

MANUAL TÉCNICO

Contenido

1.	PRO	PÓSITO	3
2.	AND	ROID STUDIO IDE	3
2.	.1.	Introducción	3
2.	.2.	Instalación	3
2.	.3.	Emulador	4
2.	.4.	Descarga e instalación de versiones	6
2.	.5.	Librerías	7
3.	SQLI	TE	8
3.	.1.	Herramientas	8
3.	.2.	Diagrama Entidad Relacion	8
4.	PRO	ҮЕСТО АРР	8
4.	.1.	Estructura	8
4.	.2.	Paquetes y clases	4
4.	.3.	Vistas	6
5.	GOC	DGLE PLAY1	6
5.	.1.	Introducción1	6
5.	.2.	Publicación de una app1	7

1. PROPÓSITO

El presente documento tiene como finalidad de proveer información sobre la estructura de la aplicación móvil Android implementada. Conocer sus partes, la manera de cómo fue construida, etc.

2. ANDROID STUDIO IDE

2.1. Introducción

Es el entorno de desarrollo integrado (IDE) exclusivo para Android desarrollado por Google Inc. Incluye el SDK y el emulador de un terminal. Además ofrece las siguientes características:

- Un entorno de desarrollo claro y robusto.
- Facilidad para testear el funcionamiento en otros tipos de dispositivos.
- Asistentes y plantillas para los elementos comunes de programación en Android.
- Un completo editor con muchas herramientas extra para agilizar el desarrollo de nuestras aplicaciones.

Al crear un nuevo proyecto en Android Studio, la estructura del proyecto aparece con casi todos los archivos dentro del directorio SRC, un cambio a un sistema de generación basado Gradle que proporcionará una mayor flexibilidad para el proceso de construcción. Además, gracias a su sistema de emulación integrado, Android Studio permite ver los cambios que realizamos en nuestra aplicación en tiempo real, pudiendo además comprobar cómo se visualiza en diferentes dispositivos Android con distintas configuraciones y resoluciones de forma simultánea.

Se seleccionó éste IDE, porque lleva más de 4 años en el mercado y ha sido catalogado cómo excelente por los programadores a nivel mundial, por encima de su predecesor el IDE Eclipse.

2.2. Instalación

Para descargar el Android Studio IDE, nos dirigimos a la página oficial de Android, <u>http://developer.android.com/intl/es/sdk/index.html</u>, y descargamos el paquete que muestra la página.



Antes de ejecutar el paquete descargado, se debe verificar que versión de JDK se tiene instalado en la computadora. La versión mínima de JAVA es la 1.8. En caso de no tener instalado JAVA, proceda a instalarlo.

En el paquete descargado se incluye el JKD y el emulador de Android para realizar los test. A continuación unas capturas de la instalación.

Android Studio Setup			Android Studio Setup		
	Welcome to Android Stu	dio Setup	<u>e</u>	hoose Components Choose which features of Androi	d Studio you want to install.
K	Setup will guide you through the installat Studio. It is recommended that you close all othe before starting Setup. This will make it pe relevant system files without having to re computer.	on of Android r applications suble to update aboot your	Check the components you wa install. Click Next to continue. Select components to install:	ent to install and uncheck the con	nponents you don't want to Description Position your mouse
Android	Click Next to continue.			Android Virtual Device	over a component to see its description.
Studio			Space required: 4.258		
Android Studio Setun	< gad light >	Cancel	Android Studio Setur	< Back	Next Cancel
	Configuration Settings Instal Locations			t alling case wait while Android Studio is	being installed.
Android Studio Installation I The location specified mu	Location at have at least 500MB of free space.		Extract: nanoxmi-2.2.3.jar 1	2015	
C: Program Piles (Androi	s. id (Android Studio	Browse	Show details		
Android SDK Installation Lo	cation				
The location specified mu Click Browse to customize	st have at least 3.208 of free space. ti				
C: Users Ismain AppDat	a Local JAndroid Ipdk	Browse			
	< Back Next 7	Cancel		< Back	Next > Cancel

2.3. Emulador

Cuando se instala el AndroidStudio, viene incluido el administrador de emulador. Para lo cual se debe hacer clic en el icono "AVD MANAGER", para abrir la ventana donde se puede administrar los diferentes emuladores de acuerdo al hardware y software que se requiera.



En esta ventana nos muestra una lista de emuladores creados, si deseas crear un emulador nuevo se debe hacer clic en el botón "CREATE VIRTUAL DEVICE".

ype	Name	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
Ц	Nexus 4 API 17 Jelly	768 × 1280: xhdpi	17	Android 4.2.2	×86	560 MB	
	Nexus 5 API 19 Kit	1080 = 1920: xxhdpi	19	Google APIs (x86 System Image)	x86	1 GB	× /
	Nexus 6 API 22 Lollipop	1440 × 2560: 560dpi	22	Android 5.1.1	x86_64	1 GB	× /
	Nexus 6 API 23 Marshmallow	1440 × 2560: 560dpi	23	Android 6.0	×86_64	1 GB	× /

A continuación vamos a crear un emulador con el hardware de un NEXUS 5 de Google.

👳 Virtual Device 🤇	Configuration				
Sel Choos	ect Hardwar se a device definition	e			
	Q.				Nerver 6
Category	Name 🔻	Size	Resolution	Density	Nexus 6
тν	Nexus S	4,0"	480×800	hdpi	
Phone	Nexus One	3,7"	480x800	hdpi	Size:
Wear	Nexus 6	5,96"	1440x2560	560dpi	Ratio: notiong Density: 560dpi
Tablet	Nexus 5	4,95"	1080x 1920	xxhdpi	5.98" 2560 px
	Nexus 4	4,7"	768x1280	xhdpi	
	Galaxy Nexus	4,65"	720x1280	xhdpi	
	5.4" FWVGA	5,4"	480x854	mdpi	
	5.1" WVGA	5,1"	480x800	mdpi	
	4.7" WXGA	4,7"	720x1280	xhdpi	
New Hardware	Profile Import H	ardware Profiles		<u> </u>	Clone Device
					Previous Next Cancel Finish

Y la versión del sistema operativo de Android, es con la que viene por defecto que es Marshmallow, Android 6.0.

Virtual Device Configuration	•			
System image Select a system image	e			
Release Name	API Level	ABI	Target	Marshmallow
Marshmallow	23	x86	Android 6.0	
Marshmallow	23	x86_64	Android 6.0	API Level
Marshmallow	23	armeabi-v7a	Google APIs (Google Inc.) - google_apis [(23
Marshmallow	23	x86	Google APIs (Google Inc.) - google_apis [(Android
Lollipop	22	armeabi-v7a	Android 5.1.1	6.0
Lollipop	22	x86	Android 5.1.1	Android Open
Lollipop	22	x86_64	Android 5.1.1	Source Project
Lollipop	22	armeabi-v7a	Google APIs (Google Inc.) - google_apis [C	System Image
Lollipop	21	armeabi-v7a	Android 5.0.1	x86
Lollipop	21	x86	Android 5.0.1	
Lollipop	21	x86_64	Android 5.0.1	
KitKat	19	armeabi-v7a	Android 4.4.2	
KitKat	19	×86	Android 4.4.2	Questions on API level?
Show downloadable system image	qes		∜€ Refreshing 🚺	See the API level distribution chart

Hacemos clic en siguiente, click en finalizar y obtendremos nuestro emulador listo para arrancarlo y realizar las pruebas de la app que se vaya a desarrollar.

WD Name	Nexus 6 API 23		AVD Name	
Nexus 6	5,96" 1440x2560 560dpi	Change	The name of this AVD.	
	Android 6.0 x86	Change		
itartup size Ind Irientation	Scale: Auto			
mulated Performance	Use Host GPU Store a snapshot for faster startup You can either use Host GPU or Snapshots			
Device Frame	Enable Device Frame			

2.4. Descarga e instalación de versiones

Ahora, que pasa si queremos hacer pruebas con versiones de Android más antiguas. Para esto debemos recurrir al botón "SDK MANAGER", como se muestra en la figura.

ts\Kichwal	.earn] - [a	pp]\ap	p/src/ı	main\java	l\cquil	umb	aqui∖	kich	walear	n∖M	ainAc
R <u>u</u> n <u>T</u> oo	ls VC <u>S</u>	$\underline{W}indow$	<u>H</u> elp					_			
📑 MainActivity 💌 🕨 🗰 🕪 🐘 😤 🖬 🗳 🗭 ? 🎩											
uilumbaq	uilumbaqui 👌 🛅 kichwalearn 🛇 🤆 MainActivity 🖉										
₩~ 10	C Mai	nActivity.ja	ava ×	C SPI	ayer.ja	va ×		🖻 tal	bhistor	y_la	yout.
Project				xf	ragme	ntTr	ansa	acti	on.re	pla	ce (R
				if (mer	nuIte	m.ge	tIte	emId	() ==	R.	id.n

En esta ventana, en la pestaña "SDK PLATFORM", nos mostrará una lista de las versiones de Android que se encuentran instaladas. Si deseas instalar o desinstalar alguna versión, solamente se debe seleccionar y dar click en "OK". Una vez finalizada las descargas correspondientes de las versiones de Android, estarán disponibles para crear en el "AVD MANAGER".

)	Appearance & Behavio	r > System Settings > Android	SDK						
Appearance & Behavior	Manager for the Androi	d SDK and Tools used by Android	Studio						
Appearance	Android SDK Location:	C:\Users\Carlos\AppData\Loca	l\Android\sdk						
Menus and Toolbars	SDK Platforms SDK T	ools SDK Update Sites							
System Settings	Each Android SDK Plat	Each Andreid CDK Deatern and each include the Andreid deatern and an annual triangle and ADI land by							
	default. Once installed	Android Studio will automatica	llv check for update	s. Check "sho	w package details" to				
HTTP Proxy	display individual SDK	components.							
Updates		Name	API Level	Revision		Status			
Usage Statistics	Android	N Preview	N	2	Not installed	515165			
Android SDK	- Android	6.0 (Marshmallow)	23	2	Update available				
Notifications	Android	5.1 (Lollipop)	22	2	Update available				
Ouick Lists	Android	5.0 (Lollipop)	21	2	Update available				
Caree Elsas	Android	4.4 (KitKat Wear)	20	2	Not installed				
Кеутар	Android	4.4 (KitKat)	19	4	Update available				
Editor	Android	4.3 (Jelly Bean)	18	3	Installed				
Plugins	🗹 Android	4.2 (Jelly Bean)	17	3	Installed				
Build, Execution, Deployment	🗹 Android	4.1 (Jelly Bean)	16	5	Installed				
Taala	Android	4.0.3 (IceCreamSandwich)	15	5	Not installed				
TOOIS	Android	4.0 (IceCreamSandwich)	14	4	Not installed				
Other Settings	🗹 Android	3.2 (Honeycomb)	13		Not installed				
	🗹 Android	3.1 (Honeycomb)	12		Not installed				
	🗹 Android	3.0 (Honeycomb)	11		Not installed				
	🗹 Android	2.3.3 (Gingerbread)	10		Not installed				
	🗹 Android	2.3 (Gingerbread)	9		Not installed				
	🗹 Android	2.2 (Froyo)	8		Not installed				
	Android	2.1 (Eclair)	7		Not installed				
						Show Package De			
	Launch Standalone SDK	Manager		Preview	packages available! <u>Sw</u>	<u>itch</u> to Preview Channel to see			

2.5. Librerías

Para la implementación de la app, recurrí a la utilización de algunas librerías que provee Google, las cuales se configura en el archivo "build.gradle", y se encuentra tal y como se muestra en la figura.





Aquí se procede a escribir las sentencias tal y como se muestra en la figura.

Una vez hecho esto, realizamos un "Build", para que se sincronice las librerías al proyecto.

3. SQLITE

3.1. Herramientas

Para el presente proyecto se realizó con el gestor de base de datos SQLite, utilizando para la administración los pequeños programas: SQLiteBrowser y Kata Kuntur.



3.2. Diagrama Entidad Relacion

El diagrama entidad relación de la app, cuyo contenido se muestra el diseño del esquema de la base de datos del aplicativo, es el siguiente:

kichwa_entry	android_metadata		
_id: INTEGER	locale: TEXT		
hw: TEXT			
gloss: TEXT			
ehw: TEXT			
phonetics: TEXT	sqlite_sequence		
pos: TEXT	name:		
ex_kichwa: TEXT	seq		
ex_xeng: TEXT			
sem1: TEXT			
see_also: TEXT			
notes to reader: TE			

En la tabla "kichwa_entry", se almacena más de 3000 registros correspondientes al diccionario Kichwa – Inglés, con todas las traducciones, sinónimos, oraciones de ejemplos, etc.

4. PROYECTO APP

4.1. Estructura

En el IDE Android Studio, en la parte de la izquierda desplegamos las carpetas del proyecto y observamos su estructura en forma de árbol de directorios. Sin embargo por defecto lo vemos de una forma un tanto peculiar que podría llevarnos a la confusión, para entender mejor la

estructura del proyecto vamos a cambiar momentáneamente la forma en que AndroidStudio nos muestra. Para ello pulsaremos sobre la lista desplegable situada en la parte superior de la izquierda, y la cambiaremos la vista de proyecto al modo "Project".



Tras hacer esto, la estructura del proyecto cambia un poco el aspecto y pasa a ser como se observa en la siguiente imagen.



A continuación describiremos los elementos principales de la estructura.

Lo primero que debemos distinguir son los conceptos de proyecto y módulo. La entidad proyecto es única, y engloba a todos los demás elementos. Dentro de un proyecto podemos incluir varios módulos, que pueden representar aplicaciones distintas, versiones diferentes de una misma aplicación, o distintos componentes de un sistema (aplicación móvil, aplicación servidor, librerías, ...). En la mayoría de los casos, trabajaremos con un proyecto que contendrá un sólo módulo correspondiente a nuestra aplicación principal. Por ejemplo en este caso que estamos creando tenemos el proyecto "KichwaLearn" que contiene al módulo "app" que contendrá todo el software de la aplicación de ejemplo.

E	🔁 KichwaLearn 🔪	
t	🗊 Project 🔹	⊕ ≑ 🖗- ⊮
Proj	🔻 📑 KichwaLearn (C:\User	s\Carlos\Dropbox\AndroidSt
÷	🗉 🕨 🖿 .gradle	
~	🕨 🖿 .idea	
e	, 🕨 🛅 app	
du	🕨 🖿 build	
Stru	🕨 🗖 gradle	
14	gitignore	
Y	💿 build.gradle	
n	a aradle.properties	

A continuación se describe los contenidos principales de la aplicación central.

Carpeta /app/src/main/java

Esta carpeta contendrá todo el código fuente de la aplicación, clases auxiliares, etc. Inicialmente, Android Studio creará por nosotros el código básico de la pantalla (actividad o activity) principal de la aplicación, que recordemos que en nuestro caso era MainActivity, y siempre bajo la estructura del paquete java definido durante la creación del proyecto.



Carpeta /app/src/main/res/

Contiene todos los ficheros de recursos necesarios para el proyecto: imágenes, layouts, cadenas de texto, etc. Los diferentes tipos de recursos se pueden distribuir entre las siguientes subcarpetas:

Carpeta	Descripción
	Contiene las imágenes y otros elementos gráficos usados por la aplicación. Para poder definir diferentes recursos dependiendo de la resolución y densidad de la pantalla del dispositivo se suele dividir en varias subcarpetas:
/res/drawable/	/drawable (recursos independientes de la densidad)
	/drawable-ldpi (densidad baja)
, ,	/drawable-mdpi (densidad media)
	/drawable-hdpi (densidad alta)
	/drawable-xhdpi (densidad muy alta)
	/drawable-xxhdpi (densidad muy muy alta :)
/res/mipmap/	Contiene los iconos de lanzamiento de la aplicación (el icono que aparecerá en el menú de aplicaciones del dispositivo) para las distintas densidades de

	pantalla existentes. Al igual que en el caso de las carpetas /drawable, se dividirá en varias subcarpetas dependiendo de la densidad de pantalla: /mipmap-mdpi, /mipmap-hdpi, /mipmap-xhdpi, etc.
/res/layout/	Contiene los ficheros de definición XML de las diferentes pantallas de la interfaz gráfica. Para definir distintos layouts dependiendo de la orientación del dispositivo se puede dividir también en subcarpetas:
	/layout (vertical), /layout-land (horizontal)
/res/anim/ /res/animator/	Contienen la definición de las animaciones utilizadas por la aplicación.
/res/color/	Contiene ficheros XML de definición de listas de colores según estado.
/res/menu/	Contiene la definición XML de los menús de la aplicación.
/res/xml/	Contiene otros ficheros XML de datos utilizados por la aplicación.
/res/raw/	Contiene recursos adicionales, normalmente en formato distinto a XML, que no se incluyan en el resto de carpetas de recursos.
/res/values/	Contiene otros ficheros XML de recursos de la aplicación, como por ejemplo cadenas de texto (strings.xml), estilos (styles.xml), colores (colors.xml), arrays de valores (arrays.xml), tamaños (dimens.xml), etc.

No todas estas carpetas tienen por qué aparecer en cada proyecto Android, tan sólo las que se necesiten. Como ejemplo, para un proyecto nuevo Android como el que hemos creado, tendremos por defecto los siguientes recursos para la aplicación:



Entre los recursos creados por defecto cabe destacar los *layouts*, en nuestro caso ya se tiene algunos layouts creados, pero el principal es el llamado "activity_main1.xml", que contienen la definición de la interfaz gráfica de la pantalla principal de la aplicación. Si hacemos doble clic

sobre este fichero Android Studio nos mostrará esta interfaz en su editor gráfico, y como podremos comprobar, contiene toda la interfaz gráfica inicial.



Pulsando sobre las pestañas inferiores "Design" y "Text" podremos alternar entre el editor gráfico (tipo arrastrar-y-soltar), mostrado en la imagen anterior, y el editor XML que se muestra en la imagen siguiente:

v	⊕ ≑ ∳• ⊮	🖸 a	ctivity_main1.xml	×	
assets			xml version=</td <td>"1.0" encoding="utf-8"?></td>	"1.0" encoding="utf-8"?>	
🕨 🛅 java		C 🕂	<linearlayout< td=""><td>xmlns:android="http://schemas.android.com/apk/res/android"</td></linearlayout<>	xmlns:android="http://schemas.android.com/apk/res/android"	
▼ Cm res		<pre>xmlns:app="http://schemas.android.com/apk/res-auto"</pre>			
drawable	e		xmlns:fab="http://schemas.android.com/tools"		
T lavout		android:orientation="vertical"			
- Idyout	the second second	android:layout_width="match_parent"			
	it i ayout xmi		android:layout_height="match_parent"		
o activ	ity_classes_details.xm		android:fi	tsSystemWindows="true"	
🔯 activ	ity_dictionary.xml		android:ba	ckground="@color/fondo_app">	
🔯 activ	ity_dictionary_details.		(android a	www.st.w?.widget Toolbay	
🔯 activ	ity_game_how_to.xm		xmlne.	apportandroid.com/ant/res/android"	
🔯 activ	ity_game_principal.xr		androi	1:layout width="match parent"	
🧰 activ	ity_game_splayer.xml		androi	i:layout height="wrap content"	
🙆 activ	ity_list_category.xml		androi	d:background="@color/orange"	
activ	ity main.xml		androi	d:id="@+id/toolbar"	
activ	ity main1 yml		androi	d:theme="@style/ThemeOverlay.AppCompat.Dark.ActionBar"/>	
actu	alidadhiston (lavout v				
	anuaumstory_layout.x		<android.s< td=""><td>upport.v4.widget.DrawerLayout</td></android.s<>	upport.v4.widget.DrawerLayout	
Dasio	class_layout.xml		xmlns:	android="http://schemas.android.com/apk/res/android"	
o hom	e1_layout.xml		<pre>xmlns:app="http://schemas.android.com/apk/res-auto"</pre>		
🙍 inty_	layout.xml		android:layout_height="match_parent"		
🙋 item	_lista_categorias.xml		androi	d:layout_width="match_parent"	
🔯 item	_lista_categorias1.xml		androi	1:1d="6+1d/drawerLayout">	
🔯 item	_todo.xml		Frame	arout	
🔯 kapa	k_layout.xml		an	droid:orientation="wertical"	
kich	waecua layout.xml		an	droid:layout width="match parent"	
o killa	lavout xml		an	droid:layout height="match parent"	
o mail	lavout xml		an	droid:id="0+id/containerView">	
	ar lavout vml				
pawi	kar_iayout.xmi		<td>eLayout></td>	eLayout>	
😐 quec	inua_iayout.xml				
🙋 taba	ncestral_layout.xml		<andro< td=""><td>id.support.design.widget.NavigationView</td></andro<>	id.support.design.widget.NavigationView	
💆 tabb	asicclass_layout.xml		xm	<pre>lns:android="http://schemas.android.com/apk/res/android"</pre>	
💁 tabc	ategory_layout.xml		xm	<pre>lns:app="http://schemas.android.com/apk/res-auto"</pre>	
🔯 tabh	tabhistory_layout.xml android:layout_width="wrap_content"			droid:layout_width="wrap_content"	
🕨 💼 menu			an	droid:layout_height="match_parent"	
🔻 🛅 mipmap	-hdpi		an	aroid:iayout gravity="start"	
		Te	ext Design		

Fichero /app/src/main/AndroidManifest.xml

Contiene la definición en XML de muchos de los aspectos principales de la aplicación, como por ejemplo su identificación (nombre, icono, ...), sus componentes (pantallas, servicios, ...), o los permisos necesarios para su ejecución. Veremos más adelante más detalles de este fichero.



Fichero /app/build.gradle

Contiene información necesaria para la compilación del proyecto, por ejemplo la versión del SDK de Android utilizada para compilar, la mínima versión de Android que soportará la aplicación, referencias a las librerías externas utilizadas, etc. Más adelante veremos también más detalles de este fichero.



En un proyecto pueden existir varios ficheros *build.gradle*, para definir determinados parámetros a distintos niveles. Por ejemplo, en nuestro proyecto podemos ver que existe un fichero*build.gradle* a nivel de proyecto, y otro a nivel de módulo dentro de la carpeta /app. El primero de ellos definirá parámetros globales a todos los módulos del proyecto, y el segundo sólo tendrá efecto para cada módulo en particular.

Carpeta /app/build/

Contiene una serie de elementos de código generados automáticamente al compilar el proyecto. Cada vez que compilamos nuestro proyecto, la maquinaria de compilación de Android genera por nosotros una serie de ficheros fuente java dirigidos, entre otras muchas cosas, al control de los recursos de la aplicación. **Importante**: dado que estos ficheros se generan automáticamente tras cada compilación del proyecto es importante que no se modifiquen manualmente bajo ninguna circunstancia.



A destacar sobre todo el fichero que aparece desplegado en la imagen anterior, llamado "R.java", donde se define la clase R. Esta clase R contendrá en todo momento una serie de constantes con los identificadores (ID) de todos los recursos de la aplicación incluidos en la carpeta /app/src/main/res/, de forma que podamos acceder fácilmente a estos recursos desde través de nuestro código java а dicho dato. Así, por ejemplo, la constante R.layout.activity_main contendrá el ID del layout "activity_main1.xml" contenido en la carpeta /app/src/main/res/layout/.

4.2. Paquetes y clases

El proyecto contiene 7 paquetes que contienen distintas clases JAVA que siguiendo el patrón MVC otorgan funcionalidad al aplicativo.



Prosiguiendo se explicarán cada uno de los paquetes a continuación:

 about.- posee las clases cuyos métodos consisten en el envío de un mail (sugerencia, contacto, etc.) al desarrollador desde la app, además de mostrar la interfaz gráfica de los desarrolladores de la aplicación.

v.	🖿 java
	🔻 🖻 cquilumbaqui.kichwalearn
	🔻 💼 about
	🕒 🚡 AboutFragment
	🕒 🚡 MailFragment

 basic_class.- posee las clases clasificadas con el patrón MVC, donde se implementa las "clases básicas" de aprendizaje del Kichwa. Ésta diseñada en un Tab y con Fragments.



 category_dictionary.- contiene las clases JAVA divididas de acuerdo al patrón MVC, donde se implementa la funcionalidad del diccionario Kichwa – inglés con la utilización de la base de datos sqlite.



 celebrations.- posee las clases donde se implementa las interfaces gráficas informativas de las cuatro celebraciones ancestrales del pueblo Kichwa Otavalo. Se utiliza para el diseño Tabs y Fragments.



 game.- posee las clases donde se implementa el popular juego "el ahorcado", dividido en dos capas: controlador y la vista. En el paquete de controlador se implementa el teclado portátil y como aparecen las palabras kichwa de manera aleatoria dentro de una lista predefinida.



 history.- posee las clases donde se implementa las interfaces gráficas informativas divididas en tres etapas de la historia de la lengua Kichwa Otavalo. Se utiliza para el diseño Tabs y Fragments.



 View.- posee la clase java home y principal de todo el proyecto donde se implementa, los floating button action para los diferentes módulos de la app, y demás funcionalidades.



4.3. Vistas

Las pantallas o interfaces gráficas de los diferentes módulos de la app se encuentran dentro de la carpeta de **res/layout**, además dentro de la carpeta **res/values**, encontramos lo concerniente a valores de los colores, fuentes, dimensiones, etc. En la carpeta **res/drawable**, se encuentra las imágenes que se utilizaron en la app.

🔻 🗖 res	V 📴 res	
🔻 🗖 drawable	🕨 🛅 drawable	
📓 ahorcado0.png	▼ 🗖 layout	
📓 ahorcado1.png	about1_layout.xml	
📓 ahorcado2.png	activity_classes_details.xml	
📓 ahorcado3.png	activity_dictionary.xml	
📓 ahorcado4.png	💁 activity_dictionary_details.xml	
📓 ahorcado5.png	💁 activity_game_how_to.xml	
📓 ahorcado6.png	activity_game_principal.xml	
📓 ahorcado7.png	activity_game_splayer.xml	drawable
ahorcadowin.png	activity_list_category.xml	
ahrcd.png	activity_main.xml	v in menu
📓 and_you4.png	activity_main1.xml	drawermenu.xml
arrow.png	actualidadhistory_layout.xml	🔯 menu_main.xml
background_label_fab.xml	basicclass_layout.xml	mipmap-hdpi
in.png	home1_layout.xml	🕨 🛅 mipmap-mdpi
compose.png	inty_layout.xml	🕨 🖻 mipmap-xhdpi
i composea.png	🔯 item_lista_categorias.xml	🔻 🖻 mipmap-xxhdpi
draft.png	🔯 item_lista_categorias1.xml	📓 ic_launcher.png
goodafternoon4.png	🙆 item_todo.xml	🕨 💼 raw
goodmorning3.png	kapak_layout.xml	V 🗖 values
goodnight5.png	kichwaecua_layout.xml	🔯 colors.xml
hello2.png	🕺 killa_layout.xml	dimens.xml
help.png	🔯 mail_layout.xml	strings.xml
historia_kichwa1.PNG	pawkar_layout.xml	🔯 styles.xml
iii historia_kichwa2.PNG	🙆 quechua_layout.xml	values-w820dp
📓 historia_kichwa3.PNG	🔯 tabancestral_layout.xml	🙋 AndroidManifest.xml

5. GOOGLE PLAY

5.1. Introducción

Google Play Store (anteriormente Android Market) es una plataforma de distribución digital de aplicaciones móviles para los dispositivos con sistema operativo Android, así como una tienda en línea desarrollada y operada por Google. Esta plataforma permite a los usuarios navegar y descargar aplicaciones (desarrolladas mediante Android SDK), juegos, música, libros, revistas y películas. También se pueden adquirir dispositivos móviles como ordenadores Chromebook, teléfonos inteligentes Nexus, Google Chromecast, entre otros.1

Las aplicaciones se encuentran disponibles de forma gratuita, así como también con costo. Pueden ser descargadas directamente desde un dispositivo con Android a través de la aplicación móvil Play Store.

5.2. Publicación de una app

Lo primero que se debe realizar es convertir la cuenta de Google en una cuenta de desarrolladores. Para ello, se accede a Google Play Developer Console por primera vez en el siguiente link: <u>https://play.google.com/apps/publish/signup/</u>



Click en pago, y llenar el formulario de acuerdo a los requerimientos de facturación y compra online.



Llenar el formulario con los datos que se mostrará en las aplicaciones a subir a Google Play.

Accede con tu cuenta de Google.	Aceptación del Acuerdo pagar tarifa de registro Completa los datos de tu para desarrolladores Completa los datos de tu cuenta		
YA CASI TERMINAS			
Solo tienes que completar la información que	e se indica a continuación. Si quieres, puedes modificar estos datos más tarde en la configuración de la cu		
PERFIL DEL PROGRAMADOR	Tienes que com		
Nombre del programador *	CARLOS EDISON QUILUMBAQUI SANTACRUZ		
	35 de 50 caracteres Los usuarios verán el nombre del programador debajo del nombre de tu aplicación.		
Dirección de correo electrónico *	karlosq2004@hotmail.com		
Sitio web	1		
Número de teléfono *	+593-99-1745482		
	Incluye el signo más, el código de país y el código de área. Por ejemplo: +1-800-555-0199 ¿Por qué te solicitamos tu número de teléfono?		
Preferencias de correo electrónico	✓ Me gustaría obtener anuncios sobre nuevas funciones y sugerencias para mejorar mis apps.		
	Me gustaría enviar comentarios para meiorar Google Play Developer Console		

Aquí se debe realizar el pago de \$25,00 dólares, por única vez, con lo que ya se podrá ser un desarrollador que distribuye apps en Google Play.

Una vez realizado el pago, podemos entrar en Google Play Developer Console y nos mostrará el centro de gestión e información como desarrolladores, en el cual se puede observar lo siguiente:

- Listado de nuestras aplicaciones.
- Servicios para Google Play Games.
- Informes de nuestros beneficios.
- Configuración
- Anuncios
- Alertas



Para agregar una aplicación Android nos vamos al botón "Publicar una aplicación para Android en Google Play", y luego en agregar nueva aplicación.

#	TODAS LAS APLICACIONES	+ Agregar nueva aplicación
P9		
8		
٥	Aún no tienes ninguna aplicación.	
	Agrega tu primera aplicación.	

Nos llevará a un cuadro de diálogo donde deberemos escoger el lenguaje por defecto de la aplicación y el título de la misma.

AGREGAR NUEVA APLICACIÓN					
Idioma predeterminado	Idioma predeterminado *				
Español (Latinoamérica) – es-419 ▼					
Título *					
Learn about Kichwa					
18 de 30 caracteres					
¿Con qué te gustaría empezar?					
Cargar archivo APK Preparar entrada en Play Store Cancelar					

Una vez aquí, se abrirá toda la información acerca de la aplicación. Llenamos todo el formulario, y subimos el APK de la aplicación. Además se puede incluirlo en una fase de testeo alpha o beta, si nos conviene, o directamente a productivo, que lo que aparecerá en Google Play.

-		EL ANDROIDE LIBR	E		
АРК	0	APK			
Store Listing Pricing & Distribution In-app Products Services & APIs	0	PRODUCTION Publish your app on Googla Play	BETA TESTING Set up Beta testing for your app	ALPHA TESTING Set up Alpha testing for your app	
Optimization Tips	D	License keys a If your applicatio APK expansion f	re now managed for each application in n uses licensing services (e.g. if your app i lifes), get your new license key on the Serv	dividually. s a paid app, or if it uses in-app billing or icos & APIs page.	*Poros
				Upload your first	APK to Production

Una vez subido, veremos que en APK aparece un **tick verde**, de que esa parte está ya ok para publicar. Ahora pasamos a **Store Listing**, donde rellenaremos la descripción, texto de promoción, pantallazos para diferentes tamaños de pantalla, icono de la aplicación, la categoría de la aplicación, datos de contacto, política de privacidad propia, entre otras cosas.

fl-res icon *	Feature Graphic	Promo Graphic	
Default – English (United States) – en-US	Default – English (United States) – en-US	Default – English (United States) – en-US	
i12 x 512	1024 w x 500 h	180 w x 120 h	
i2-bit PNG (with alpha)	JPG or 24-bit PNG (no alpha)	JPG or 24-bit PNG (no alpha)	
+	+	+	
Add high-res icon	Add feature graphic	Add promo graphic	

ł

Phone		
+ Add screenshot		
7-inch tablet		
+ Add screenshot	Add at least one 7-inch screenshot here to help tablet users see how your app will look on their device.	
10-inch tablet		+ Poros
+ Add screenshot	Add at least one 10-inch screenshot here to help tablet users see how your app will look on their device.	

A continuación, pasamos a la sección **Pricing & Distribution**, donde elegiremos los países donde queremos que la aplicación esté disponible, así como si será gratis o de pago.

APK	0	PRICING & DISTRIBUTION Saved	
Store Listing	0	This application is	
Pricing & Distribution	0	1	To sublish said applications, you need to set up a merchant account. Set up a merchant
In app Products			account now or Learn more
Services & APIs		DISTRIBUTE IN THESE COUNTRIES	
Optimization Tips	10	You have not selected any countries	
		SELECT ALL COUNTRIES	
		🖾 Albania	
		🖾 Algeria	
		🖾 Angola	
		III Antigua and Barbuda	
		III Argentina	

Una vez rellenemos esta pestaña, ya estamos listos para publicar la aplicación, cambiando el estado Borrador (*Draft*). También tendremos una pestaña con **consejos para optimizar la información sobre nuestra app en Google Play**.