



UNIVERSIDAD TECNOLÓGICA DE PANAMÁ
FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES
NOMBRE DEL DEPARTAMENTO
GUÍA DE ACTIVIDAD N° 1



LABORATORIO

FC-FISC-1-8-2016

Facilitador(a): Juan Antonio Zamora Arosemena Asignatura: Herramientas de Programación Aplicada IV
Estudiante: Pablo Palacios 8-975-537, Roberto Bethancourt 8-978-783, David Fabbioni 8-927-258 Fecha: 10-4-23
Grupo: 1LS131

A. TÍTULO DE LA EXPERIENCIA: Introducción a la programación en Android

B. TEMAS:

- a. Estructura de un Proyecto Android
- b. Componentes de una Aplicación Android

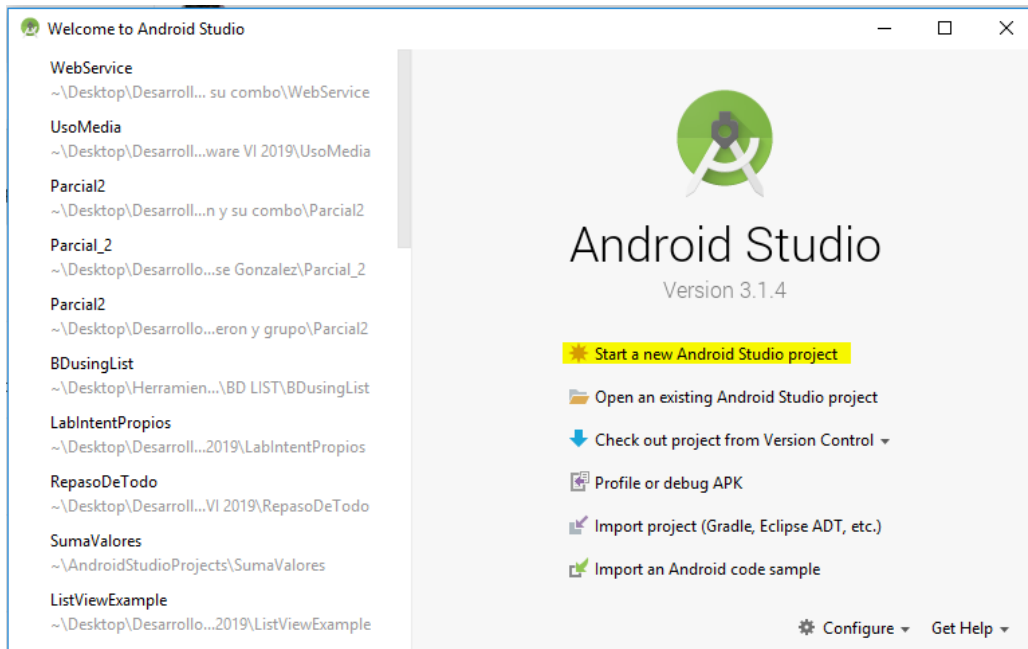
C. OBJETIVO(S): Familiarizarse con la creación de diferentes tipos de proyectos plantillas que nos presenta el Android studio

D. METODOLOGÍA: Siga las instrucciones dadas por el profesor o los pasos que contiene esta guía.

E. PROCEDIMIENTO O ENUNCIADO DE LA EXPERIENCIA:

Cree un nuevo proyecto en Android siguiendo:

- a. Al iniciar Android Studio, le aparecerá una pantalla similar a la siguiente donde debe elegir la opción de Crear Nuevo Proyecto Android Studio (Start a new Android Studio Project).



- b. En la ventana siguiente, deberá colocar el nombre de la aplicación y asegurar de que el proyecto se cree en una ubicación Física dentro de la PC (no crear el proyecto desde una memoria USB o externa), además debe prestar atención al contenido del recuadro Rojo (La dirección donde se guarda

el proyecto preferiblemente no debe tener espacios en blanco)

Create New Project

Create Android Project

Application name
Zapps

Company domain
jotaz.example.com

Project location
C:\Users\jotaz\Desktop\Desarrollo de Software VI 2019\Parcial2\Johan y su combo\Zapps

Package name
com.example.jotaz.zapps

☐ Include C++ support

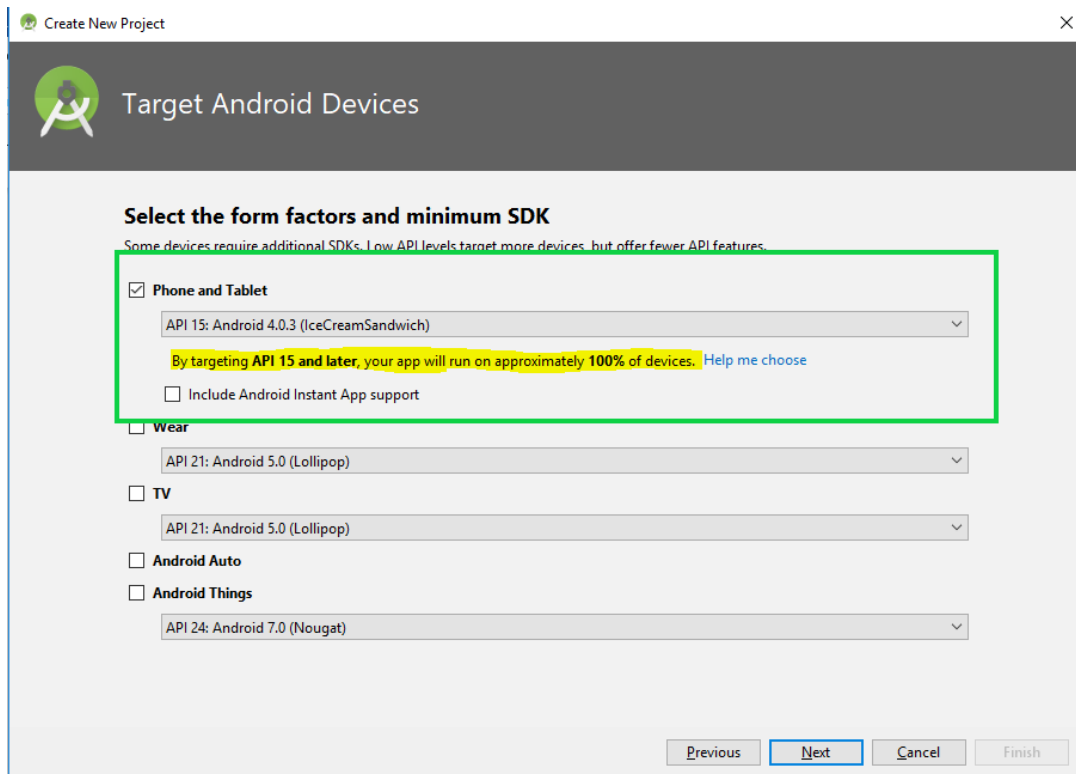
☐ Include Kotlin support

⚠ project location should not contain whitespace, as this can cause problems with the NDK tools.

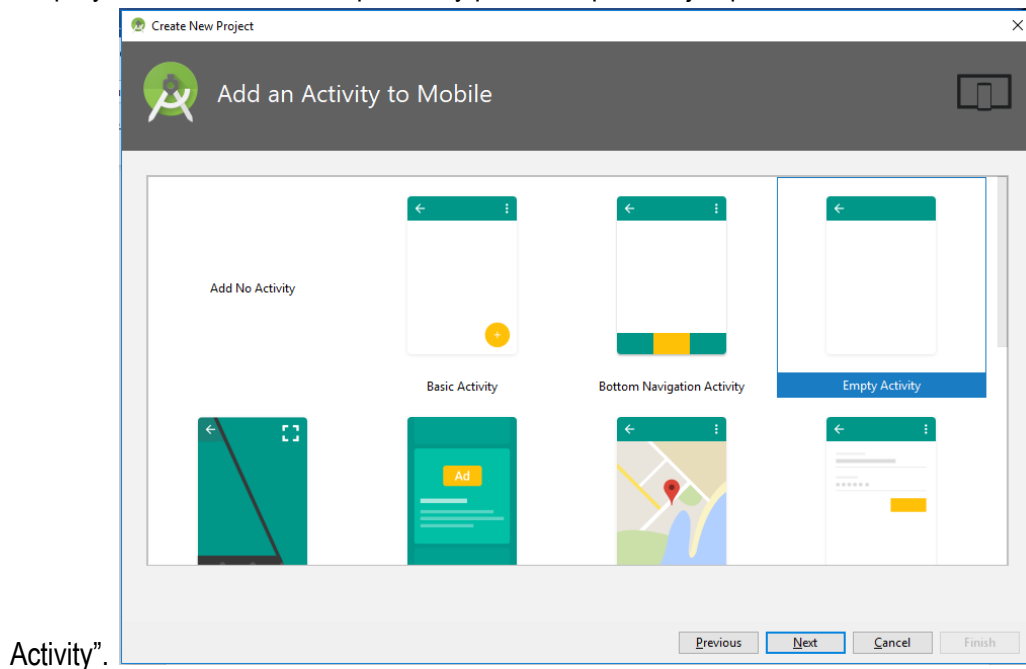
Previous Next Cancel Finish

- c. En la siguiente pantalla debemos seleccionar el API target para el cual deseamos desarrollar, aquí es muy importante realizar el hecho de que entre mas bajo seleccionemos el API, mas compatibilidad tendremos con el 100% de los dispositivos que existen en el mercado. Por defecto se selecciona el API 15. Tambien se puede mencionar que en esta pantalla podemos seleccionar el tipo de aplicación

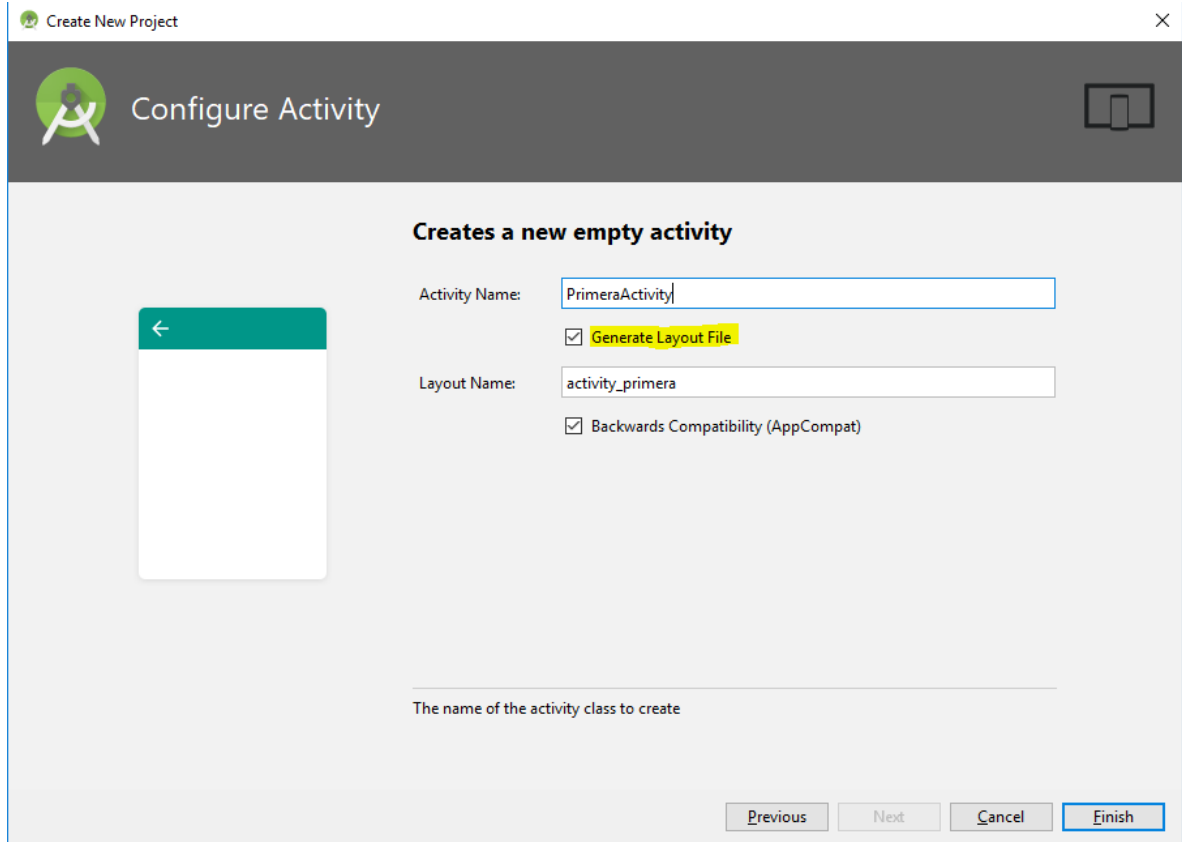
que queremos realizar (Aplicación móvil, para Android TV, Android Things y Android Wears).



- d. En esta pantalla debemos seleccionar la plantilla de Activities con la que queremos trabajar al inicio del proyecto. Existen varias opciones y para este primer ejemplo utilizaremos Actividad Vacía o “Empty Activity”.



- e. En la clase anterior, el profesor menciona que los archivos de Diseño (Interfaz Graficas) y Código fuente están separados, sin embargo, deben cumplir una regla de nomenclatura y llevar nombres similares. Colocaremos el nombre de PrimeraActivity (el nombre de los activities siempre debe llevar la palabra Activity) para el código fuente y activity_primera para el código de diseño. El check resaltado en la imagen, indica que queremos realizar el set automáticamente del diseño con el código fuente.



- f. Ya se ha creado nuestro primer proyecto en Android.

F. **RECURSOS:** Android Studio, Teléfono celular con Android (físico o virtual).

G. **RESULTADOS:** una vez realizado el código, responda las siguientes preguntas:

- Repita los pasos, pero en el punto F quite la selección del campo sombreado. ¿Indique que ha sucedido en el código? ¿Indique como lo pudo arreglar (comparar proyectos)?
- Repita los pasos, pero en el punto D seleccione otros tipos de plantillas (3 más), compare los códigos de ambos proyectos y muestre las diferencias, además de mencionar si se han creados archivos adicionales. (captura de pantalla de código con explicación y carpeta con archivos).

NOTA: De existir alguna diferencia entre la versión instalada y los pasos de esta guía, méncionelo y haga captura de pantalla explicando.

Plantilla de “Empty Activity”

La plantilla de actividad vacía nos da una estructura base simple con los archivos suficientes para generar un “Hola Mundo”



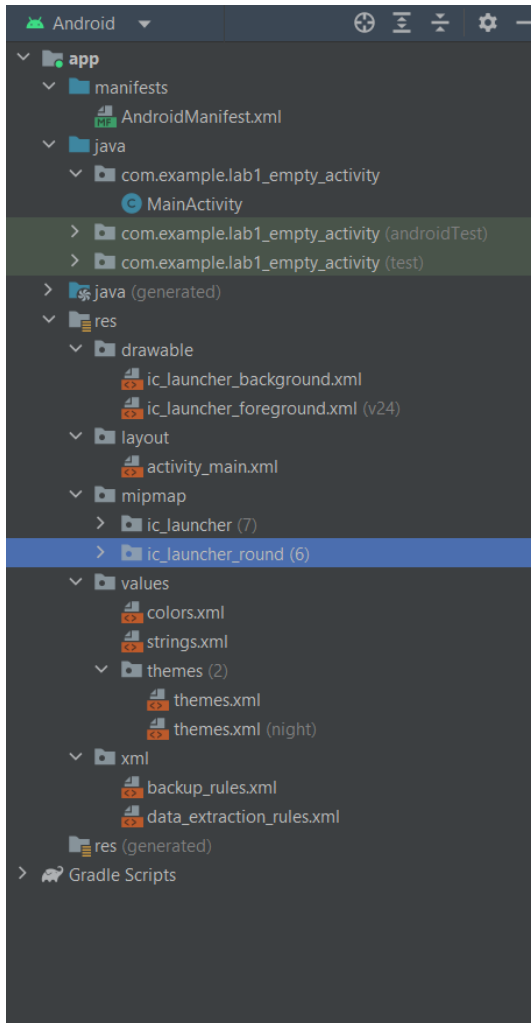
```
package com.example.lab1_empty_activity;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;

2 usages
public class MainActivity extends AppCompatActivity {

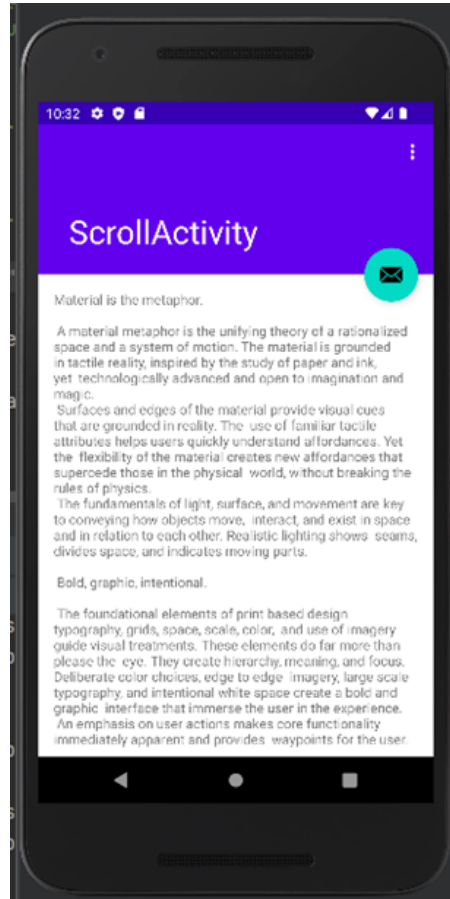
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Siendo este, el código principal, con una clase main que hereda de AppCompatActivity y el método super que crea los componentes. Además se importan 2 dependencias nada más a comparación del resto de pantallas de actividades que podemos generar



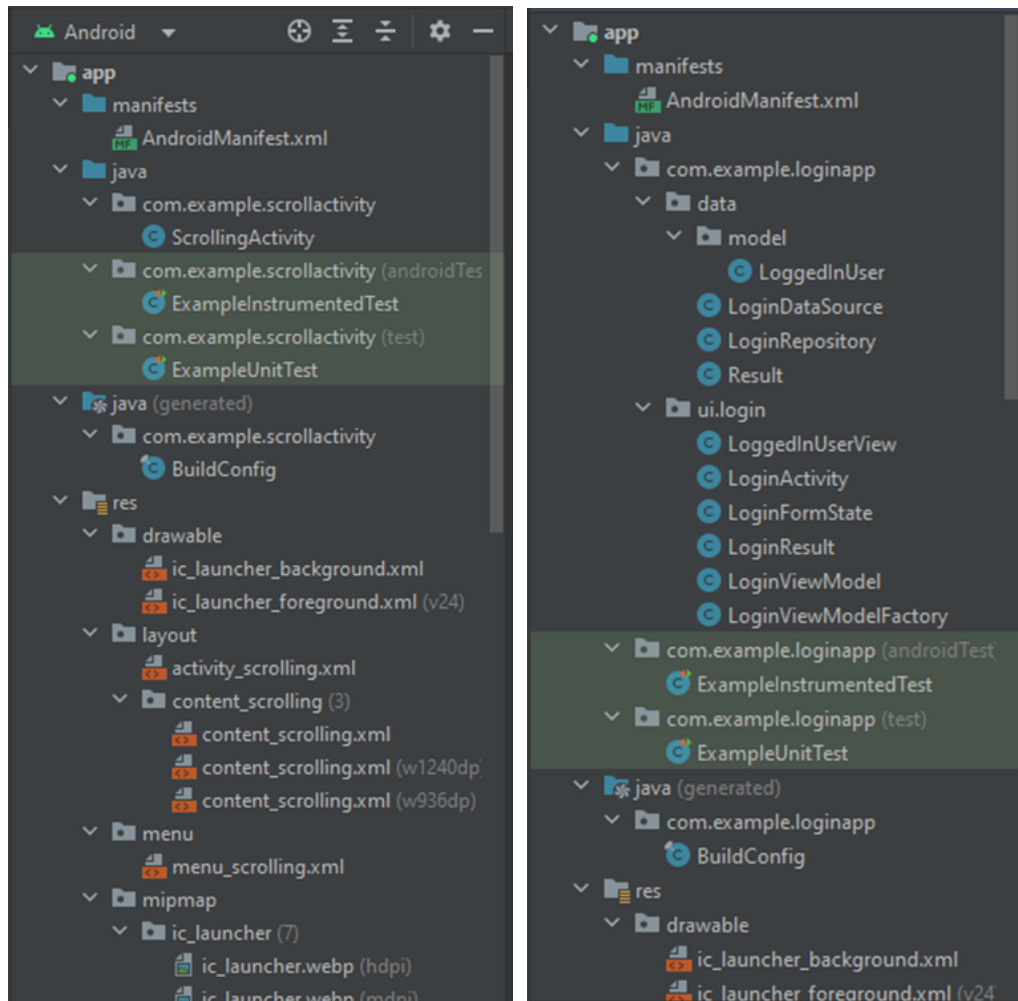
La estructura de archivos nos da la clase main, y una carpeta res con los componentes para colores, strings y el resto de elementos para la construcción del app

Plantilla de “Scroll Activity.”



Como ya se ha mencionado, la se estará mostrando la plantilla de Scroll y se mostrarán las diferencias entre la plantilla y la otras plantillas, en el caso de ScrollActivity, comparado con la plantilla de “Login”, esta tiene mucho menos carpetas “documentos” es decir que comparando las carpetas de “Login” esta tiene 20 carpetas en total y la otra plantilla 22 mayormente son aumentado esta cifra por las conocidas “Subcarpetas”.

A la hora de comparar estás plantillas puede ser un poco difícil ya que cada plantilla tiene su función final o mejor dicho tiene una finalidad para cada cosa, es decir que no es lo mismo la plantilla de login, como la de FullScreen o la scroll, pero se puede utilizar para adjuntarse y funcionar de manera de en conjunto.



Es una pequeña comparación entre la plantilla de “Login” y la plantilla de “Scroll” entre la cantidad de archivos que tiene uno del otro, ya que uno puede tener más archivos que el otro.

En el archivo de “Android Manifest” se puede notar una pequeña diferencia entre los códigos de uno del otro, ya que en uno tiene escrito en el Código “*android:theme="@style/Theme.ScrollActivity.NoActionBar"*”

```
android:theme="@style/Theme.ScrollActivity.NoActionBar">
```

Más que nada se utiliza para la emulación para remover la ActionBar

librerías de la plantilla de Scroll

```
package com.example.scrollactivity;

import android.os.Bundle;

import com.google.android.material.appbar.CollapsingToolbarLayout;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.snackbar.Snackbar;

import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;

import android.view.View;
import android.view.Menu;
import android.view.MenuItem;

import com.example.scrollactivity.databinding.ActivityScrollingBinding;
```

Librería del login

```
package com.example.loginapp.ui.login;

import android.app.Activity;

import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProvider;

import android.os.Bundle;

import androidx.annotation.Nullable;
import androidx.annotation.StringRes;
import androidx.appcompat.app.AppCompatActivity;

import android.text.Editable;
import android.text.TextWatcher;
import android.view.KeyEvent;
import android.view.View;
import android.view.inputmethod.EditorInfo;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import com.example.loginapp.R;
import com.example.loginapp.ui.login.LoginViewModel;
import com.example.loginapp.ui.login.LoginViewModelFactory;
import com.example.loginapp.databinding.ActivityLoginBinding;
```

Hay diferencias entre cada programa, ya sea la de full Screen y la de “La de pantalla completa”

Como se sabe que java o que es son las librerías que pueden facilitar las operaciones y tareas para la persona que esté por llamar, además se muestra como una se está utilizando las librerías de texto o de botón.

```
5 usages
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_scrolling, menu);
    return true;
}
```

Una clase Booleana que se utiliza para poder modificar los botones de la opción de configuración en la sección de arriba del menú de inicio.

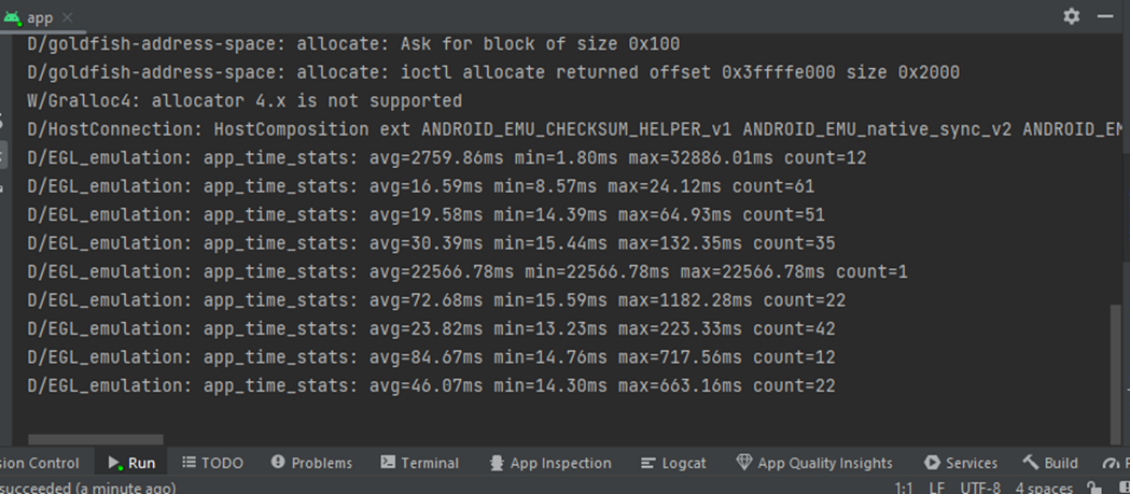
```
BuildConfig.java x ScrollingActivity.java x
Files under the "build" folder are generated and should not be edited.
1  /.../
4  package com.example.scrollactivity;
5
6  public final class BuildConfig {
7      public static final boolean DEBUG = Boolean.parseBoolean("true");
8      public static final String APPLICATION_ID = "com.example.scrollactivity";
9      public static final String BUILD_TYPE = "debug";
10     public static final int VERSION_CODE = 1;
11     public static final String VERSION_NAME = "1.0";
12 }
13
```

Esta es para los distintos estados estáticos que en sí no se pueden modificar después de obtener un dato predeterminado

Por ejemplo:

```
public static final String VERSION_NAME = "1.0";
```

Este es un dato que no se podrá editar, ya que está mostrando el nombre de la versión en este caso “1.0”



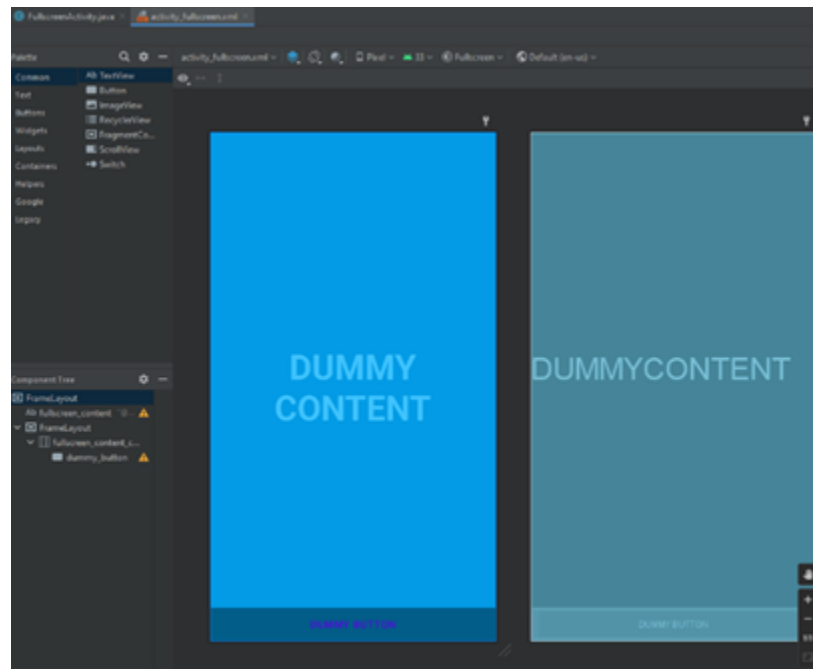
The screenshot shows the terminal window of an Android Studio interface. The terminal displays a series of log messages from an application running on an emulator. The messages include allocation requests, warnings about unsupported allocators, and performance statistics for EGL emulation. The bottom of the image shows the Android Studio toolbar with the 'Run' button highlighted, and a status bar indicating the build was successful.

```
app x
D/goldfish-address-space: allocate: Ask for block of size 0x100
D/goldfish-address-space: allocate: ioctl allocate returned offset 0x3ffffe000 size 0x2000
W/Gralloc4: allocator 4.x is not supported
D/HostConnection: HostComposition ext ANDROID_EMU_CHECKSUM_HELPER_v1 ANDROID_EMU_native_sync_v2 ANDROID_EMU
D/EGL_emulation: app_time_stats: avg=2759.86ms min=1.80ms max=32886.01ms count=12
D/EGL_emulation: app_time_stats: avg=16.59ms min=8.57ms max=24.12ms count=61
D/EGL_emulation: app_time_stats: avg=19.58ms min=14.39ms max=64.93ms count=51
D/EGL_emulation: app_time_stats: avg=30.39ms min=15.44ms max=132.35ms count=35
D/EGL_emulation: app_time_stats: avg=22566.78ms min=22566.78ms max=22566.78ms count=1
D/EGL_emulation: app_time_stats: avg=72.68ms min=15.59ms max=1182.28ms count=22
D/EGL_emulation: app_time_stats: avg=23.82ms min=13.23ms max=223.33ms count=42
D/EGL_emulation: app_time_stats: avg=84.67ms min=14.76ms max=717.56ms count=12
D/EGL_emulation: app_time_stats: avg=46.07ms min=14.30ms max=663.16ms count=22

Run
succeeded (a minute ago)
1:1 | LF | UTF-8 | 4 spaces
```

Imagen final de la consola a la hora de correr en el emulador

Fullscreen Activity



La plantilla de fullscreen (fullscreen activity) nos provee los archivos necesarios para facilitar el control de la visibilidad de las interfaces del dispositivo. Hay una gran cantidad de diferencias al compararlo con el empty activity, por ejemplo, hay muchísimo más código java y xml dentro de layout, attrs y styles.

```

1 package com.example.myapplication;
2
3 import androidx.appcompat.app.AppCompatActivity;
4
5 /**
6  * An example full-screen activity that shows and hides the system UI (i.e.
7  * status bar and navigation/system bar) with user interaction.
8  */
9
10 @SuppressWarnings("deprecation")
11 public class FullscreenActivity extends AppCompatActivity {
12
13     /**
14      * Whether or not the system UI should be auto-hidden after
15      * {@link #AUTO\_HIDE\_DELAY\_MILLIS} milliseconds.
16      */
17     @SuppressWarnings("deprecation")
18     private static final boolean AUTO_HIDE = true;
19
20     /**
21      * If {@link #AUTO\_HIDE} is set, the number of milliseconds to wait after
22      * user interaction before hiding the system UI.
23      */
24     @SuppressWarnings("deprecation")
25     private static final int AUTO_HIDE_DELAY_MILLIS = 3000;
26
27     /**
28      * Some older devices needs a small delay between UI widget updates
29      * and a change of the status and navigation bar.
30      */
31     @SuppressWarnings("deprecation")
32     private static final int UI_ANIMATION_DELAY = 300;
33
34     private final Handler mHideHandler = new Handler(Looper.myLooper());
35
36     private View mContentView;
37
38     @Override
39     protected void onCreate(Bundle savedInstanceState) {
40         super.onCreate(savedInstanceState);
41         setContentView(R.layout.activity_main);
42
43         // Hide the system UI.
44         hideSystemUI();
45
46         // Hide the system UI.
47         hideSystemUI();
48
49         // Hide the system UI.
50         hideSystemUI();
51
52         // Hide the system UI.
53         hideSystemUI();
54
55         // Hide the system UI.
56         hideSystemUI();
57
58         // Hide the system UI.
59         hideSystemUI();
60
61         // Hide the system UI.
62         hideSystemUI();
63
64         // Hide the system UI.
65         hideSystemUI();
66
67         // Hide the system UI.
68         hideSystemUI();
69
70         // Hide the system UI.
71         hideSystemUI();
72
73         // Hide the system UI.
74         hideSystemUI();
75
76         // Hide the system UI.
77         hideSystemUI();
78
79         // Hide the system UI.
80         hideSystemUI();
81
82         // Hide the system UI.
83         hideSystemUI();
84
85         // Hide the system UI.
86         hideSystemUI();
87
88         // Hide the system UI.
89         hideSystemUI();
90
91         // Hide the system UI.
92         hideSystemUI();
93
94         // Hide the system UI.
95         hideSystemUI();
96
97         // Hide the system UI.
98         hideSystemUI();
99
100        // Hide the system UI.
101        hideSystemUI();
102
103        // Hide the system UI.
104        hideSystemUI();
105
106        // Hide the system UI.
107        hideSystemUI();
108
109        // Hide the system UI.
110        hideSystemUI();
111
112        // Hide the system UI.
113        hideSystemUI();
114
115        // Hide the system UI.
116        hideSystemUI();
117
118        // Hide the system UI.
119        hideSystemUI();
120
121        // Hide the system UI.
122        hideSystemUI();
123
124        // Hide the system UI.
125        hideSystemUI();
126
127        // Hide the system UI.
128        hideSystemUI();
129
130        // Hide the system UI.
131        hideSystemUI();
132
133        // Hide the system UI.
134        hideSystemUI();
135
136        // Hide the system UI.
137        hideSystemUI();
138
139        // Hide the system UI.
140        hideSystemUI();
141
142        // Hide the system UI.
143        hideSystemUI();
144
145        // Hide the system UI.
146        hideSystemUI();
147
148        // Hide the system UI.
149        hideSystemUI();
150
151        // Hide the system UI.
152        hideSystemUI();
153
154        // Hide the system UI.
155        hideSystemUI();
156
157        // Hide the system UI.
158        hideSystemUI();
159
160        // Hide the system UI.
161        hideSystemUI();
162
163        // Hide the system UI.
164        hideSystemUI();
165
166        // Hide the system UI.
167        hideSystemUI();
168
169        // Hide the system UI.
170        hideSystemUI();
171
172        // Hide the system UI.
173        hideSystemUI();
174
175        // Hide the system UI.
176        hideSystemUI();
177
178        // Hide the system UI.
179        hideSystemUI();
180
181        // Hide the system UI.
182        hideSystemUI();
183
184        // Hide the system UI.
185        hideSystemUI();
186
187        // Hide the system UI.
188        hideSystemUI();
189
190        // Hide the system UI.
191        hideSystemUI();
192
193        // Hide the system UI.
194        hideSystemUI();
195
196        // Hide the system UI.
197        hideSystemUI();
198
199        // Hide the system UI.
200        hideSystemUI();
201
202        // Hide the system UI.
203        hideSystemUI();
204
205        // Hide the system UI.
206        hideSystemUI();
207
208        // Hide the system UI.
209        hideSystemUI();
210
211        // Hide the system UI.
212        hideSystemUI();
213
214        // Hide the system UI.
215        hideSystemUI();
216
217        // Hide the system UI.
218        hideSystemUI();
219
220        // Hide the system UI.
221        hideSystemUI();
222
223        // Hide the system UI.
224        hideSystemUI();
225
226        // Hide the system UI.
227        hideSystemUI();
228
229        // Hide the system UI.
230        hideSystemUI();
231
232        // Hide the system UI.
233        hideSystemUI();
234
235        // Hide the system UI.
236        hideSystemUI();
237
238        // Hide the system UI.
239        hideSystemUI();
240
241        // Hide the system UI.
242        hideSystemUI();
243
244        // Hide the system UI.
245        hideSystemUI();
246
247        // Hide the system UI.
248        hideSystemUI();
249
250        // Hide the system UI.
251        hideSystemUI();
252
253        // Hide the system UI.
254        hideSystemUI();
255
256        // Hide the system UI.
257        hideSystemUI();
258
259        // Hide the system UI.
260        hideSystemUI();
261
262        // Hide the system UI.
263        hideSystemUI();
264
265        // Hide the system UI.
266        hideSystemUI();
267
268        // Hide the system UI.
269        hideSystemUI();
270
271        // Hide the system UI.
272        hideSystemUI();
273
274        // Hide the system UI.
275        hideSystemUI();
276
277        // Hide the system UI.
278        hideSystemUI();
279
280        // Hide the system UI.
281        hideSystemUI();
282
283        // Hide the system UI.
284        hideSystemUI();
285
286        // Hide the system UI.
287        hideSystemUI();
288
289        // Hide the system UI.
290        hideSystemUI();
291
292        // Hide the system UI.
293        hideSystemUI();
294
295        // Hide the system UI.
296        hideSystemUI();
297
298        // Hide the system UI.
299        hideSystemUI();
300
301        // Hide the system UI.
302        hideSystemUI();
303
304        // Hide the system UI.
305        hideSystemUI();
306
307        // Hide the system UI.
308        hideSystemUI();
309
310        // Hide the system UI.
311        hideSystemUI();
312
313        // Hide the system UI.
314        hideSystemUI();
315
316        // Hide the system UI.
317        hideSystemUI();
318
319        // Hide the system UI.
320        hideSystemUI();
321
322        // Hide the system UI.
323        hideSystemUI();
324
325        // Hide the system UI.
326        hideSystemUI();
327
328        // Hide the system UI.
329        hideSystemUI();
330
331        // Hide the system UI.
332        hideSystemUI();
333
334        // Hide the system UI.
335        hideSystemUI();
336
337        // Hide the system UI.
338        hideSystemUI();
339
340        // Hide the system UI.
341        hideSystemUI();
342
343        // Hide the system UI.
344        hideSystemUI();
345
346        // Hide the system UI.
347        hideSystemUI();
348
349        // Hide the system UI.
350        hideSystemUI();
351
352        // Hide the system UI.
353        hideSystemUI();
354
355        // Hide the system UI.
356        hideSystemUI();
357
358        // Hide the system UI.
359        hideSystemUI();
360
361        // Hide the system UI.
362        hideSystemUI();
363
364        // Hide the system UI.
365        hideSystemUI();
366
367        // Hide the system UI.
368        hideSystemUI();
369
370        // Hide the system UI.
371        hideSystemUI();
372
373        // Hide the system UI.
374        hideSystemUI();
375
376        // Hide the system UI.
377        hideSystemUI();
378
379        // Hide the system UI.
380        hideSystemUI();
381
382        // Hide the system UI.
383        hideSystemUI();
384
385        // Hide the system UI.
386        hideSystemUI();
387
388        // Hide the system UI.
389        hideSystemUI();
390
391        // Hide the system UI.
392        hideSystemUI();
393
394        // Hide the system UI.
395        hideSystemUI();
396
397        // Hide the system UI.
398        hideSystemUI();
399
400        // Hide the system UI.
401        hideSystemUI();
402
403        // Hide the system UI.
404        hideSystemUI();
405
406        // Hide the system UI.
407        hideSystemUI();
408
409        // Hide the system UI.
410        hideSystemUI();
411
412        // Hide the system UI.
413        hideSystemUI();
414
415        // Hide the system UI.
416        hideSystemUI();
417
418        // Hide the system UI.
419        hideSystemUI();
420
421        // Hide the system UI.
422        hideSystemUI();
423
424        // Hide the system UI.
425        hideSystemUI();
426
427        // Hide the system UI.
428        hideSystemUI();
429
430        // Hide the system UI.
431        hideSystemUI();
432
433        // Hide the system UI.
434        hideSystemUI();
435
436        // Hide the system UI.
437        hideSystemUI();
438
439        // Hide the system UI.
440        hideSystemUI();
441
442        // Hide the system UI.
443        hideSystemUI();
444
445        // Hide the system UI.
446        hideSystemUI();
447
448        // Hide the system UI.
449        hideSystemUI();
450
451        // Hide the system UI.
452        hideSystemUI();
453
454        // Hide the system UI.
455        hideSystemUI();
456
457        // Hide the system UI.
458        hideSystemUI();
459
460        // Hide the system UI.
461        hideSystemUI();
462
463        // Hide the system UI.
464        hideSystemUI();
465
466        // Hide the system UI.
467        hideSystemUI();
468
469        // Hide the system UI.
470        hideSystemUI();
471
472        // Hide the system UI.
473        hideSystemUI();
474
475        // Hide the system UI.
476        hideSystemUI();
477
478        // Hide the system UI.
479        hideSystemUI();
480
481        // Hide the system UI.
482        hideSystemUI();
483
484        // Hide the system UI.
485        hideSystemUI();
486
487        // Hide the system UI.
488        hideSystemUI();
489
490        // Hide the system UI.
491        hideSystemUI();
492
493        // Hide the system UI.
494        hideSystemUI();
495
496        // Hide the system UI.
497        hideSystemUI();
498
499        // Hide the system UI.
500        hideSystemUI();
501
502        // Hide the system UI.
503        hideSystemUI();
504
505        // Hide the system UI.
506        hideSystemUI();
507
508        // Hide the system UI.
509        hideSystemUI();
510
511        // Hide the system UI.
512        hideSystem
```

El código generado por la plantilla incluye comentarios que explican el propósito de las variables constantes.

Por defecto la plantilla está configurada para automáticamente ocultar el interfaz del sistema después de 3000 milisegundos.

También toma en cuenta el delay necesario para dispositivos más viejos (300ms)

```

102 private final Runnable mHidePart2Runnable = new Runnable() {
103     @SuppressWarnings("InlinedApi")
104     @Override
105     public void run() {
106         // Delayed removal of status and navigation bar
107         if (Build.VERSION.SDK_INT >= 30) {
108             mContentView.getWindowInsetsController().hide(
109                 WindowInsets.Type.statusBars() | WindowInsets.Type.navigationBars());
110         } else {
111             // Note that some of these constants are new as of API 16 (Jelly Bean)
112             // and API 19 (KitKat). It is safe to use them, as they are inlined
113             // at compile-time and do nothing on earlier devices.
114             mContentView.setSystemUiVisibility(View.SYSTEM_UI_FLAG_LOW_PROFILE
115                 | View.SYSTEM_UI_FLAG_FULLSCREEN
116                 | View.SYSTEM_UI_FLAG_LAYOUT_STABLE
117                 | View.SYSTEM_UI_FLAG_IMMERSIVE_STICKY
118                 | View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION
119                 | View.SYSTEM_UI_FLAG_HIDE_NAVIGATION);
120         }
121     }
122 };
123 // 3 usages
124 private View mControlsView;
125 // 2 usages
126 private final Runnable mShowPart2Runnable = new Runnable() {
127     @Override
128     public void run() {
129         // Delayed display of UI elements
130         ActionBar actionBar = getSupportActionBar();
131         if (actionBar != null) {
132             actionBar.show();
133         }
134         mControlsView.setVisibility(View.VISIBLE);
135     }
136 };
137 // 4 usages
138 private boolean mVisible;
139 private final Runnable mHideRunnable = new Runnable() {
140     @Override
141     public void run() { hide(); }
142 };
143 /**
144  * Touch listener to use for in-layout UI controls to delay hiding the
145  * system UI. This is to prevent the jarring behavior of controls going away
146  * while interacting with activity UI.
147  */
148 // 1 usage
149 private final View.OnTouchListener mDelayHideTouchListener = new View.OnTouchListener() {
150     @Override
151     public boolean onTouch(View view, MotionEvent motionEvent) {
152         switch (motionEvent.getAction()) {
153             case MotionEvent.ACTION_DOWN:
154                 if (AUTO_HIDE) {
155                     delayedHide(AUTO_HIDE_DELAY_MILLIS);
156                 }
157                 break;
158             case MotionEvent.ACTION_UP:
159                 view.performClick();
160                 break;
161             default:
162                 break;
163         }
164         return false;
165     }
166 };
167 // 5 usages
168 private ActivityFullscreenBinding binding;

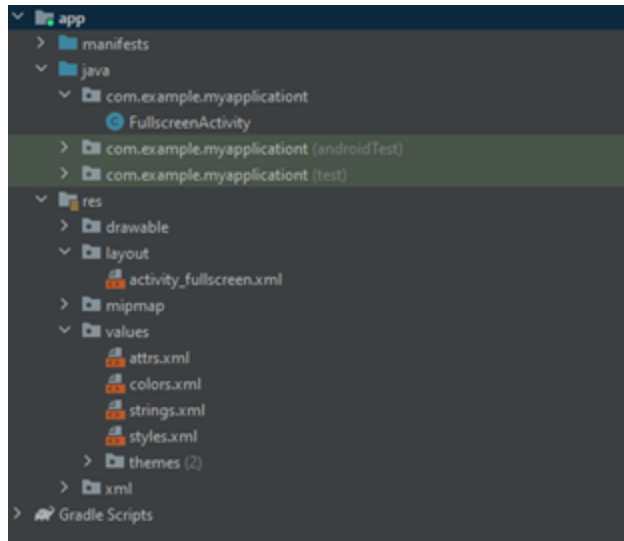
```

```

169 @Override
170 protected void onCreate(Bundle savedInstanceState) {
171     super.onCreate(savedInstanceState);
172
173     binding = ActivityFullscreenBinding.inflate(getLayoutInflater());
174     setContentView(binding.getRoot());
175
176     mVisible = true;
177     mControlsView = binding.fullscreenContentControls;
178     mContentView = binding.fullscreenContent;
179
180     // Set up the user interaction to manually show or hide the system UI.
181     mContentView.setOnClickListener(new View.OnClickListener() {
182         @Override
183         public void onClick(View view) { toggle(); }
184     });
185
186     // Upon interacting with UI controls, delay any scheduled hide()
187     // operations to prevent the jarring behavior of controls going away
188     // while interacting with the UI.
189     binding.dummyButton.setOnTouchListener(mDelayHideTouchListener);
190 }
191 // 5 usages
192 @Override
193 protected void onStart(Bundle savedInstanceState) {
194     super.onStart(savedInstanceState);
195
196     // Trigger the initial hide() shortly after the activity has been
197     // created, to briefly hint to the user that UI controls
198     // are available.
199     delayedHide( delayMillis: 100);
200 }
201 private void toggle() {
202     if (mVisible) {
203         hide();
204     } else {
205         show();
206     }
207 }
208 // 2 usages
209 private void hide() {
210     // Hide UI first
211     ActionBar actionBar = getSupportActionBar();
212     if (actionBar != null) {
213         actionBar.hide();
214     }
215     mControlsView.setVisibility(View.GONE);
216     mVisible = false;
217
218     // Schedule a runnable to remove the status and navigation bar after a delay
219     mHideHandler.removeCallbacks(mHidePart2Runnable);
220     mHideHandler.postDelayed(mHidePart2Runnable, UI_ANIMATION_DELAY);
221 }
222 private void show() {
223     // Show the system bar
224     if (Build.VERSION.SDK_INT >= 30) {
225         mContentView.getWindowInsetsController().show(
226             WindowInsets.Type.statusBars() | WindowInsets.Type.navigationBars());
227     } else {
228         mContentView.setSystemUiVisibility(View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN
229             | View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION);
230     }
231     mVisible = true;
232
233     // Schedule a runnable to display UI elements after a delay
234     mHideHandler.removeCallbacks(mHidePart2Runnable);
235     mHideHandler.postDelayed(mShowPart2Runnable, UI_ANIMATION_DELAY);
236 }
237 /**
238  * Schedules a call to hide() in delay milliseconds, canceling any
239  * previously scheduled calls.
240  */
241 // 2 usages
242 private void delayedHide(int delayMillis) {
243     mHideHandler.removeCallbacks(mHideRunnable);
244     mHideHandler.postDelayed(mHideRunnable, delayMillis);
245 }
246 }

```

Aquí tenemos las funciones que controlan el comportamiento de los controles del interfaz (delay al mostrar, delay para ocultar dependiendo en las acciones del usuario).



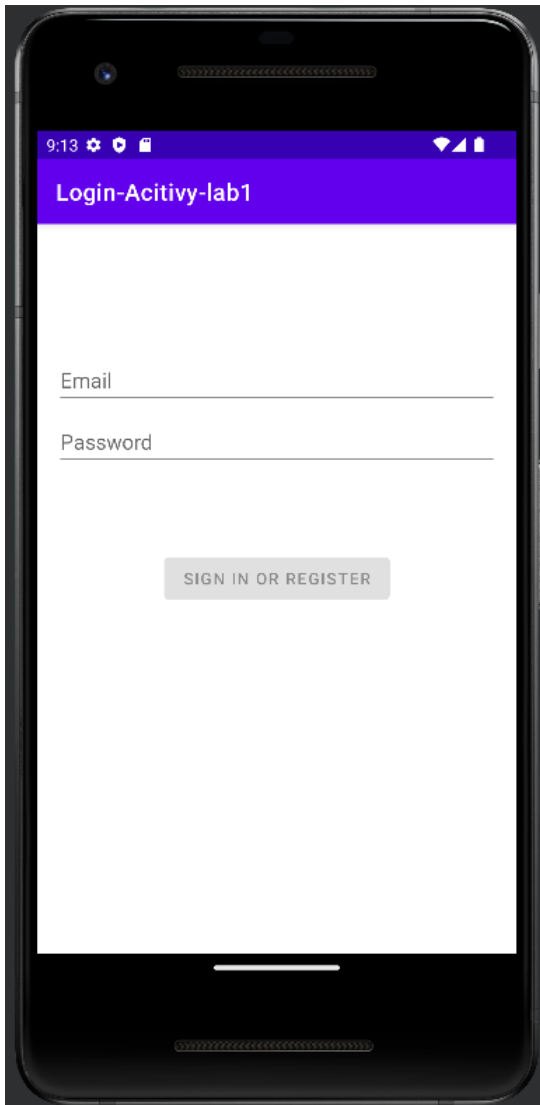
Los archivos nuevos son:

activity_fullscreen (vista previa del UI)

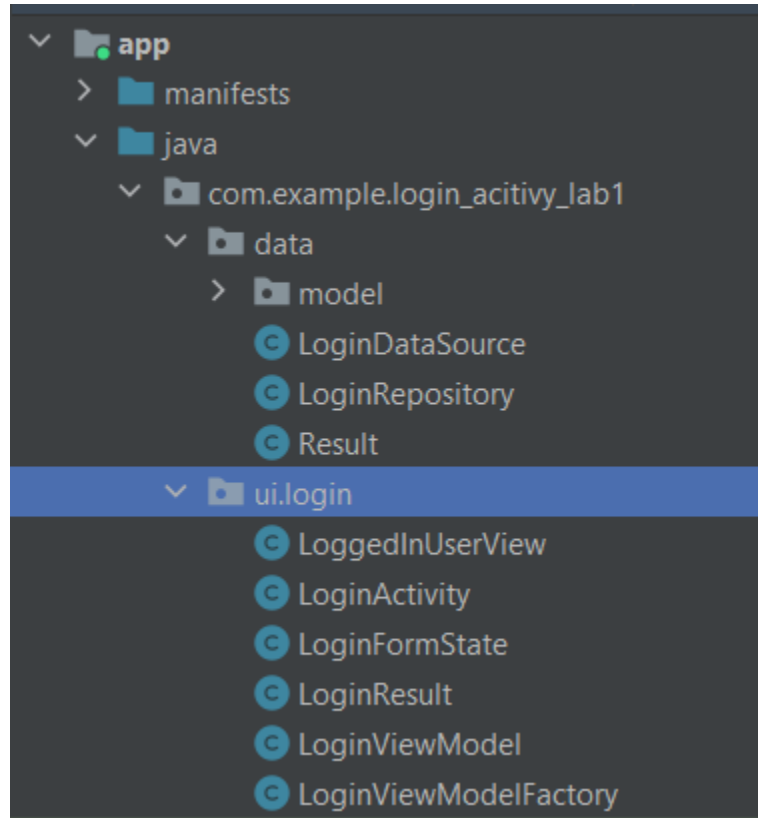
attrs (background y text color)

styles (colores del actionbar y buttonbar)

Login Activity



En la pantalla de login, tenemos a diferencia de las otras, un modelo simple de inicio de sesión



Tenemos aquí que se generan 3 clases que manejan los datos que introduce el usuario siendo LoginDataSource, aquí se autenticar las credenciales del usuario

A diferencia de otras pantallas, en el login tenemos 3 clases que manejan por separado la autenticación, la persistencia de los datos y el resultado de esta persistencia.

También tenemos que toda la interfaz de usuario está segmentada en varias clases y en otra carpeta, siendo ui.login.

```

package com.example.login_activiy_lab1.data;

import com.example.login_activiy_lab1.data.model.LoggedInUser;

import java.io.IOException;

/**
 * Class that handles authentication w/ login credentials and retrieves user information.
 */
5 usages
public class LoginDataSource {

    public Result<LoggedInUser> login(String username, String password) {

        try {
            // TODO: handle loggedInUser authentication
            LoggedInUser fakeUser =
                new LoggedInUser(
                    java.util.UUID.randomUUID().toString(),
                    displayName: "Jane Doe");
            return new Result.Success<>(fakeUser);
        } catch (Exception e) {
            return new Result.Error(new IOException("Error logging in", e));
        }
    }

    1 usage
    public void logout() {
        // TODO: revoke authentication
    }
}

```

Aquí podemos ver cómo se maneja la autenticación y el control de la sesión, así como la posible excepción de fallar el inicio de sesión

Bitácoras de aprendizaje

- Tema: Introducción a la programación en Android
- Estructura de un Proyecto Android
- Componentes de una Aplicación Android

Reflexiones:

1. Roberto Bethancourt

- **Dificultades Encontradas:**

Una de las dificultades encontradas en el Lab ha sido poder instalar el Android Studio ya que el documento adquirido es antiguo respecto con la versión actual del ya dicho Android Studio y correr el sistema de emulación de dispositivos.

- **Solución Establecida:**

Utilizar las famosas soluciones de las páginas de StackOverflow y Youtube para poder instalar y solucionar el problema con la emulación de los dispositivos. ya que por ejemplo el problema de la emulación era por que no tenía activada la opción para emular en el procesador AMD.

- **Conocimiento Adquirido:**

El conocimiento adquirido sobre esta actividad ha sido utilizar brevemente el programa de Android Studio, ya que puede ser similar a lo que es Netbeans, VS, Intelly entre otros programas.

2. Pablo Palacios

- **Dificultades Encontradas:**

Identificar ciertos pasos con respecto a la nueva interfaz de Android Studio Electron y problemas con dependencias/clases duplicadas

- **Solución Establecida:**

Segui los pasos segun la guia intuitivamente y tuve que agregar una linea al gradle.properties

- **Conocimiento Adquirido:**

Importancia del manejo de dependencias y su correcto control

3. David Fabbroni

- **Dificultades Encontradas:**

No tuve dificultades, solo me confundí con un error relacionado a intel HAXM al instalar android studio y el emulador.

- **Solución Establecida:**

Intel HAXM ha sido descontinuado, no se instala.

- **Conocimiento Adquirido:**

Aprendí sobre las plantillas y el IDE (parecido a IntelliJ).