

Android

Android Studio

Kotlin Language – xml for design

Eng. Ibrahim AlDekhail

BCT - TVTC

Introduction

1. What is Kotlin?

Kotlin is a modern programming language developed by JetBrains. It is statically typed and runs on the Java Virtual Machine (JVM). The Kotlin language is designed to be interoperable with Java, so you can use Kotlin code alongside Java code seamlessly. Aside from Android app development, it is also used for server-side and web development.

2. How is Kotlin different from Java?

Kotlin and Java both programming languages that run on the Java Virtual Machine (JVM).

But Kotlin is designed to be more concise and expressive, reducing unnecessary code.

One notable feature is its built-in null safety, helping to avoid common programming errors related to null values.

Kotlin also introduces modern language features like extension functions and coroutines, offering developers more flexibility and improved productivity compared to Java.

Introduction

1. Java vs Kotlin: Difference Between Kotlin and Java

Java and Kotlin are two widely known programming languages used for developing Android applications. Both these languages are quite similar; however, while drawing the comparison between Java vs Kotlin, there are some of the key differences that you should consider. Let's have a look at them.

Comparison Factors	Java	Kotlin
Popularity	Widely Used	Steadily Growing
Ease of Use	More Concise	Mature Development Environment
Scalability	Yes	Yes
Community Support	Well-established Community	Still Expanding

Introduction

Java is Better For:

- Large applications that require multiple features and functionalities and need to function across all platforms, such as Windows, iOS, Android, or Linux.
- Java also has mature libraries that support this type of application development well and provide the integration of features with the latest updates.

Kotlin is Better For:

- Android app development with Kotlin is better suited for apps where performance matters a lot, such as those that are required to run smoothly on older Android phones or the ones that are used for photo editing.
- In Java vs Kotlin, Kotlin has a more streamlined and efficient design than Java, so it will perform better in these types of situations, particularly when scaling is concerned
- Kotlin is able to perform these functions while Java cannot because of its use of bytecode which can only compile code for one specific platform at once.

Key Features of Java

Platform Independence

Object-Oriented Programming

Security

Standard Library

Large and Active Community

Key Features of Kotlin

Conciseness and
Readability

Null Safety

Interoperability
with Java

Functional
Programming
Support

Coroutines

Q5: What is the difference between `var` and `val` in Kotlin?

Answer

- `var` is like `general` variable and it's known as a *mutable* variable in kotlin and can be assigned multiple times.
- `val` is like `Final` variable and it's known as *immutable* in Kotlin and can be initialized only single time.

	<code>val</code>	<code>var</code>
Reference type	Immutable(once initialized can't be reassigned)	Mutable(can able to change value)
Example	<code>val n = 20</code>	<code>var n = 20</code>
In Java	<code>final int n = 20;</code>	<code>int n = 20;</code>

Q6: Where should I use `var` and where `val` ?

Answer

Use `var` where value is changing *frequently*. For example while getting location of android device:

```
var integerValue : Int? = null
```

Use `val` where there is *no change* in value in whole class. For example you want set textview or button's text programmatically.

```
val stringVariables : String = "Button's Constant or final Text"
```

Q17: What is `lateinit` in Kotlin and when would you use it?

Answer

`lateinit` means *late initialization*. If you do not want to initialize a variable in the constructor instead you want to initialize it later on and if you can guarantee the initialization before using it, then declare that variable with `lateinit` keyword. It will not allocate memory until initialized. You cannot use `lateinit` for primitive type properties like `Int`, `Long` etc.


```
lateinit var test: String

fun doSomething() {
    test = "Some value"
    println("Length of string is "+test.length)
    test = "change value"
}
```

There are a handful of use cases where this is extremely helpful, for example:

- Android: variables that get initialized in lifecycle methods;
- Using Dagger for DI: injected class variables are initialized outside and independently from the constructor;
- Setup for unit tests: test environment variables are initialized in a `@Before` - annotated method;
- Spring Boot annotations (eg. `@Autowired`).

Google Play Console



الرئيسية

حالة السياسة

المستخدمون والأدوات

إدارة الطلبات

تنزيل التقارير

حساب المطور

حسابات المطورين المرتبطة

سجل الأنشطة

إعدادات

حالة السياسة

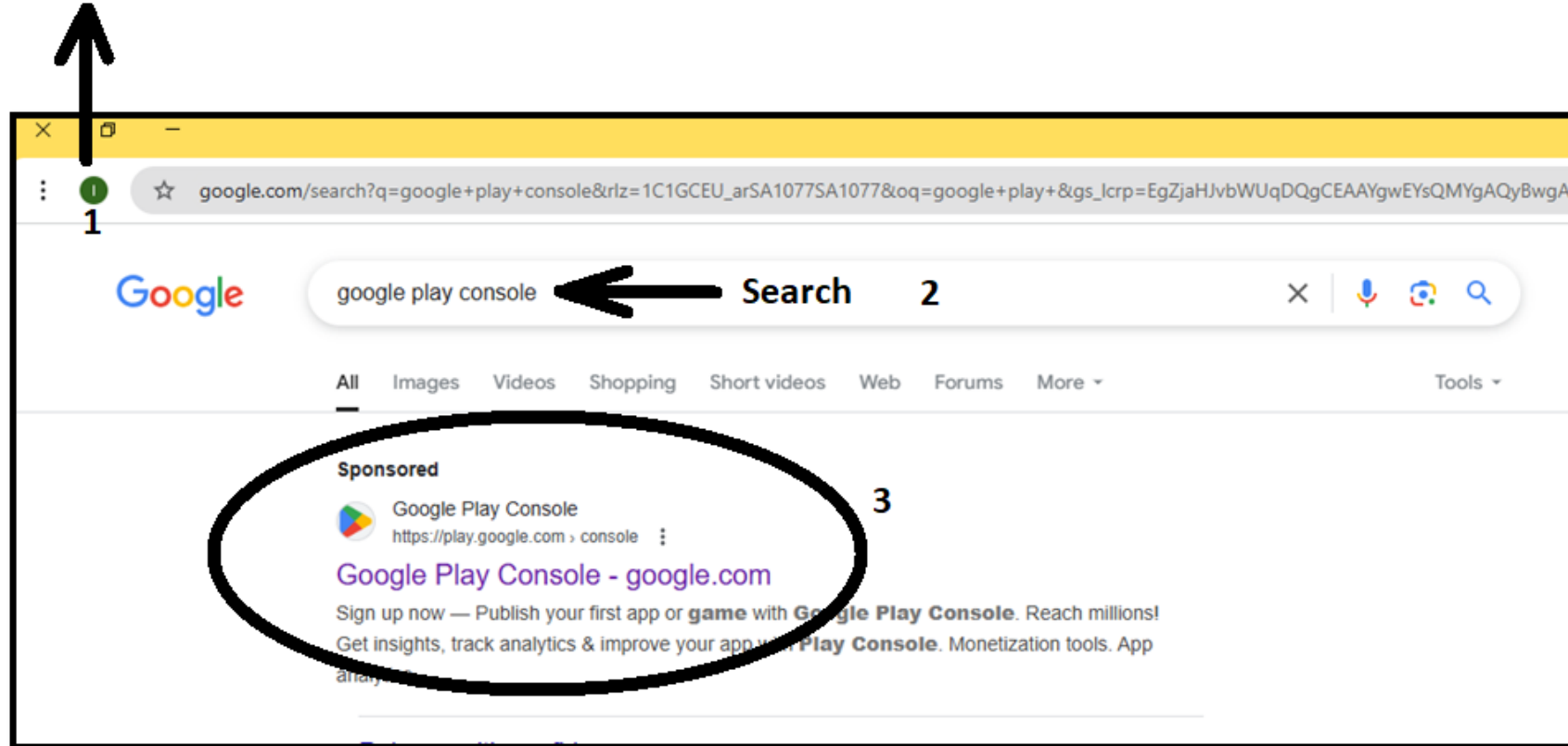
يمكنك إدارة مدى امتثال تطبيقك لسياسات Google Play للمطورين. [مزيد من المعلومات](#)

ملخص

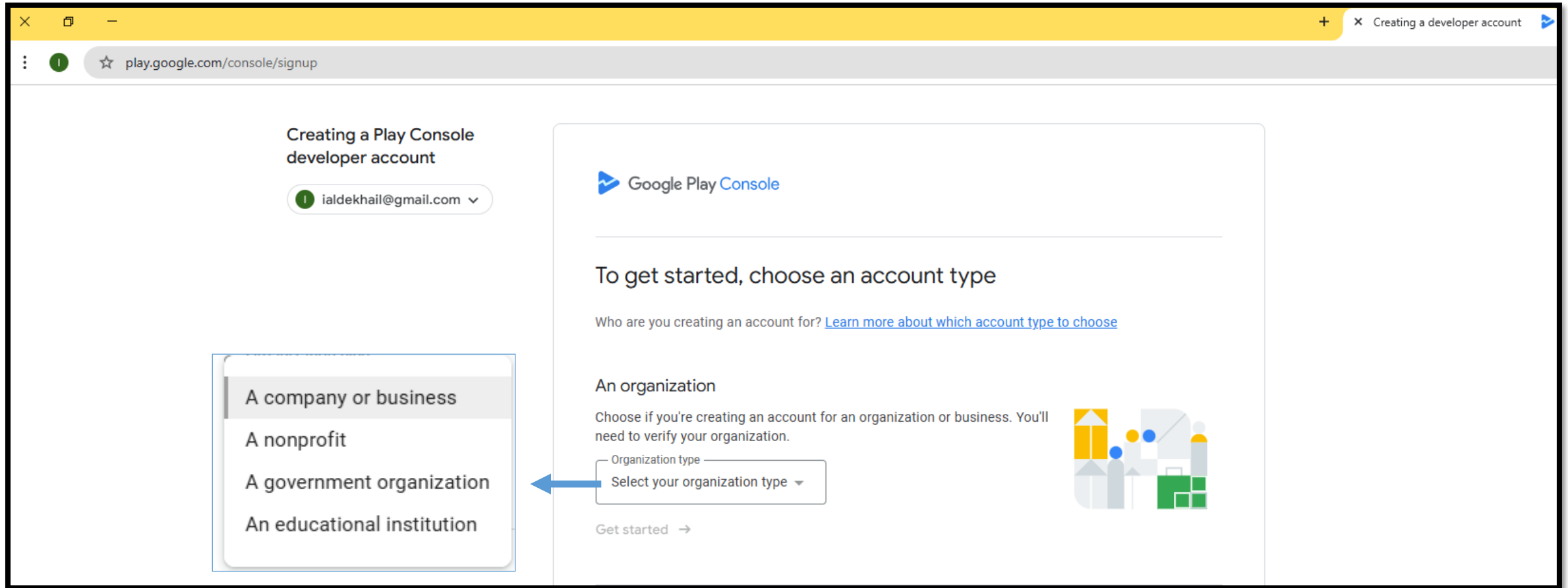
الحالة	تم إغلاق الحساب.
مخلق	20 أكتوبر 2024, 6:51 م
تاريخ إرسال التحذير	14 أغسطس 2024
الموعد النهائي للتعامل مع التحذير	13 أكتوبر 2024
التفاصيل	<p>تم إغلاق حساب المطور الخاص بك نظراً لعدم استخدامه. لقد تم إرسال التحذيرات والمعلومات حول هذه السياسة إلى عنوان البريد الإلكتروني الخاص بصاحب الحساب وأي شخص لديه إذن مشرف الحساب.</p> <p>إنّ رسوم تسجيل حساب المطور غير قابلة للاسترداد. لبدء نشر التطبيقات على Google Play، يجب إنشاء حساب جديد. لمزيد من المعلومات حول هذه السياسة، يُرجى الانتقال إلى "مركز السياسات"</p>

Create new Google Play Console Account

Signin to your google account



Create new Google Play Console Account



The screenshot shows the Google Play Console signup page. The browser's address bar displays `play.google.com/console/signup`. The page title is "Creating a Play Console developer account". A dropdown menu shows the email `ialdekhail@gmail.com`. The main heading is "To get started, choose an account type". Below this, it asks "Who are you creating an account for?" with a link to "Learn more about which account type to choose". The "An organization" section is active, with the text "Choose if you're creating an account for an organization or business. You'll need to verify your organization." and a "Get started" button. A list of organization types is shown on the left, with "A company or business" selected. A blue arrow points from this selection to the "Organization type" dropdown menu on the right, which currently displays "Select your organization type".

Creating a Play Console developer account

ialdekhail@gmail.com

To get started, choose an account type

Who are you creating an account for? [Learn more about which account type to choose](#)

An organization

Choose if you're creating an account for an organization or business. You'll need to verify your organization.

Organization type

Select your organization type

Get started →


- A company or business
- A nonprofit
- A government organization
- An educational institution

<https://play.google.com/console/signup>

Create new Google Play Console Account

Yourself

Choose if you're creating an account for yourself, and you don't currently have an organization or business. For example, if you're an amateur developer, student, or hobbyist. You'll still be able to earn money on Google Play, and invite others to join your account.

 Some types of apps can only be distributed by organizations.

[Learn more](#)

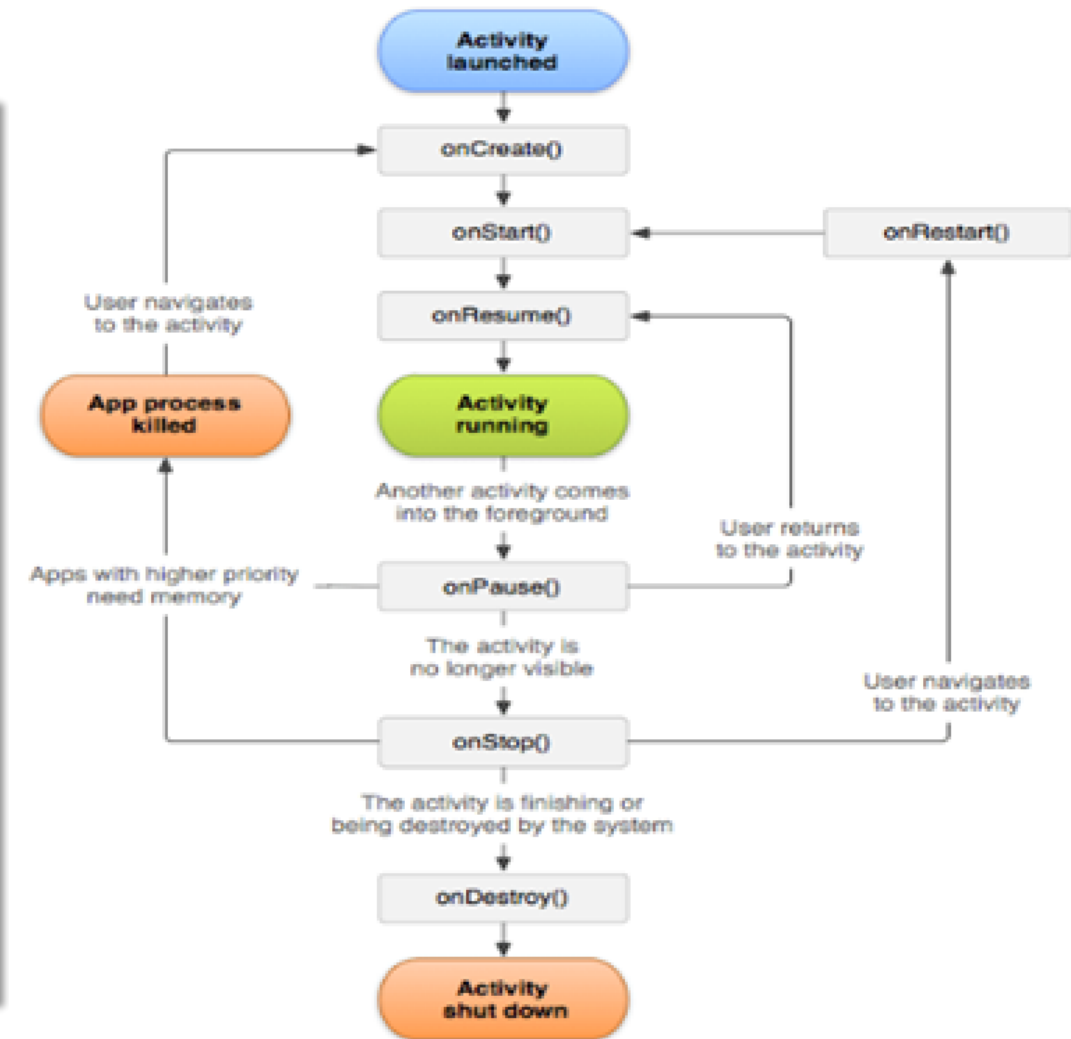
[Get started](#) →

Android activity Lifecycle methods

Android Activity Lifecycle methods

Let's see the 7 lifecycle methods of android activity.

Method	Description
<code>onCreate</code>	called when activity is first created.
<code>onStart</code>	called when activity is becoming visible to the user.
<code>onResume</code>	called when activity will start interacting with the user.
<code>onPause</code>	called when activity is not visible to the user.
<code>onStop</code>	called when activity is no longer visible to the user.
<code>onRestart</code>	called after your activity is stopped, prior to start.
<code>onDestroy</code>	called before the activity is destroyed.



الدالة	متى تستدعاء
onCreate	تستدعاء عند أول انشاء للشاشة
onStart	تستدعاء عندما يصبح التطبيق ظاهر للمستخدم
onResume	تستدعاء عندما يستطيع المستخدم التفاعل مع التطبيق
onPause	تستدعاء عندما يكون التطبيق غير ظاهر للمستخدم (مثل ذهاب التطبيق للخلفية او فتح تطبيق اخر)
onStop	تستدعاء عندما التطبيق تأخر بالظهور
onRestart	تستدعاء عندما ترجع للشاشة بعد التوقف
onDestroy	تستدعاء عندما يغلق الشاشة بشكل نهائي

Introduction (Units)

Screen size

Actual physical size, measured as the screen's diagonal. For simplicity, Android groups all actual screen sizes into four generalized sizes: small, normal, large, and extra-large.

Orientation

The orientation of the screen from the user's point of view. This is either landscape or portrait.

Orientation

The orientation of the screen from the user's point of view. This is either landscape or portrait.

PX (Pixels)

Corresponds to actual pixels on the screen.

Screen density

The number of pixels within a physical area of the screen.

Density-independent pixel (dp)

A virtual pixel unit that you should use when defining UI layout.

Unit	Description	Units Per Physical Inch	Density Independent	Same Physical Size On Every Screen
px	Pixels	Varies	No	No
in	Inches	1	Yes	Yes
mm	Millimeters	25.4	Yes	Yes
pt	Points	72	Yes	Yes
dp	Density Independent Pixels	~160	Yes	No
sp	Scale Independent Pixels	~160	Yes	No

AndroidManifest.xml

مدير المشروع

يوضع فيه جميع البيانات الاساسية للمشروع، مثل:

- اسم التطبيق
- ايقونة التطبيق
- الاذونات

PHONE_CALL ❖

INTERNET ❖

LOCATION ❖

WIFI ❖

CAMERA ❖

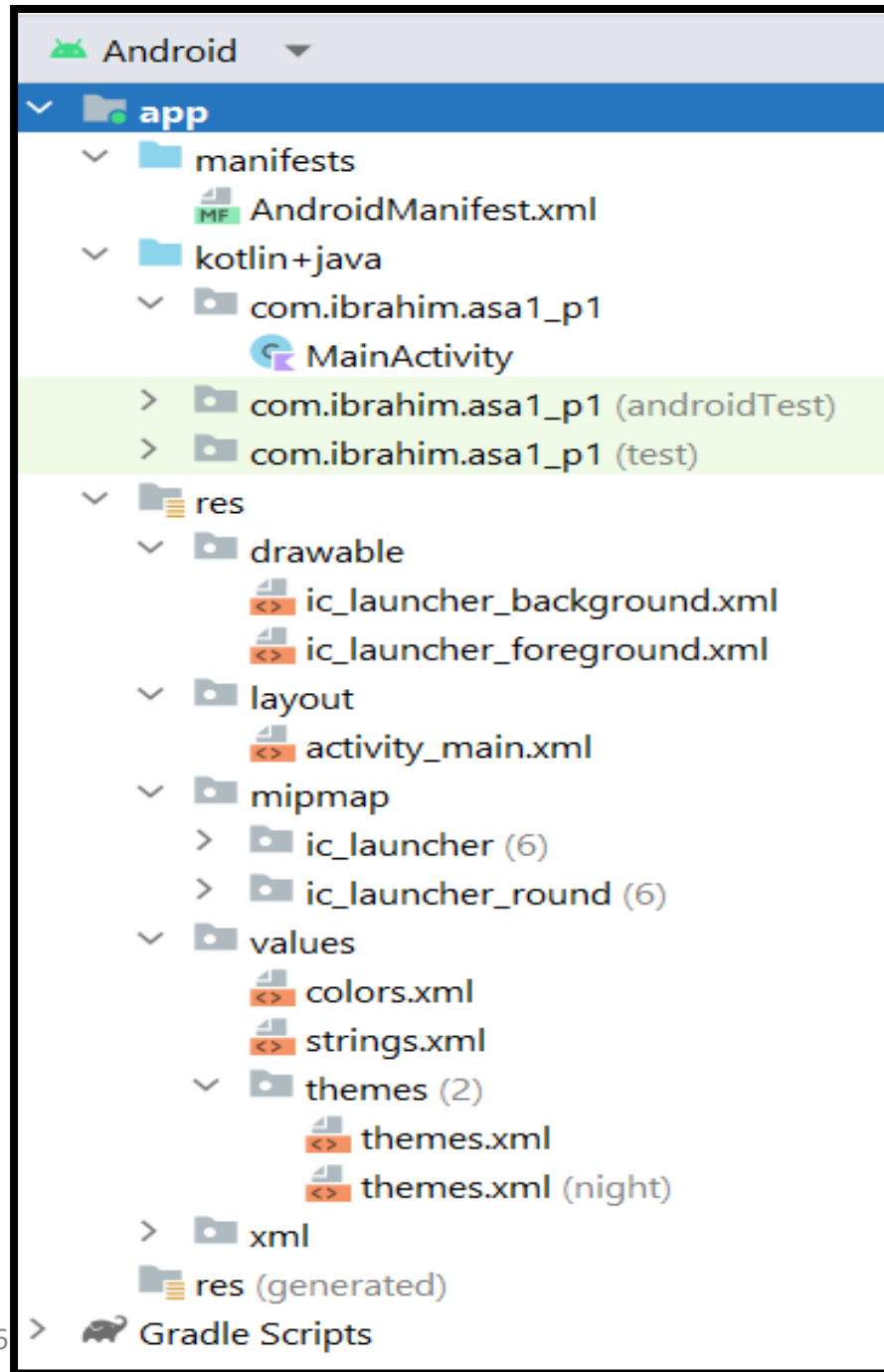
• أسماء الشاشات

• شاشة البداية

• الثيم

• الإصدار المستهدف

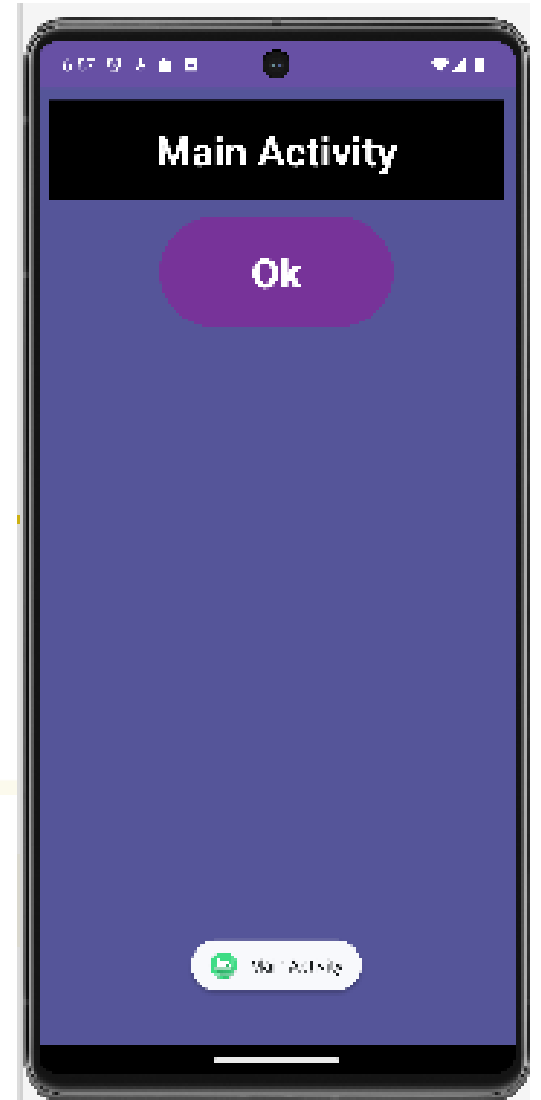
Project



Activity_main.xml

```
MainActivity.kt x activity_main.xml x
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:background="#555599"
8     android:layout_height="match_parent"
9     tools:context=".MainActivity">
10
11     <TextView
12         android:id="@+id/show"
13         android:layout_width="match_parent"
14         android:layout_height="wrap_content"
15         android:layout_margin="10dp"
16         android:background="@color/black"
17         android:gravity="center"
18         android:padding="20dp"
19         android:text="Main Activity"
20         android:textColor="@color/white"
21         android:textSize="35dp"
22         android:textStyle="bold"
23         app:layout_constraintEnd_toEndOf="parent"
24         app:layout_constraintStart_toStartOf="parent"
25         app:layout_constraintTop_toTopOf="parent" />
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44 </androidx.constraintlayout.widget.ConstraintLayout>

<Button
    android:id="@+id/ok"
    android:layout_width="201dp"
    android:layout_height="103dp"
    android:layout_margin="10dp"
    android:layout_marginTop="100dp"
    android:backgroundTint="#773399"
    android:gravity="center"
    android:padding="20dp"
    android:text="Ok"
    android:textColor="@color/white"
    android:textSize="35dp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/show" />
```



MainActivity.kt

```
1 package com.ibrahim.asa1_p1
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5 import android.widget.Button
6 import android.widget.TextView
7 import android.widget.Toast
8
9 class MainActivity : AppCompatActivity() {
10     override fun onCreate(savedInstanceState: Bundle?) {
11         super.onCreate(savedInstanceState)
12         setContentView(R.layout.activity_main)
13
14         val show = findViewById<TextView>(R.id.show)
15         var ok = findViewById<Button>(R.id.ok)
16
17         ok.setOnClickListener { it: View!
18             Toast.makeText(context: this, show.text.toString(), Toast.LENGTH_LONG).show()
19             Toast.makeText(context: this, text: "مرحبا", Toast.LENGTH_LONG).show()
20         }
21     }
22 }
```

المكتبات

الملف البرمجي

دالة إنشاء التصميم على شاشة الجوال

جملة الربط بين الملف البرمجي وملف التصميم

جملة الربط بين العناصر في التصميم والملف البرمجي باستخدام id

حدث عند الضغط على زر يقوم بتنفيذ ما بين الاقواس

رسالتين تنبيه تضرها على الشاشة لفترة مؤقتة ثم تختفيان

تظهر الاولى وبعد ما تنتهي تظهر الثانية

Design for Android

<https://developer.android.com/design/ui>

Design beautiful and modern Android apps that meet your user

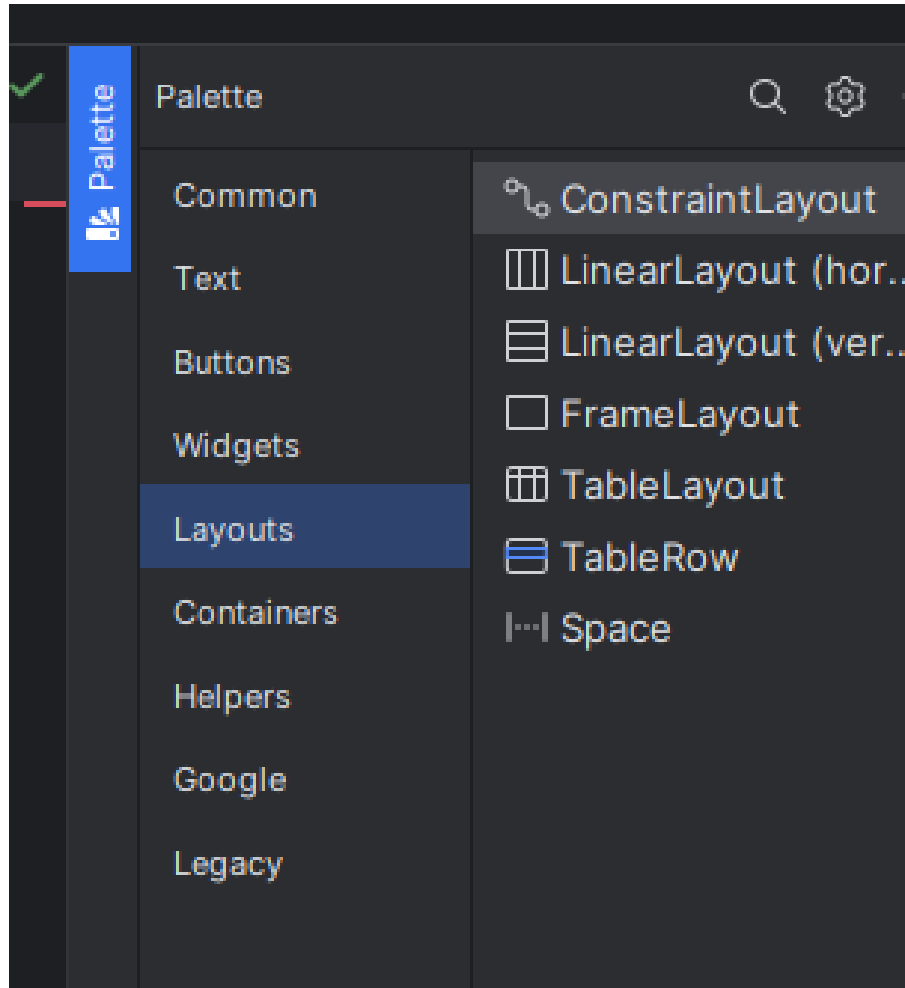
- where they are,
- whether browsing their phone,
- reading on their tablet,
- glancing at their wrist,
- or watching TV.

Top 6 Android UI Design Tools in 2024

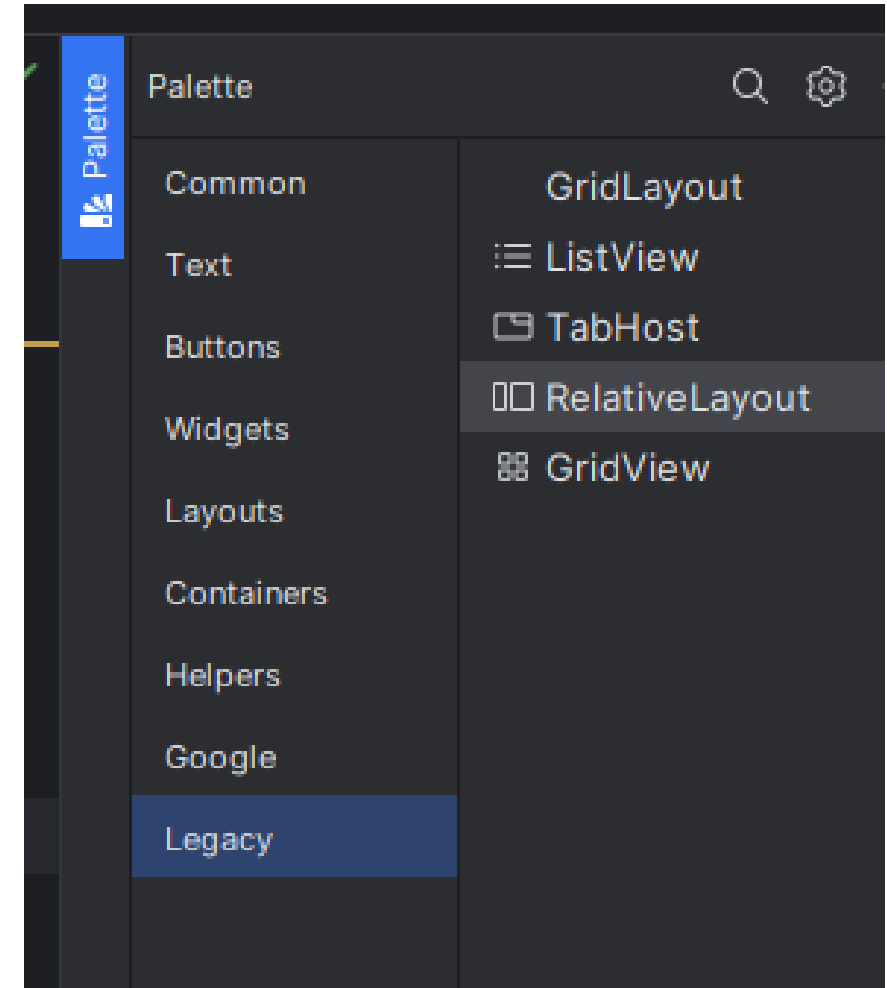
<https://mockitt.com/ui-ux-design/android-ui-design.html>

Design for Android

Layout Types in Android



Legacy Types in Android



Design for Android

```
</> activity_next.xml x
1  <?xml version="1.0" encoding="utf-8"?>
2
```

```
</> activity_next.xml x
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:id="@+id/main"
7      android:layout_width="match_parent"
8      android:layout_height="match_parent"
9      tools:context=".NextActivity">
10
11  </androidx.constraintlayout.widget.ConstraintLayout>
```

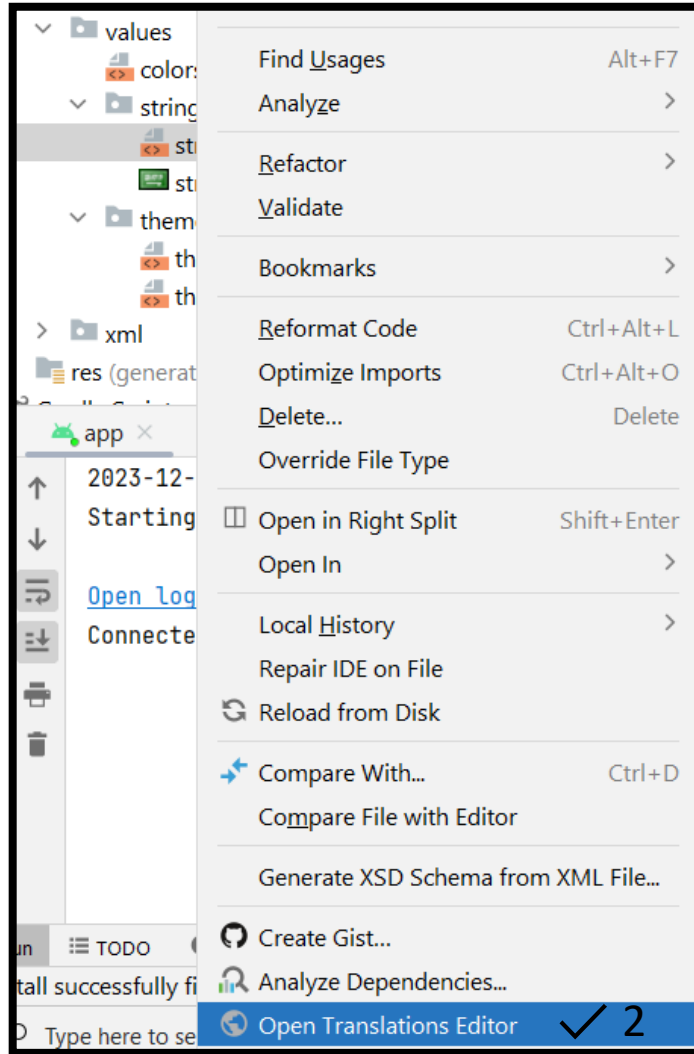
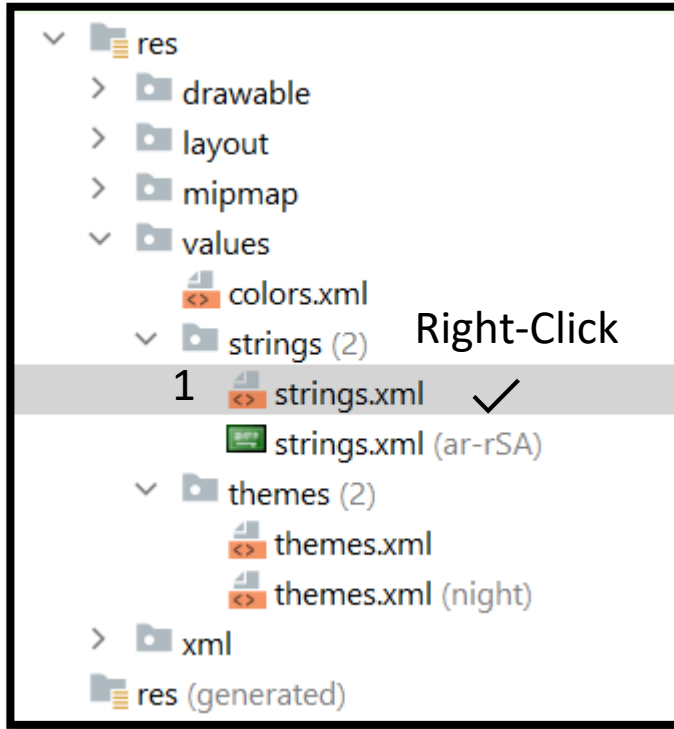
Design for Android

```
<?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="match_parent"
    android:orientation="vertical" >
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a Button" />
</LinearLayout>
```

Android studio run on physical device

- Run apps on a hardware device
- Set up a device for development
 - ❖ Before you can start debugging on your device, decide whether you want to connect to the device using a USB cable or Wi-Fi. Then do the following:
 - ❖ On the device, open the **Settings** app, select **Developer options**, and then enable **USB debugging** (if applicable).
 - ❖ **Windows:** Install a USB driver for ADB (if applicable). For an installation guide and links to OEM drivers, see [Install OEM USB drivers](#).

Localization (التطبيق يعمل بلغة الجهاز)



خطوات إضافة ملف لدعم اللغة العربية

Localization (التطبيق يعمل بلغة الجهاز)

لغة التطبيق تعمل بلغة الجهاز

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:app="http://schemas.android.com/apk/res-auto"
5     xmlns:tools="http://schemas.android.com/tools"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     tools:context=".MainActivity">
9     <TextView
10         android:id="@+id/show"
11         android:layout_width="match_parent"
12         android:layout_height="200dp"
13         android:gravity="center"
14         android:text="@string/show"
15         android:textSize="36dp"
16         android:textStyle="bold"
17         app:layout_constraintEnd_toEndOf="parent"
18         app:layout_constraintStart_toStartOf="parent"
19         app:layout_constraintTop_toTopOf="parent" />
20     <Button
21         android:id="@+id/close"
22         android:layout_width="172dp"
23         android:layout_height="153dp"
24         android:layout_marginTop="96dp"
25         android:text="@string/close"
26         android:textSize="30dp"
27         android:textStyle="italic"
28         app:layout_constraintEnd_toEndOf="parent"
29         app:layout_constraintHorizontal_bias="0.497"
30         app:layout_constraintStart_toStartOf="parent"
31         app:layout_constraintTop_toBottomOf="@+id/show" />
32 </androidx.constraintlayout.widget.ConstraintLayout>
```

activity_main.xml x strings.xml x

Edit translations for all locales in the translations editor.

```
1 <resources>
2     <string name="app_name">ASA1_Icon</string>
3     <string name="show">show</string>
4     <string name="close">close</string>
5 </resources>
```

activity_main.xml x values\strings.xml x ar-rSA\strings.xml x

Visual layout of bidirectional text can depend on the base direction (View | Bidi Text Base Direction)

Edit translations for all locales in the translations editor.

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <resources>
3     <string name="app_name">تصميم ايقونة</string>
4     <string name="show">عرض</string>
5     <string name="close">أغلق</string>
6 </resources>
```

Full Screen [NoActionBar – No Title]

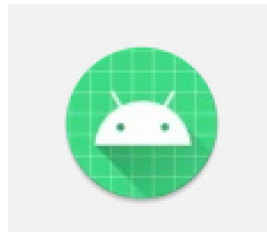
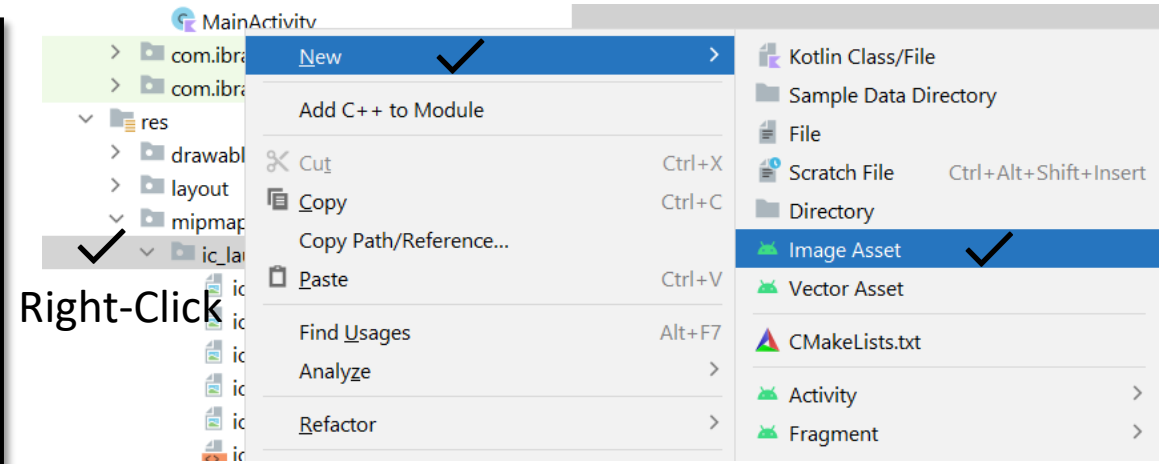
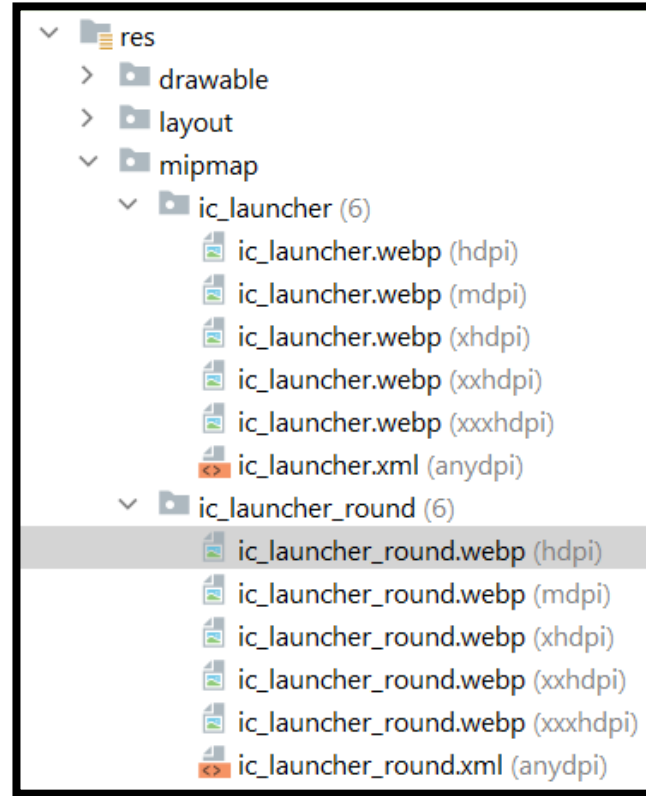


```
MainActivity.kt x activity_main.xml x themes.xml x
1 package com.ibrahim.asa_localization
2 import androidx.appcompat.app.AppCompatActivity
3 import android.os.Bundle
4 import android.view.Window
5 import android.view.WindowManager
6 class MainActivity : AppCompatActivity() {
7     override fun onCreate(savedInstanceState: Bundle?) {
8         super.onCreate(savedInstanceState)
9         // Hide Title
10        getWindow().requestFeature(Window.FEATURE_NO_TITLE)
11        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
12                               WindowManager.LayoutParams.FLAG_FULLSCREEN)
13        setContentView(R.layout.activity_main)
14    }
15 }
```

```
MainActivity.kt x activity_main.xml x themes.xml x
1 <resources xmlns:tools="http://schemas.android.com/tools">
2     <style name="Base.Theme.ASA_Localization" parent="Theme.Material3.DayNight.NoActionBar">
3     </style>
4     <style name="Theme.ASA_Localization" parent="Base.Theme.ASA_Localization" />
5 </resources>
```




Create App Icon



Google Play icon design specifications

<https://developer.android.com/distribute/google-play/resources/icon-design-specifications>

Create App Icon ... Cont

 Configure Image Asset

Icon type: Launcher Icons (Adaptive and Legacy) ▾

Preview xhdpi ▾ ☒ Show safe zone ☐ Show grid

Name: ic_launcher

Foreground Layer Background Layer Options

Layer name: ic_launcher_foreground

Source Asset


Asset type: ☒ Image ☐ Clip art ☐ Text


Path: :t_studio\ic_launcher_foreground.xml


Scaling


Trim: ☐ Yes ☒ No


Resize: 100 %


 Circle


 Squircle


 Rounded Square


 Square


 Full Bleed Layers

 Legacy Icon

 Round Icon

 Google Play Store

 An icon with the same name already exists and will be overwritten.

 28

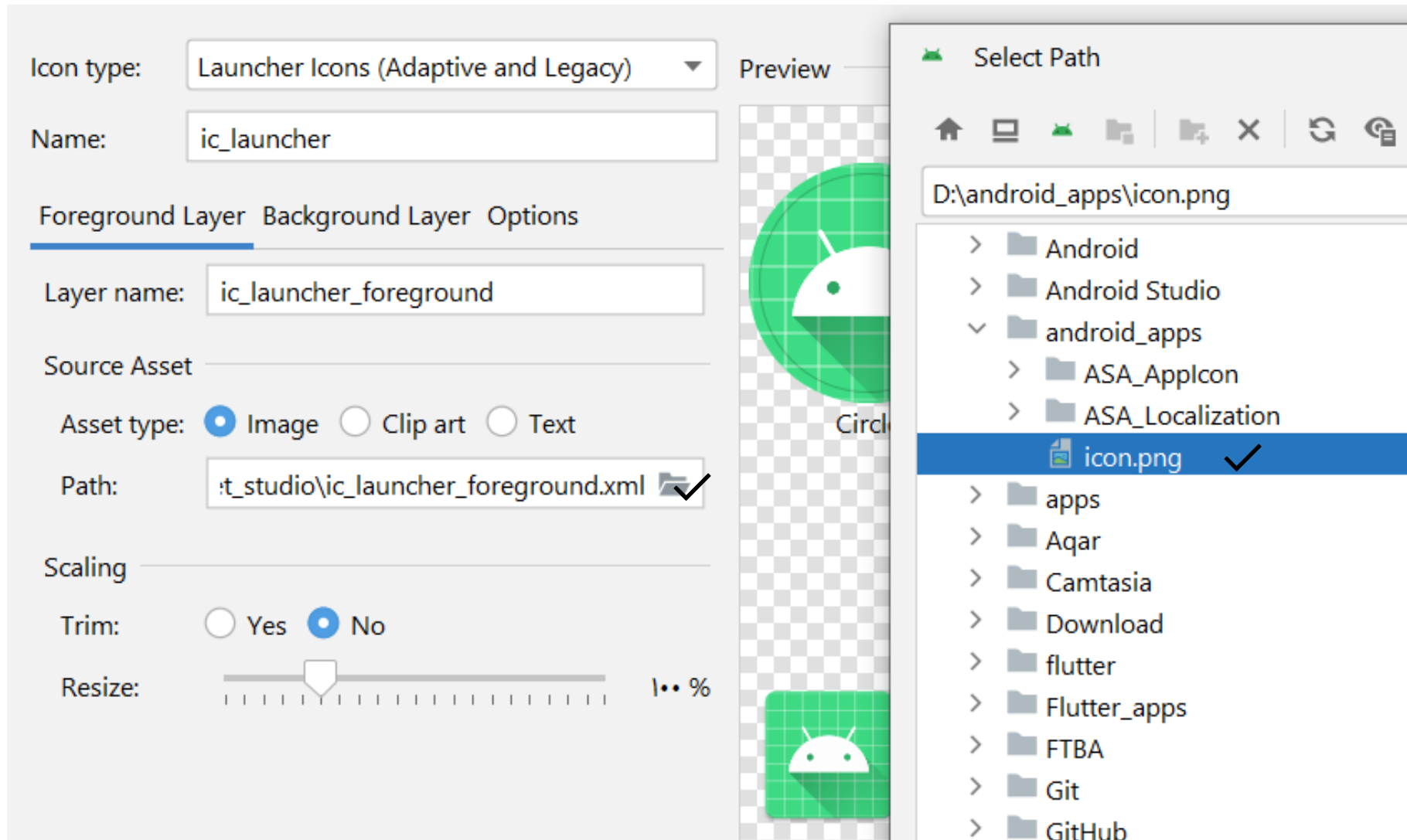
Previous

Next


Cancel

Finish

Create App Icon ... Cont



Create App Icon ... Cont

 Configure Image Asset

Icon type: Launcher Icons (Adaptive and Legacy) ▾

Name: ic_launcher

Foreground Layer Background Layer Options

Layer name: ic_launcher_foreground

Source Asset

Asset type: ☒ Image ☐ Clip art ☐ Text


Path: D:\android_apps\icon.png


Scaling


Trim: ☐ Yes ☒ No


Resize: 100 %


Preview xhdpi ☒ Show safe zone ☐ Show grid


 Circle


 Squircle


 Rounded Square


 Square


 Full Bleed Layers

 Legacy Icon


 Round Icon

 Google Play Store

 An icon with the same name already exists and will be overwritten.

 ?

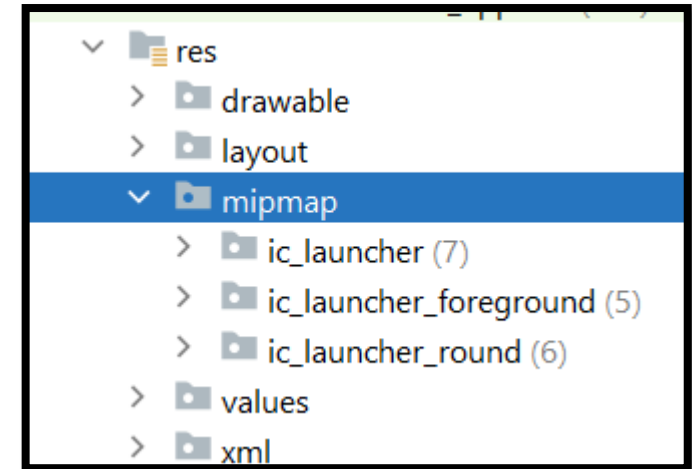
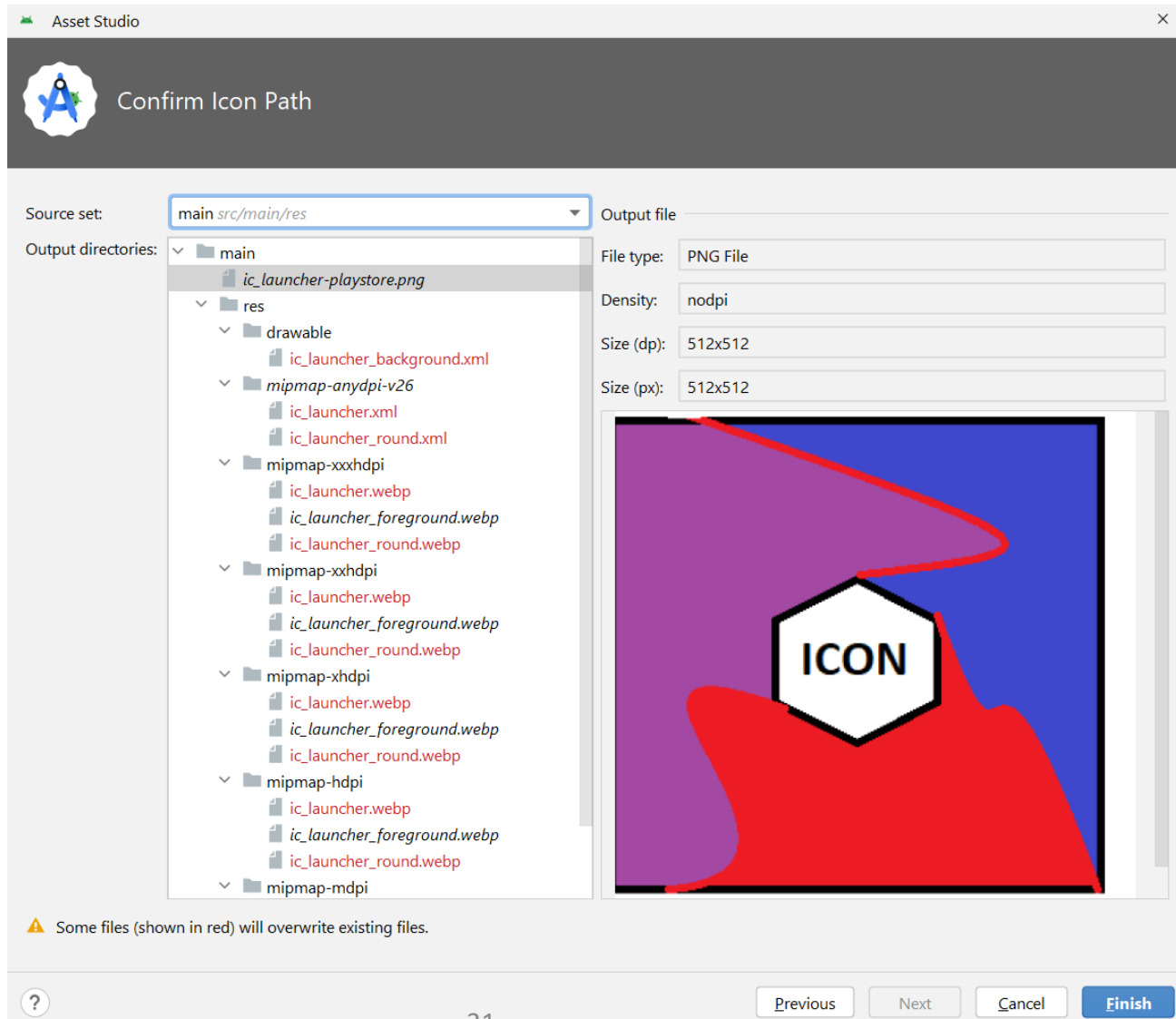
Previous

 Next

Cancel

Finish

Create App Icon ... Cont



Create App Icon ... Cont

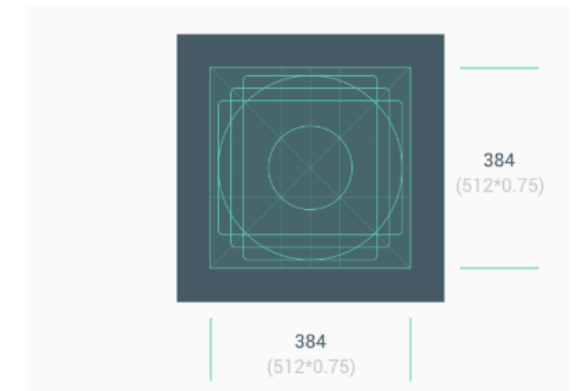
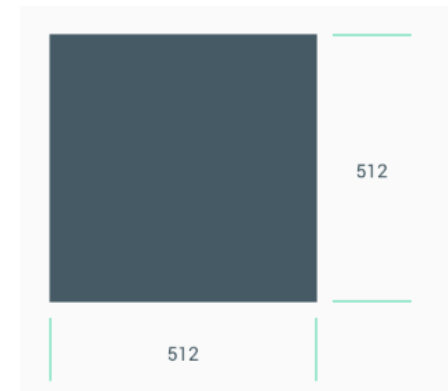
What is WebP in Android Studio?

WebP is an image file format from Google that provides lossy compression (like JPEG) as well as transparency (like PNG) but can provide better compression than either JPEG or PNG.

Images

You can import your own images and adjust them for the icon type. Image Asset Studio supports the following file types: PNG (preferred), JPG (acceptable), and GIF (discouraged).

Image Asset Studio places the icons in the proper locations in the `res/mipmap-density` directories. It also creates a 512 x 512 pixel image that's appropriate for the Google Play store.

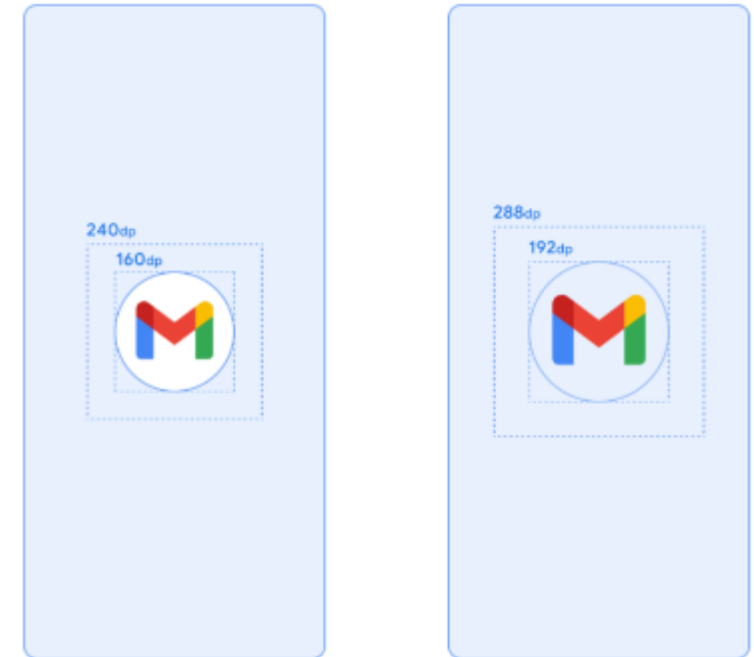


Splash Screen

Splash screen dimensions

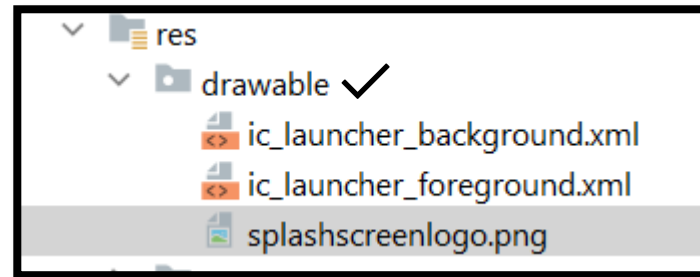
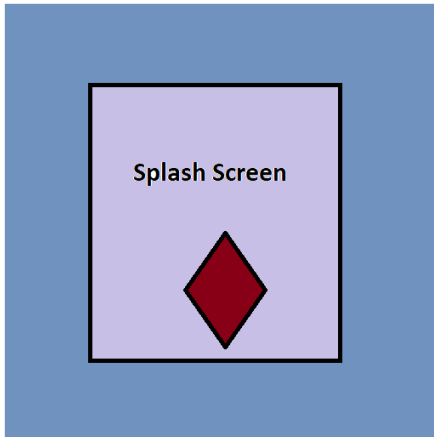
The splash screen icon uses the same specifications as adaptive icons, as follows:

- Branded image: this must be 200×80 dp.
- App icon with an icon background: this must be 240×240 dp and fit within a circle 160 dp in diameter.
- App icon without an icon background: this must be 288×288 dp and fit within a circle 192 dp in diameter.
- For example, if the full size of an image is 300×300 dp, the icon needs to fit within a circle with a diameter of 200 dp. Everything outside the circle turns invisible (masked).
- 1/3 - 2/3



Splash Screen

Step 1: Design logo then **Copy/Paste** into drawable folder



Splash Screen

Step 2: Add the following snippet to your **build.gradle** file

<https://developer.android.com/develop/ui/views/launch/splash-screen#kts>

Get started ↗

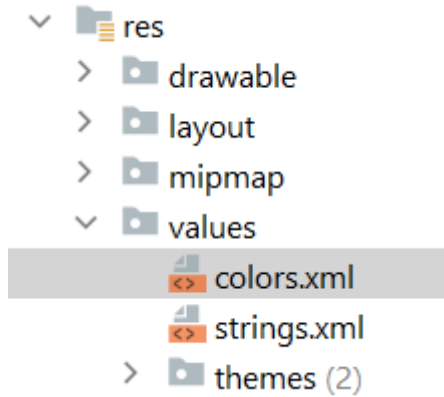
The core `SplashScreen` library brings the Android 12 splash screen to all devices from API 23. To add it to your project, add the following snippet to your `build.gradle` file:

```
Groovy Kotlin
dependencies {
    implementation("androidx.core:core-splashscreen:1.0.0")
}
```

```
build.gradle.kts (:app) x
Gradle files have changed since last project sync. A project sync may be necess... Sync Now Ignore these changes
35 }
36 }
37
38 dependencies { this: DependencyHandlerScope
39
40     implementation("androidx.core:core-ktx:1.12.0")
41     implementation("androidx.appcompat:appcompat:1.6.1")
42     implementation("com.google.android.material:material:1.11.0")
43     implementation("androidx.constraintlayout:constraintlayout:2.1.4")
44     testImplementation("junit:junit:4.13.2")
45     androidTestImplementation("androidx.test.ext:junit:1.1.5")
46     androidTestImplementation("androidx.test.espresso:espresso-core:3.5.1")
47     implementation("androidx.core:core-splashscreen:1.0.0")
48 }
```

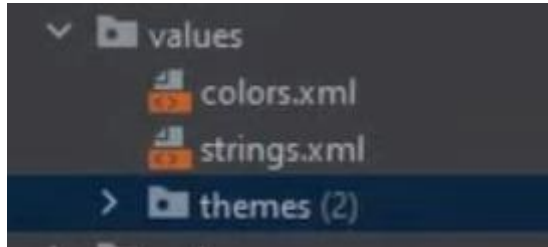
Splash Screen

Step 3: Add background color



Splash Screen

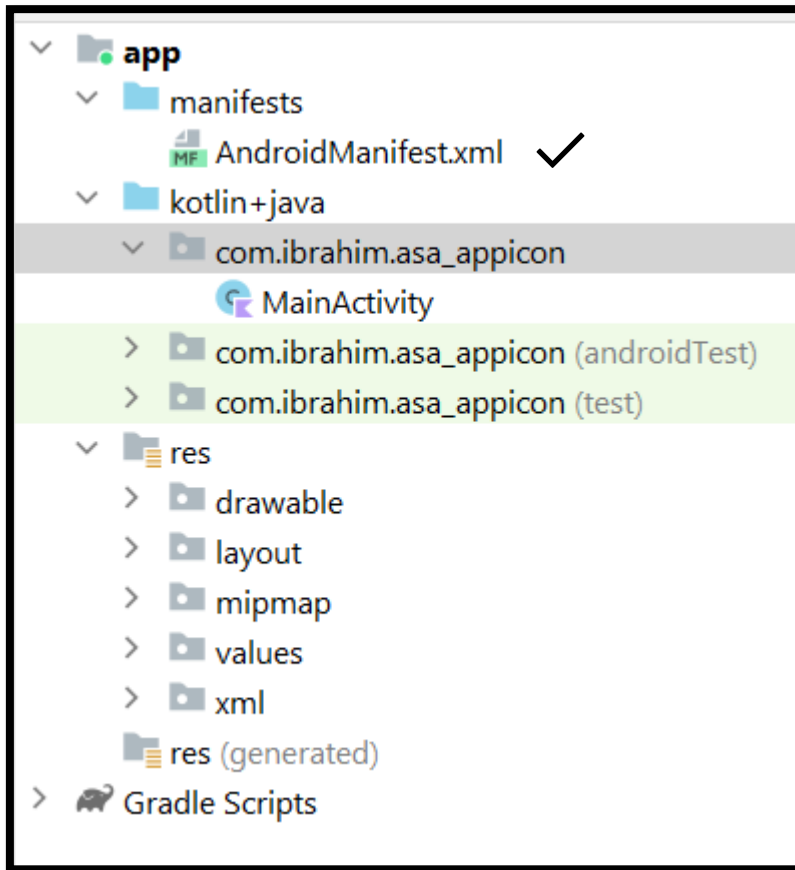
Step 4: Add Theme to themes.xml file



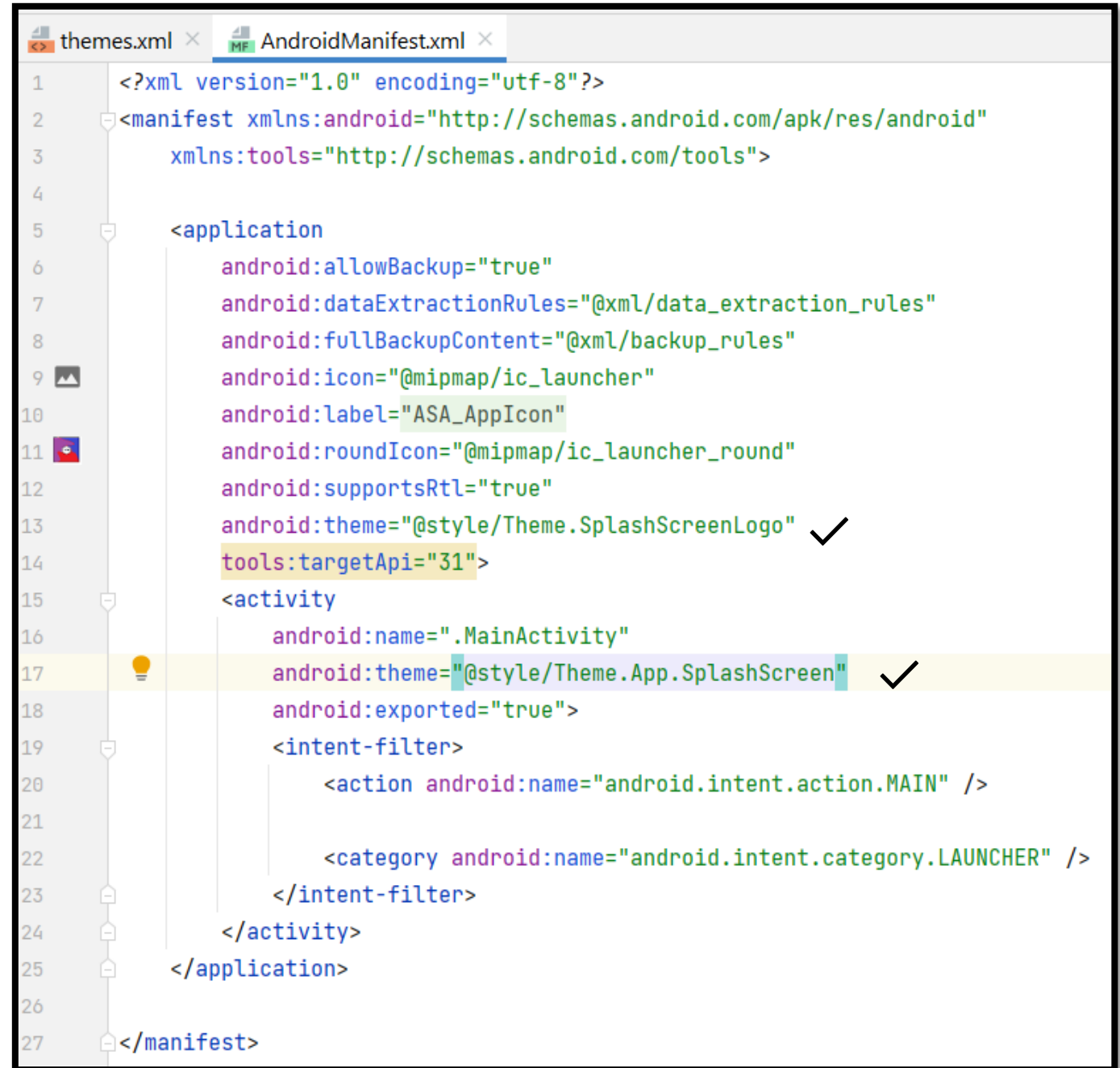
```
themes.xml x
1 <resources xmlns:tools="http://schemas.android.com/tools">
2   <style name="Theme.ASA_AppIcon" parent="Base.Theme.ASA_AppIcon" />
3
4   <!-- Base application theme. -->
5   <style name="Base.Theme.SplashScreenLogo" parent="Theme.MaterialComponents.DayNight.DarkActionBar">
6     <!-- Customize your light theme here. -->
7     <item name="colorPrimary">@color/gold</item>
8     <item name="colorPrimaryVariant">@color/gold</item>
9     <item name="colorOnPrimary">@color/gold</item>
10    <item name="android:statusBarColor">?attr/colorPrimaryVariant</item>
11  </style>
12
13  <style name="Theme.SplashScreenLogo" parent="Base.Theme.SplashScreenLogo" />
14
15  <style name="Theme.App.SplashScreen" parent="Theme.SplashScreen">
16    <item name="windowSplashScreenBackground">@color/gold</item>
17    <item name="windowSplashScreenAnimatedIcon">@drawable/splashscreenlogo</item>
18    <item name="postSplashScreenTheme">@style/Base.Theme.SplashScreenLogo</item>
19  </style>
20</resources>
```

Splash Screen

Step 5: Add Theme to AndroidManifest.xml file

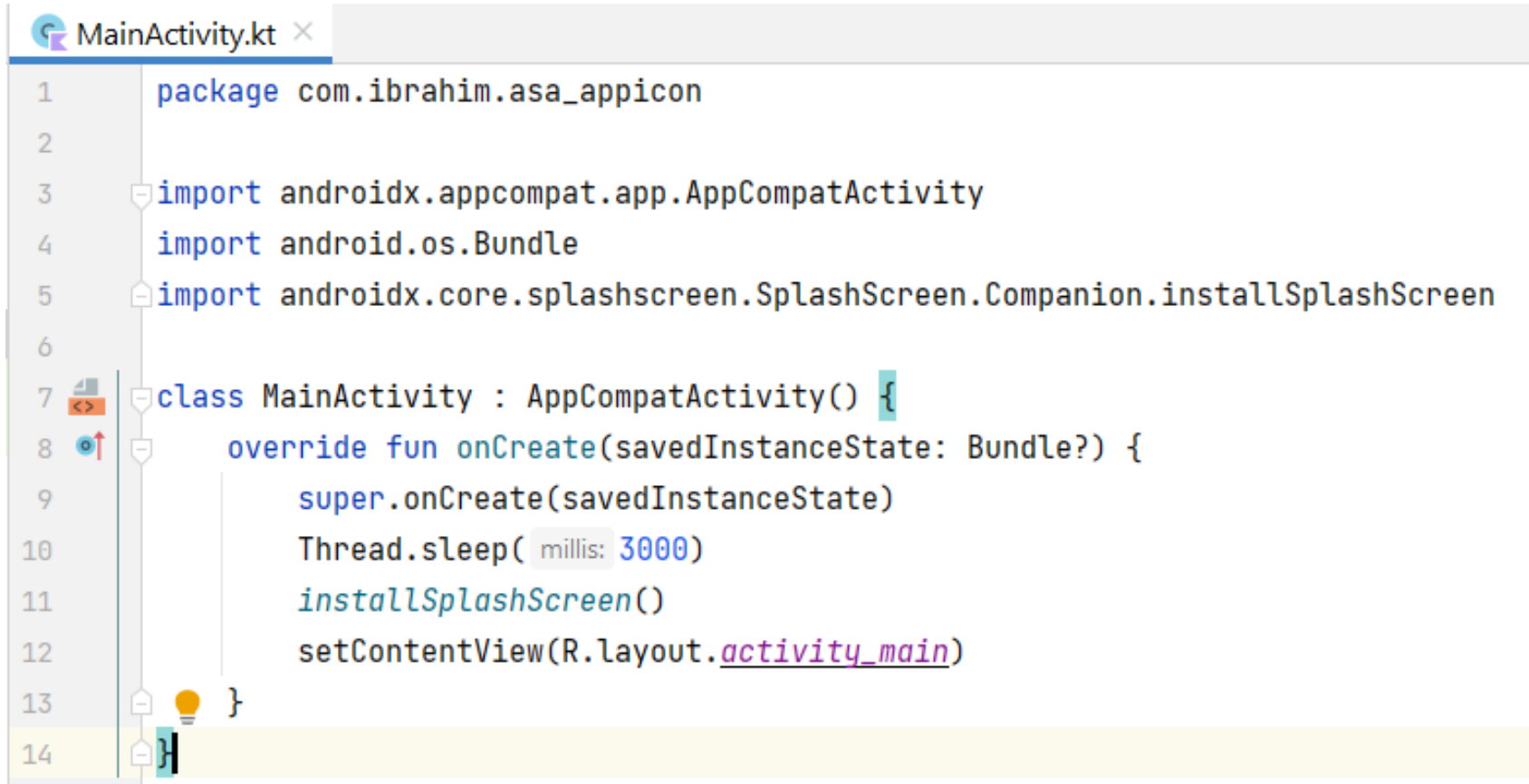


38



Splash Screen

Step 6: Change MainActivity

A screenshot of an IDE showing the MainActivity.kt file. The code is in Kotlin and defines the MainActivity class, which inherits from AppCompatActivity. The onCreate method is overridden to call super.onCreate, sleep for 3000 milliseconds, call installSplashScreen, and set the content view to R.layout.activity_main. The file name 'MainActivity.kt' is visible in the tab at the top. The code is as follows:

```
1 package com.ibrahim.asa_appicon
2
3 import androidx.appcompat.app.AppCompatActivity
4 import android.os.Bundle
5 import androidx.core.splashscreen.SplashScreen.Companion.installSplashScreen
6
7 class MainActivity : AppCompatActivity() {
8     override fun onCreate(savedInstanceState: Bundle?) {
9         super.onCreate(savedInstanceState)
10        Thread.sleep(3000)
11        installSplashScreen()
12        setContentView(R.layout.activity_main)
13    }
14 }
```

Android Animation

Animation in android apps is:

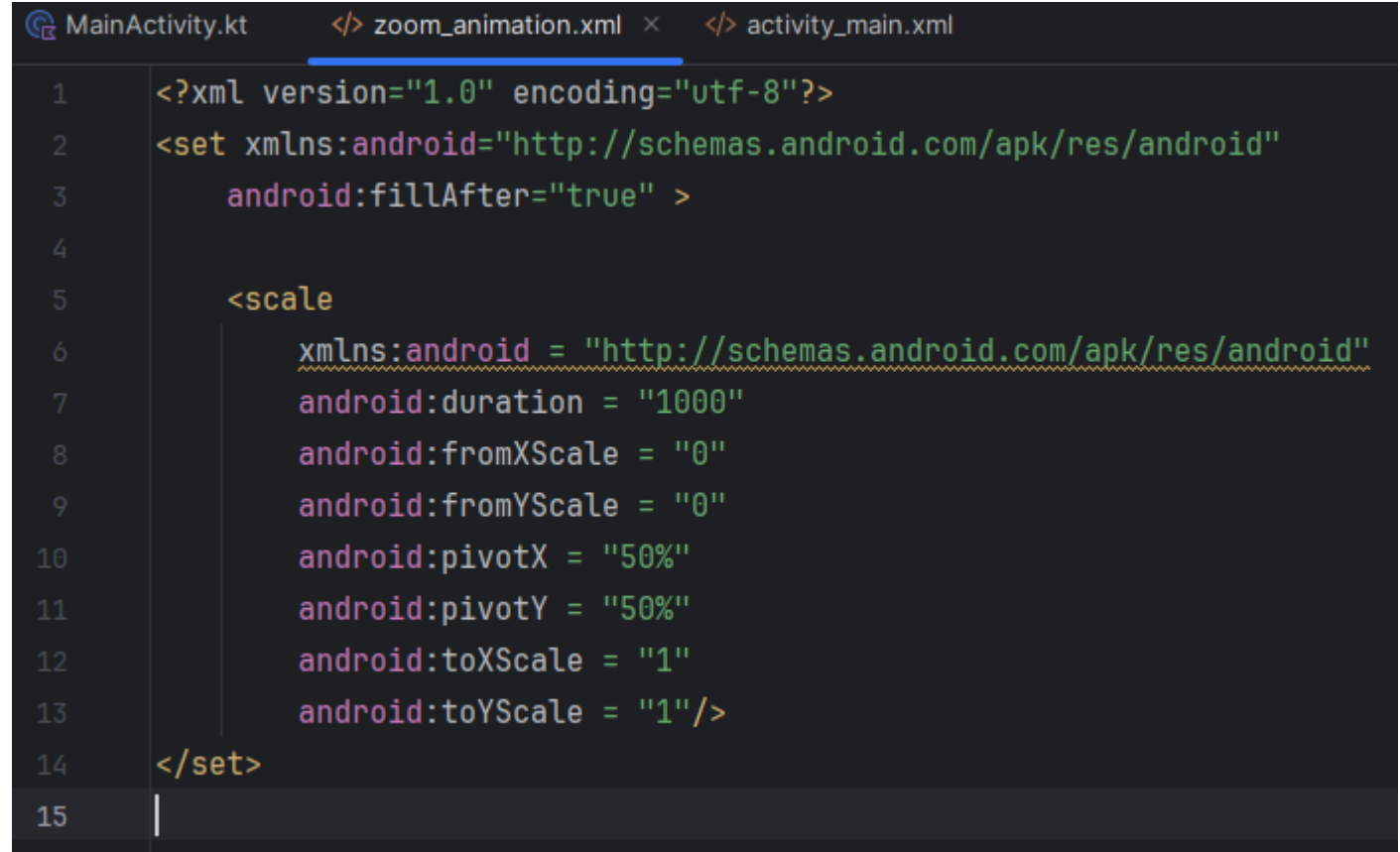
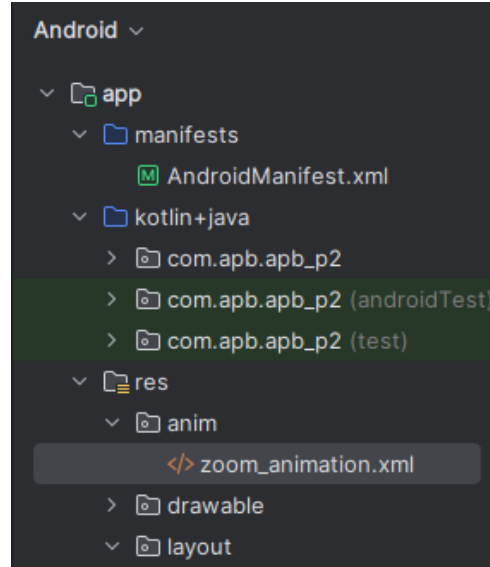
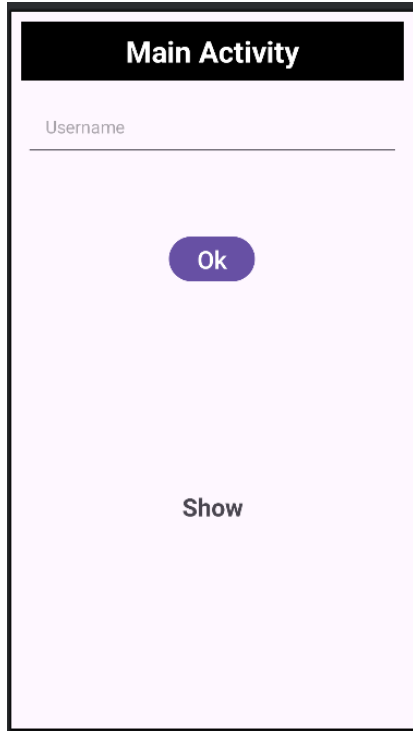
- the process of creating motion
- and shape change.

The basic ways of animation that we'll look upon in this tutorial are:

1. Fade In Animation
2. Fade Out Animation
3. Cross Fading Animation
4. Blink Animation
5. Zoom In Animation
6. Zoom Out Animation
7. Rotate Animation
8. Move Animation
9. Slide Up Animation
10. Slide Down Animation
11. Bounce Animation
12. Sequential Animation
13. Together Animation

https://github.com/idekhail/APA_Animation1

Animation



https://github.com/idekhail/APB_P2-Animation-EditText

https://www.geeksforgeeks.org/implement-zoom-in-or-zoom-out-in-android/?ref=ml_lbp

Exam Reference

- <https://www.interviewbit.com/android-mcq/>
- <https://www.javaguides.net/2023/12/android-quiz-mcq-questions-and-answers.html>