Presentation layer

Procedures for mobile development

*(preliminary document)*

*May 2018*

Contents

[1. Instalación del ambiente de desarrollo. 3](#_Toc516481152)

[2. Creación y prueba de un proyecto tipo “Hola Mundo”. 6](#_Toc516481153)

[3. Diseño de una vista (layout) 10](#_Toc516481154)

[4. Control de versiones con git (crear repositorio y push inicial) 14](#_Toc516481155)

[5. Override Life cycle methods 17](#_Toc516481156)

[6. Log 18](#_Toc516481157)

[7. Printing setting 19](#_Toc516481158)

[8. Crear una nueva actividad / agregar un una nueva escena. 20](#_Toc516481159)

[9. Responder al click a un botón. 22](#_Toc516481160)

[10. Iniciar una actividad / mostrar una nueva escena. 24](#_Toc516481161)

[11. Hacer referencia a un elemento de la UI. 25](#_Toc516481162)

[12. Entradas del usuario a través de elementos de la UI. 27](#_Toc516481163)

[13. Send a Broadcast / Register a receiver of a Broadcast 29](#_Toc516481164)

[14. Threads and Asynchronous communication 30](#_Toc516481165)

[15. Crear una DB para guardar y recuperar datos 31](#_Toc516481166)

[16. Desplegar elementos de una lista. 36](#_Toc516481167)

[17. Adding a Toolbar Back Button / Navigation Controller. 40](#_Toc516481168)

[20. Solución de problemas 42](#_Toc516481169)

|  |
| --- |
| 1. Instalación del ambiente de desarrollo. |
| Angular |
| **EQUIPO WINDOWS**  **Instalación de node.js y npm.**  **npm**: administrador de paquetes para JS, podemos encontrar bibliotecas como bootstrap, react y angular.   * Ir al sitio <https://nodejs.org/en/download/> y seguir las indicaciones. * Verificar la instalación en una ventana de comandos:   > node -v  > npm -v  **Instalación de Visual Studio Code.**  Editor de código fuente, ligero, potente disponible para Windows, MacOS y Linux, con soporte integrado para TS y Node.js y con una terminal embebida.   * Ir al sitio <https://code.visualstudio.com> y seguir las indicaciones.   **Instalación de la CLI (Command Line Interface) de angular.**  Herramienta para la ejecución de comandos para la creación y prueba de proyectos angular.   * En una ventana de comandos ejecutar:   > npm install -g @angular/cli  **Si aún no tiene git instalado, instalar git.**  Ir al sitio <https://git-scm.com/downloads> y seguir las indicaciones.  **EQUIPO MAC**  *No se ha probado, por ahora el curso se dará con equipos Windows.*  *May 28, 2018. LGF.* |
| Android |
| **EQUIPO WINDOWS**  **Instalación de Android Studio.**  IDE oficial para el desarrollo de aplicaciones para Android.   * Ir al sitio <https://developer.android.com/studio/#downloads> y seguir las indicaciones.   *Ya no es necesario instalar de manera individual Android SDK o Google Android emulator.*  **Si aún no tiene git instalado, instalar git.**  Ir al sitio <https://git-scm.com/downloads> y seguir las indicaciones.  **EQUIPO MAC**  **Instalación de Android Studio.**  *No se ha probado, por ahora el curso se dará con equipos Windows.*  *May 28, 2018. LGF.* |

|  |
| --- |
| iOS |
| **EQUIPO MAC**  **Instalación de XCode.**  Abrir la App Store, buscar **Xcode**, dar click en obtener.  También se pueder ir al siguiente sitio y seguir las indicaciones.  Es posible que se requiera crear un Appli  **Si aún no tiene git instalado, instalar git.**  Ir al sitio <https://git-scm.com/downloads> y seguir las indicaciones.  **EQUIPO WINDOWS**  *Con una máquina virtual, por ahora se planea dar el curso en equipo MAC.*  *May 28, 2018. LGF.* |
| Xamarin.Android |
| **EQUIPO WINDOWS (El equipo se volvió extremadamente lento)**  **Instalación de Visual Studio Community 2017.**  Xamarin está incluido en todas las ediciones de Visual Studio sin costo, no requiere licencia adicional.   * Ir al sitio <https://www.visualstudio.com/vs/> y descargar Visual Studio Community 2017. * Instalar Visual Studio agregando “Mobile development with .NET”. * Verificar la instalación dando click a Help en el menú.   *Se instala Android SDK API level 27, JDK8 (requerido a partir de API level 24, JDK9 no es soportado) y Google Android emulator.*  **Agregar Xamarin a Visual Studio, en caso de ya tener instalado Visual Studio.**  Si ya tiene instalado Visual Studio y no realizó el paso anterior.   * Ejecutar Visual Studio Installer, dar click en Modify, select “Mobile development with .NET” y click Modify.   **Si aún no tiene git instalado, instalar git.**  Ir al sitio <https://git-scm.com/downloads> y seguir las indicaciones.  **EQUIPO MAC**  *Se optó por trabajar en la MAC, el equipo Windows es muy lento.*  *May 28, 2018. LGF.* |
| Xamarin.iOS |
| **EQUIPO MAC**  **Instalación de XCode.**  **Instalación de Visual Studio Community 2017 para MAC.**  *Para trabajar desarrollo para iOS se requiere tener instalado previamente xCode.*  Desde el equipo MAC Ir al sitio <https://www.visualstudio.com/>, descargar Community 2017 para Mac.  Dar doble-click al archivo con extensión dmg descargado.  Dar doble-click al Installer y seguir las indicaciones.  **Si aún no tiene git instalado, instalar git.**  **https://visualstudio.github.com/index.html**  **EQUIPO WINDOWS**  *\*Requiere una conexión remota a una MAC.*  *\*Se probó, es muy lento, por ahora el curso se dará con la MAC.*  **Instalación de Visual Studio Community 2017.**  **Agregar Xamarin a Visual Studio, en caso de ya tener instalado Visual Studio.**  **Crear una conexión a un equipo MAC que tenga instalado xCode y VisualStudio for MAC**  Habilitar *remote login* en la MAC   * On the Mac, open System Preferences and go to the **Sharing (Compartir)** pane. * Check **Remote Login (Sesión remota)** in the **Service** list. * Permitir a todos los usuarios   **Si aún no tiene git instalado, instalar git.**  *May 28, 2018. LGF.* |
| Xamarin.Forms |
| **EQUIPO WINDOWS**  *Mismas indicaciones que Xamarin.iOS*  **EQUIPO MAC**  *No se ha probado, por ahora el curso se dará con equipos Windows.*  *May 28, 2018. LGF.* |

|  |
| --- |
| 2. Creación y prueba de un proyecto tipo “Hola Mundo”. |
| Angular |
| **Creación del proyecto.**  Abrir el IDE:  Visual Studio Code.  Abrir la carpeta donde se desea crear el proyecto.   * [Ctrl – k Ctrl – o] o menú: File y Open Folder…   Abrir la CLI.   * [Ctrl – ñ] o menú: Terminal Integrado[[1]](#footnote-1)…   Asegurarse de estar en la carpeta correcta y teclear:  > ng new <name-app> (*it may take a while*)  **Prueba del proyecto.**  Iniciar la aplicación:   * En la CLI, cambiarse a la carpeta del proyecto que acaba de crear:   > cd <name-app>   * Teclear:   > ng serve --open (ng serve –o, ng serve)[[2]](#footnote-2)  Abrir la aplicación:   * En un navegador, teclear en la barra de direcciones:   <http://localhost:4200/>  Detener la aplicación:   * En la CLI:   [Ctrl – c]  *May 28, 2018. LGF..* |
| Android |
| **Creación del proyecto.**  Abrir el IDE:   * Android Studio.   Da click a menu: File, New, New project…, para ejecutar el Create Android Project Wizard...   * Teclea el nombre del proyecto <name-app>. * Navegar a la carpeta donde se creará la carpeta del proyecto y completar con el nombre del proyecto. * Deja los valores por omisión en la pantalla Target Android Devices. * Selecciona una Empty Activity en la pantalla Add an Activity to Mobile. * Deja los valores por omisión en la pantalla Configure Activity.   **Prueba de la app en el emulador.**   * En el IDE, dar click al ícono Run (triangulo verde), o Shift F10. * Si es la primera vez, puede crear un nuevo dispositivo virtual. * Seleccionar el dispositivo (*me ha funcionado bien: Pixel\_2\_API\_27*).   **Prueba de la app en un dispositivo real.**  Conecta el dispositivo mediante cable USB.  Si es necesario instala el driver USB apropiado:   * Descargar el driver: * <https://developer.android.com/studio/run/oem-usb#Drivers> * Descomprimirlo. * Abrir desde el panel de control el Device Manager y localizar Portable Devices. * Dar click-derecho al device and select Update Driver Software. * Selecciona la carpeta que acabas de descomprimir.   Habilita USB debugging en tu dispositivo:   * Open the Settings app. * Scroll and select About phone. * Tap seven-times Build number. * Return to the previous screen to find Developer options. * Open Developer options and enable USB debugging.   Instala y ejecuta la app.   * En Android Studio dar click al ícono Run (triangulo verde), [Shift + F10]. * Selecciona tu dispositivo y click OK.   *May 28, 2018. LGF.* |
| iOS |
| **Creación del proyecto.**  Abrir el IDE:   * xCode.   Dar click a File> New> Project:   * Select iOS. * Select Single View App. * Teclear el nombre de la app. * Selecciona el lenguaje Swift. * (Por ahora no incluiré pruebas). * Seleccionar Create Git repository on my Mac.   **Prueba de la app en el emulador.**  Dar click al ícono run.  *May 28, 2018. LGF.* |
| Xamarin.Android |
| **Creación del proyecto.**  Abrir el IDE:  Visual Studio 2017 Community.  Da click a menu: File, New, Project…, para ejecutar el Create Android Project Wizard...   * Seleccionar Visual C#> Android> Aplicación de Android (Xamarin) * Teclea el nombre del proyecto <name-app>. * Navegar a la carpeta donde se creará la carpeta del proyecto. * Teclea el nombre de la solución. * Check a “Crear directorio para la solución” y click a Aceptar. * Seleccionar *Empty Activity.*   **Prueba de la app en el emulador.**  Ejecutar el emulador:   * Abrir Android Device Manager.   Menú: Tools> Android> Administrador de dispositivos Android.   * Seleccionar Android\_Accelerated\_x86\_Oreo y dar click Start. * Esperar a que esté listo.   Seleccionar el emulador y dar click al combo/ícono Run, [Ctrl F5].  *May 28, 2018. LGF.* |
| Xamarin.iOS |
| **Creación del proyecto.**  Abrir el IDE:  Visual Studio 2017 Community para MAC.  Da click a menu: Archivo> Nueva solución   * Seleccionar iOS> Aplicación, Aplicación de una vista y dar click a Siguiente. * Teclea el nombre del proyecto <name-app>. * Navegar a la carpeta donde se creará la carpeta del proyecto. * Teclea el nombre de la solución. * Check a “Crear directorio para la solución”. * Usar Git para el control de versiones y click a Crear.   **Prueba de la app en el emulador.**  Dar click al ícono Run.  *May 28, 2018. LGF.* |
| Xamarin.Forms |
| **Creación del proyecto.**  Abrir el IDE:  Visual Studio 2017 Community.  Da click a menu: File, New, Project…, para ejecutar el Create Android Project Wizard...   * Seleccionar Visual C#> Cross-Platfform> Aplicación de Android (Xamarin.Forms) * Teclea el nombre del proyecto <AppName>.   **Importante: note que a diferencia de los ejercicios anteriores, da problemas al incluir un guion medio en el nombre de la aplicación.**   * Navegar a la carpeta donde se creará la carpeta del proyecto. * Teclea el nombre de la solución. * Check a “Crear directorio para la solución” y click a Aceptar. * Seleccionar *Aplicación en blanco*, marcar las plataformas Android y iOS. * Seleccionar la estrategia .NET Standard y dar click a OK.   **Prueba de la app en el emulador.**  Probar la aplicación para Android.   * Dar click derecho a la solución y dar click a Restaurar paquetes de NuGet (quizá no restaure nada). * Dar click derecho a la solución y dar click a Administración de la configuración…, uncheck la compilación del proyecto para iOS. * Dar click derecho al proyecto Android y dar click a Establecer como proyecto de inicio. * Seleccionar el dispositivo y dar click al botón de Run (puede tardar un poco abrir el emulador).   Probar la aplicación para iOS.   * Establecer comunicación con una MAC, como se especifica arriba. * En esta ocasión check la compilación para iOS. * Dar click derecho al proyecto iOS y dar click a Establecer como proyecto de inicio. * Seleccionar el dispositivo y dar click al botón de Run, alternatively, choose Product > Run or press Command-R (tarda mucho).   *May 28, 2018. LGF.* |

|  |
| --- |
| 3. Diseño de una vista (layout) |
| Angular |
| **Utilizar bootstrap para el acomodo de las vistas y otros elementos html.** |
| Android *(Con el Layout Editor y modificando el archivo xml).* |
| **Verificar/Agregar soporte para ConstraintLayout.**  Verificar que el archivo **build.gradle** tenga la siguiente línea en el bloque de **dependencies** {  Implementation ‘com.android.support,constraint:constraint-layout:1.1.0’  **Utilizar el Layout Editor para el diseño.**   * Abrir el layout (archivo xml) y seleccionar la vista de diseño. * Agregar los elementos desde **Palette Windos** al *blueprint*. * Agregar los *constraint* mediante el menú contextual o la **Attributes Window**. * Identificar cada elemento, incluir en el nombre el tipo de elemento y una parte significativa. * Definir los recursos *string* que se utilizarán para los atributos de texto. * Establecer el *text attribute* utilizando los recursos string que definió. * Establecer otras propiedades como tamaño de letra, etc. * Verificar/modificar que haga sentido el archivo *xml* generado. * Ejecutar la app para ver que todo salió OK.   **Elementos (Widgets).**  Etiqueta.  <TextView/> Botón.  <Button/>  …  **Propiedades de los Widgets.**  android:id="@+id/TextViewName" android:text="@string/TextToShow"  android:textSize="24sp"  android:textAlignment="textStart" android:textAllCaps="false"  …  **Propiedades del Layout.**  android:layout\_width="wrap\_content" android:layout\_height="0dp" android:layout\_marginTop="24dp"  android:layout\_marginBottom="24dp" android:layout\_marginEnd="24dp"  …  **Propiedades del ConstraintLayout.**  *Utilizar el Layout Editor.*  *May 28, 2018. LGF.* |
| iOS *(Con el* [*Interface Builder*](https://developer.apple.com/library/content/referencelibrary/GettingStarted/DevelopiOSAppsSwift/GlossaryDefinitions.html#//apple_ref/doc/uid/TP40015214-CH12-SW47)*, se genera código).* |
| **Utilizar el** [**Interface Builder**](https://developer.apple.com/library/content/referencelibrary/GettingStarted/DevelopiOSAppsSwift/GlossaryDefinitions.html#//apple_ref/doc/uid/TP40015214-CH12-SW47) **(visual interface editor) para el diseño del storyboard.**  *\*El* ***storyboard*** *contiene una* [***scene***](https://developer.apple.com/library/content/referencelibrary/GettingStarted/DevelopiOSAppsSwift/GlossaryDefinitions.html#//apple_ref/doc/uid/TP40015214-CH12-SW62)*, which represents a screen of content in your app.*   * Habilitar mostrar el marco de los elementos en la escena.   Choose Editor > Canvas, and make sure Show Bounds Rectangles is selected.  *\*This setting causes Interface Builder to draw a blue bounding box around all the views in the canvas.*   * Mostrar la librería de objetos:   View > Utilities > Show Object Library.  Agregar elementos a la escena:   * Buscar los elementos en el filtro del Object Library   *Por ejemplo escribir Text View.*   * Arrastrar los elementos a la scene. * Modificar su tamaño y posición. * Utilizar el **Atrributes inspector** para modificar atributos: Text, Color, Font, Placeholder, ….   View> Utilities> Show Attributes Inspector (cuarto ícono en la ventana de la derecho).  Utilizar los botones de la parte inferior del *canvas* para:   * Agrupar elementos (Embed in Stack). * Agregar constraints y alineamiento.   **Revise el Outline View**  Muestra como organizan los elementos que se han agregado a la scene.  El primer ícono en la parte inferior del canvas oculta y muestra el outline view.  **Elementos (views objects: UIView)**  Text View  Button  **Propiedades de los elementos.**  Establecer en el Attributes inspector:  Title (double-click)  Text (double-click)  **Alineamiento y constraints.**  Establecer en el Canvas (íconos inferiores).  *May 28, 2018. LGF.* |
| Xamarin.Android *(Editando el archivo axml y ver resultado en la vista de diseño)*. |
| **Utilizando ConstraintLayout**  **Agregar soporte para ConstraintLayout:**  Descargar el paquete de [*https://www.nuget.org/packages/Xamarin.Android.Support.Constraint.Layout/*](https://www.nuget.org/packages/Xamarin.Android.Support.Constraint.Layout/)  Place the nugpkg file in a specific directory.  Ctrl-click to Packages, then click Add Packages.  Click the combo and change to Configure Sources…  Click Add to select the directory where you place the nugpkg file and click OK.  Select the Xamarin.Android.Support.Constraint.Layout and click Add Package.  **Editar el archivo axml**  Abrir el layout (archivo axml), note que el layout activity\_main.axml por omisión utiliza un RelativeLayout.  Abrir la vista de diseño para agregar el ConstraintLayout si es necesario.  Se pueden arrastrar los elementos desde el Toolbox al Document Outline.  Abrir la vista de origen, editar el código del archivo axml para agregar los elementos y sus parámetros.  Ver el resultado en la vista de diseño.  ***\*Nota: puede utilizar el código xml del proyecto Android.***  **Utilizando RelativeLayout:**  Abrir el layout (archivo axml), note que el layout activity\_main.axml por omisión utiliza un RelativeLayout.  Abrir la vista de diseño para agregar el RelativeLayout si es necesario.  Se pueden arrastrar los elementos desde el Toolbox al Document Outline.  Abrir la vista de origen, editar el código del archivo axml para agregar los elementos y sus parámetros.  Ver el resultado en la vista de diseño.  Consultar los parámetros de android:layout en:  <https://developer.android.com/reference/android/widget/RelativeLayout.LayoutParams>  Ejecutar la app para ver que todo salió OK.  Abrir la ventana de propiedades: Ver> Ventana de propiedades (F4)  **Elementos (Widgets).**  Etiqueta.  <TextView/> Botón.  <Button/>  …  **Propiedades de los Widgets.**  android:id="@+id/TextViewName" android:text="@string/TextToShow"  android:textSize="24sp"  android:textAlignment="textStart" android:textAllCaps="false"  …  **Propiedades del Layout.**  android:layout\_width="wrap\_content" android:layout\_height="0dp" android:layout\_marginTop="24dp"  android:layout\_marginBottom="24dp" android:layout\_marginEnd="24dp"  android:layout\_centerHorizontal="true"  android:layout\_below="@id/TextViewSummary"  android:layout\_alignStart="@id/TextViewTaskToDoCount"  android:layout\_alignParentBottom="true"  android:layout\_alignParentEnd="true"  *May 28, 2018. LGF.* |
| Xamarin.iOS *(De forma gráfica, se genera código).* |
| Abrir el storyboard (Main.storyboard).  Agregar los elementos.  Establecer propiedades y constraints. |
| Xamarin.Forms *(Editando el archivo xaml).* |
| **Utilizando StackLayout:**  <StackLayout> </StackLayout>  Abrir el archivo MainPage.xaml, editar el código xaml para agregar los elementos y sus propiedades.  Para ver un previo de la página utilizar Xamarin.Forms Previer  Menú: View > Other Windows > Xamarin.Forms Previewer  **Controles (Views).**  <Label/>  <Button/>  …  **Propiedades de los Controles.**  x:Name="LabelSummary"  Text="Summary"  Font="30"Text="Summary"  **Propiedades del Layout.**  RelativeLayout.XConstraint=  "{ConstraintExpression Type=RelativeToParent,  Property=X, Factor=1, Constant=120}"  RelativeLayout.YConstraint=  "{ConstraintExpression Type=RelativeToParent,  Property=Y, Factor=1, Constant=80 }" />  RelativeLayout.XConstraint=  "{ConstraintExpression Type=RelativeToView, ElementName=LabelSummary,  Property=X, Factor=1, Constant=-10}"  RelativeLayout.YConstraint=  "{ConstraintExpression Type=RelativeToView, ElementName=LabelSummary,  Property=Y, Factor=1, Constant=100}" /> |

|  |
| --- |
| 4. Control de versiones con git (crear repositorio y push inicial) |
| Angular |
|  |
| Android |
| **Crear el repositorio.**   * Dar click a menú: VCD > Import into Version Control > Create Git Repository… * Seleccionar la carpeta del proyecto.   Android ejecuta por ti el comando **git init** para crear el repositorio, es decir, agrega la carpeta .git y un archivo **HEAD** inicial para la **master branch**.  **Hacer un push inicial.**  Hacer un commit inicial.   * En la Version Control Window [Alt + 9], dar click a commit para agregar todos los archivos y hacer un primer commit. * Agregar el comentario “Initial Commit” y firmar.   Crear el repositorio remoto.   * Si es necesario crear una cuenta de github. * Crear un New Empty repository without readme file en github.   Hacer el push al repositorio remoto.   * Dar click a menu: VCD > Git > Push… * Indicar el nombre del remote y la URL. * Dar click a Push.   Android ejecuta por ti los siguientes comandos:   * git remote add origin https://github.com/LilianaGF/task-app.git * git push -u origin master   **Clonar un proyecto remoto a un repositorio local.**  *El siguiente procedimiento es independiente de la tecnología, aplica a todos los casos.*   * Abrir una bash terminal y ubicarse en la carpeta donde se clonará el proyecto. * Ejecutar el comando   git clone <https://github.com/>...  *May 28, 2018. LGF.* |
| iOS |
| **Crear el repositorio.**   * Al crear el proyecto, Seleccionar Create Git repository on my Mac.   **Hacer un push inicial.**  Crear el repositorio remoto.   * Si es necesario crear una cuenta de github. * Crear un New Empty repository without readme file en github.   Agregar la cuenta de github en Xcode   * Xcode> Preferences…Accounts. * Click al ícno + para gregar una cuenta.   Agregar el remote:   * Abrir el Souce Control Navigator. * Click ícono junto al del Project Navigator (segundo ícono). * Control-click a la raíz del árbol que se muestra. * Seleccionar Add Existing Remote… * Configurar la dirección del repositorio creado.   Hacer el commit y el push.   * Ir al menú Source Control > Commit y hacer el commit dejando un mensaje. * Ir al menú Source Control> Push, dar click a push.   **Clonar un proyecto remoto a un repositorio local.**  *Ver arriba.*  *May 28, 2018. LGF.* |
| Xamarin.Android |
| **Crear el repositorio.**  Al crear la solución, se puede marcar la casilla Create new Git repository.  Para una solución donde no se creó el repositorio al inicio, dar click al botón Publish en la esquina inferior derecha de Visual Studio, esto creará un Git repository en la carpeta de la solución.  **Hacer un push inicial.**  **Desde Visual Studio Windows**  Crear el repositorio remoto.   * Si es necesario crear una cuenta de github. * Crear un New Empty repository without readme file en github.   Abrir la ventana Team Explorer.  Menú: Ver > Team Explorer  Para commmit, dar click a Changes.  Para push, dar click a Sync,  En Synchonization dar click a Push ( en español aparece como Confirmaciones de salida-Insertar)  **Desde Visual Studio Mac**  **Clonar un proyecto remoto a un repositorio local.**  *Ver arriba.*  *May 28, 2018. LGF.* |
| Xamarin.iOS |
| **Crear el repositorio.**  Al crear la solución.  **Hacer un push inicial.**  Crear el repositorio remoto:   * Crear un New Empty repository without readme file en github.   Desde Visual Studio Community para Mac:   * Ir al menú: Control de versiones> Revisar la solución y confirmar. * Dar click a confirmar y agregar un mensaje. * Hacer el push desde una terminal, ubicarse en la carpeta del proyecto y ejecutar los siguientes comandos:   git remote add origin <https://github.com/LilianaGF/xamarin_ios>  git push origin master  **Clonar un proyecto remoto a un repositorio local.**  *Ver arriba.*  *May 28, 2018. LGF.* |
| Xamarin.Forms |
| *Mismas indicaciones que Xamarin.Android.* |

|  |
| --- |
| 5. Override Life cycle methods |
| Angular |
|  |
| Android |
| **Override the following methods in the activity file (java):**  **protected void** onCreate(Bundle savedInstanceState)  **protected void** onStart()  **protected void** onResume()  **private void** updateTaskCount()  **protected void** onPause()  **protected void** onStop()  **public void** onDestroy()  *May 28, 2018. LGF.* |
| iOS |
| **Override the following methods:**  To perform any additional setup required by your view controller.  override func viewDidLoad()  Before the content view is presented onscreen.  viewWillAppear()  viewDidAppear()  viewWillDisappear()  viewDidDisappear() |
| Xamarin.Android |
| **Override the following methods in the activity file (cs):**  protected override void OnCreate(Bundle savedInstanceState)  protected override void OnStart()  protected override void OnResume()  protected override void OnPause()  protected override void OnStop()  protected override void OnDestroy()  *May 28, 2018. LGF.* |
| Xamarin.iOS |
| **Override the following methods:** |
| Xamarin.Forms |
| **Override the following methods:** |

|  |
| --- |
| 6. Log |
| Angular |
|  |
| Android |
| **Use the following methods:**  Log.*e*(***tag***, **"this is an error message"**); Log.*w*(***tag***, **"this is a warning message"**); Log.*i*(***tag***, **"this is an info message"**); Log.*d*(***tag***, **"this is a debug message"**); Log.*v*(***tag***, **"this is a verbose message"**); Log.*wtf*(***tag***, **"this is a wtf message"**);  *May 28, 2018. LGF.* |
| iOS |
| print(“this is a message”); |
| Xamarin.Android |
| **Use the following methods:**  Log.Debug(tag, " this is a debug message ");  Log.Info(tag, "this is an info message");  Log.Warn(tag, "this is a warning message");  Log.Error(tag, "this is an error message");  *May 28, 2018. LGF.* |
| Xamarin.iOS |
| **Use the following methods:** |
| Xamarin.Forms |
| **Use the following methods:** |

|  |
| --- |
| 7. Printing setting |
| Angular |
|  |
| Android |
| **For Android Studio:**  Go to menu: File> Print…   1. Paper size: Letter (8 ½ x 11 in). 2. Font: Consolas 10. 3. (Check) Show line numbers. 4. Orientation: Portrait. 5. Margins (inches): Left 0.5, Right 0.5.   *May 28, 2018. LGF.* |
| iOS |
|  |
| Xamarin.Android |
|  |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 8. Crear una nueva actividad / agregar un una nueva escena. |
| Angular |
|  |
| Android |
| Create the activity:   * (right-click) on the package * New> Activity> Empty Activity * Name it, end the name with Activity (e.g. NewTaskFormActivity) and click OK * Leave the name for the layout (e.g. activity\_new\_task\_form) and click Finish.   *May 28, 2018. LGF.* |
| iOS |
| Add a new View Controller Scene to the storyboard:   * Drag from the Object Library a View Controller to the Canvas.   Create the swif file:   * Menú: File> New> File… * Choose Cocoa Touch Class (aunque parezca raro), dar click a next. * Name it, end the name with ViewController (e.g. NewTaskFormViewController). * Subclass of: UIViewController. * Language: Swift, and click Next.   Relacionar la clase con el ViewController:   * Ir al Identity Inspector (ícono a la derecha del Attributes Inspector) * En Custom Class, seleccionar la clase que acaba de crear.   *May 28, 2018. LGF.* |
| Xamarin.Android |
| **En Windows:**  Create the activity:   * (right-click) on the project. * Agregar> Nuevo element... * Seleccionar Visual C#, android, Actividad. * Name it, end the name with Activity (e.g. NewTaskFormActivity) and click Agregar.   Create the layout:   * (right-click) on Resources/layout folder * Agregar> Nuevo element... * Seleccionar Visual C#, android, Diseño de Android. * Name it (e.g. activity\_new\_task\_form.axml) and click Agregar.   Set the layout to the activity:  Add the following line to the onCreate method.  SetContentView(Resource.Layout.activity\_new\_task\_form);  **En MAC:**  Create the activity:   * (ctrl-click) on the project * Add> New File… * Seleccionar Android, Activity. * Name it.   Create the layout:   * (ctrl-click) on Resources/layout folder * Add> Nuevo File... * Seleccionar Android, Layout. * Name it and click New.   Set the layout to the activity:  Add the following line to the onCreate method.  SetContentView(Resource.Layout.activity\_xxx);  *May 28, 2018. LGF.* |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 9. Responder al click a un botón. |
| Angular |
|  |
| Android |
| Agregar el método que se ejecutarán cuando se de click al botón:    **public void** InResponseToButtonXxx(View view) {  *// //To verify the click get the method.* Log.*d*(***tag***, **"click ButtonNewTask"**);}  Registrar el evento:  Opción 1.   * En la vista del diseño, dar click al botón. * En la ventana de atributos, seleccionar onClick * Seleccionar el método que acaba de crear.   Opción 2.  En el método onCreate (java):  final Button button = (Button) findViewById(R.id.button\_id);  button.setOnClickListener(new View.OnClickListener(){  public void onClick(View view{  InResponseToButtonXxx (view);  }  });  Opción 3.  En el layout (xml)  <Button  …  Android:onClick:” InResponseToButtonXxx” />  *May 28, 2018. LGF.* |
| iOS |
| **Use an Assistant Editor:**  View> Assistant Editor> Show Assistant Editor.  (ícono con dos círculos en la esquina superior derecho)  Muestre en el primer editor el storyboard y en el segundo el ViewController.  Ctrl+drag the button from the scene to the code to add an **Action**.  Type the name for the var.  **Code Examples:**  @IBAction func InResponseToButtonXxx(\_ sender: Any){  print(“New Button clicked”);  }  *May 28, 2018. LGF.* |
| Xamarin.Android |
| Agregar el método que se ejecutarán cuando se de click al botón:    **public void** InResponseToButtonXxx(?) {  *// //To verify the click get the method.* Log.Debug(tag, "click ButtonNewTask");  }  Registrar el evento:  Opción 1. (da problemas)   * En la vista del diseño, dar click al botón. * En la ventana de Propiedades, buscar On Click * Teclear el nombre del método que acaba de crear.   Opción 2. (utilizar esta opción)  En el método onCreate (cs):  Button buttonXxx = FindViewById<Button>(Resource.Id.ButtonXxx);  buttonXxx.Click += (sender, args) =>  {  InResponseToButtonXxx (sender, args);  };  Opción 3. (da problemas)  En el layout (axml)  <Button  …  Android:onClick:” InResponseToButtonXxx” />  *May 28, 2018. LGF.* |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 10. Iniciar una actividad / mostrar una nueva escena. |
| Angular |
|  |
| Android |
| Utilizar el método startActivity para iniciar la actividad, pasándole el :Intent.  Intent intent = **new** Intent(getApplicationContext(), NewTaskFormActivity.**class**); startActivity(intent);  *May 28, 2018. LGF.* |
| iOS |
| **En el storyboard:**   * Hacer ctrl+drag desde el botón y arrastrarlo hasta la escena que se desea mostrar, * Crear un Action Segue, Show.   *May 28, 2018. LGF.* |
| Xamarin.Android |
| Utilizar el método StartActivity para iniciar la actividad, pasándole el :Intent.  Intent intent = new Intent(this, typeof(NewTaskFormActivity));  StartActivity(intent);  *May 28, 2018. LGF.* |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 11. Hacer referencia a un elemento de la UI. |
| Angular |
|  |
| Android |
| findViewById(R.id.***xx***)  **R** stands for **R**esources, contains resource ids for all the resources of res directory.  Ejemplos:  **SeekBar:**  SeekBar seekbarPercentage = findViewById(R.id.***SeekBarPercentage***);  seekbarPercentage.getProgress();  seekbarPercentage.setProgress(x);  seekbarPercentage.setMax(100);  **TextView**:  TextView textViewPercentage = findViewById(R.id.***TextViewPercentage***);  textViewPercentage.setText(“…”);  TextView editTextLongDescription = findViewById(R.id.EditTextLongDescription);  String strLongDescription = editTextLongDescription.getText().toString();  **Switch**:  Switch switchDone = findViewById(R.id.***SwitchDone***);  switchDone.isChecked();  *May 28, 2018. LGF.* |
| iOS |
| **Use an Assistant Editor:**  View> Assistant Editor> Show Assistant Editor.  (ícono con dos círculos en la esquina superior derecho)  Muestre en el primer editor el storyboard y en el segundo el ViewController.  Ctrl+drag the view from the scene to the code to add an **Outlet**.  Type the name for the var  **Code Examples:**  **TextView**:  @IBOutlet weak var TaskDoneText: UITextView!  TaskDoneText.text= “…”;    *May 28, 2018. LGF.* |
| Xamarin.Android |
| FindViewById<Button>(Resource.Id.xx);  Ejemplos:  **Button:**  Button buttonNewTask = FindViewById<Button>(Resource.Id.ButtonNewTask);  **SeekBar:**  SeekBar seekBarPercentage = FindViewById<SeekBar>(Resource.Id.SeekBarPercentage);  **TextView:**  TextView textPercentage = FindViewById<TextView>(Resource.Id.textPercentage);  textPercentage.Text = “…”;  **Switch:**  Switch switchDone = FindViewById<Switch>(Resource.Id.SwitchDone);  *May 28, 2018. LGF.* |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 12. Entradas del usuario a través de elementos de la UI. |
| Angular |
|  |
| Android |
| **Button:**  Button button = (Button) findViewById(R.id.button\_id);  button.setOnClickListener(new View.OnClickListener(){  public void onClick(View view{ }  });  **SeekBar:**  SeekBar seekbarPercentage = findViewById(R.id.***SeekBarPercentage***);  seekbarPercentage.setOnSeekBarChangeListener(**new** SeekBar.OnSeekBarChangeListener(){  **public void** onProgressChanged(  SeekBar seekBar, **int** progress, **boolean** fromUser){}  **public void** onStartTrackingTouch(SeekBar seekBar){}  **public void** onStopTrackingTouch(SeekBar seekBar){} });  **Switch:**  Switch switchDone = findViewById(R.id.***SwitchDone***); switchDone.setOnCheckedChangeListener(**new** CompoundButton.OnCheckedChangeListener(){  **public void** onCheckedChanged(  CompoundButton buttonView, **boolean** isChecked){} });  *May 28, 2018. LGF.* |
| iOS |
| **Button:**  *\*Ver procedimiento 9.*  **Slider:**  @IBOutlet weak var PercentageText: UITextView!  @IBAction func PercentageSliderAction(\_ sender: UISlider) {  strPercentage = String(Int(self.PercentageSlider.value)) + "%";  self.PercentageText.text = strPercentage;  }  **Switch:**  @IBAction func DoneSwitchAction(\_ sender: UISwitch) {  if(self.DoneSwitch.isOn){  }  }  *May 28, 2018. LGF.* |
| Xamarin.Android |
| **Button:**  Button buttonNewTask = FindViewById<Button>(Resource.Id.ButtonNewTask);  buttonNewTask.Click += (sender, args) =>  {  };  **SeekBar:**  SeekBar seekBarPercentage = FindViewById<SeekBar>(Resource.Id.seekBarPercentage);  TextView textPercentage = FindViewById<TextView>(Resource.Id.textPercentage);  seekBarPercentage.ProgressChanged += (object sender,  SeekBar.ProgressChangedEventArgs e) =>{  textPercentage.Text = string.Format("{0}%", e.Progress);  };  **Switch:**  Switch switchDone = FindViewById<Switch>(Resource.Id.SwitchDone);  swtchDone.CheckedChange += (object sender,  CompoundButton.CheckedChangeEventArgs e) =>{  if(e.IsChecked)  {  }  }; |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 13. Send a Broadcast / Register a receiver of a Broadcast |
| Angular |
|  |
| Android |
| Send the broadcast:  Log.*d*(**"LGF "**, **"Sending a broadcast"**); Intent intent = **new** Intent(); intent.setAction(**"com.LGF.CUSTOM\_INTENT.Xxxx"**); **context**.sendBroadcast(intent);  Create the receiver:   * Crear una clase que derive de BroadclassReceiver. * Sobreescribir el método onReceive()   *//-------------------------------------------------------------------------------* **private class Xxx**Receiver **extends** BroadcastReceiver {  @Override  **public void** onReceive(Context context, Intent intent) {  // TODO here.  } }  *//-------------------------------------------------------------------------------*  BroadcastReceiver XxxReceiver = **new** XxxReceiver();  Register the receiver (onStart)  IntentFilter intentFilter = **new** IntentFilter(**"com.LGF.CUSTOM\_INTENT.Xxx"**); **this**.registerReceiver(**this**.XxxReceiver, intentFilter);  Unregister the receiver (onPause)  BroadcastReceiver XxxReceiver = **new** XxxReceiver (); **this**.unregisterReceiver(**this**.XxxReceiver);  *May 28, 2018. LGF.* |
| iOS |
|  |
| Xamarin.Android |
|  |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 14. Threads and Asynchronous communication |
| Angular |
|  |
| Android |
| * Crear una clase interna que derive de AsyncTask. * Sobreescribir el método doInBackground para realizar la operación a la DB. * Crear un método public static que instancié la clase interna y levante un thread mediante el método execute(). * Sobreescribir el método onPostExecute para avisar cuando se haya recuperado la información.   *May 28, 2018. LGF.* |
| iOS |
|  |
| Xamarin.Android |
|  |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 15. Crear una DB para guardar y recuperar datos |
| Angular |
|  |
| Android |
| **Agregar Room Persistence Library into your Android project**    Agregar las siguientes líneas al archivo build.gradle (Module:app), dentro del bloque  dependencies {  *//Room components* **def** room\_version = **"1.1.0"** implementation **"android.arch.persistence.room:runtime:**$room\_version**"** annotationProcessor **"android.arch.persistence.room:compiler:**$room\_version**"**  Sincronizar el proyecto.  **Crear una clase para modelar la entidad.**  Agregar decoradores a la clase para indicar entidad, PK y columnas.  **import** android.arch.persistence.room.ColumnInfo; **import** android.arch.persistence.room.Entity; **import** android.arch.persistence.room.PrimaryKey; @Entity **public class** Task {  @PrimaryKey(autoGenerate = **true**)  **private int id**;  @ColumnInfo(name = **"shortDescription"**)  **private** String **shortDescription**;  @ColumnInfo(name = **"longDescription"**)  **private** String **longDescription**;  @ColumnInfo(name = **"percentage"**)  **private int percentage**;  *//getters y setters* }  **Crear una interface a la DB.**  **import** android.arch.persistence.room.Dao; **import** android.arch.persistence.room.Delete; **import** android.arch.persistence.room.Insert; **import** android.arch.persistence.room.Query; **import** java.util.List; @Dao **public interface** TaskDAO {  @Query(**"SELECT \* FROM Task"**)  List<Task> getAll();  @Query(**"SELECT \* FROM Task WHERE percentage = 0"**)  List<Task> getToDo();  @Query(**"SELECT \* FROM Task WHERE percentage > 0 and percentage < 100"**)  List<Task> getDoing();  @Query(**"SELECT \* FROM Task WHERE percentage = 100"**)  List<Task> getDone();  @Insert  **void** insertTask(Task task);  @Delete  **void** deleteTask(Task task); }  **Crear una instancia de la DB.**  @Database(entities = {Task.**class**}, version = 1) **public abstract class** TaskDB **extends** RoomDatabase {  **public abstract** TaskDAO taskDAO();  **private static** TaskDB *INSTANCE*;  **public static** TaskDB getTaskDB(**final** Context context) {  **if** (*INSTANCE* == **null**) {  *INSTANCE* = Room.*databaseBuilder*(  context.getApplicationContext(),  TaskDB.**class**, **"task-database"**).build();  }  **return** *INSTANCE*;  }  **public static void** destroyInstance() {  *INSTANCE* = **null**;  } }  **Crear una clase DBUtil con métodos que hagan operaciones a la DB.**  **Para insertar:**   * Crear una clase interna que derive de AsyncTask. * Sobreescribir el método doInBackground para realizar la operación a la DB. * Crear un método public static que instancié la clase interna y levante un thread mediante el método execute().   **public class** DBUtil {  **public static void** DBSaveNewTask(TaskDB taskDBInstance, Task task){  SaveNewTask saveNewTask = **new** SaveNewTask(taskDBInstance, task);  saveNewTask.execute();  }  *//---------------------------------------------------------------------------* **private static class** SaveNewTask **extends** AsyncTask<Void, Void, Void> {  TaskDB **taskDBInstance**;  Task **task**;   **public** SaveNewTask(TaskDB taskDBInstance, Task task) {  **this**.**taskDBInstance** = taskDBInstance;  **this**.**task** = task;  }  @Override  **protected** Void doInBackground(**final** Void... params) {  **taskDBInstance**.taskDAO().insertTask(**task**);  Log.*d*(**"LGF "**, **"Saving new task "**);  **return null**;  }  }  *//---------------------------------------------------------------------------* }  **Para recuperar datos:**   * Hacer lo mismo que para insertar. * Adicionalmente, sobreescribir el método onPostExecute para avisar cuando se haya recuperado la información.   **public class** DBUtil {  **static** List<Task> *tasks* = **new** ArrayList<>();  **public static** List<Task> getTasks() {  **return** *tasks*;  }  **public static** List<Task> DBGetAllTask(  TaskDB taskDBInstance, Context context){  GetAllTask getAllTask = **new** GetAllTask(taskDBInstance, context);  getAllTask.execute();  **return** *tasks*;  }  *//---------------------------------------------------------------------------*  **private static class** GetAllTask **extends** AsyncTask<Void, Void, Void> {  TaskDB **taskDBInstance**;  Context **context**;   **public** GetAllTask(TaskDB taskDBInstance, Context context) {  **this**.**taskDBInstance** = taskDBInstance;  **this**.**context** = context;  }  @Override  **protected** Void doInBackground(**final** Void... params) {  *tasks* = **taskDBInstance**.taskDAO().getAll();  Log.*d*(**"LGF "**, *tasks*.size() + **" Tasks in DB"**);  **return null**;  }  @Override  **protected void** onPostExecute(Void v)  {  Log.*d*(**"LGF Broadcast "**, **"TasksReady"**);  Intent intent = **new** Intent();  intent.setAction(**"com.LGF.CUSTOM\_INTENT.TasksReady"**);  **context**.sendBroadcast(intent);  }  }  *//---------------------------------------------------------------------------* }  **Utilizar los métodos de la clase DBUtil:**  **Para insertar:**  *// //Asking to the DB to save.* TaskDB taskDBInstance = TaskDB.*getTaskDB*(getApplicationContext()); DBUtil.*DBSaveNewTask*(taskDBInstance, task);   * Destruir la instancia en el método public void onDestroy()de la actividad.   TaskDB.destroyInstance();  **Parea desplegar datos recuperados:**   * Crear la instancia de la DB (por ejemplo en onResume).   *// //Asking to the DB for data.* TaskDB taskDBInstance = TaskDB.*getTaskDB*(getApplicationContext()); DBUtil.*DBGetAllTask*(taskDBInstance, getApplicationContext());   * Crear el receiver en la actividad que utilizara los datos.   *//-------------------------------------------------------------------------------* **private class** ShowTaskReceiver **extends** BroadcastReceiver {  @Override  **public void** onReceive(Context context, Intent intent) { List<Task> listOfTask = DBUtil.*getTasks*();  **for** (Task task: listOfTask){  Log.*d*(**"LGF - Tasks "**, task.getShortDescription() + **", "** +  String.*valueOf*(task.getPercentage()));  }  } }  *//-------------------------------------------------------------------------------*   * Declarar el receiver.   BroadcastReceiver showTaskReceiver = **new** ShowTaskReceiver();   * Registrar el receiver (en el método onStart).   IntentFilter intentFilter = **new** IntentFilter(**"com.LGF.CUSTOM\_INTENT.TasksReady"**); **this**.registerReceiver(showTaskReceiver, intentFilter);   * Unregistrar el receiver (en el método onPause).   **this**.unregisterReceiver(**this**.**showTaskReceiver**);   * Destruir la instancia de la DB (en el método onDestroy).   TaskDB.destroyInstance();  *May 28, 2018. LGF.* |
| iOS |
|  |
| Xamarin.Android |
| **SQLite-net-pcl**  **En Windows** (en la consola del administrador de paquetes):  PM> Install-Package sqlite-net-pcl  **En MAC** (Solution Window)  Packages, ctrl+click, Add Packages…  Buscar (exactamente) sqlite-net-pcl y agregar.  Agregar using SQLite;  // //path string for the database file  string dbpath = Path.Combine(  System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal), "dbTest.db");  // //setup the db connection  var db = new SQLiteConnection(dbpath);  // //setup a table  db.CreateTable<Task>();  // //store the task in the table  db.Insert(task);  // //acceder a los datos  var table = db.Table<Task>();  foreach(var row in table){  Task PersistenceTask =  new Task(row.shortDescription, row.longDescription, row.percentage);  Log.Debug(tag, "FROM THE DB TASK: " + PersistenceTask.ToString());  } |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 16. Desplegar elementos de una lista. |
| Angular |
|  |
| Android |
| **Agregar soporte para RecyclerView.**    Agregar la siguiente línea al archivo archivo build.gradle (Module:app), dentro del bloque  dependencies {  *//RecyclerView //The RecyclerView widget is a more advanced and flexible version of ListView.* implementation **'com.android.support:recyclerview-v7:27.1.1'**  **Crear una nueva actividad e incluir un RecyclerView.**  *Ver procedimiento 8 para agregar la nueva actividad.*  Agregar el RecyclerView al layout asociado a la tarea.  **Disponer de la lista de datos a mostrar.**  *(Por ejemplo pedirlos a una DB y esperar por ellos en un receiver)*  **Crear un *layout* para modelar un elemento de la lista.**   * (click-derecho) sobre la carpeta layout. * New> Layout resource file. * Nombrarlo single\_task\_item. * Utilizar un LinearLayout y agregar los elementos necesarios, por ejemplo:   *<?***xml version="1.0" encoding="utf-8"***?>* <**LinearLayout  xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical"**>  <**TextView  android:id="@+id/taskItem"  android:layout\_width="match\_parent"  android:layout\_height="70dp"  android:textSize="22sp"** />  *<!--Line separator between list items-->* <**View  android:layout\_width="match\_parent"  android:layout\_height="3dp"  android:background="#000"** /> </**LinearLayout**>  **Crear un adaptador para el RecyclerView.**  Crear una clase que derive de Adapter.  Agregar una variable para la lista a desplegar.  Agregar un constructor que reciba la lista e inicialice la variable.  Sobreescribir los métodos:   * onCreateViewHolder() * onBindViewHolder() * getItemCount()   **public class** ListTaskAdapter **extends** Adapter<ListTaskAdapter.TaskHolder> {  **private** List<Task> **listOfTasks**;  **public** ListTaskAdapter(List<Task> listOfTasks) {  **this**.**listOfTasks** = listOfTasks;  }   *//--------------------------------------------------------------------------------------* @Override  **public** ListTaskAdapter.TaskHolder onCreateViewHolder(ViewGroup parent, **int** viewType)  {  LayoutInflater inflater = LayoutInflater.*from*(parent.getContext());  View view = inflater.inflate(R.layout.***single\_task\_item***, parent, **false**);   TaskHolder taskHolder = **new** TaskHolder(view);  **return** taskHolder;  }  *//- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -* **class** TaskHolder **extends** ViewHolder {  TextView **textView**;   **public** TaskHolder(View view) {  **super**(view);   **textView** = (TextView) **itemView**.findViewById(R.id.***taskItem***);  }  }  *//--------------------------------------------------------------------------------------* @Override  **public void** onBindViewHolder(ListTaskAdapter.TaskHolder holder, **int** position) {   Task task = **listOfTasks**.get(position);  String strTask = task.getShortDescription() + **" "** +  String.*valueOf*(task.getPercentage())+ **"%"**;  holder.**textView**.setText(strTask);  }  *//--------------------------------------------------------------------------------------* @Override  **public int** getItemCount() {  **return listOfTasks**.size();  }  *//--------------------------------------------------------------------------------------* } */\*END-ADAPTER\*/*  **Establecer el *adaptador y el manager para el* RecyclerView.**  *// //Referencing the RecyclerView* RecyclerView recyclerView = findViewById(R.id.recycletViewTasks);  *// //Setting the adapter*  ListTaskAdapter listTaskAdapter = **new** ListTaskAdapter(listOfTask); recyclerView.setAdapter(listTaskAdapter);  *// //Setting the manager* LinearLayoutManager manager = **new** LinearLayoutManager(  getApplicationContext(), LinearLayoutManager.***VERTICAL***, **false**); recyclerView.setLayoutManager(manager);  *May 28, 2018. LGF.* |
| iOS |
|  |
| Xamarin.Android |
| **En Mac (Utilizando un ListView):**  **Crear una nueva actividad para incluir .**  *Ver procedimiento 8 para agregar la nueva actividad.*  Agregar un ListView, *drag* from the Toolbox to the Document Outline and name it.  **Disponer de la lista de datos a mostrar.**  *(Por ejemplo recuperarlo desde SQLite)*  **Crear un *layout* para modelar un elemento de la lista.**   * (ctrl-click) sobre la carpeta layout. * Add> New File… * Escoger Android Layout. * Nombrarlo single\_task\_item y click New. * Utilizar un LinearLayout y agregar los elementos necesarios, por ejemplo:     **Crear un adaptador para el ListView.**  Crear una clase que derive de BaseAdapter.  Agregar una variable para la lista a desplegar.  Agregar un constructor que reciba la lista e inicialice la variable.  Sobreescribir los fields y methods:   * Count * GetItemId() * GetItem() * GetView()   public class ListTaskAdapter: BaseAdapter{  List<Task> listOfTask;  public ListTaskAdapter(List<Task> listOfTask){  this.listOfTask = listOfTask;  }  public override int Count {  get { return listOfTask.Count; }  }  public override long GetItemId(int position){  throw new NotImplementedException();  }  public override Java.Lang.Object GetItem(int position){  throw new NotImplementedException();  }  public override View GetView(int position, View convertView, ViewGroup parent){  LayoutInflater inflater = LayoutInflater.From(parent.Context);  var view = inflater.Inflate(Resource.Layout.single\_task\_item, parent, false);  var textView = view.FindViewById<TextView>(Resource.Id.taskItem);  Task task = listOfTask[position];  textView.Text = task.shortDescription + " " + task.percentage + "%";  return view;  }  }  **Establecer el adaptador el ListView.**  *// //Referencing ListView*  var taskListView = FindViewById<ListView>(Resource.Id.listViewTaskList);  *// //Setting the adapter*  var listTaskAdapter = new ListTaskAdapter(listOfTask);  taskListView.Adapter = listTaskAdapter;  **En Mac (Utilizando un RecyclerView):**  **Agregar soporte para RecyclerView.**  Descargar el paquete de [*https://www.nuget.org/packages/Xamarin.Android.Support.v7.RecyclerView/*](https://www.nuget.org/packages/Xamarin.Android.Support.v7.RecyclerView/)  Place the nugpkg file in a specific directory.  Ctrl-click to Packages, then click Add Packages.  Click the combo and change to Configure Sources…  Click Add to select the directory where you place the nugpkg file and click OK.  Select the Xamarin.Android.Support.v7.RecyclerView and click Add Package.  \*Nota: De aquí en eadelante, mismo procedimiento que para Android.  *May 28, 2018. LGF.* |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 17. Adding a Toolbar Back Button / Navigation Controller. |
| Angular |
|  |
| Android |
| En el archivo styles.xml, cambia en parent theme a:  <**style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar"**>  Cree un archivo de ícono  Click on res and then right click on drawable > New > Vector Asset.  Select the Back Arrow from the Clip Art  Modify the color in the source if desired  <**vector xmlns:android="http://schemas.android.com/apk/res/android"  android:width="24dp"  android:height="24dp"  android:viewportWidth="24.0"  android:viewportHeight="24.0"**>  <**path  android:fillColor="#FFffffff"  android:pathData="M20,11H7.83l5.59,-5.59L12,4l-8,8 8,8 1.41,-1.41L7.83,13H20v-2z"**/> </**vector**>  Modifique el layout, abra el archivo en la vista de diseño para…   * Arrastrar un Toolbar. * Agregar los constraints necesarios. * Si se requiere, modificar los constraints de los otros controles para lograr la vista deseada. * Establecer el atributo title. * Establecer el atributo NavigationIcon. * Modify the textcolor if desired.   Cree el método para responder al click, si es la actividad principal, utilice un método como el siguiente:  *//-------------------------------------------------------------------------------* **public void** FinishApp() *// //onClick-toolbar (Toolbar back button)* {  Log.*d*(***tag***, **"Bye Li..."**);  finish(); }  \*Se deben modificar todos los layouts.  *May 28, 2018. LGF.* |
| iOS |
| * Agregar un NavigationController, desde el Object Library al canvas del storyboard. * Elimina el Root View Controller que trae asociado el NavigationController. * Haz que el NavigationController sea el Initial View Controller * Especifica cual será el Root View Controller, ctrl+drag desde el NavigationController al ViewController deseado y seleccionar root view controller.   *May 28, 2018. LGF.* |
| Xamarin.Android |
|  |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

|  |
| --- |
| 20. Solución de problemas |
| Angular |
| **Se presenta al teclear:**  > ng serve --open (ng serve –o, ng serve)  **Se manifiesta:**  Error message: node\_modules appears empty.  **Explicación:**  Not in the app directory.  **Solution:**  cd to the app directory. |
| Android |
| **Se presenta al teclear:**  **Se manifiesta:**  se obtiene el mensaje:  Installation failed with message Invalid File: C:\AndroidStudioProjects\taskapp\app\build\intermediates\split-apk\debug\slices\slice\_2.apk. It is possible that this issue is resolved by uninstalling an existing version of the apk if it is present, and then re-installing.  **Solution:**  Menu: Build > Clean Project  Menú: Build > Build APK(s)  Run again. |
| iOS |
|  |
| Xamarin.Android |
|  |
| Xamarin.iOS |
|  |
| Xamarin.Forms |
|  |

1. Pendiente checar nombre que aparece en inglés. [↑](#footnote-ref-1)
2. angular CLI calls tsc under the hood, bundles are injected at runtime. [↑](#footnote-ref-2)