido validada.
 - Un elemento que permita mostrar el valor del recurso cadena **Status**.



- Un elemento que permita mostrar el valor del estatus devuelto después de validar tu actividad.
- Un elemento que permita mostrar el valor del recurso cadena **Token**.
- Un elemento que permita mostrar el valor del Token devuelto después de validar tu actividad.

El diseño de la interfaz de usuario deberá ser similar al siguiente.



3. Agrega el código necesario para que al ejecutar la aplicación se muestren los datos de validación de tu actividad. La pantalla deberá ser similar a la siguiente.



Cuando tu actividad se haya validado exitosamente puedes ver el estatus en el siguiente enlace: <https://ticapacitacion.com/evidencias/xamarin30>.

4. Regresa a Visual Studio y detén la ejecución de la aplicación.

Nota: Es probable que recibas un correo similar al siguiente.

Tu código de lab no es válido, revisa que estés utilizando un código de reto válido. Si tienes preguntas o dudas por favor contacta a dxaudmx@microsoft.com

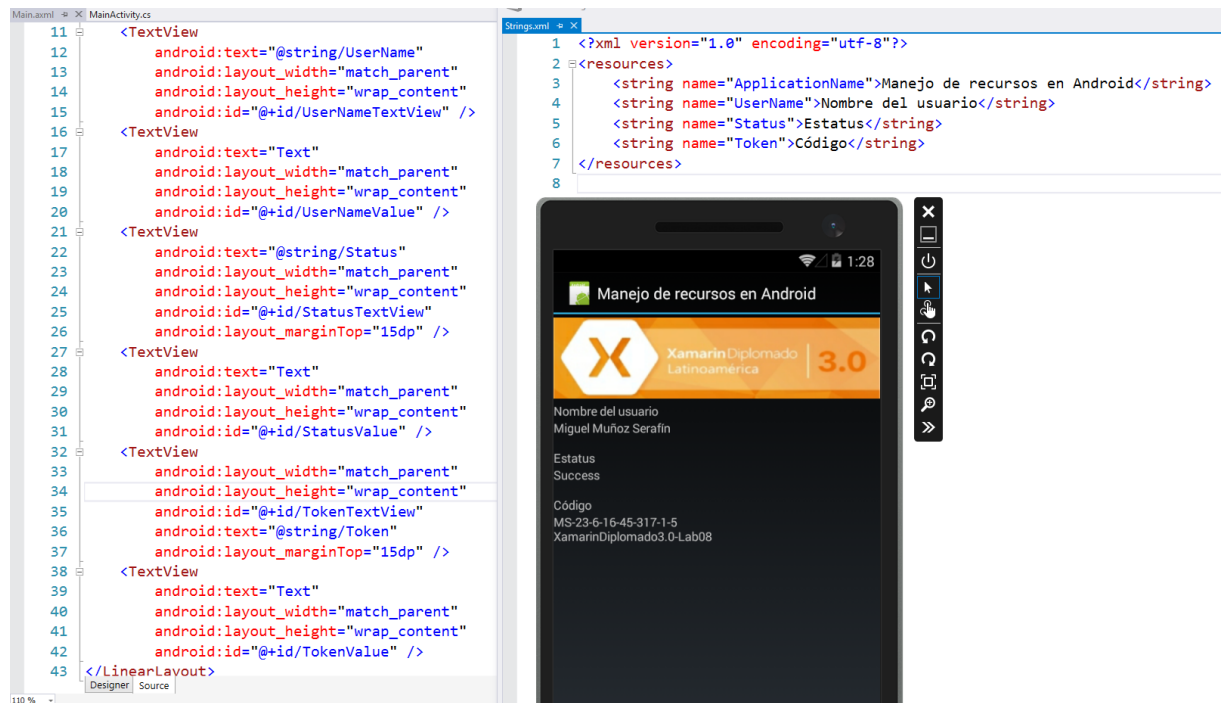
Puedes hacer caso omiso al mensaje.

5. Para verificar que hayas realizado tu actividad, sube una imagen mostrando lo siguiente:
 - a. El código XML del Diseñador de Android mostrando el uso de los recursos *UserName*, *Status* y *Token*.
 - b. El contenido del archivo Strings.xml.



- c. La pantalla de la aplicación con los datos de validación legibles.

La imagen deberá ser similar a la siguiente:



La evidencia donde debes subir la imagen es “Imagen laboratorio 8”.

Imagen laboratorio 8

Fecha Límite: 30 de Junio de 2017

No ha enviado evidencia

Los puntos que se evalúan en la evidencia son los siguientes:

- La imagen debe mostrar el código XML del diseñador de Android mostrando el uso de los recursos *UserName*, *Status* y *Token*.
- La imagen debe mostrar el contenido del archivo *Strings.xml*.
- La imagen debe mostrar la pantalla de la aplicación con los datos de validación legibles.

Si encuentras problemas durante la realización de este laboratorio, puedes solicitar apoyo en los grupos de Facebook siguientes:

<https://www.facebook.com/groups/iniciandoconxamarin/>

<https://www.facebook.com/groups/xamarindiplomadoitc/>



Resumen

En este laboratorio exploraste los conceptos y tareas básicas del manejo de recursos en Android. En los siguientes laboratorios de este módulo explorarás los Recursos Alternativos de las aplicaciones Android para poder crear aplicaciones que respondan apropiadamente a múltiples tamaños de pantalla y a la configuración regional del usuario.

¿Qué te pareció este laboratorio?

Comparte tus comentarios en twitter y Facebook utilizando el hashtag **#XamarinDiplomado**.