

TRƯỜNG ĐẠI HỌC THỦY LỢI
KHOA CÔNG NGHỆ THÔNG TIN



GIÁO TRÌNH

THỰC HÀNH PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG

Hà Nội, 2.2025

MỤC LỤC

CHƯƠNG 1.	Làm quen	3
Bài 1)	Tạo ứng dụng đầu tiên	3
1.1)	Android Studio và Hello World.....	3
1.2)	Giao diện người dùng tương tác đầu tiên	27
1.3)	Trình chỉnh sửa bố cục	Error! Bookmark not defined.
1.4)	Văn bản và các chế độ cuộn	58
1.5)	Tài nguyên có sẵn	71
Bài 2)	Activities	71
2.1)	Activity và Intent	71
2.2)	Vòng đời của Activity và trạng thái	71
2.3)	Intent ngầm định.....	71
Bài 3)	Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ.....	71
3.1)	Trình gỡ lỗi.....	71
3.2)	Kiểm thử đơn vị.....	71
3.3)	Thư viện hỗ trợ.....	71
CHƯƠNG 2.	Trải nghiệm người dùng.....	72
Bài 1)	Tương tác người dùng.....	72
1.1)	Hình ảnh có thể chọn	72
1.2)	Các điều khiển nhập liệu	72
1.3)	Menu và bộ chọn	72
1.4)	Điều hướng người dùng.....	72
1.5)	RecyclerView.....	72
Bài 2)	Trải nghiệm người dùng thú vị	72
2.1)	Hình vẽ, định kiểu và chủ đề	72
2.2)	Thẻ và màu sắc.....	72

2.3)	Bố cục thích ứng.....	72
Bài 3)	Kiểm thử giao diện người dùng	72
3.1)	Espresso cho việc kiểm tra UI.....	72
CHƯƠNG 3.	Làm việc trong nền	72
Bài 1)	Các tác vụ nền	72
1.1)	AsyncTask	72
1.2)	AsyncTask và AsyncTaskLoader	72
1.3)	Broadcast receivers	72
Bài 2)	Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền.....	72
2.1)	Thông báo.....	72
2.2)	Trình quản lý cảnh báo	72
2.3)	JobScheduler	72
CHƯƠNG 4.	Lưu dữ liệu người dùng	73
Bài 1)	Tùy chọn và cài đặt.....	73
1.1)	Shared preferences	73
1.2)	Cài đặt ứng dụng	73
Bài 2)	Lưu trữ dữ liệu với Room	73
2.1)	Room, LiveData và ViewModel	73
2.2)	Room, LiveData và ViewModel	73

3.1) Trính gowx loi

CHƯƠNG 1. LÀM QUEN

Bài 1) Tạo ứng dụng đầu tiên

1.1) Android Studio và Hello World

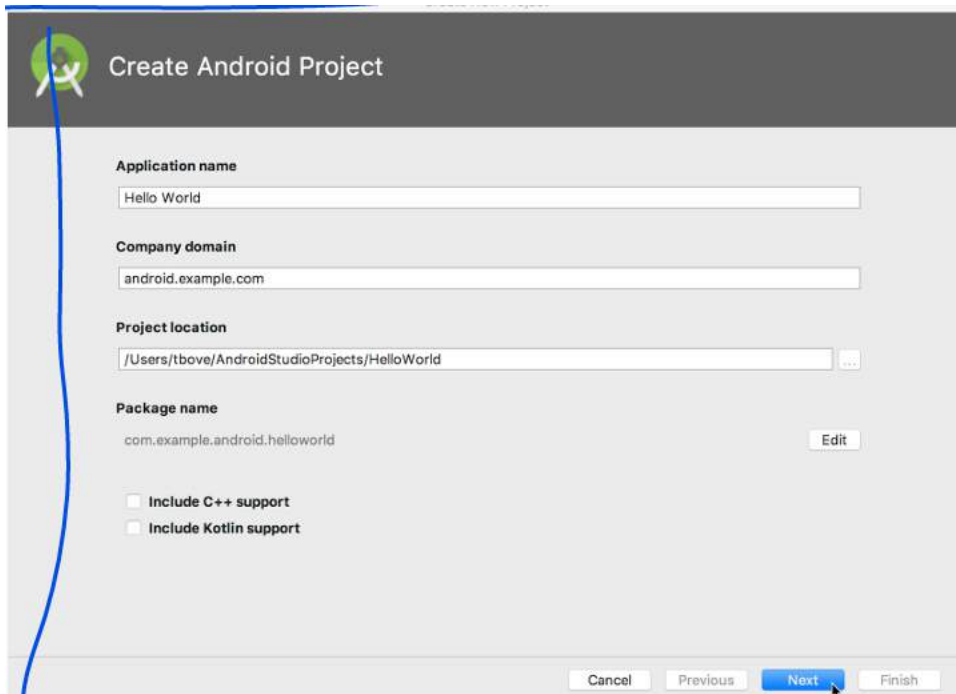
Giới thiệu

Trong bài thực hành này, bạn sẽ tìm hiểu cách cài đặt Android Studio, môi trường phát triển Android. Bạn cũng sẽ tạo và chạy ứng dụng Android đầu tiên của mình, Hello World, trên một trình giả lập và trên một thiết bị vật lý.

Những gì Bạn nên biết

Bạn nên có khả năng:

- Hiểu quy trình phát triển phần mềm tổng quát cho các ứng dụng lập trình hướng đối tượng sử dụng một IDE (môi trường phát triển tích hợp) như Android Studio.
- Chứng minh rằng bạn có ít nhất 1-3 năm kinh nghiệm trong lập trình hướng đối tượng, với một phần trong số đó tập trung vào ngôn ngữ lập trình Java. (Các bài thực hành này sẽ không giải thích về lập trình hướng đối tượng hoặc ngôn ngữ Java.



Những gì Bạn sẽ cần:

- Một máy tính chạy Windows hoặc Linux, hoặc một Mac chạy macOS. Xem trang tải xuống Android Studio để biết yêu cầu hệ thống cập nhật.
- Truy cập Internet hoặc một phương pháp thay thế để tải các cài đặt mới nhất của Android Studio và Java lên máy tính của bạn.

Những gì bạn sẽ học

- Cách cài đặt và sử dụng IDE Android Studio.
- Cách sử dụng quy trình phát triển để xây dựng ứng dụng Android.
- Cách tạo một dự án Android từ một mẫu.
- Cách thêm thông điệp ghi lại vào ứng dụng của bạn để phục vụ mục đích gỡ lỗi.

Những gì bạn sẽ làm

- Cài đặt môi trường phát triển **Android Studio**.
- Tạo một trình giả lập (thiết bị ảo) để chạy ứng dụng của bạn trên máy tính.
- Tạo và chạy ứng dụng **Hello World** trên các thiết bị ảo và vật lý.

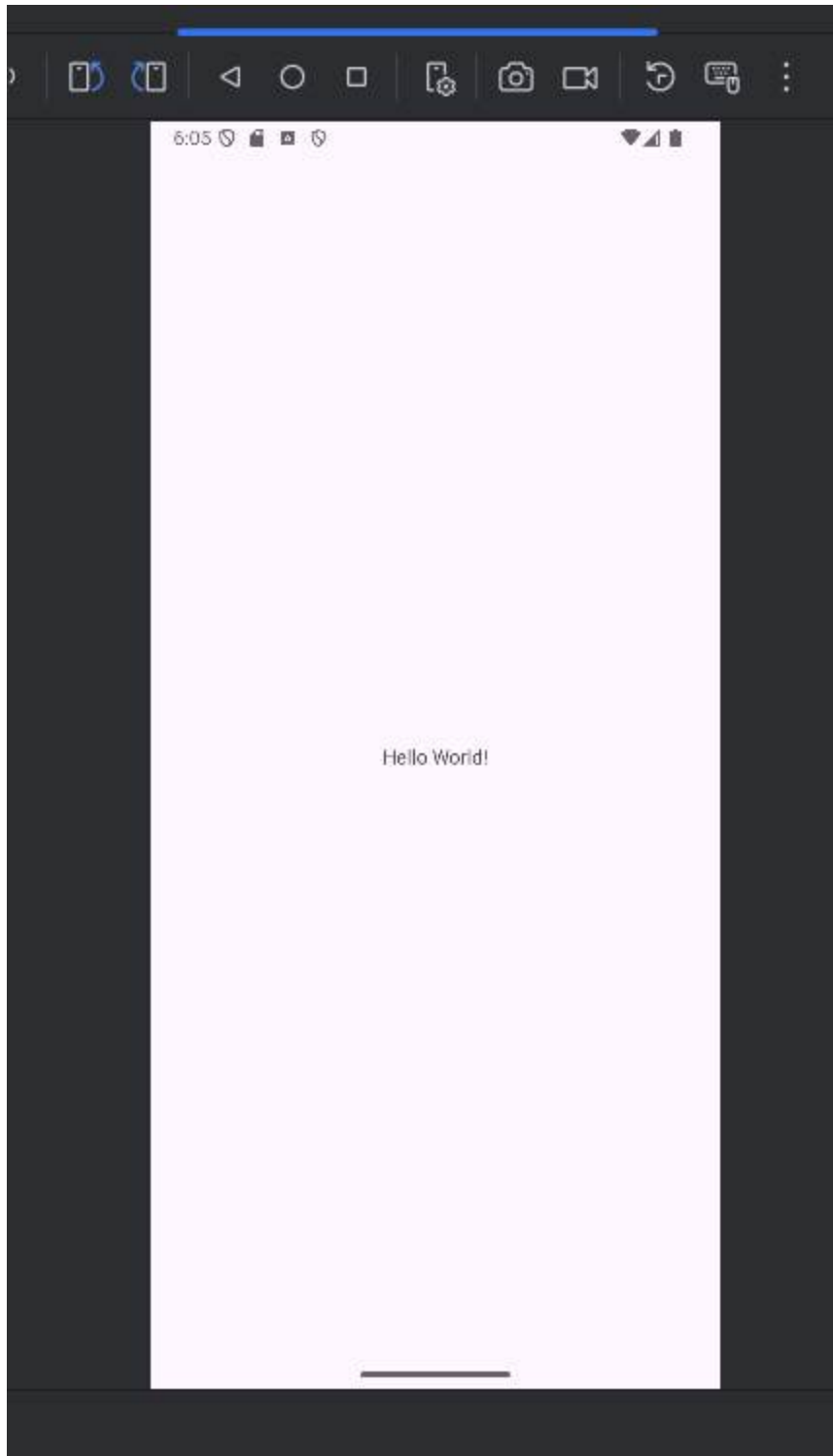
Khám phá cấu trúc dự án.

- Tạo và xem các thông điệp ghi lại từ ứng dụng của bạn.
- Khám phá tệp **AndroidManifest.xml**

Tổng quan về ứng dụng

Sau khi cài đặt thành công Android Studio, bạn sẽ tạo, từ một mẫu, một dự án mới cho ứng dụng Hello World. Ứng dụng đơn giản để hiển thị chuỗi “Hello World” trên màn hình máy ảo hoặc thiết bị vật lý.

Ứng dụng khi hoàn thành sẽ trông như thế này:



Nhiệm vụ 1: Cài đặt Android Studio

Android Studio cung cấp môi trường phát triển tích hợp (IDE) hoàn chỉnh bao gồm các trình chỉnh sửa mã nâng cao và một tập mẫu ứng dụng. Ngoài ra, nó còn chứa các công cụ để phát triển, sửa lỗi, kiểm thử và hiệu suất giúp phát triển ứng dụng nhanh và dễ dàng hơn. Bạn có thể kiểm tra ứng dụng của mình bằng nhiều trình mô phỏng được cấu hình sẵn hoặc trên thiết bị di động của riêng mình, tạo ứng dụng và phát hành trên cửa hàng Google Play.

Lưu ý: Android Studio liên tục được cải thiện. Để biết thông tin mới nhất về yêu cầu hệ thống và hướng dẫn cài đặt, hãy xem Android Studio.

Android Studio có sẵn cho máy chạy Windows hoặc Linux, và với Macs đang chạy macOS. OpenJDK (Java Development Kit) mới nhất được đi kèm với Android Studio.

Để thiết lập và chạy Android Studio, trước tiên hãy kiểm tra các yêu cầu hệ thống để đảm bảo hệ thống của bạn đáp ứng các yêu cầu đó. Việc cài đặt tương tự cho tất cả các nền tảng. Bất kỳ sự khác biệt nào được lưu ý bên dưới.

1. Truy cập trang web Android developers và xem hướng dẫn tải và cài đặt Android Studio.
2. Chấp nhận cấu hình mặc định cho tất cả các bước, và đảm bảo tất cả các thành phần được chọn để cấu hình.
3. Sau khi cài đặt xong, trình hướng dẫn cài đặt sẽ tải xuống và cài đặt thêm một số thành phần bổ sung bao gồm Android SDK. Hãy kiên nhẫn, quá trình này có thể mất một thời gian tùy thuộc vào tốc độ mạng của bạn và một số bước dư thừa.
4. Khi quá trình tải xuống hoàn tất, Android Studio sẽ khởi động và bạn sẵn sàng tạo dự án đầu tiên.

Khắc phục sự cố: Nếu bạn gặp sự cố khi cài đặt, hãy xem ghi chú phát hành Android Studio hoặc nhờ giúp đỡ.

Nhiệm vụ 2: Tạo ứng dụng Hello World

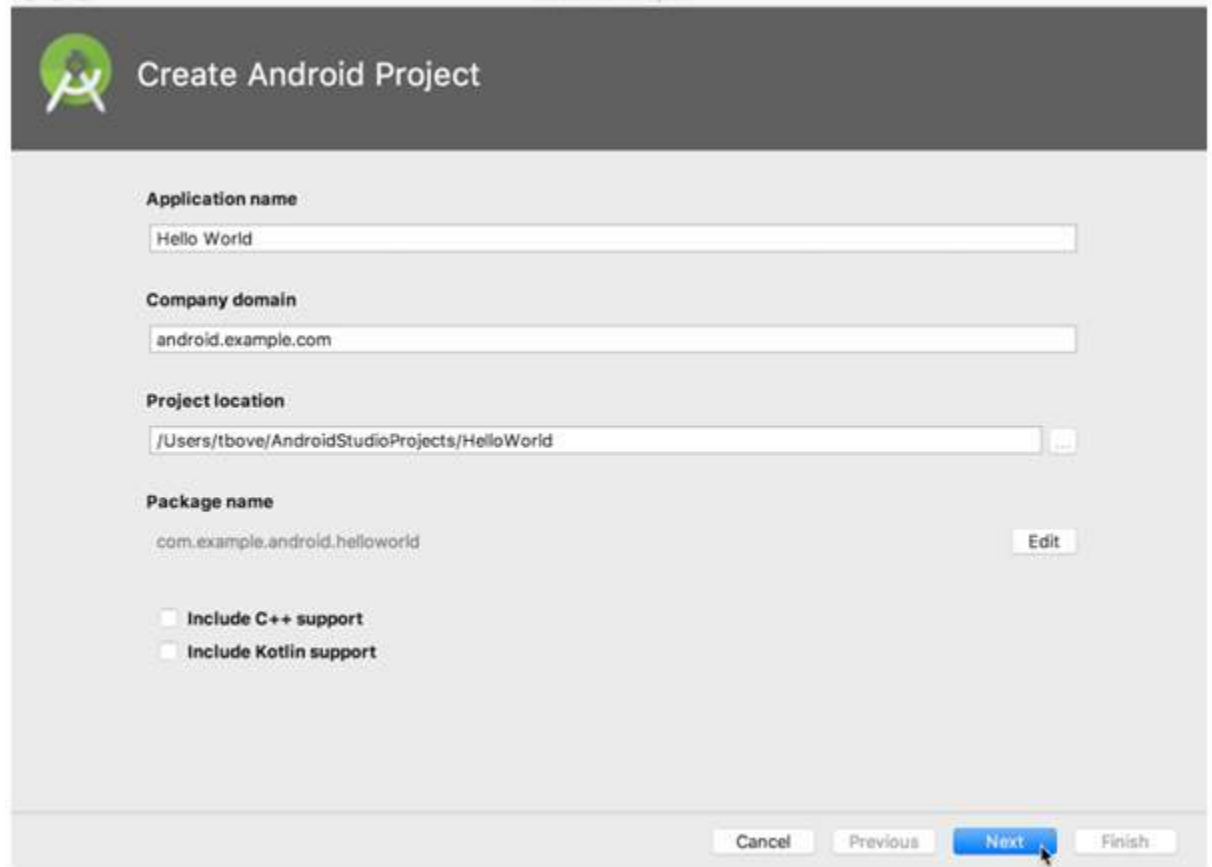
Trong nhiệm vụ này, bạn sẽ tạo ứng dụng hiển thị “Hello World” để xác minh Android Studio đã được cài đặt chính xác và tìm hiểu những điều cơ bản về phát triển Android Studio.

2.1 Tạo dự án ứng dụng

1. Mở Android Studio nếu chưa mở.

2. Trong cửa sổ chính Welcome to Android Studio, ấn Start a new Android Studio project

3. Trong cửa sổ Create Android Project, nhập Hello World cho tên ứng dụng.

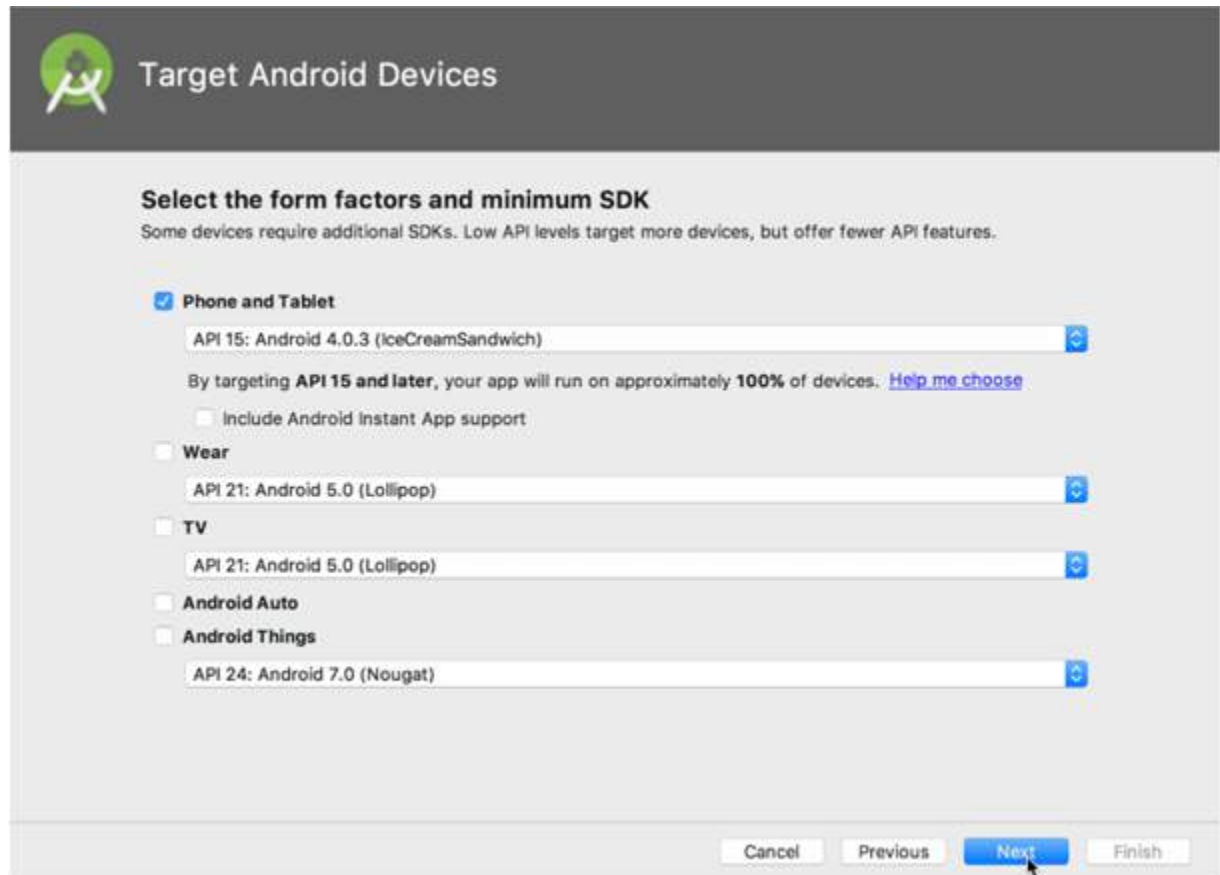


4. Xác minh rằng vị trí Dự án mặc định là nơi bạn muốn lưu trữ ứng dụng Hello World và các dự án Android Studio khác hoặc thay đổi vị trí đó thành thư mục ưa thích của bạn.

5. Chấp nhận android.example.com cho tên miền công ty, hoặc tạo miền công ty riêng. Nếu không có kế hoạch phát hành ứng dụng của mình, bạn có thể chấp nhận mặc định. Lưu ý rằng việc thay đổi tên gói của ứng dụng sau này là một công việc bổ sung.

6. Bỏ chọn các tùy chọn **Include C++ support** và **Include Kotlin support**, và chọn Next.

7. Trên màn hình **Target Android Devices**, nên chọn **Phone and Tablet**. Đảm bảo **API 15: Android 4.0.3 IceCreamSandwich** được đặt làm SDK tối thiểu; Nếu không, hãy sử dụng menu bật lên để thiết lập.

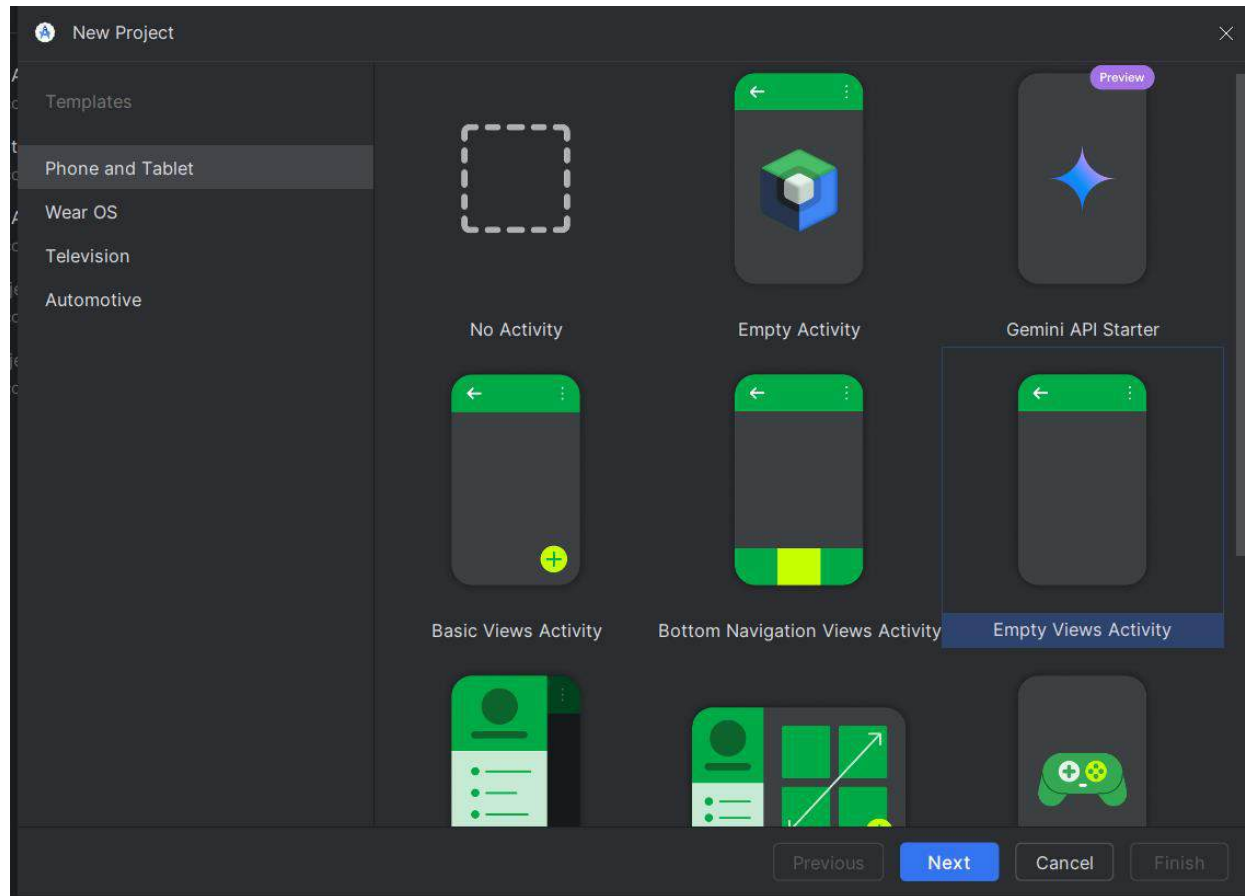


Đây là các cài đặt được sử dụng bởi các ví dụ trong các bài học cho khóa học này. Khi viết bài này, các cài đặt này làm cho ứng dụng Hello World của bạn tương thích với 97% thiết bị Android đang hoạt động trên Cửa hàng Google Play.

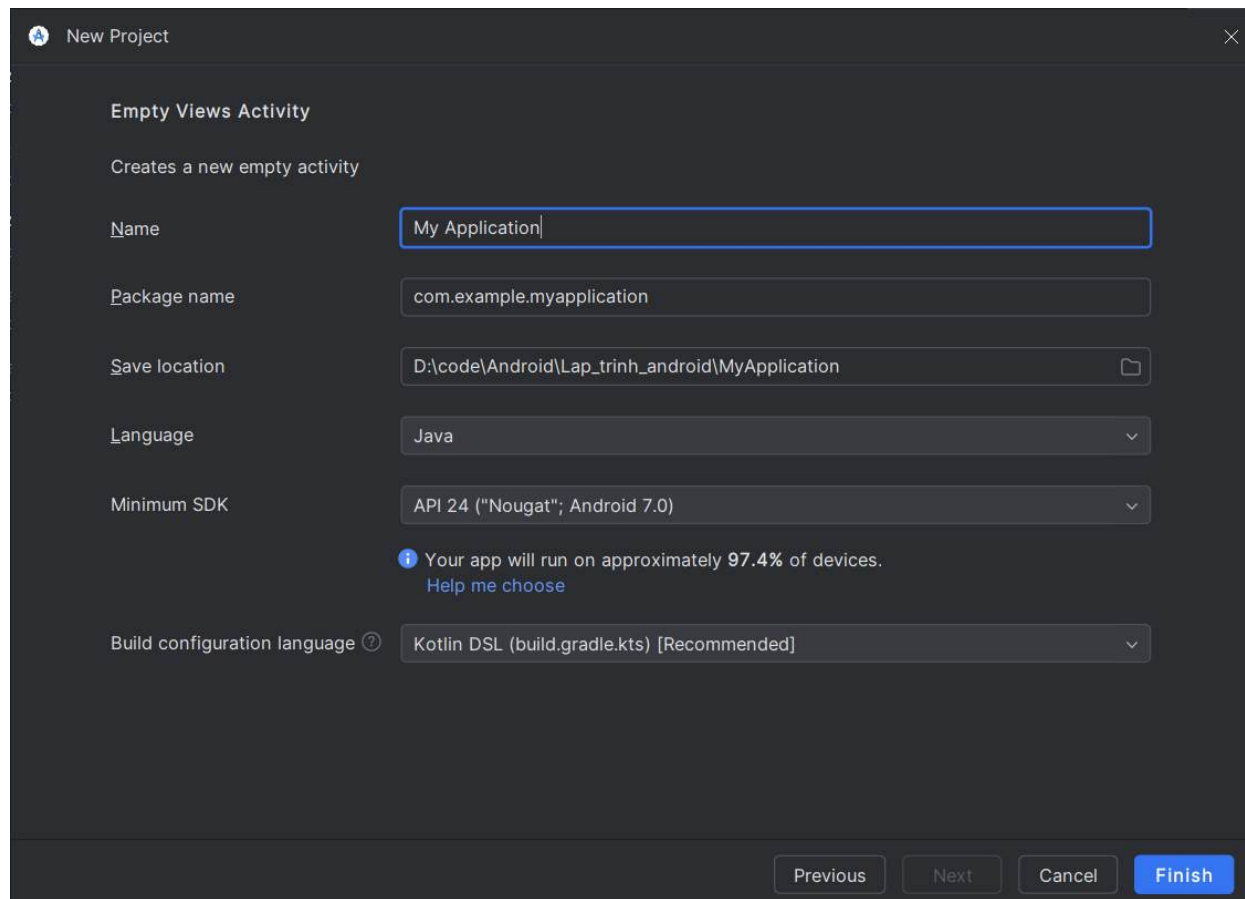
8. Bỏ chọn **Include Instant App support** và tất cả các tùy chọn khác. Sau đó nhấp vào **Next** . Nếu dự án của bạn yêu cầu các thành phần bổ sung cho SDK mục tiêu bạn đã chọn, Android Studio sẽ tự động cài đặt các thành phần đó.

9. Cửa sổ **Add to Activity** xuất hiện. Hoạt động là một việc tập trung duy nhất mà người dùng có thể làm. Nó là một thành phần quan trọng của bất kỳ ứng dụng Android nào. Hoạt động thường có bố cục được liên kết với nó xác định cách các thành phần giao diện người dùng xuất hiện trên màn hình. Android Studio cung cấp các mẫu Hoạt động để giúp bạn bắt đầu. Đối với

dự án Hello World, hãy chọn **Empty Views Activity** như hình dưới đây và nhấp vào **Next** .



10. Màn hình **Configure Activity** xuất hiện (khác nhau tùy thuộc vào mẫu bạn đã chọn ở bước trước). Theo mặc định, Hoạt động trống do mẫu cung cấp có tên là **MainActivity** . Bạn có thể thay đổi điều này nếu muốn, nhưng bài học này sử dụng **MainActivity**.



11. Đảm bảo rằng tùy chọn **Generate Layout file** được chọn. Tên bố cục theo mặc định là `activity_main`. Bạn có thể thay đổi điều này nếu muốn, nhưng bài học này sử dụng `activity_main`.

12. Đảm bảo rằng tùy chọn **Backwards Compatibility (App Compat)** được chọn. Điều này đảm bảo rằng ứng dụng của bạn sẽ tương thích ngược với các phiên bản Android trước đó.

13. Chọn **Finish**.

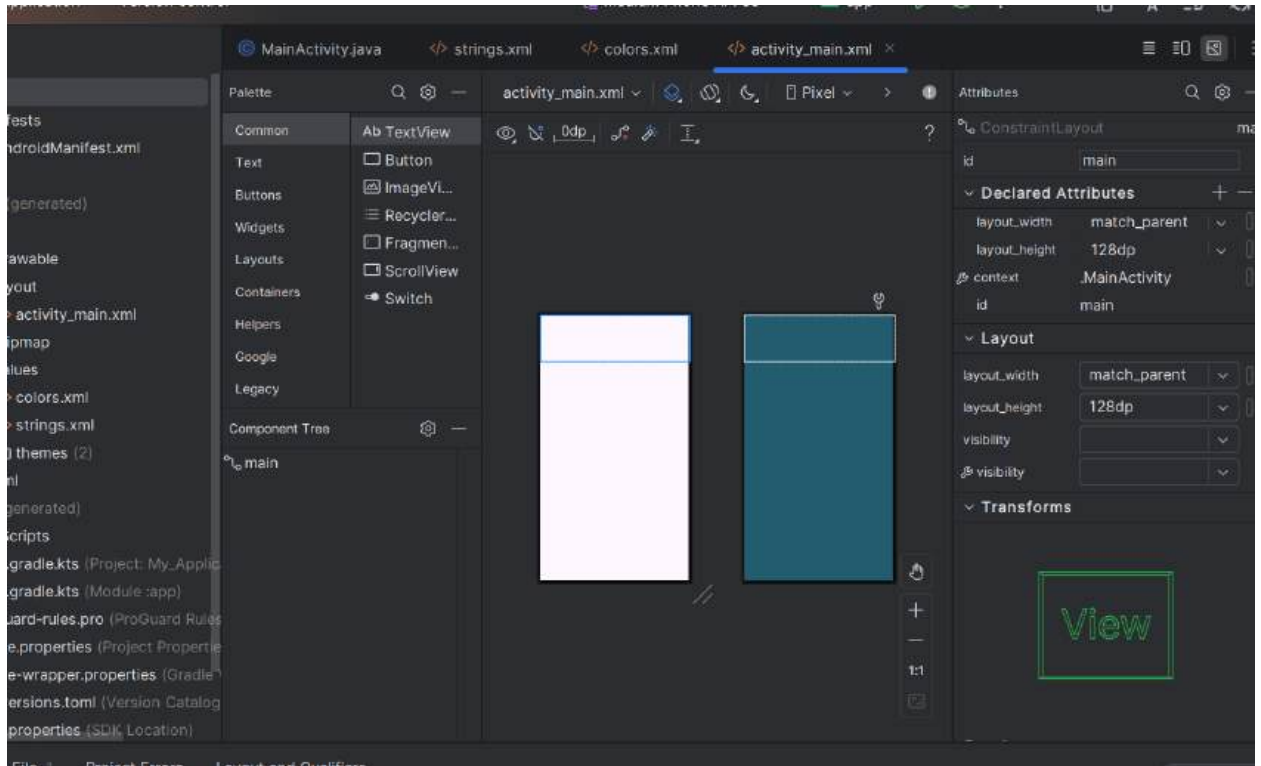
Android Studio sẽ tạo một thư mục cho các dự án của bạn và xây dựng dự án bằng Gradle (quá trình này có thể mất vài phút).

Mẹo: Xem trang [Configure your build](#) để biết thông tin chi tiết.

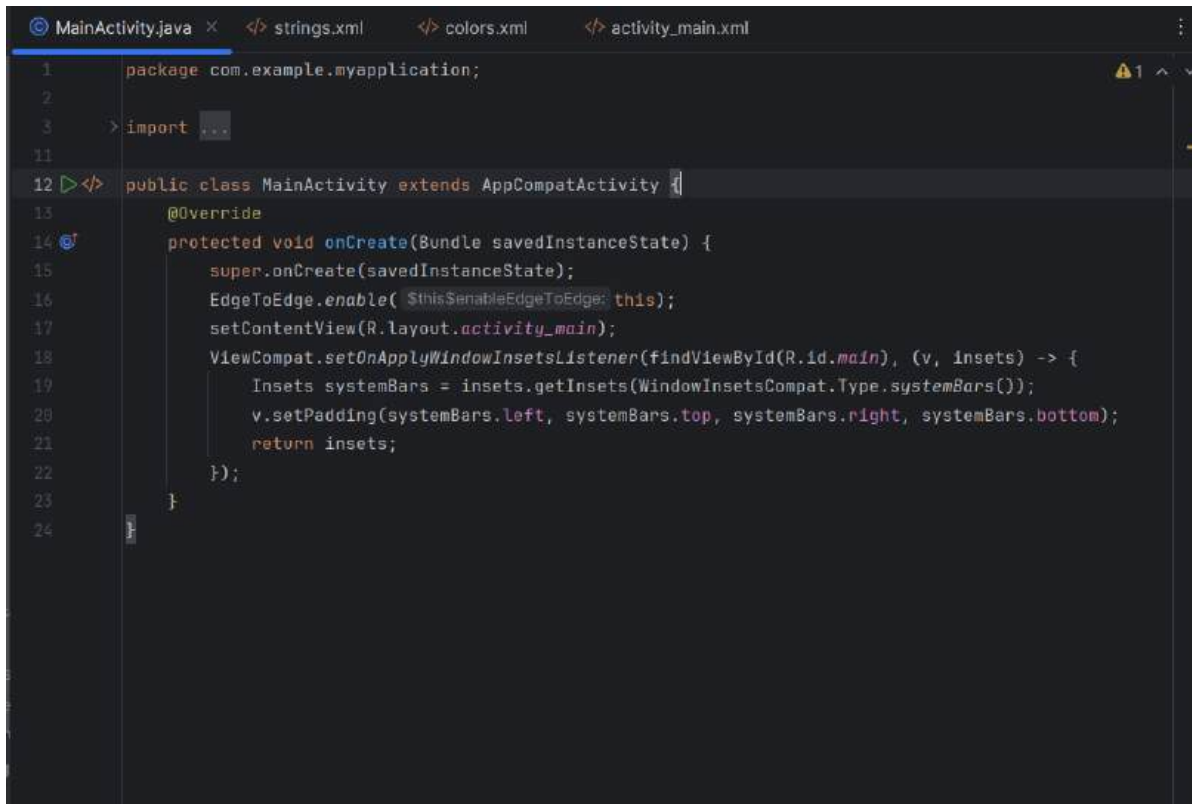
Bạn cũng có thể thấy thông báo "**Tip of the day**" với các phím tắt và các mẹo hữu ích khác. Nhấp vào **Close** để đóng thư.

Trình chỉnh sửa Android Studio sẽ xuất hiện. Làm theo các bước sau:

1. Nhấp vào **activity_main.xml** để xem trình chỉnh sửa bố cục.
2. Nhấp vào **Design** chỉnh sửa bố cục, nếu chưa được chọn, để hiển thị biểu diễn đồ họa của bố cục như hình dưới đây.



3. Nhấp vào **MainActivity.java** để xem trình chỉnh sửa mã như hình bên dưới.

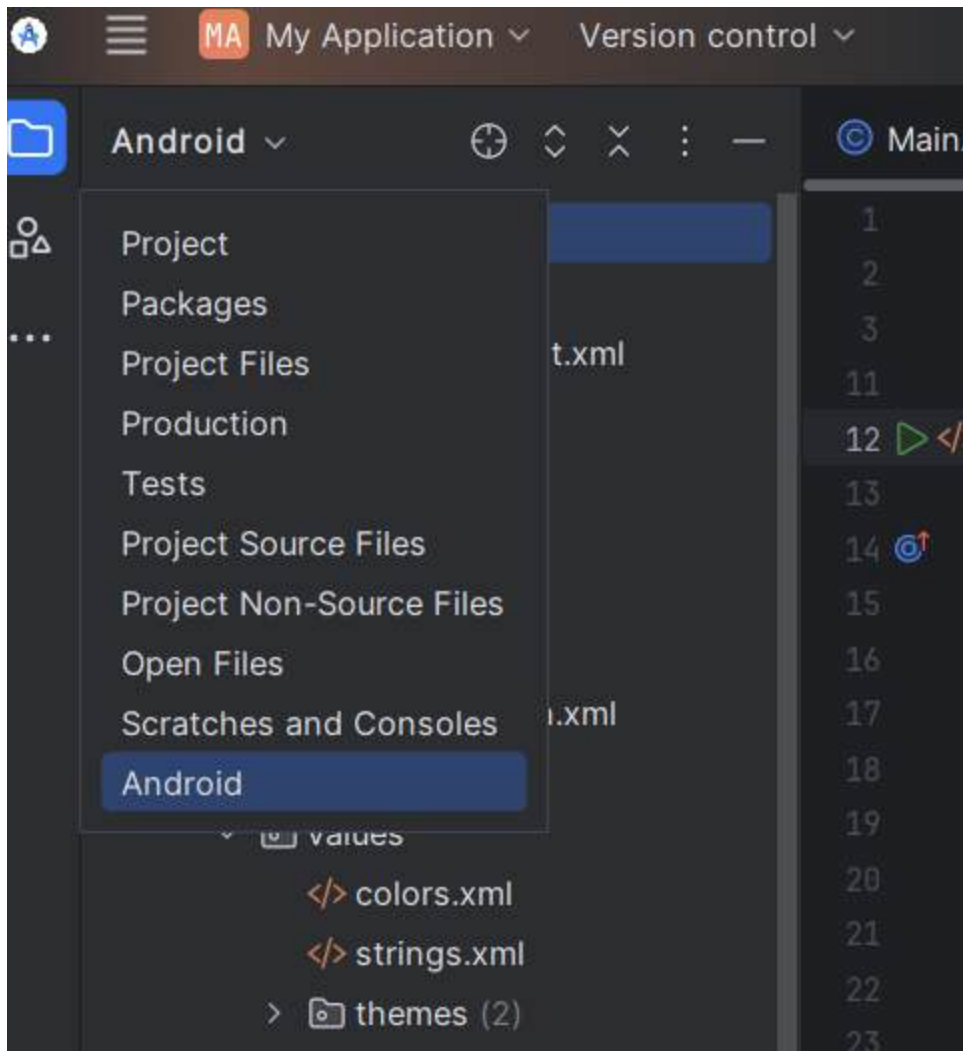


```
1 package com.example.myapplication;
2
3 > import ...
4
5
6
7
8
9
10
11
12 public class MainActivity extends AppCompatActivity {
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         EdgeToEdge.enable(this);
17         setContentView(R.layout.activity_main);
18         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
19             Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
20             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
21             return insets;
22         });
23     }
24 }
```

2.2 Khám phá khung Project > Android

Trong bài này, bạn sẽ khám phá cách tổ chức dự án trong Android Studio.

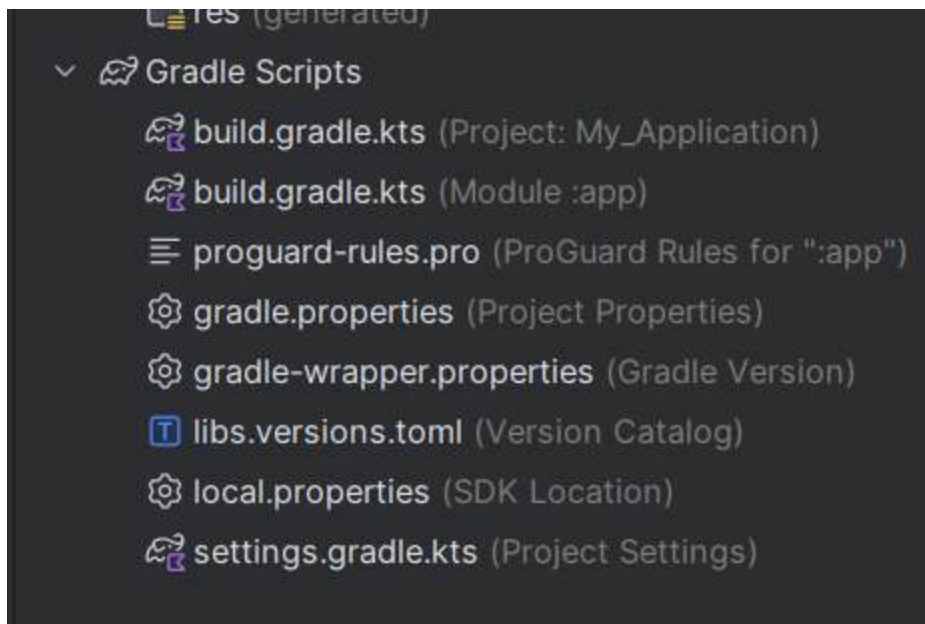
1. Nếu chưa chọn, hãy nhấp vào **Project** trong cột tab dọc ở bên trái cửa sổ Android Studio. Ngăn Dự án xuất hiện.
2. Để xem dự án trong hệ thống phân cấp dự án Android tiêu chuẩn, hãy chọn **Android** từ menu bật lên ở đầu ngăn Dự án, như được hiển thị bên dưới.



2.3 Khám phá thư mục Tập lệnh Gradle

Hệ thống bản dựng Gradle trong Android Studio giúp bạn dễ dàng đưa các tệp nhị phân bên ngoài hoặc các mô-đun thư viện khác vào bản dựng dưới dạng phần phụ thuộc.

Khi bạn tạo dự án ứng dụng lần đầu tiên, ngăn **Project > Android** sẽ xuất hiện với thư mục **Gradle Scripts** được mở rộng như hình bên dưới.



Làm theo các bước sau để khám phá hệ thống Gradle:

1. Nếu thư mục **Gradle Scripts** không được mở rộng, hãy nhấp vào hình tam giác để mở rộng thư mục đó. Thư mục này chứa tất cả các tệp cần thiết cho hệ thống xây dựng.

2. Tìm tệp **build.gradle(Project: HelloWorld)**.

Đây là nơi bạn sẽ tìm thấy các tùy chọn cấu hình chung cho tất cả các mô-đun tạo nên dự án của bạn. Mỗi dự án Android Studio đều chứa một tệp bản dựng Gradle cấp cao nhất. Hầu hết thời gian, bạn sẽ không cần thực hiện bất kỳ thay đổi nào đối với tệp này, nhưng vẫn hữu ích khi hiểu nội dung của nó.

Theo mặc định, tệp bản dựng cấp cao nhất sử dụng khối buildscript để xác định kho lưu trữ Gradle và các phần phụ thuộc chung cho tất cả các mô-đun trong dự án. Khi phần phụ thuộc của bạn không phải là thư viện cục bộ hoặc cây tệp, Gradle sẽ tìm kiếm các tệp trong bất kỳ kho lưu trữ trực tuyến nào được chỉ định trong khối kho lưu trữ của tệp này. Theo mặc định, các dự án Android Studio mới khai báo JCenter và Google (bao gồm kho lưu trữ Google Maven) là vị trí kho lưu trữ:

3. Tìm tệp **build.gradle(Module:app)**.

Ngoài tệp build.gradle cấp dự án, mỗi mô-đun có một tệp build.gradle riêng cho phép bạn định cấu hình cài đặt bản dựng cho từng mô-đun cụ thể (ứng dụng HelloWorld chỉ có một mô-đun). Việc định cấu hình các tùy chọn cài đặt bản dựng

này cho phép bạn cung cấp các tùy chọn đóng gói tùy chỉnh, chẳng hạn như các loại bản dựng bổ sung và hương vị sản phẩm. Bạn cũng có thể ghi đè cài đặt trong tệp `AndroidManifest.xml` hoặc tệp `build.gradle` cao cấp.

Tệp này thường là tệp cần chỉnh sửa khi thay đổi cấu hình cấp ứng dụng, chẳng hạn như khai báo các phần phụ thuộc trong phần phần phụ thuộc. Bạn có thể khai báo phần phụ thuộc thư viện bằng cách sử dụng một trong một số cấu hình phần phụ thuộc khác nhau. Mỗi cấu hình phần phụ thuộc cung cấp cho Gradle các hướng dẫn khác nhau về cách sử dụng thư viện. Ví dụ: câu lệnh thực hiện `fileTree(dir: 'libs', include: ['*.jar'])` thêm phần phụ thuộc của tất cả các tệp ".jar" bên trong thư mục `libs`.

Sau đây là tệp **`build.gradle(Module:app)`** cho ứng dụng HelloWorld:

```
1  plugins {
2      alias(libs.plugins.android.application)
3  }
4
5  android {
6      namespace = "com.example.myapplication"
7      compileSdk = 35
8
9      defaultConfig {
10         applicationId = "com.example.myapplication"
11         minSdk = 24
12         targetSdk = 35
13         versionCode = 1
14         versionName = "1.0"
15
16         testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
17     }
18
19     buildTypes {
20         release {
21             isMinifyEnabled = false
22             proguardFiles(
23                 getDefaultProguardFile("name: "proguard-android-optimize.txt"),
24                 "proguard-rules.pro"
25             )
26         }
27     }
28 }
```



```

26     }
27 }
28 compileOptions {
29     sourceCompatibility = JavaVersion.VERSION_11
30     targetCompatibility = JavaVersion.VERSION_11
31 }
32 }
33
34 dependencies {
35
36     implementation(libs.appcompat)
37     implementation(libs.material)
38     implementation(libs.activity)
39     implementation(libs.constraintlayout)
40     testImplementation(libs.junit)
41     androidTestImplementation(libs.ext.junit)
42     androidTestImplementation(libs.espresso.core)
43 }

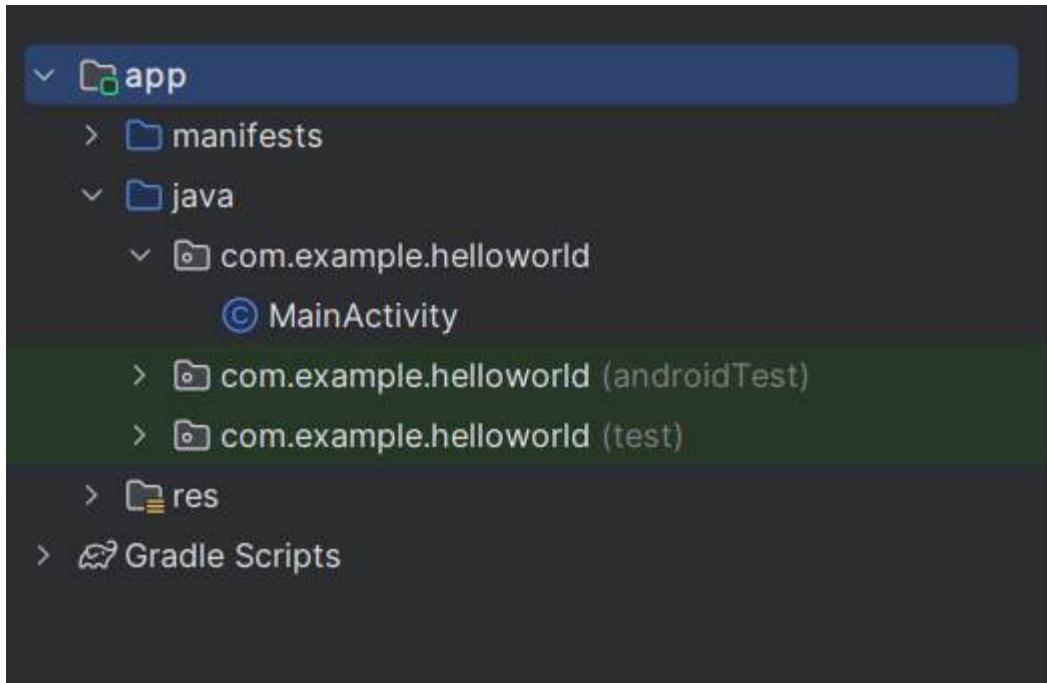
```

4. Nhấp vào hình tam giác để đóng **Gradle Scripts**.

2.4 Khám phá ứng dụng và thư mục res

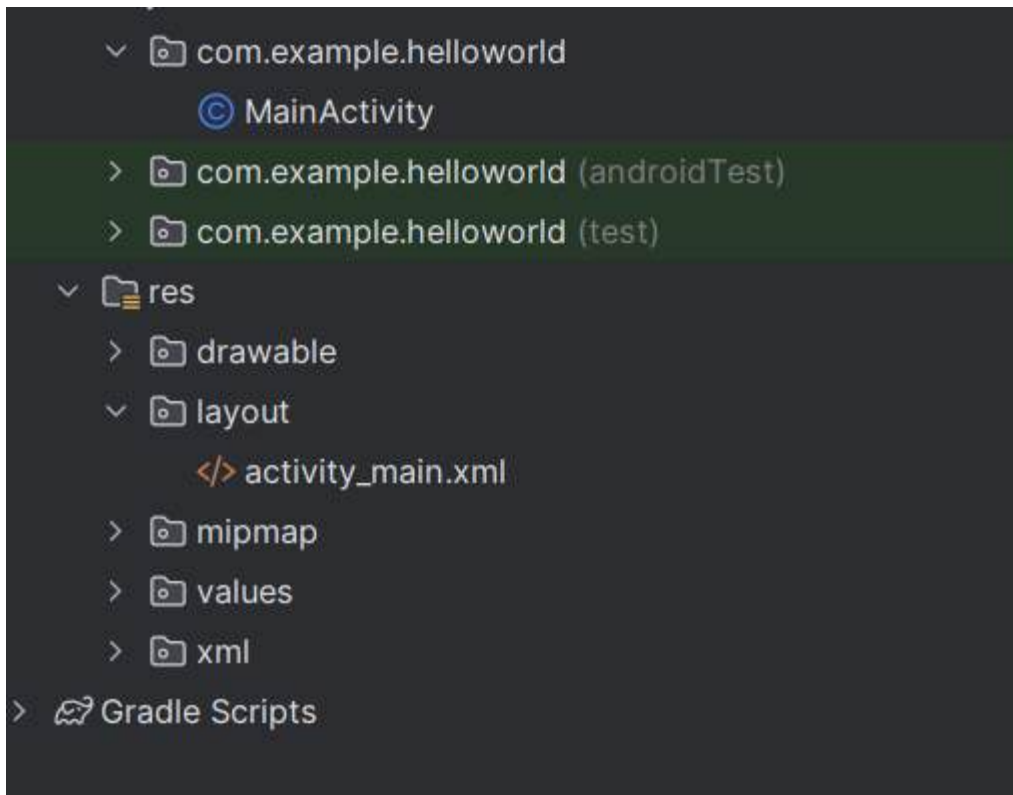
Tất cả mã và tài nguyên cho ứng dụng nằm trong thư mục app và res

1. Mở rộng thư mục **app**, thư mục **java**, thư mục **com.example.android.helloworld** để xem tệp MainActivity. Nhấn đúp chuột vào tệp sẽ mở tệp đó trong trình soạn thảo.



Thư mục java bao gồm các tệp lớp Java trong ba thư mục con, như thể hiện trong hình trên. Thư mục **com.example.helloworld** chứa tất cả các tệp cho một gói ứng dụng. Hai thư mục còn lại dùng để kiểm tra và sẽ được mô tả trong bài học khác. Đối với ứng dụng Hello World, chỉ có một gói và nó chứa tệp MainActivity.java. Tên của màn hình đầu tiên mà người dùng nhìn thấy, cũng là khởi tạo tài nguyên. Trên toàn ứng dụng, thường gọi là MainActivity (phần mở rộng tệp bị bỏ trong ngăn **Project > Android**).

2. Mở rộng thư mục **res** và thư mục **layout**, và nhấn đúp chuột vào tệp **activity_main.xml** để mở nó trong trình chỉnh sửa bố cục.



Thư mục **res** chứa các tài nguyên, chẳng hạn như bố cục, chuỗi và hình ảnh.

Hoạt động thường được liên kết với bố cục của các chế độ xem giao diện người dùng được xác định dưới dạng tệp xml. Tệp này thường được đặt tên theo hoạt động của nó.

2.5 Khám phá thư mục manifests

Thư mục kê khai chứa các tệp cung cấp thông tin cần thiết về ứng dụng của bạn cho hệ thống Android, hệ thống phải có thông tin này trước khi có thể chạy bất kỳ mã nào của ứng dụng.

1. Mở rộng thư mục **manifests**
2. Mở tệp **AndroidManifests.xml**

Tệp **AndroidManifests.xml** sẽ mô tả tất cả các thành phần của ứng dụng Android của bạn.

Tất cả các thành phần cho ứng dụng, chẳng hạn như mỗi Activity, đều phải khai báo trong tệp XML này. Trong các bài học này, bạn sẽ sửa đổi tệp này để thêm các tính năng và quyền tính năng.

Nhiệm vụ 3: Sử dụng máy ảo (trình giả lập)

Trong nhiệm vụ này, bạn sẽ sử dụng **Android Virtual Device(AVD) manager** để tạo máy ảo(còn được gọi là trình giả lập) mô phỏng cấu hình cho một loại thiết bị android cụ thể và sử dụng nó để chạy ứng dụng. Lưu ý rằng trình mô phỏng Android có các yêu cầu bổ sung ngoài các yêu cầu hệ thống cơ bản đối với Android Studio.

Sử dụng AVD manager, bạn xác định các đặc điểm phần cứng của thiết bị, cấp độ API, bộ nhớ, giao diện và các thuộc tính khác và lưu nó dưới dạng thiết bị ảo. Với thiết bị ảo, bạn có thể kiểm tra ứng dụng trên các cấu hình thiết bị khác nhau (chẳng hạn như máy tính bảng và điện thoại) với các cấp độ API khác nhau mà không cần phải sử dụng thiết bị vật lý.

3.1 Tạo thiết bị ảo Android (AVD)

Để chạy trình mô phỏng trên máy tính, bạn phải tạo một cấu hình mô tả thiết bị ảo.

1.Trong Android Studio, chọn **Tools > Android > AVD Manager**, hoặc nhấn vào AVD



Manager có biểu tượng trên thanh công cụ. Màn hình thiết bị ảo sẽ xuất hiện . Nếu bạn đã tạo thiết bị ảo, màn hình sẽ hiển thị chúng (như trong hình bên dưới); nếu không, bạn sẽ thấy một danh sách trống.

2. Nhấp vào + **Create Virtual Device**. Cửa sổ **Select Hardware** xuất hiện hiển thị danh sách các thiết bị phần cứng được định cấu hình sẵn. Đối với mỗi thiết bị, bảng cung cấp một cột cho kích thước hiển thị đường chéo (**Size**), độ phân giải màn hình tính bằng pixel (**Resolution**) và mật độ điểm ảnh (**Density**).

3. Chọn một thiết bị như **Nexus 5x** hoặc **Pixel XL** và nhấp vào **Next**. Màn hình Hình ảnh hệ thống xuất hiện.

4. Nhấp vào **Recommended** nếu nó chưa được chọn và chọn phiên bản hệ thống Android để chạy trên thiết bị ảo (chẳng hạn như **Oreo**).


Có nhiều phiên bản hơn được hiển thị trong **Recommended**. Nhìn vào tab **x86 Images** và **Other Images** để xem chúng.

Nếu liên kết Tải xuống hiển thị bên cạnh hình ảnh hệ thống bạn muốn sử dụng, thì liên kết đó chưa được cài đặt. Nhấp vào liên kết để bắt đầu tải xuống và nhấp vào Kết thúc khi hoàn tất.

5. Sau khi chọn hình ảnh hệ thống, hãy nhấp vào **Next**. Cửa sổ **Android Virtual Device (AVD)** sẽ xuất hiện. Bạn cũng có thể thay đổi tên của AVD. Kiểm tra cấu hình của bạn và nhấp vào **Finish**.

3.2 Chạy ứng dụng trên thiết bị ảo

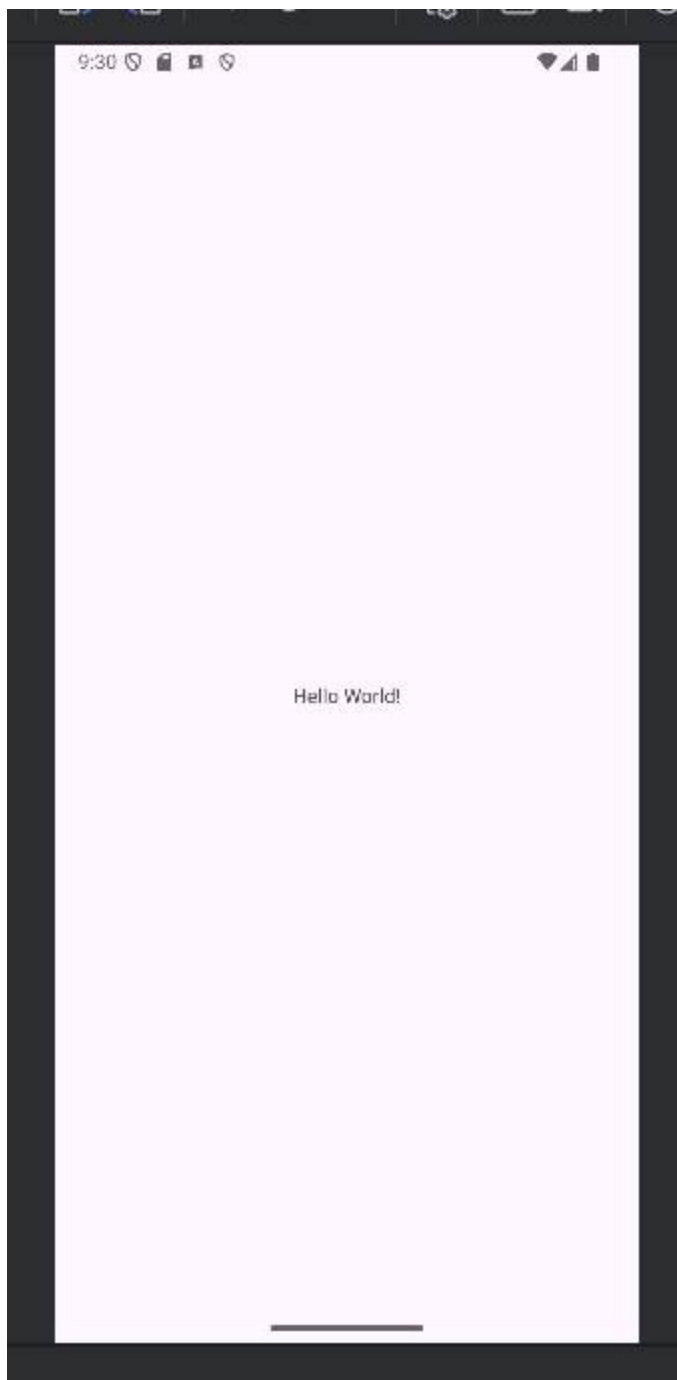
Trong nhiệm vụ này, cuối cùng bạn sẽ chạy ứng dụng Hello World của mình.

1. Trong Android Studio, chọn **Run > Run app** hoặc nhấp vào biểu tượng  trên thanh công cụ.

2. Cửa sổ **Select Deployment Target**, bên dưới **Available Virtual Devices** chọn thiết bị máy ảo bạn vừa tạo và ấn **OK**.

Trình giả lập khởi động và khởi động giống như một thiết bị vật lý. Tùy thuộc vào tốc độ máy tính của bạn, quá trình này có thể mất một lúc. Ứng dụng của bạn sẽ được xây dựng và sau khi trình mô phỏng đã sẵn sàng, Android Studio sẽ tải ứng dụng lên trình mô phỏng và chạy ứng dụng đó.

Bạn sẽ thấy ứng dụng Hello World như trong hình sau:



Mẹo: Khi thử nghiệm trên một thiết bị ảo, bạn nên khởi động nó một lần, ngay từ đầu phiên của bạn. Bạn không nên đóng ứng dụng cho đến khi hoàn tất việc kiểm tra ứng dụng của mình để ứng dụng của bạn không phải trải qua quá trình khởi động thiết bị một lần nữa. Để đóng thiết bị ảo, hãy nhấp vào nút **X** ở đầu trình giả lập, chọn **Quit** từ menu hoặc nhấn **Control-Q** trong Windows hoặc **Command-Q** trong macOS.

Nhiệm vụ 4: (Tùy chọn) sử dụng thiết bị vật lý

Trong nhiệm vụ cuối cùng này, bạn sẽ chạy ứng dụng của mình trên thiết bị di động vật lý như điện thoại hoặc máy tính bảng. Bạn phải luôn kiểm tra ứng dụng của mình trên cả thiết bị ảo và thiết bị vật lý.

Những gì bạn cần:

- Thiết bị Android như điện thoại hoặc máy tính bảng.
- Cáp dữ liệu để kết nối thiết bị Android với máy tính qua cổng USB.
- Nếu bạn đang sử dụng hệ thống Linux hoặc Windows, bạn có thể cần thực hiện các bước bổ sung để chạy trên thiết bị phần cứng. Kiểm tra tài liệu Sử dụng thiết bị phần cứng. Bạn cũng có thể cần cài đặt trình điều khiển USB thích hợp cho thiết bị của mình. Đối với trình điều khiển USB dựa trên Windows, hãy xem Trình điều khiển USB OEM .

4.1 Bật gỡ lỗi USB

Để cho phép Android Studio giao tiếp với thiết bị của bạn, bạn phải bật tính năng Gỡ lỗi USB trên thiết bị Android của mình. Tính năng này được bật trong cài đặt **Developer options** trên thiết bị của bạn.


Trên Android 4.2 trở lên, màn hình **Developer options** bị ẩn theo mặc định. Để hiển thị các tùy chọn dành cho nhà phát triển và bật Gỡ lỗi USB:

1. Trên thiết bị của bạn, mở **Settings**, tìm kiếm **About phone**, nhấp vào **About phone** và nhấn vào **Build number** bảy lần.
2. Quay lại màn hình trước đó (**Settings** / **System**). **Developer options** xuất hiện trong danh sách. Nhấn vào **Developer options**.
3. Chọn **USB Debugging**.

4.2 Chạy ứng dụng của bạn trên thiết bị

Giờ đây bạn có thể kết nối điện thoại của mình và chạy ứng dụng.

1. Kết nối thiết bị của bạn với máy phát triển bằng cáp USB.

2. Nhấn **Run** ở nút  trên thanh công cụ. Cửa sổ **Select Deployment Target** sẽ mở với danh sách các trình giả lập có sẵn và thiết bị kết nối.

3. Chọn thiết bị của bạn và ấn **OK**. Android Studio sẽ tải và chạy ứng dụng trên thiết bị của bạn.

Khắc phục sự cố

Nếu Android Studio không nhận dạng được thiết bị của bạn, hãy thử các cách sau:

1. Rút và cắm lại thiết bị của bạn
2. Khởi động lại Android Studio

Nếu máy của bạn vẫn không tìm thấy thiết bị hoặc báo là “Không được phép”, hãy làm theo các bước:

1. Rút phích cắm của thiết bị.
2. Trên thiết bị, mở **Developer Options in Settings app**.
3. Nhấn vào nút **USB Debugging**
4. Khi được nhắc, hãy cấp ủy quyền.

Nhiệm vụ 5: Thay đổi cấu hình Gradle của ứng dụng

Trong tác vụ này, bạn sẽ thay đổi điều gì đó về cấu hình ứng dụng trong tệp `build.gradle(Module:app)` để tìm hiểu cách thực hiện các thay đổi và đồng bộ hóa chúng với dự án Android Studio của bạn.

5.1 Thay đổi phiên bản SDK tối thiểu cho ứng dụng

Hãy làm theo các bước:

1. Mở rộng thư mục **Gradle Scripts** nếu chưa được mở, và nhấn đúp chuột vào tệp **build.gradle(Module:app)**
Nội dung của tệp xuất hiện trong trình soạn thảo mã.
2. Trong khối `defaultConfig`, hãy thay đổi giá trị của `minSdkVersion` thành 17 như hình dưới đây (ban đầu giá trị này được đặt thành 15)


```

android {
    namespace = "com.example.helloworld"
    compileSdk = 35

    defaultConfig {
        applicationId = "com.example.helloworld"
        minSdk = 24
        targetSdk = 35
        versionCode = 1
        versionName = "1.0"
    }
}

```

5.2 Đồng bộ hóa cấu hình Gradle mới

Khi bạn thực hiện các thay đổi đối với các tệp cấu hình bản dựng trong một dự án, Android Studio yêu cầu bạn đồng bộ hóa các tệp dự án để có thể nhập các thay đổi về cấu hình bản dựng và chạy một số kiểm tra để đảm bảo cấu hình sẽ không tạo ra lỗi bản dựng.

Để đồng bộ hóa các tệp dự án, hãy bấm vào **Sync Now** trong thanh thông báo xuất hiện khi thực hiện thay đổi (như trong hình trước) hoặc nhấp vào **Sync Project with Gradle**



Files biểu tượng trong thanh công cụ.

Khi quá trình đồng bộ hóa Gradle kết thúc, thông báo Bản dựng Gradle đã hoàn tất sẽ xuất hiện ở góc dưới cùng bên trái của cửa sổ Android Studio.

Nhiệm vụ 6: Thêm câu lệnh nhật ký vào ứng dụng của bạn

Trong nhiệm vụ này, bạn sẽ thêm câu lệnh Nhật ký vào ứng dụng của mình, cụm từ này hiển thị thông báo trong ngăn Logcat. Thông báo nhật ký là một công cụ gỡ lỗi mạnh mẽ mà bạn có thể sử dụng để kiểm tra các giá trị, đường dẫn thực thi và báo cáo ngoại lệ

6.1 Xem ngăn Logcat

Để xem ngăn Logcat, hãy nhấp vào thẻ Logcat ở cuối cửa sổ Android Studio như trong hình bên dưới.

6.2 Thêm câu lệnh nhật ký vào ứng dụng của bạn

Tìm hiểu thêm

Tài liệu Android Studio

- Android Studio download page
- Android Studio release notes
- Meet Android Studio
- Logcat command-line tool
- Android Virtual Device (AVD) manager
- App Manifest Overview
- Configure your build
- Log class
- Create and Manage Virtual Devices

Khác:

- How do I install Java?
- Installing the JDK Software and Setting JAVA_HOME
- Gradle site
- Apache Groovy syntax
- Gradle Wikipedia page

Bài tập về nhà

Xây dựng và chạy ứng dụng

- Tạo một dự án Android mới từ Empty Template.
- Thêm câu lệnh ghi nhật ký cho các cấp độ nhật ký khác nhau trong onCreate() trong hoạt động chính.
- Tạo trình mô phỏng cho thiết bị, nhắm mục tiêu đến bất kỳ phiên bản Android nào bạn thích và chạy ứng dụng.
- Sử dụng tính năng lọc trong Logcat để tìm các câu lệnh nhật ký của bạn và điều chỉnh các cấp độ để chỉ hiển thị các câu lệnh gỡ lỗi hoặc ghi nhật ký lỗi

Trả lời câu hỏi

Câu 1

Tên của tệp bố cục cho hoạt động chính là gì?

- MainActivity.java

Câu 2

Tên của tài nguyên chuỗi chỉ định tên của ứng dụng là gì?

- app_name

Câu 3

Bạn sử dụng công cụ nào để tạo trình giả lập mới?

- AVD Manager

Câu 4

Giả sử rằng ứng dụng của bạn bao gồm câu lệnh ghi nhật ký sau:

```
Log.i("MainActivity", "MainActivity layout is complete");
```

Bạn thấy câu lệnh "Bố cục MainActivity đã hoàn tất" trong ngăn Logcat nếu trình đơn Cấp nhật ký được đặt thành tùy chọn nào sau đây? (Gợi ý: nhiều câu trả lời là được.)

- Info

1.2) Giao diện người dùng tương tác đầu tiên

Giới thiệu

Giao diện người dùng (UI) xuất hiện trên màn hình của thiết bị Android bao gồm một hệ thống phân cấp các đối tượng được gọi là chế độ **View** — mọi phần tử của màn hình là một Chế độ xem. Lớp View đại diện cho khối xây dựng cơ bản cho tất cả các thành phần giao diện người dùng và lớp cơ sở cho các lớp cung cấp các thành phần giao diện người dùng tương tác như nút, hộp kiểm và trường nhập văn bản. Các lớp con View thường được sử dụng được mô tả trong một số bài học bao gồm:

- **TextView** để hiển thị văn bản
- **EditText** để cho phép người dùng nhập và chỉnh sửa văn bản

- **Button** và các yếu tố có thể nhấp khác (chẳng hạn như **RadioButton**, **CheckBox** và **Spinner**) cung cấp hành vi tương tác
- **ScrollView** và **RecyclerView** để hiển thị các mục có thể cuộn
- **ImageView** để hiển thị hình ảnh
- **ConstraintLayout** và **LinearLayout** để chứa các phần tử View khác và định vị chúng

Mã Java hiển thị và điều khiển giao diện người dùng được chứa trong một lớp mở rộng Activity. Hoạt động thường được liên kết với bố cục của chế độ xem giao diện người dùng được xác định dưới dạng tệp XML (Ngôn ngữ đánh dấu mở rộng). Tệp XML này thường được đặt tên theo Activity của nó và xác định bố cục của các phần tử View trên màn hình.

Ví dụ: mã MainActivity trong ứng dụng Hello World hiển thị bố cục được xác định trong tệp bố cục activity_main.xml, bao gồm TextView với văn bản "Hello World".

Trong các ứng dụng phức tạp hơn, Hoạt động có thể triển khai các hành động để phản hồi các thao tác nhấn của người dùng, vẽ nội dung đồ họa hoặc yêu cầu dữ liệu từ cơ sở dữ liệu hoặc internet. Bạn tìm hiểu thêm về lớp Sinh Hoạt trong một bài học khác.

Trong thực tế này, bạn sẽ tìm hiểu cách tạo ứng dụng tương tác đầu tiên của mình — một ứng dụng cho phép tương tác với người dùng. Bạn tạo một ứng dụng bằng cách sử dụng mẫu Hoạt động trống. Bạn cũng học cách sử dụng trình soạn thảo bố cục để thiết kế bố cục và cách chỉnh sửa bố cục trong XML. Bạn cần phát triển những kỹ năng này để có thể hoàn thành các bài thực hành khác trong khóa học này.

Những điều bạn nên biết

Bạn nên làm quen với:

- Cài đặt và mở Android Studio
- Cách tạo ứng dụng HelloWorld
- Cách chạy chương trình HelloWorld

Những gì bạn sẽ học

- Cách tạo ứng dụng với hành vi tương tác

- **Cách** sử dụng trình chỉnh sửa bố cục để thiết kế bố cục
- Cách chỉnh sửa bố cục trong XML

Bạn sẽ làm những gì

- Tạo một ứng dụng và thêm hai phần tử Button và một TextView vào bố cục.
- Thao tác từng phần tử trong ConstraintLayout để hạn chế chúng ở lề và các phần tử khác.
- Thay đổi thuộc tính thành phần giao diện người dùng.
- Chỉnh sửa bố cục của ứng dụng trong XML.
- Trích xuất các chuỗi được mã hóa cứng vào tài nguyên chuỗi.
- Triển khai các phương thức xử lý nhấp chuột để hiển thị thông báo trên màn hình khi người dùng nhấn vào từng nút

Tổng quan về ứng dụng

Ứng dụng HelloToast bao gồm hai phần tử Button và một TextView . Khi người dùng nhấn vào Nút đầu tiên, nó sẽ hiển thị một thông báo ngắn (Toast) trên màn hình. Nhấn vào Nút thứ hai sẽ tăng bộ đếm "nhấp chuột" được hiển thị trong TextView , bắt đầu từ không.

Đây là những gì ứng dụng đã hoàn thành trông như thế nào:

1.3) Bố cục


Tác vụ 1: Tạo biến thể bố cục

Trong bài học trước, thử thách mã hóa yêu cầu thay đổi bố cục của ứng dụng Hello Toast để nó vừa vặn với hướng ngang hoặc dọc. Trong nhiệm vụ này, bạn sẽ học một cách dễ dàng hơn để tạo các biến thể bố cục của bạn cho hướng ngang (còn được gọi là ngang) và dọc (còn được gọi là dọc) cho điện thoại và cho màn hình lớn hơn như máy tính bảng.

Trong nhiệm vụ này, bạn sẽ sử dụng một số nút trong hai thanh công cụ trên cùng của trình soạn thảo bố cục. Thanh công cụ trên cùng cho phép bạn định cấu hình giao diện của bản xem trước bố cục trong trình chỉnh sửa bố cục:

1.1 Xem trước bố cục theo hướng ngang

Để xem trước bố cục ứng dụng Hello Toast theo hướng ngang, hãy làm theo các bước sau:


1. Mở ứng dụng Hello Toast từ bài học trước.
2. Mở tệp bố cục `activity_main.xml`. Nhấn Design nếu chưa chọn
3. Nhấn vào nút **Orientation in Editor**  trên thanh công cụ.
4. Chọn **Switch to Landscape** trong menu thả xuống . Bố cục xuất hiện theo hướng ngang như hình dưới đây. Để trở lại hướng dọc, chọn **Switch to Portrait**.

1.2 Tạo biến thể bố cục cho chiều ngang

Sự khác biệt trực quan giữa hướng dọc và chiều ngang cho bố cục này là chữ số (0) trong phần tử `TextView show_count` quá thấp so với hướng ngang — quá gần với nút **Count** .

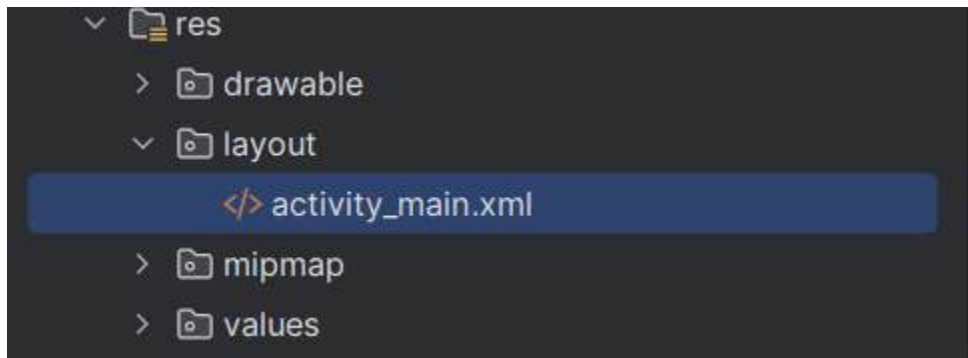
Tùy thuộc vào thiết bị hoặc trình mô phỏng bạn sử dụng, phần tử `TextView` có thể xuất hiện quá lớn hoặc không ở giữa vì kích thước văn bản được cố định ở mức 160sp.

Để khắc phục điều này đối với các hướng ngang trong khi vẫn giữ nguyên hướng dọc, bạn có thể tạo biến thể của bố cục ứng dụng Hello Toast khác với hướng ngang. Làm theo các bước sau:

1. Nhấn vào **Orientation in Editor** nút  phía trên thanh công cụ.
2. Chọn **Create Landscape Variation**.

Một cửa sổ trình chỉnh sửa mới mở ra với tab **land / activity_main.xml** hiển thị bố cục cho hướng ngang (ngang). Bạn có thể thay đổi bố cục này, dành riêng cho hướng ngang, mà không cần thay đổi hướng dọc (dọc) ban đầu

3. Trong ngăn **Project > Android**, hãy nhìn vào bên trong thư mục `res > layout` và bạn sẽ thấy rằng Android Studio đã tự động tạo biến thể cho bạn, được gọi là **activity_main.xml**.



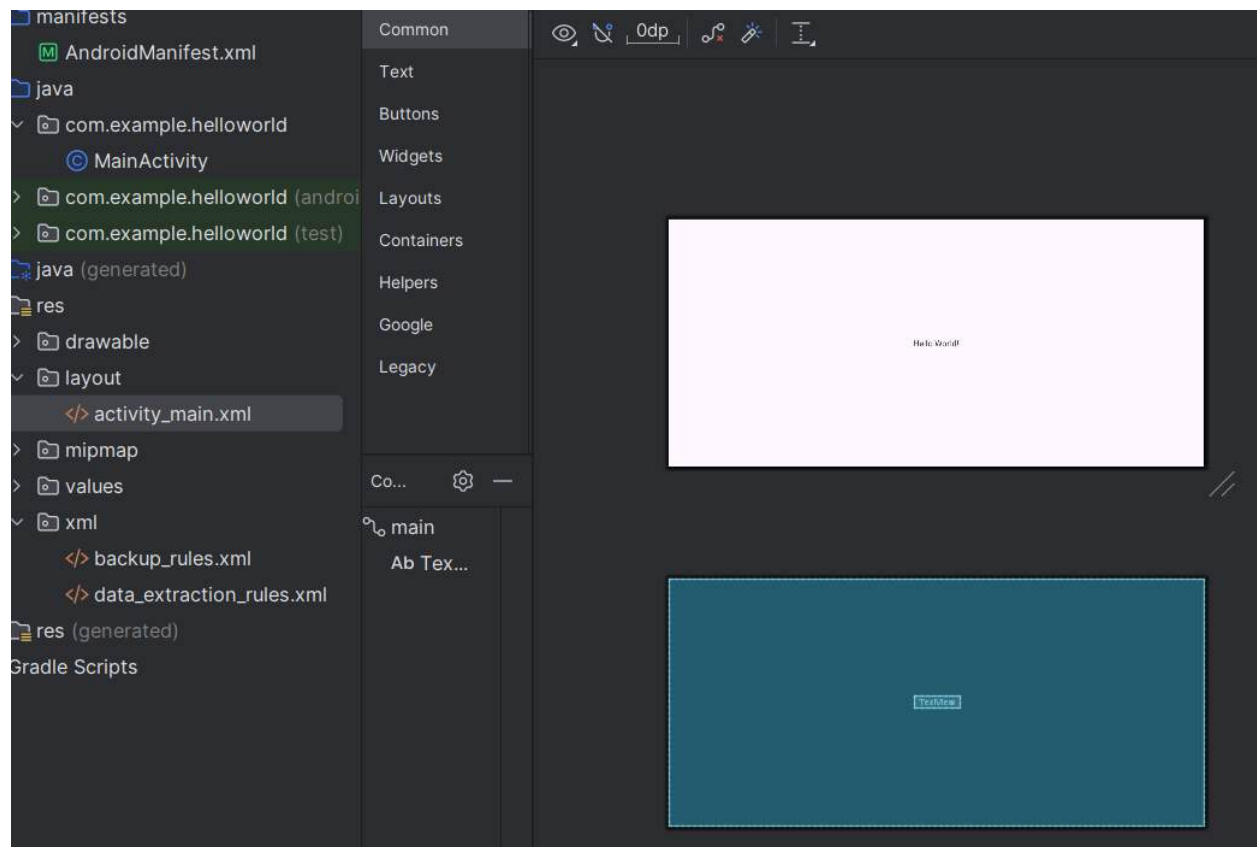
1.3 Xem trước bố cục cho các thiết bị khác nhau

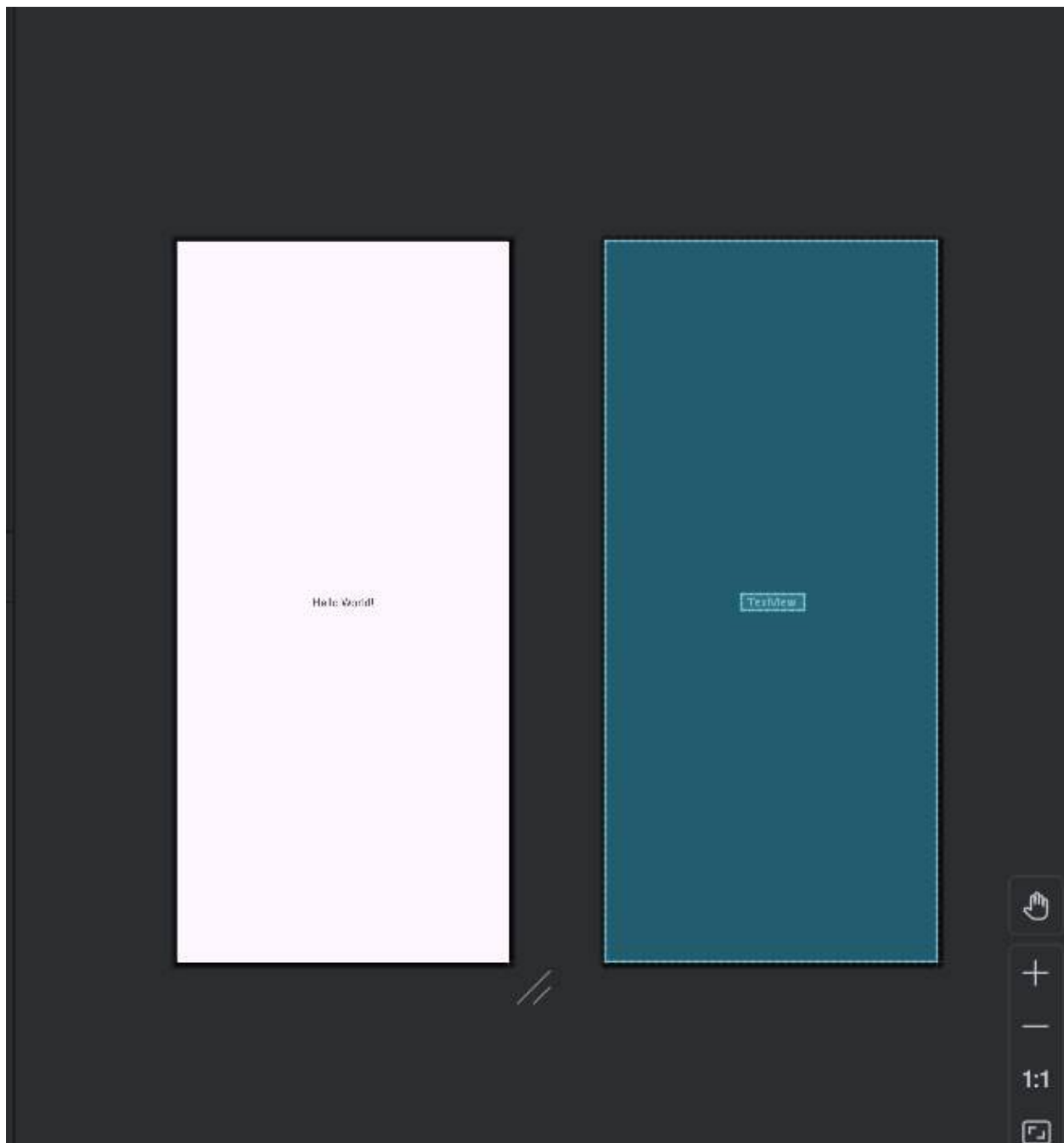
Bạn có thể xem trước bố cục cho các thiết bị khác nhau mà không cần phải chạy ứng dụng trên thiết bị hoặc trình mô phỏng. Làm theo các bước sau:

1. Trong **project** view, điều hướng đến thư mục **res/layout** và mở tệp layout xml mà bạn muốn xem trước.
2. Bạn sẽ thấy 2 tab **design** và **code** trên cửa sổ layout. Nhấn vào **design** để xem giao diện đồ họa.
3. Ở cửa sổ **design**, nhìn vào thanh bên phải bạn sẽ thấy phần **Preview**.

1.4 Thay đổi bố cục theo hướng ngang

Bạn có thể sử dụng ngăn Thuộc tính trong tab **Design** để đặt hoặc thay đổi thuộc tính, nhưng đôi khi có thể nhanh hơn nếu sử dụng tab **Text** để chỉnh sửa mã XML trực tiếp. Tab **Text** hiển thị mã XML và cung cấp tab **Preview** ở phía bên phải của cửa sổ để hiển thị bản xem trước bố cục, như được hiển thị trong phần bên dưới.



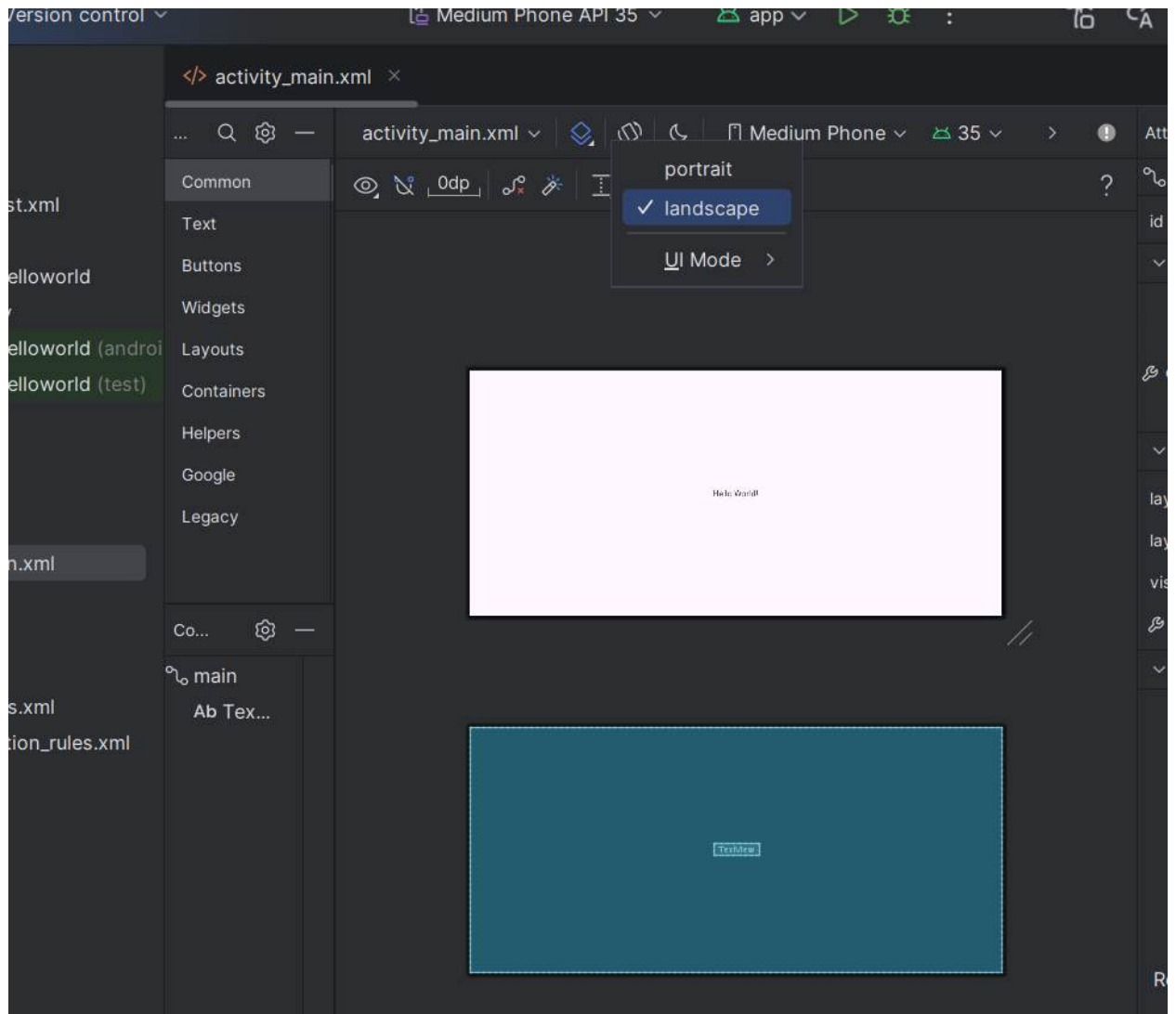


1.5 Tạo bố cục cơ bản tính

1. Nhấn vào tab **Design** (nếu chưa được chọn) để hiển thị các ngăn thiết kế và bảng thiết kế.


2. Nhấn vào nút  trên thanh công cụ.

3. Chọn **landscape**.

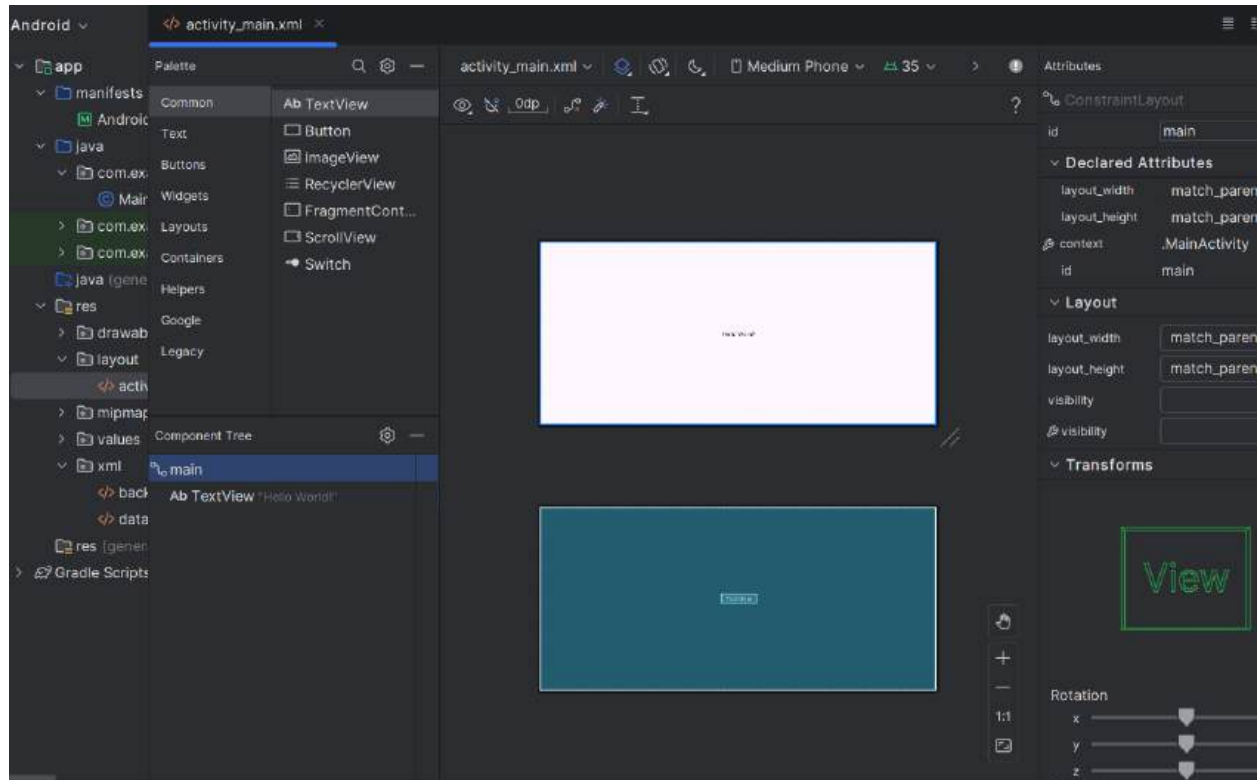


1.6 Thay đổi biến thể bố cục cho máy tính bảng

Bạn có thể sử dụng thuộc tính **Design** để thay đổi bố cục

1. Tắt công cụ Tự động kết nối trên thanh công cụ. Đối với bước này, hãy đảm bảo rằng công cụ đã bị tắt.
2. Xóa tất cả các ràng buộc trong bố cục bằng cách nhấp vào nút  trên thanh công cụ.
Với các ràng buộc được loại bỏ, bạn có thể di chuyển và thay đổi kích thước các phần tử trên bố cục một cách tự do.

3. Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước trên cả bốn góc của một phần tử để thay đổi kích thước của nó. Trong **Component Tree**, chọn TextView được gọi là show_count. Để loại bỏ TextView để bạn có thể tự do kéo các phần tử Button, hãy kéo một góc của nó để thay đổi kích thước, như trong hình động bên dưới.



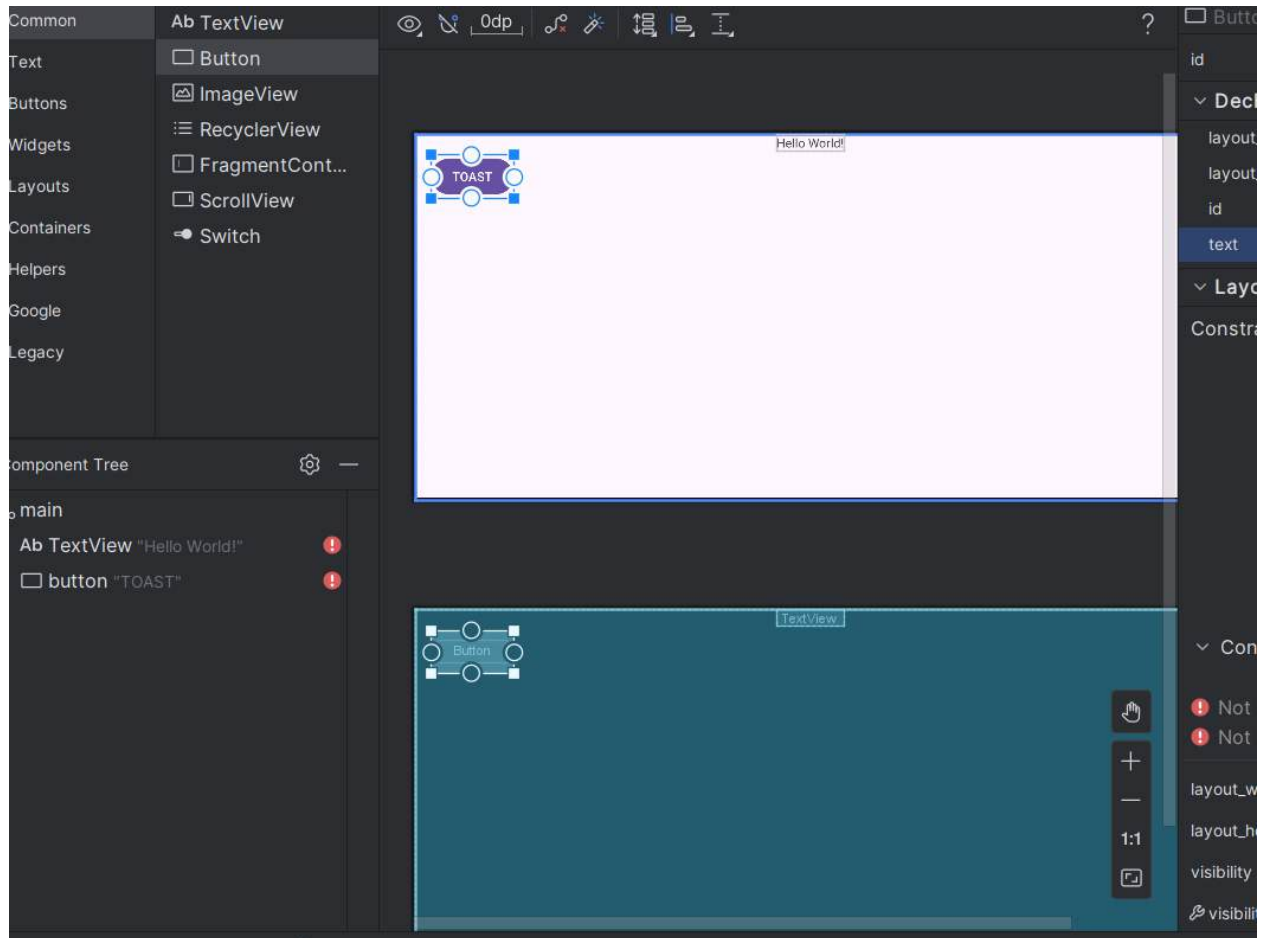
Thay đổi kích thước phần tử mã hóa cứng kích thước chiều rộng và chiều cao. Tránh mã hóa cứng kích thước cho hầu hết các yếu tố, vì bạn không thể dự đoán kích thước được mã hóa cứng sẽ trông như thế nào trên màn hình có kích thước và mật độ khác nhau. Bạn đang làm điều này bây giờ chỉ để di chuyển phần tử ra khỏi đường và bạn sẽ thay đổi kích thước trong một bước khác.

1.7 Sử dụng ràng buộc đường cơ sở

Bạn có thể căn chỉnh một phần tử giao diện người dùng có chứa văn bản, chẳng hạn như TextView hoặc Button với một phần tử giao diện người dùng khác có chứa văn bản. Một ràng buộc đường cơ sở cho phép bạn hạn chế các phần tử để đường cơ sở văn bản khớp nhau.

1. Hạn chế Nút button_toast ở phía trên và bên trái của bố cục, kéo Nút button_count vào một khoảng trống gần Nút button_toast và hạn chế Nút

button_count ở phía bên trái của Nút button_toast , như được hiển thị trong hình ảnh động:



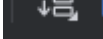
2. Sử dụng ràng buộc đường cơ sở , bạn có thể hạn chế Nút button_count để đường cơ sở văn bản của nó khớp với đường cơ sở văn bản của Nút button_toast . Chọn phần tử button_count, sau đó di con trỏ của bạn lên phần tử cho đến khi nút ràng buộc đường cơ sở xuất hiện bên dưới phần tử.
3. Nhấp vào nút ràng buộc đường cơ sở. Tay cầm cơ sở xuất hiện, nhấp nháy màu xanh lục như trong hình động. Nhấp và kéo một đường ràng buộc đường cơ sở vào đường cơ sở của phần tử button_toast.

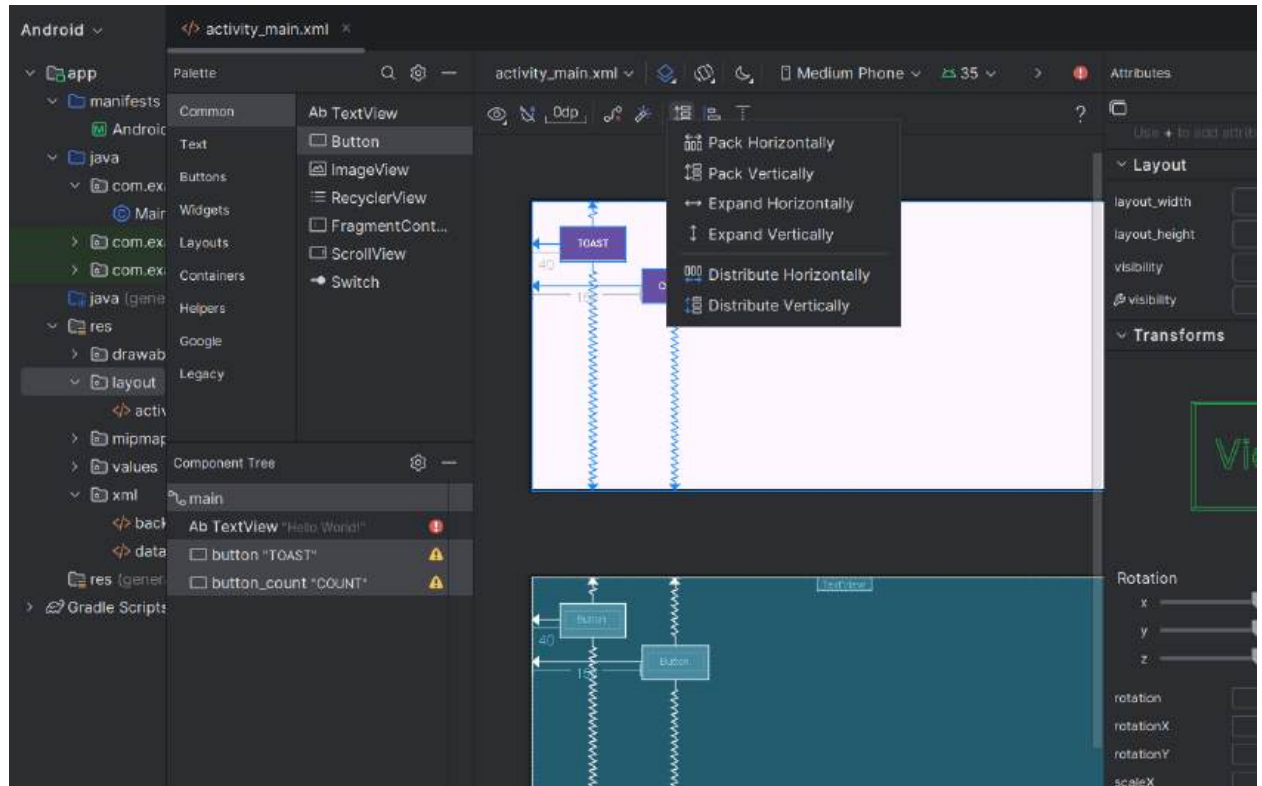
1.8 Mở rộng các nút theo chiều ngang

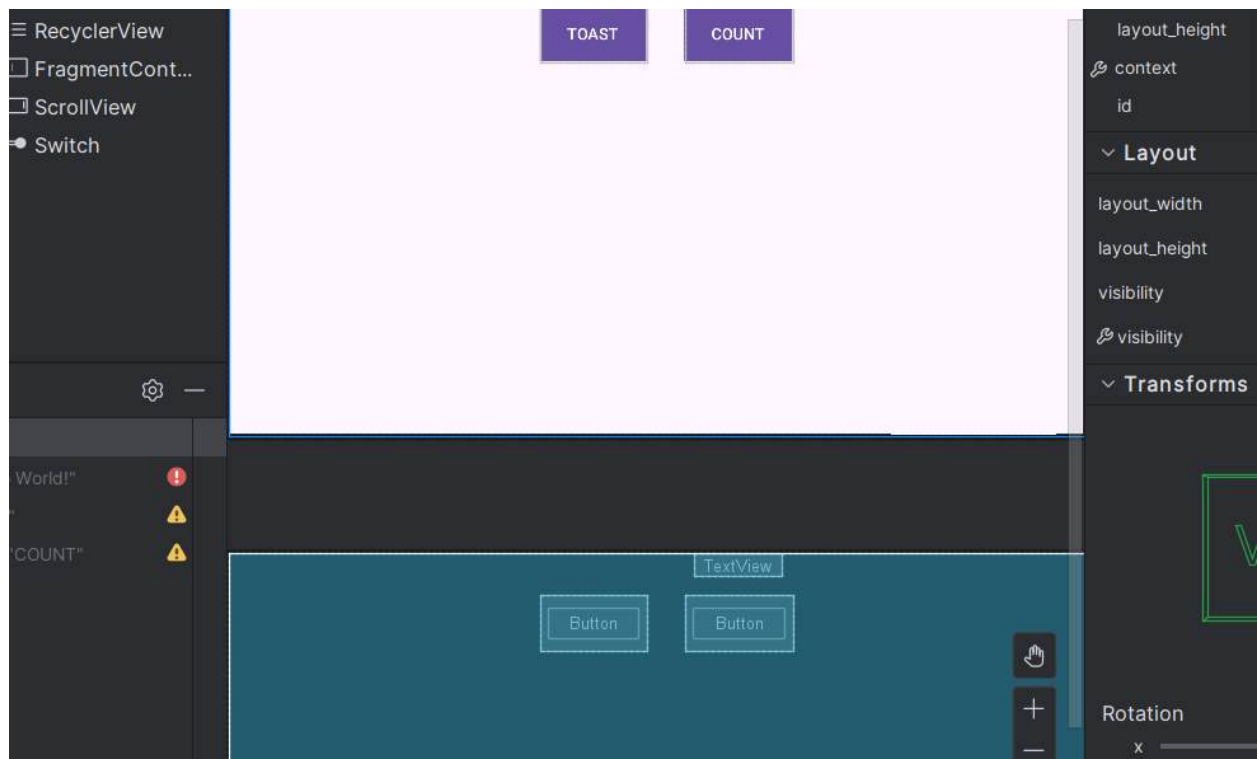


Nút đóng gói trên thanh công cụ cung cấp các tùy chọn để đóng gói hoặc mở rộng các phần tử giao diện người dùng đã chọn. Bạn có thể sử dụng nó để sắp xếp các phần tử Button theo chiều ngang trên bố cục.

1. Chọn Nút button_count trong **Component Tree** và Shift-chọn Nút button_toast để cả hai đều được chọn.

2. Nhấp vào nút gói  trên thanh công cụ và chọn Mở rộng theo chiều ngang như trong hình bên dưới.





- Để hoàn tất bố cục, hãy hạn chế TextView show_count ở cuối Nút button_toast và ở hai bên và dưới cùng của bố cục, như trong hình động bên dưới.

Giải mã tác vụ 1:

Dự án Android Studio: [HelloToast](#)

Thử thách lập trình 1

Tác vụ 2: Thay đổi bố cục thành các

LinearLayout là một ViewGroup sắp xếp bộ sưu tập các chế độ xem của nó theo hàng ngang hoặc dọc. LinearLayout là một trong những bố cục phổ biến nhất vì nó đơn giản và nhanh chóng. Nó thường được sử dụng trong một nhóm chế độ xem khác để sắp xếp các phần tử giao diện người dùng theo chiều ngang hoặc chiều dọc.

LinearLayout bắt buộc phải có các thuộc tính sau:

- layout_width
- layout_height
- orientation

Các layout_width và layout_height có thể lấy một trong các giá trị sau:

- `match_parent` : Mở rộng chế độ xem để lấp đầy chế độ xem của nó theo chiều rộng hoặc chiều cao. Khi `LinearLayout` là chế độ xem gốc, nó sẽ mở rộng đến kích thước của màn hình (chế độ xem mẹ).
- `wrap_content` : Thu nhỏ kích thước chế độ xem để chế độ xem vừa đủ lớn để bao bọc nội dung của nó. Nếu không có nội dung, chế độ xem sẽ trở nên vô hình.
- Số lượng cố định dp (pixel không phụ thuộc vào mật độ): Chỉ định kích thước cố định, được điều chỉnh cho mật độ màn hình của thiết bị. Ví dụ: 16dp có nghĩa là 16 pixel không phụ thuộc vào mật độ.

orientation có thể có:

- `horizontal` : Chế độ xem được sắp xếp từ trái sang phải.
- `vertical`: Views được sắp xếp từ trên xuống dưới.

Trong tác vụ này, bạn sẽ thay đổi nhóm chế độ xem gốc `ConstraintLayout` cho ứng dụng Hello Toast thành `LinearLayout` để bạn có thể thực hành sử dụng `LinearLayout`

2.1 Thay đổi nhóm chế độ xem gốc thành `LinearLayout`

1. Mở ứng dụng HelloToast từ tác vụ trước
2. Mở tệp `activity_main` (nếu chưa được mở), và nhấp vào **Text** ở dưới cùng ngăn chỉnh sửa để xem mã XML. Ở trên cùng của mã có dòng thẻ sau:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

3. Thay đổi thẻ `<android.support.constraint.ConstraintLayout` thành `<LinearLayout` để mã trông như thế này:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

4. Đảm bảo thẻ đóng ở cuối mã đã thay đổi thành `</LinearLayout>` (Android Studio tự động thay đổi thẻ đóng nếu bạn thay đổi thẻ mở). Nếu nó không tự động thay đổi, hãy thay đổi nó theo cách thủ công.

5. Trong dòng thẻ `<LinearLayout`, hãy thêm thuộc tính sau thuộc tính `android:layout_height`:

```
android:layout_height="match_parent"
android:orientation="vertical"
tools:context=".MainActivity">
```

Sau khi thực hiện những thay đổi này, một số thuộc tính XML cho các phần tử khác được gạch chân màu đỏ vì chúng được sử dụng với ConstraintLayout và không liên quan đến LinearLayout .

2.2 Thay đổi thuộc tính phần tử cho LinearLayout

Làm theo các bước sau để thay đổi các thuộc tính phần tử giao diện người dùng để chúng hoạt động với LinearLayout

1. Mở ứng dụng HelloToast từ tác vụ trước
2. Mở tệp bố cục activity_main.xml (nếu chưa được mở), và nhấn vào tab **Text**.
3. Tìm phần tử button_toast Button và thay đổi thuộc tính sau:

Original	Change to
<hr/> <small>This work is licensed under a Creative Commons Attribution 4.0 International License. This PDF is a one-time snapshot. See developer.android.com/courses/fundamentals-training/toc-v2 for the latest updates.</small> <hr/> <div>Page 94</div> <hr/> <div>Android Developer Fundamentals Course (V2) – Unit 1</div> <hr/>	
android:layout_width="0dp"	android:layout_width="match_parent"

4. Xóa các thuộc tính sau khỏi phần tử button_toast:

```
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
```

5. Tìm phần tử button_count Button và thay đổi thuộc tính sau:

Original	Change to
android:layout_width="0dp"	android:layout_width="match_parent"

6. Xóa các thuộc tính sau khỏi phần tử button_count:


```
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent"
```

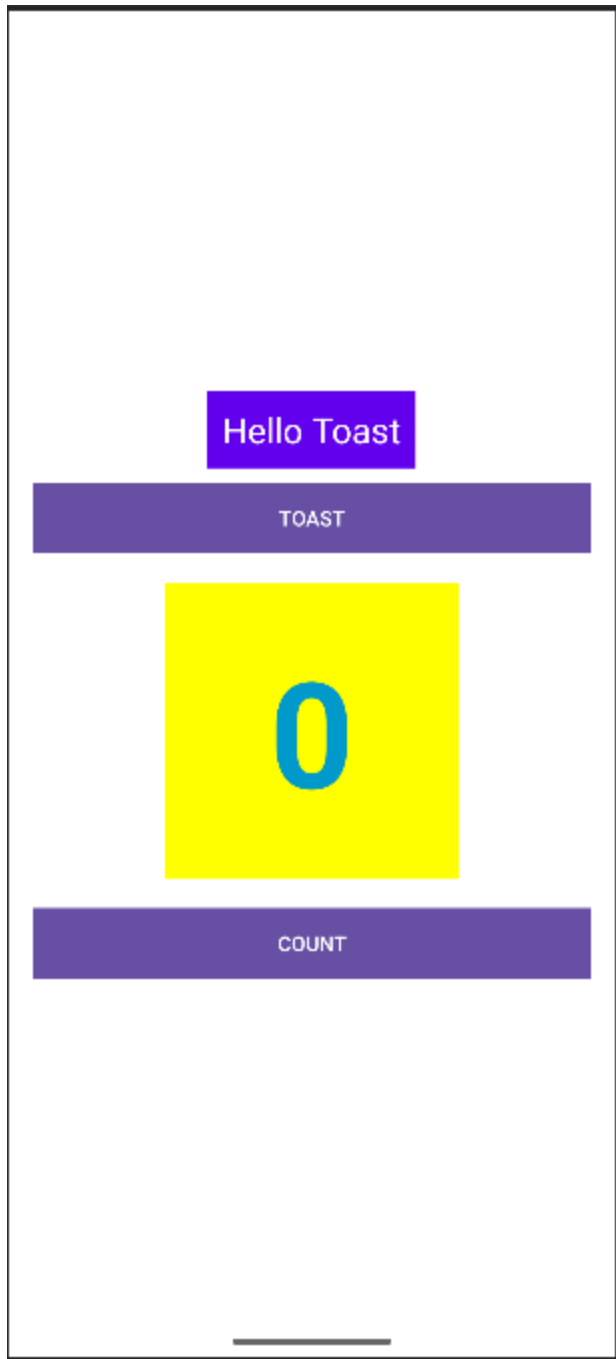
7. Tìm phần tử TextView show_count và thay đổi các thuộc tính sau:

Original	Change to
android:layout_width="0dp"	android:layout_width="match_parent"
android:layout_width="0dp"	android:layout_height="wrap_content"

8. Xóa các thuộc tính sau khỏi phần tử show_count:

```
app:layout_constraintBottom_toTopOf="@+id/button_count"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/button_toast"
```

9. Nhấp vào tab Preview ở bên phải của cửa sổ Android Studio (nếu chưa được chọn) để xem bản xem trước bố cục cho đến nay:



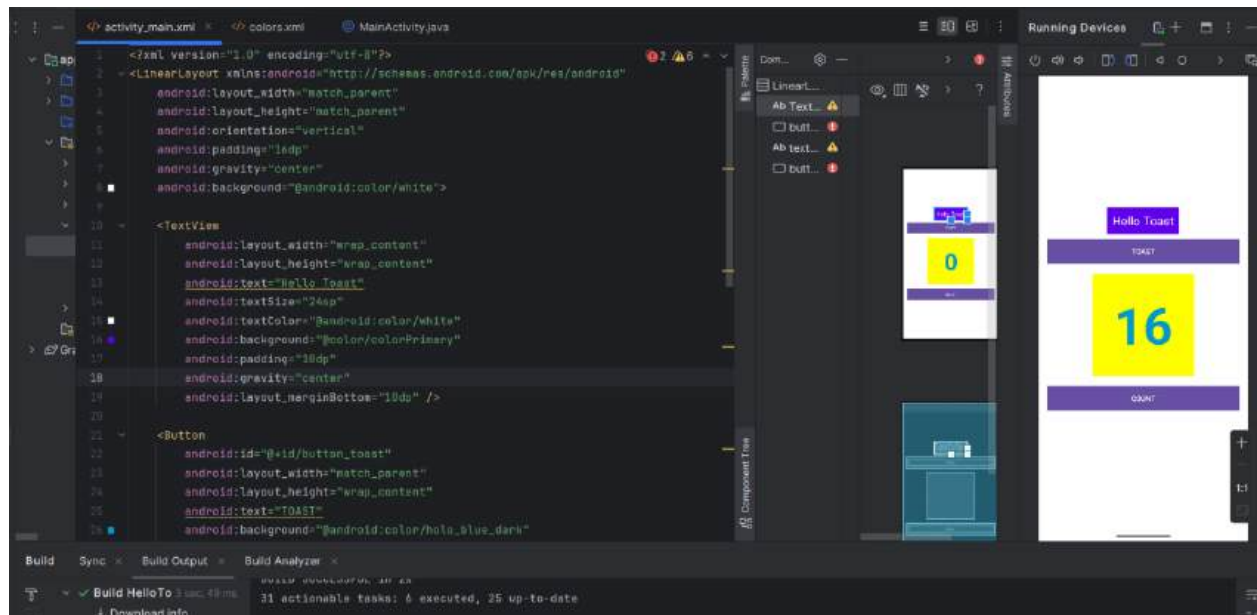
2.3 Thay đổi vị trí các phần tử trong LinearLayout

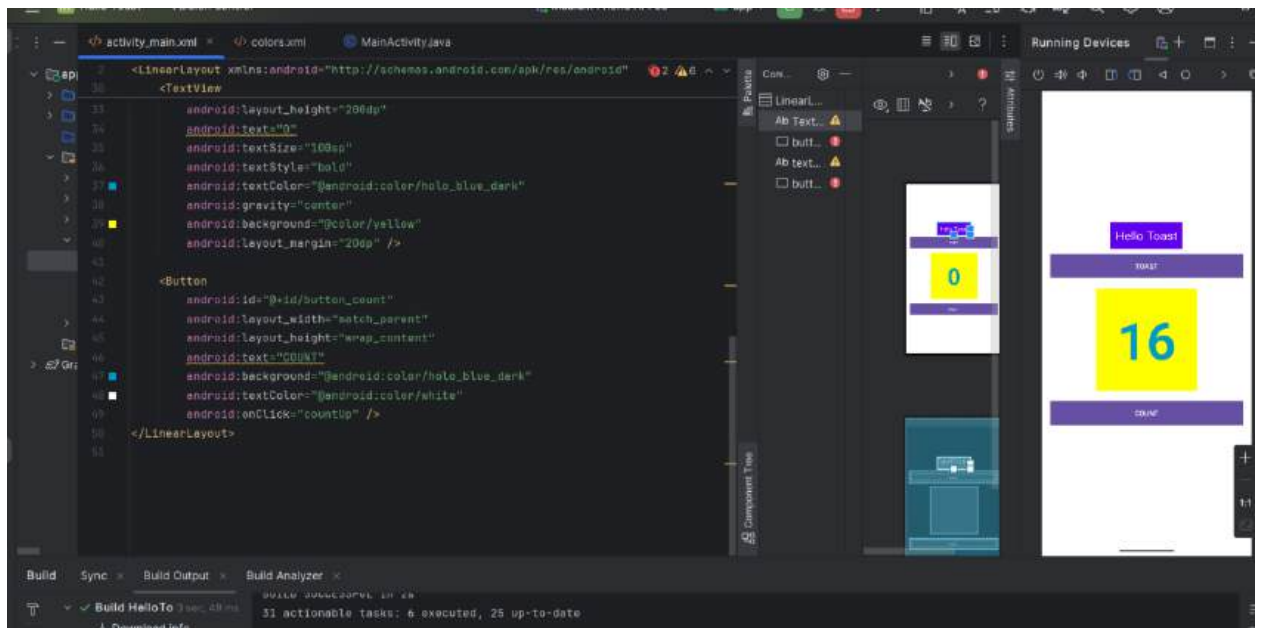
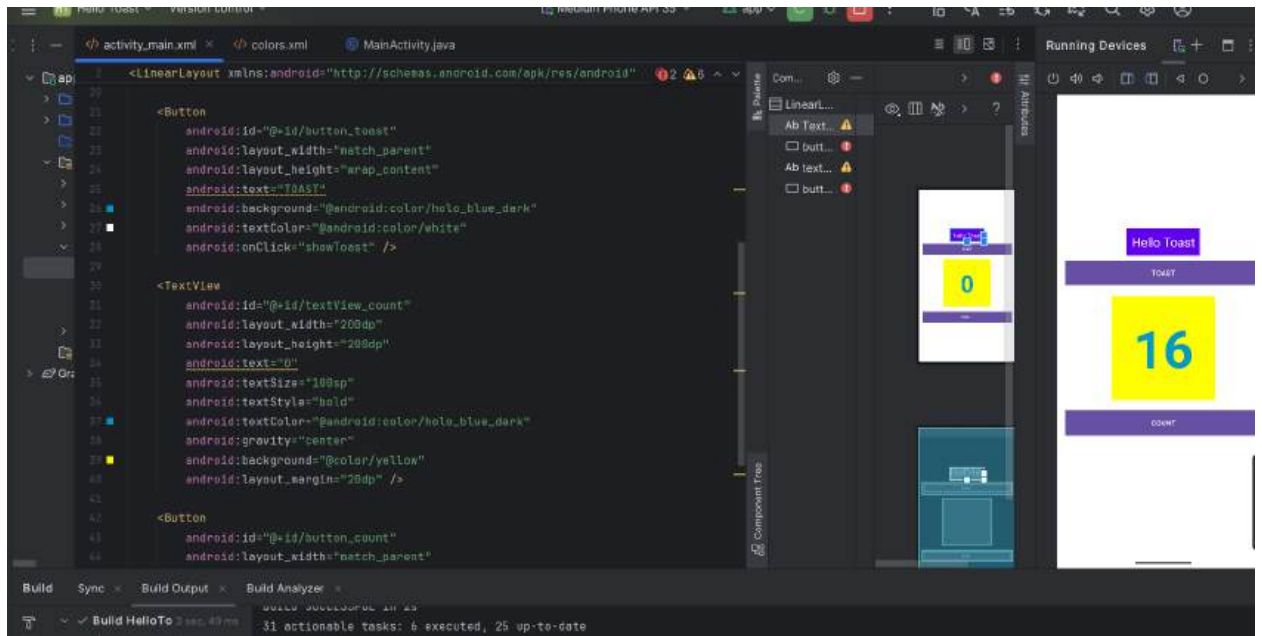
LinearLayout sắp xếp các phần tử của nó trong một hàng ngang hoặc dọc. Bạn đã thêm thuộc tính `android:orientation="vertical"` cho LinearLayout, vì vậy các phần tử được xếp chồng lên nhau theo chiều dọc như trong hình trước.

Để thay đổi vị trí của chúng để nút Đếm ở dưới cùng, hãy làm theo các bước sau:

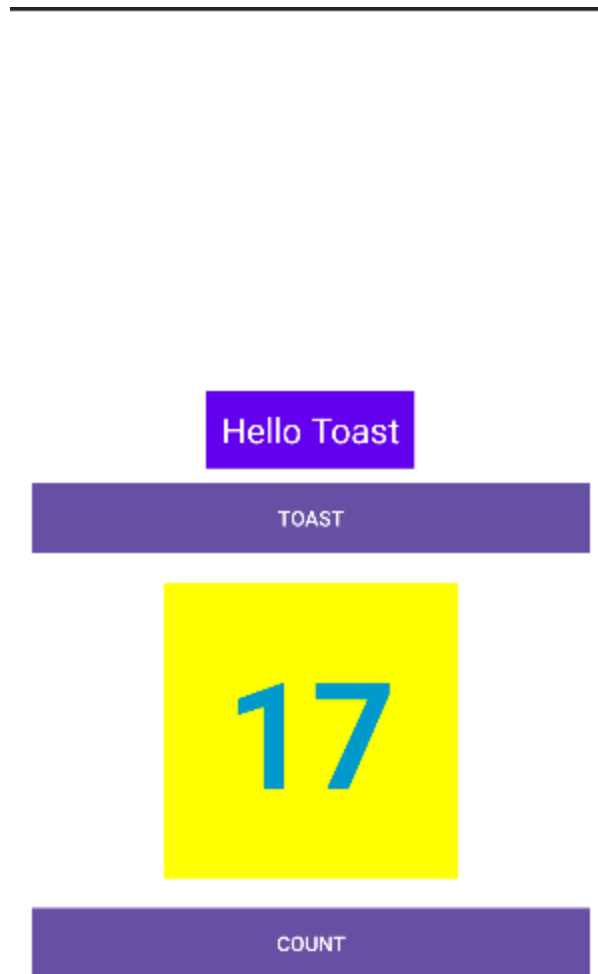
1. Mở ứng dụng HelloToast ở tác vụ trước đó.
2. Mở tệp bố cục activity_main.xml (nếu chưa được mở), và nhấp vào tab **Text**.
3. Chọn Nút button_count và tất cả các thuộc tính của nó, từ thẻ <Button đến và bao gồm closing / tag> đóng và chọn **Edit > Cut**.
4. Nhấp vào sau closing /> tag của phần tử TextView nhưng trước thẻ đóng và </LinearLayout> chọn **Edit > Paste**.
5. (Tùy chọn) Để khắc phục bất kỳ vấn đề thụt lề hoặc khoảng cách nào cho mục đích thẩm mỹ, hãy chọn **Code > Reformat Code** để định dạng lại mã XML với khoảng cách và thụt lề thích hợp.

Mã XML cho các phần tử giao diện người dùng bây giờ trông như sau:





Bằng cách di chuyển Nút button_count bên dưới TextView , bố cục bây giờ gần với những gì bạn đã có trước đây, với nút Count ở dưới cùng. Bản xem trước của bố cục bây giờ trông như sau:



2.4 Thêm trọng lượng cho phần tử TextView

Việc chỉ định các thuộc tính trọng lực và trọng lượng cho phép bạn kiểm soát thêm việc sắp xếp các chế độ xem và nội dung trong LinearLayout.

Thuộc tính `android:gravity` chỉ định căn chỉnh nội dung của Chế độ xem trong chính Chế độ xem. Trong bài học trước, bạn thiết lập thuộc tính này cho TextView `show_count` để căn giữa nội dung (chữ số 0) ở giữa Text View

```
android:gravity="center_vertical"  
android:text="Hello World!"  
app:layout_constraintEnd_toEndOf="parent"  
app:layout_constraintStart_toStartOf="parent" />
```

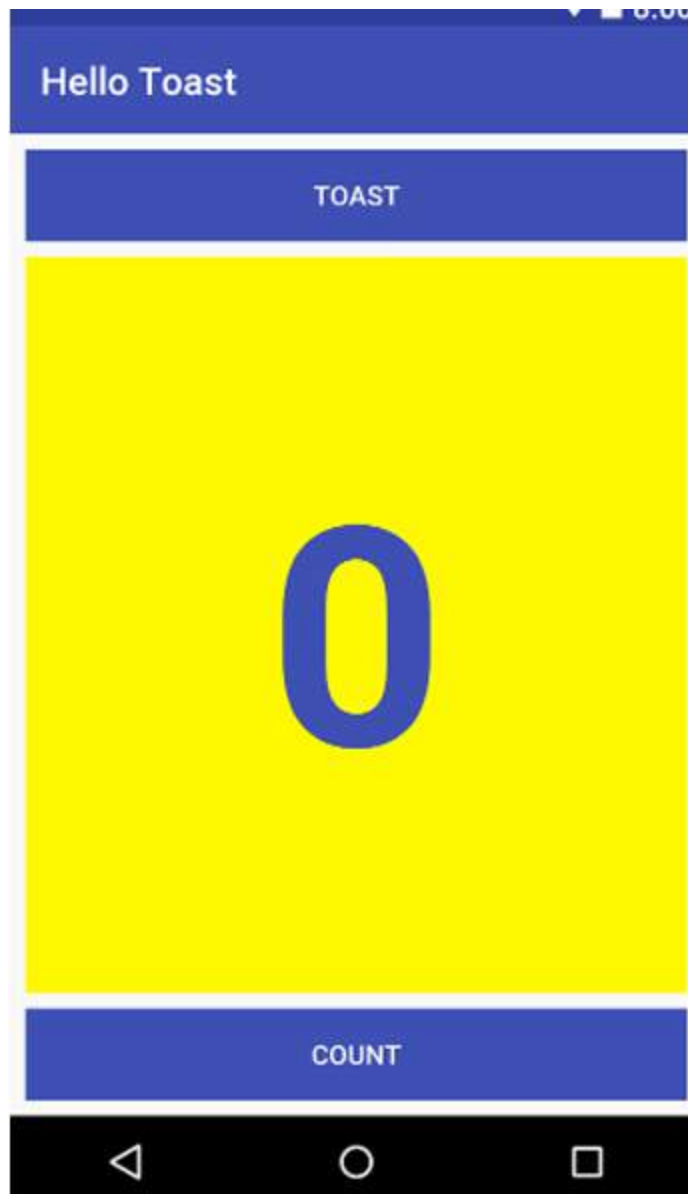
Thuộc tính `android:layout_weight` cho biết bao nhiêu không gian bổ sung trong `LinearLayout` sẽ được phân bổ cho View. Nếu chỉ có một Chế độ xem có thuộc tính này, nó sẽ nhận được tất cả không gian màn hình bổ sung. Đối với nhiều phần tử View, không gian được chia theo tỷ lệ. Ví dụ: nếu mỗi phần tử Button có trọng lượng là 1 và TextView là 2, tổng cộng là 4, thì các phần tử Button nhận được 1/4 khoảng trống mỗi phần tử và một nửa TextView.

Trên các thiết bị khác nhau, bố cục có thể hiển thị phần tử TextView `show_count` như một phần hoặc phần lớn khoảng trống giữa các nút Toast và Count. Để mở rộng TextView để lấp đầy không gian có sẵn bất kể thiết bị nào được sử dụng, hãy chỉ định thuộc tính `android:gravity` cho TextView. Làm theo các bước sau:

1. Mở ứng dụng HelloToast ở tác vụ trước đó.
2. Mở tệp bố cục `activity_main.xml` (nếu chưa được mở), và nhấp vào tab **Text**.
3. Tìm phần tử TextView `show_count` và thêm thuộc tính sau:

```
android:layout_weight="1"
```

Bản xem trước bây giờ trông giống như hình sau.



Giải mã tác vụ 2

Mã XML trong activity_main.xml

```

<?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"
    android:padding="16dp"
    android:gravity="center"
    android:background="@android:color/white">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello Toast"
        android:textSize="24sp"
        android:textColor="@android:color/white"
        android:background="@color/colorPrimary"
        android:padding="10dp"
        android:gravity="center"
        android:layout_marginBottom="10dp" />

    <Button
        android:id="@+id/button_toast"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="TOAST"
        android:background="@android:color/holo_blue_dark"
        android:textColor="@android:color/white"
        android:onClick="showToast" />

```

Build Output x Build Analyzer x


```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    <TextView
        android:gravity="center"
        android:layout_marginBottom="10dp" />

    <Button
        android:id="@+id/button_toast"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="TOAST"
        android:background="@android:color/ho_lo_blue_dark"
        android:textColor="@android:color/white"
        android:onClick="showToast" />

    <TextView
        android:id="@+id/textView_count"
        android:layout_width="200dp"
        android:layout_height="200dp"
        android:text="0"
        android:textSize="100sp"
        android:textStyle="bold"
        android:textColor="@android:color/ho_lo_blue_dark"
        android:gravity="center"
        android:background="@color/yellow"
        android:layout_margin="20dp" />

    <Button
        android:id="@+id/button_count"
```

```
activity_main.xml x colors.xml MainActivity.java
? <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
30 <TextView
31     android:id="@+id/textView_count"
32     android:layout_width="200dp"
33     android:layout_height="200dp"
34     android:text="0"
35     android:textSize="100sp"
36     android:textStyle="bold"
37     android:textColor="@android:color/ho_lo_blue_dark"
38     android:gravity="center"
39     android:background="@color/yellow"
40     android:layout_margin="20dp" />
41
42 <Button
43     android:id="@+id/button_count"
44     android:layout_width="match_parent"
45     android:layout_height="wrap_content"
46     android:text="COUNT"
47     android:background="@android:color/ho_lo_blue_dark"
48     android:textColor="@android:color/white"
49     android:onClick="countUp" />
50 </LinearLayout>
51
```

Tác vụ 3: Thay đổi bố cục thành RelativeLayout

RelativeLayout là một nhóm chế độ xem trong đó mỗi chế độ xem được định vị và căn chỉnh tương đối với các dạng xem khác trong nhóm. Trong nhiệm vụ này, bạn sẽ học cách xây dựng bố cục với RelativeLayout

3.1 Thay đổi LinearLayout thành RelativeLayout

Một cách dễ dàng để thay đổi LinearLayout thành RelativeLayout là thêm các thuộc tính XML vào tab Text.

1. Mở tệp bố cục activity_main.xml và nhấp vào tab Văn bản ở cuối ngăn chỉnh sửa để xem mã XML.
2. Thay đổi <LinearLayout ở trên cùng thành < RelativeLayout để câu lệnh trông như sau:

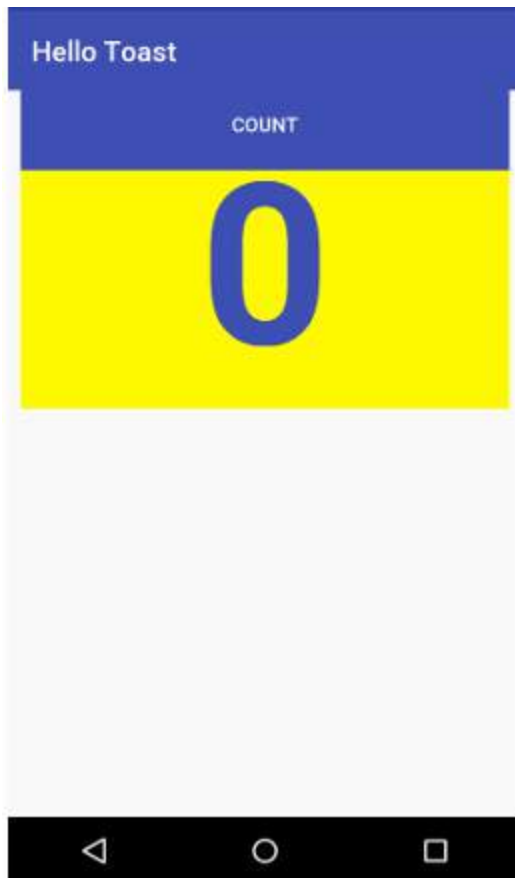
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

3. Cuộn xuống để đảm bảo rằng thẻ kết thúc </LinearLayout> cũng đã thay đổi thành </RelativeLayout> ; nếu chưa, hãy thay đổi nó theo cách thủ công.

3.2 Sắp xếp lại các chế độ xem trong RelativeLayout

Một cách dễ dàng để thay đổi LinearLayout thành RelativeLayout là thêm các thuộc tính XML vào tab Text.

1. Nhấp vào tab Preview ở bên cạnh trình chỉnh sửa (nếu nó chưa được chọn) để xem bản xem trước bố cục, bây giờ trông giống như hình bên dưới.



Với sự thay đổi thành RelativeLayout, trình soạn thảo bố cục cũng thay đổi một số thuộc tính chế độ xem. Chẳng hạn:

- Nút đếm (`button_count`) phủ lên Nút Toast , đó là lý do tại sao bạn không thể nhìn thấy Nút Toast (`button_toast`).
- Phần trên cùng của TextView (`show_count`) phủ các phần tử Button.

2. Thêm thuộc tính `android:layout_below` vào `button_count` Button để đặt Button ngay bên dưới `show_count` TextView . Thuộc tính này là một trong một số thuộc tính để định vị các dạng xem trong RelativeLayout —bạn đặt các dạng xem liên quan đến các dạng xem khác.

```
android:layout_below="@+id/show_count"
```

3. Thêm thuộc tính `android:layout_centerHorizontal` vào cùng một Button để căn giữa chế độ xem theo chiều ngang trong khung cảnh cha của nó, trong trường hợp này là nhóm chế độ xem RelativeLayout.

```
android:layout_centerHorizontal="true"
```

Mã XML đầy đủ cho Nút button_count như sau:

```
<Button
    android:id="@+id/button_count"
    android:layout_below="@+id/show_count"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#2F45BF"
    android:backgroundTint="#2196F3"
    android:text="COUNT"
    app:layout_constraintVertical_bias="0.099" />
```

4. Thêm các thuộc tính sau vào TextView show_count

```
android:layout_below="@+id/button_toast"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true"
```

5. Xóa thuộc tính android:layout_weight="1" khỏi TextView show_count , thuộc tính này không liên quan đến RelativeLayout . Bản xem trước bố cục bây giờ trông giống như hình sau:



Giải mã tác vụ 3

Mã XML trong activity_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity"
    tools:ignore="ExtraText">

    <Button
        android:id="@+id/button_toast"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:background="@color/colorPrimary"
        android:onClick="showToast"
        android:layout_centerHorizontal="true"
        android:text="@string/button_label_toast"
        android:textColor="@android:color/white" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center_vertical"

```



```
        android:gravity="center_vertical"
        android:text="Hello World!"
        android:layout_below="@+id/button_toast"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="272dp"
    android:background="#3F51B5"
    android:text="TOAST"
    app:layout_constraintVertical_bias="0.099" />

<Button
    android:id="@+id/button_count"
    android:layout_below="@+id/show_count"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#2F45BF"
    android:backgroundTint="#2196F3"
    Formats: color
    Tint to apply to the background.
    </Relati
```

Tóm tắt

Sử dụng trình chỉnh sửa bố cục để xem trước và tạo mã:

- Để xem trước bố cục ứng dụng với hướng ngang trong trình chỉnh sửa bố cục, hãy bấm vào nút Hướng trong trình chỉnh sửa trên thanh công cụ trên cùng và chọn Chuyển sang ngang . Chọn Chuyển sang Dọc để quay lại hướng dọc.
- Để tạo biến thể của bố cục khác nhau cho hướng ngang, hãy nhấp vào nút Orientation in Editor và chọn Create Landscape Variation . Một cửa sổ trình chỉnh sửa mới mở ra với tab đất liền / activity_main.xml hiển thị bố cục cho hướng ngang (ngang).
- Để xem trước bố cục cho các thiết bị khác nhau mà không cần phải chạy ứng dụng trên thiết bị hoặc trình mô phỏng, hãy nhấp vào thiết bị nút Thiết bị trong Trình chỉnh sửa.
- Để tạo biến thể của bố cục khác nhau cho máy tính bảng (màn hình lớn hơn), hãy nhấp vào nút Định hướng trong Trình chỉnh sửa và chọn Tạo bố cục x-large Variation

Sử dụng ConstraintLayout

- Để xóa tất cả các ràng buộc trong bố cục có gốc ConstraintLayout. Nhấp vào nút **Clear All Constraints** trên thanh công cụ
- Bạn có thể căn chỉnh một phần tử giao diện người dùng có chứa văn bản, chẳng hạn như TextView hoặc Button với một phần tử giao diện người dùng khác có chứa văn bản. Một ràng buộc đường cơ sở cho phép bạn hạn chế các phần tử để đường cơ sở văn bản khớp nhau.
- Để tạo ràng buộc đường cơ sở, hãy di con trỏ qua phần tử giao diện người dùng cho đến khi nút ràng buộc đường cơ sở xuất hiện bên dưới phần tử.
- Nút đóng gói trên thanh công cụ cung cấp các tùy chọn để đóng gói hoặc mở rộng các phần tử giao diện người dùng đã chọn. Bạn có thể sử dụng nó để sắp xếp các phần tử Button theo chiều ngang trên bố cục.

Sử dụng LinearLayout:

- LinearLayout là một ViewGroup sắp xếp bộ sưu tập các chế độ xem của nó theo hàng ngang hoặc dọc.
- LinearLayout là bắt buộc phải có các thuộc tính layout_width , layout_height và orientation

- `match_parent` cho `layout_width` hoặc `layout_height` : Mở rộng Chế độ xem để lấp đầy cha của nó theo chiều rộng hoặc chiều cao. Khi `LinearLayout` là `View` gốc, nó mở rộng đến kích thước của màn hình (`View`)
- `Wrap_content` cho `layout_width` hoặc `layout_height` : Thu nhỏ các kích thước để Chế độ xem vừa đủ lớn để bao bọc nội dung của nó. Nếu không có nội dung, Chế độ xem sẽ trở nên vô hình.
- Số lượng dp cố định (pixel không phụ thuộc vào mật độ) cho `layout_width` hoặc `layout_height` : Chỉ định kích thước cố định, được điều chỉnh cho mật độ màn hình của thiết bị. Ví dụ: 16dp có nghĩa là 16 pixel không phụ thuộc vào mật độ.
- Hướng cho `LinearLayout` có thể ngang để sắp xếp các phần tử từ trái sang phải hoặc dọc để sắp xếp các phần tử từ trên xuống dưới.
- Việc chỉ định các thuộc tính trọng lực và trọng lượng cho phép bạn kiểm soát thêm việc sắp xếp các chế độ xem và nội dung trong `LinearLayout`
- Thuộc tính `android:gravity` chỉ định căn chỉnh nội dung của Chế độ xem trong chính Chế độ xem.

Sử dụng `RelativeLayout`:

- `RelativeLayout` là `ViewGroup` trong đó mỗi chế độ xem được định vị và căn chỉnh tương đối với các chế độ xem khác trong nhóm.
- Sử dụng `android:layout_alignParentTop` để căn chỉnh Chế độ xem lên trên cùng của cha mẹ.
- Sử dụng `android:layout_alignParentLeft` để căn chỉnh Chế độ xem ở phía bên trái của cha mẹ.
- Sử dụng `android:layout_alignParentStart` để làm cho cạnh bắt đầu của `View` khớp với cạnh bắt đầu của cha mẹ. Thuộc tính này hữu ích nếu bạn muốn ứng dụng của mình hoạt động trên các thiết bị sử dụng các tùy chọn ngôn ngữ hoặc ngôn ngữ khác nhau. Bắt đầu là cạnh trái của màn hình nếu tùy chọn là từ trái sang phải hoặc đó là cạnh phải của màn hình nếu tùy chọn là từ phải sang trái.

Các khái niệm liên quan

Tài liệu khái niệm liên quan có trong [1.2: Layouts and resources for the UI](#) .

Tìm hiểu thêm

Tài liệu Android Studio:

- Meet Android Studio
- Create app icons with Image Asset Studio

Tài liệu dành cho nhà phát triển Android:

- Layouts
- Build a UI with Layout Editor
- Build a Responsive UI with ConstraintLayout
- LinearLayout
- RelativeLayout
- View
- Button
- TextView

1.4) Văn bản và các chế độ cuộn

Giới thiệu

Lớp `TextView` là một lớp con của lớp `View` hiển thị văn bản trên màn hình. Bạn có thể kiểm soát cách văn bản xuất hiện bằng các thuộc tính `TextView` trong tệp bố cục XML. Thực tế này cho thấy cách làm việc với nhiều phần tử `TextView`, bao gồm cả một phần tử mà người dùng có thể cuộn nội dung theo chiều dọc.

Nếu bạn có nhiều thông tin hơn mức phù hợp trên màn hình của thiết bị, bạn có thể tạo chế độ xem cuộn để người dùng có thể cuộn theo chiều dọc bằng cách vuốt lên hoặc xuống hoặc theo chiều ngang bằng cách vuốt sang phải hoặc trái.

Bạn thường sẽ sử dụng chế độ xem cuộn cho các câu chuyện tin tức, bài báo hoặc bất kỳ văn bản dài nào không hoàn toàn phù hợp với màn hình. Bạn cũng có thể sử dụng chế độ xem cuộn để cho phép người dùng nhập nhiều dòng văn bản hoặc kết hợp các thành phần giao diện người dùng (chẳng hạn như trường văn bản và nút) trong chế độ xem cuộn.

Lớp `ScrollView` cung cấp bố cục cho chế độ xem cuộn. `ScrollView` là một lớp con của `FrameLayout`. Chỉ đặt một chế độ xem làm dạng xem con trong đó — một chế độ xem con

chứa toàn bộ nội dung cần cuộn. Bản thân chế độ xem con này có thể là một ViewGroup (chẳng hạn như LinearLayout) chứa các phần tử giao diện người dùng.

Bố cục phức tạp có thể gặp vấn đề về hiệu suất với chế độ xem con như hình ảnh. Một lựa chọn tốt cho Chế độ xem trong ScrollView là LinearLayout được sắp xếp theo hướng dọc, trình bày các mục mà người dùng có thể cuộn qua (chẳng hạn như các phần tử TextView).

Với ScrollView, tất cả các thành phần giao diện người dùng đều nằm trong bộ nhớ và trong hệ thống phân cấp chế độ xem ngay cả khi chúng không được hiển thị trên màn hình. Điều này làm cho ScrollView trở nên lý tưởng để cuộn các trang văn bản dạng tự do một cách mượt mà, vì văn bản đã có trong bộ nhớ. Tuy nhiên, ScrollView có thể sử dụng nhiều bộ nhớ, điều này có thể ảnh hưởng đến hiệu suất của phần còn lại của ứng dụng. Để hiển thị danh sách dài các mục mà người dùng có thể thêm, xóa hoặc chỉnh sửa, hãy cân nhắc sử dụng RecyclerView, được mô tả trong một bài học riêng.

Những điều bạn nên biết

Bạn sẽ có thể:

- Tạo ứng dụng HelloWorld với Android Studio.
- Chạy ứng dụng trên trình giả lập hoặc thiết bị.
- Triển khai TextView trong bố cục cho ứng dụng
- Tạo và sử dụng các tài nguyên chuỗi.

Những gì bạn sẽ học:

- Cách sử dụng mã XML để thêm nhiều phần tử TextView
- Cách sử dụng mã XML để xác định chế độ xem cuộn.
- Cách hiển thị văn bản dạng tự do với một số thẻ định dạng HTML
- Cách tạo kiểu cho màu nền TextView và màu văn bản.

Bạn sẽ làm gì:

Tạo ứng dụng ScrollingText.

Thay đổi ConstraintLayout ViewGroup thành RelativeLayout.

Thêm hai phần tử TextView cho tiêu đề và tiêu đề phụ của bài viết.

Sử dụng kiểu và màu TextAppearance cho tiêu đề và tiêu đề phụ của bài viết.

Sử dụng thẻ HTML trong chuỗi văn bản để kiểm soát định dạng.

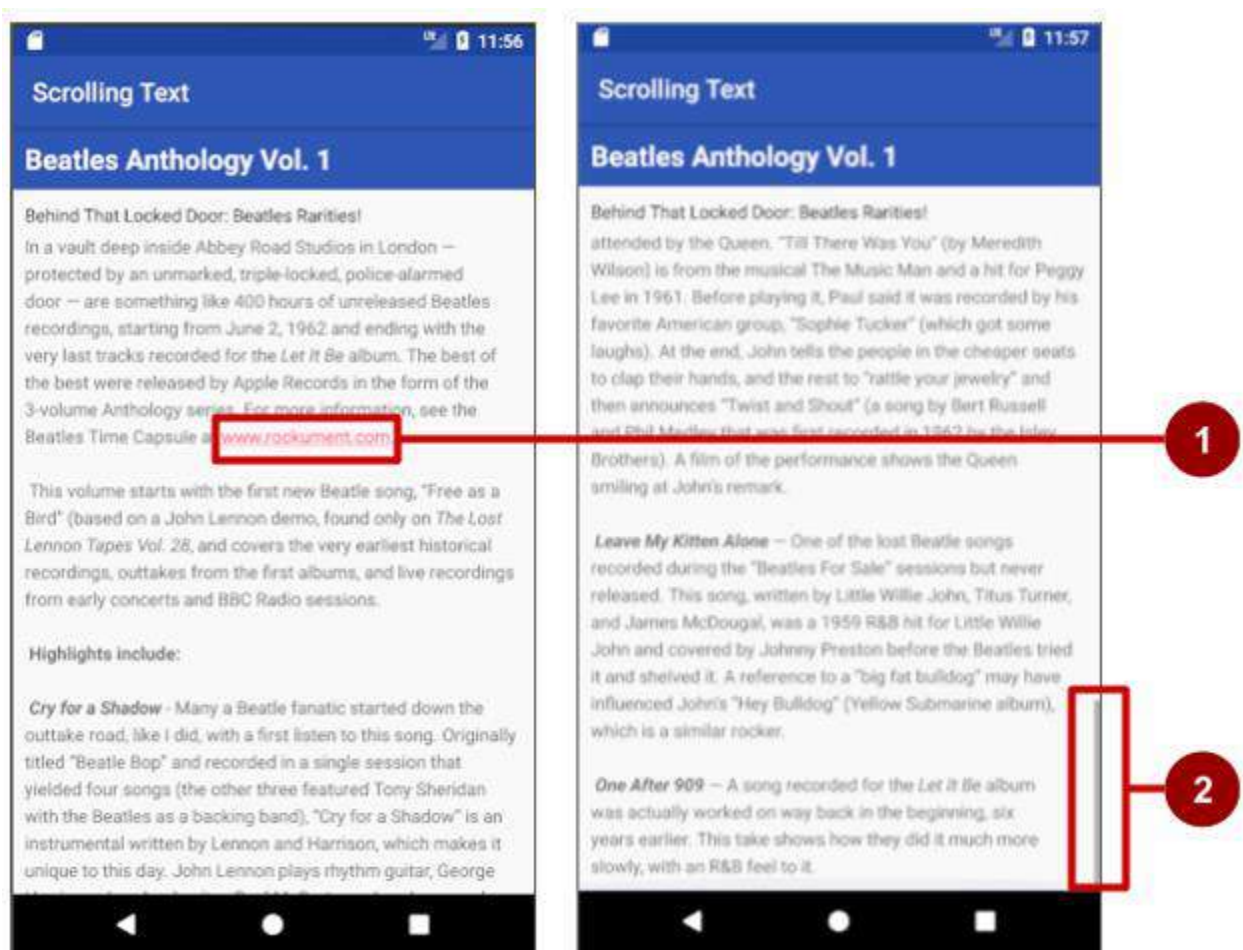
Sử dụng thuộc tính `lineSpacingExtra` để thêm khoảng cách dòng để dễ đọc.

Thêm `ScrollView` vào bố cục để cho phép cuộn phần tử `TextView`.

Thêm thuộc tính `autoLink` để cho phép URL trong văn bản hoạt động và có thể nhấp vào.

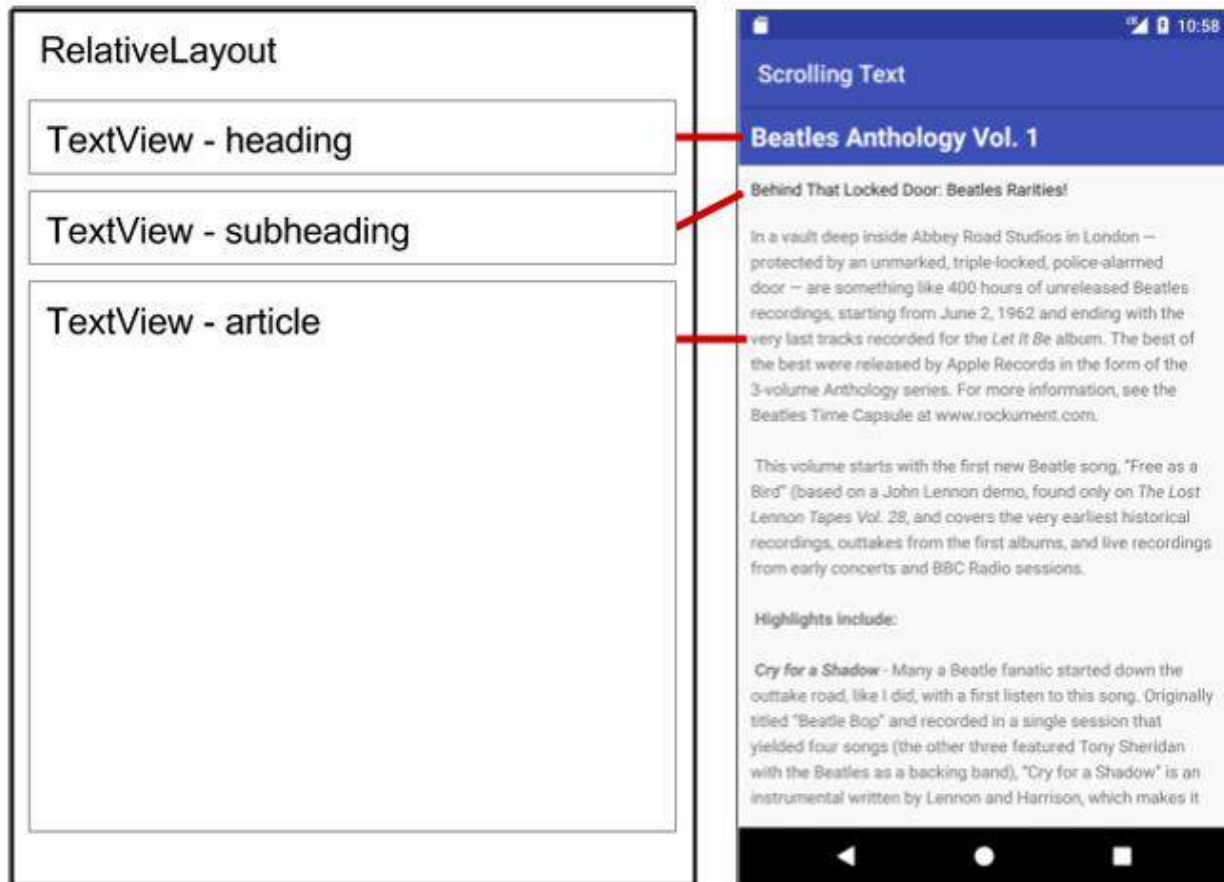
Tổng quan về ứng dụng

Ứng dụng Văn bản cuộn trình bày thành phần giao diện người dùng `ScrollView`. `ScrollView` là một `ViewGroup` mà trong ví dụ này chứa một `TextView`. Nó hiển thị một trang văn bản dài — trong trường hợp này là đánh giá album nhạc — mà người dùng có thể cuộn theo chiều dọc để đọc bằng cách vuốt lên và xuống. Một thanh cuộn xuất hiện ở lề bên phải. Ứng dụng cho thấy cách bạn có thể sử dụng văn bản được định dạng với các thẻ HTML tối thiểu để đặt văn bản thành in đậm hoặc in nghiêng và với các ký tự dòng mới để phân tách đoạn văn. Bạn cũng có thể bao gồm các liên kết web đang hoạt động trong văn bản.



Tác vụ 1: Thêm và chỉnh sửa các thành phần TextView

Trong thực tế này, bạn sẽ tạo một dự án Android cho ứng dụng ScrollingText, thêm các phần tử TextView vào bố cục cho tiêu đề và phụ đề bài viết, đồng thời thay đổi phần tử TextView "Hello World" hiện có để hiển thị một bài viết dài. Hình dưới đây là sơ đồ bố cục.



Bạn sẽ thực hiện tất cả các thay đổi này trong mã XML và trong tệp strings.xml. Bạn sẽ chỉnh sửa mã XML cho bố cục trong ngăn Văn bản, bạn hiển thị bằng cách bấm vào tab Văn bản, thay vì bấm vào tab Thiết kế cho ngăn Thiết kế. Một số thay đổi đối với các phần tử và thuộc tính giao diện người dùng để thực hiện trực tiếp trong ngăn Văn bản bằng mã nguồn XML.

1.1 Tạo dự án và các phần tử TextView

Trong tác vụ này, bạn sẽ tạo dự án và các phần tử TextView, đồng thời sử dụng các thuộc tính TextView để tạo kiểu cho văn bản và nền.

1. Trong Android Studio tạo dự án mới với các thông số như sau:

Attribute	Value
Application Name	Scrolling Text
Company Name	android.example.com (or your own domain)
Phone and Tablet Minimum SDK	API15: Android 4.0.3 IceCreamSandwich
Template	Empty Activity
Generate Layout File checkbox	Selected
Backwards Compatibility (AppCompat) checkbox	Selected

2. Trong ứng dụng > thư mục res > layout trong ngăn Project > Android, mở activity_main.xml và nhấp vào tab Text để xem mã XML.

```
android.support.constraint.ConstraintLayout
```

3. Thay đổi ViewGroup này thành RelativeLayout . Dòng mã thứ hai bây giờ trông giống như thế này:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

4. Xóa dòng mã XML sau đây, có liên quan đến ConstraintLayout.

Khối mã XML ở trên cùng bây giờ trông như thế này:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
```

5. Thêm phần tử TextView phía trên TextView "Hello World" bằng cách nhập <TextView . Một khối TextView xuất hiện kết thúc bằng /> và hiển thị các thuộc tính layout_width và layout_height, cần thiết cho TextView
6. Nhập các thuộc tính sau cho TextView . Khi bạn nhập từng thuộc tính và giá trị, các đề xuất sẽ xuất hiện để hoàn thành tên hoặc giá trị thuộc tính.

TextView #1 attribute	Value
android:layout_width	"match_parent"
android:layout_height	"wrap_content"
android:id	"@+id/article_heading"
android:background	"@color/colorPrimary"
android:textColor	"@android:color/white"
android:padding	"10dp"
android:textAppearance	"@android:style/TextAppearance.DeviceDefault.Large"
android:textStyle	"bold"
android:text	"Article Title"

```
<TextView
    android:id="@+id/article_heading"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:background="@color/colorPrimary"
    android:textColor="@android:color/white"
    android:padding="10dp"
    android:textAppearance="@android:style/TextAppearance.DeviceDefault.Large"
    android:textStyle="bold"
    android:text="Beatles Anthology Vol. 1"/>
```

7. Trích xuất tài nguyên chuỗi cho chuỗi được mã hóa cứng "Tiêu đề bài viết" của thuộc tính android:text trong TextView để tạo một mục nhập cho thuộc tính đó trong strings.xml . Đặt con trỏ vào chuỗi được mã hóa cứng, nhấn Alt-Enter (Option-Enter trên máy Mac) và chọn Trích xuất tài nguyên chuỗi . Đảm bảo rằng tùy chọn Tạo tài nguyên trong thư mục được chọn, sau đó chỉnh sửa tên tài nguyên cho giá trị chuỗi thành article_title .

8. Trích xuất tài nguyên thứ nguyên cho chuỗi mã hóa cứng "10dp" của thuộc tính `android:padding` trong `TextView` để tạo `dimens.xml` và thêm một mục nhập vào đó.
9. Thêm một phần tử `TextView` khác phía trên `TextView` "Hello World" và bên dưới `TextView` mà bạn đã tạo ở các bước trước. Thêm các thuộc tính sau vào `TextView`

TextView #2 Attribute	Value
<code>layout_width</code>	<code>"match_parent"</code>

*This work is licensed under a [Creative Commons Attribution 4.0 International License](#).
This PDF is a one-time snapshot. See developer.android.com/courses/fundamentals-training/toc-v2
for the latest updates.*

Page 120

Android Developer Fundamentals Course (V2) – Unit 1

<code>layout_height</code>	<code>"wrap_content"</code>
<code>android:id</code>	<code>"@+id/article_subheading"</code>
<code>android:layout_below</code>	<code>"@id/article_heading"</code>
<code>android:padding</code>	<code>"@dimen/padding_regular"</code>
<code>android:textAppearance</code>	<code>"@android:style/TextAppearance.DeviceDefault"</code>
<code>android:text</code>	<code>"Article Subtitle"</code>

10. Trích xuất tài nguyên chuỗi cho chuỗi được mã hóa cứng "Phụ đề bài viết" của thuộc tính android:text trong TextView để article_subtitle .

11. Trong phần tử TextView "Hello World", hãy xóa các thuộc tính layout_constraint:

```
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintLeft_toLeftOf="parent"
app:layout_constraintRight_toRightOf="parent"
app:layout_constraintTop_toTopOf="parent"
```

12. Thêm các thuộc tính TextView sau vào phần tử TextView "Hello World" và thay đổi thuộc tính android:text:

TextView Attribute	Value
android:id	"@+id/article"
android:layout_below	"@id/article_subheading"
android:lineSpacingExtra	"5sp"
android:padding	"@dimen/padding_regular"
android:text	Change to "Article text"

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_below="@id/article_subheading"
android:padding="16dp">
```

13.

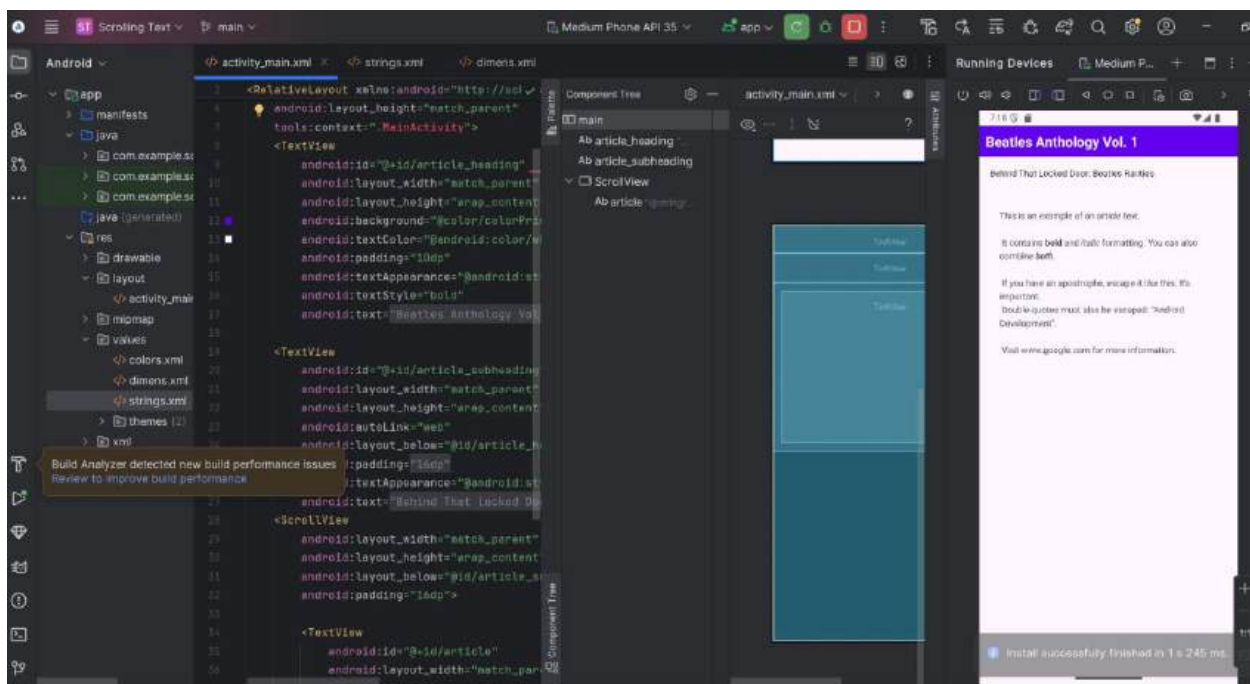
1.2 Thêm văn bản của bài viết

Bạn có thể sử dụng văn bản trong tệp văn bản của mình hoặc sử dụng văn bản được cung cấp cho chuỗi article_text trong tệp strings.xml của ứng dụng ScrollingText đã hoàn thành. Yêu cầu duy nhất cho tác vụ này là văn bản phải đủ dài để không vừa với màn hình.

```
<resources>
    <string name="app_name">Scrolling Text</string>
    <string name="article_title">Beatles Anthology Vol. 1</string>
    <string name="article_subtitle">Behind That Locked Door: Beatles Rarities</string>
    <string name="article_text">This is an example of an article text.\n\n
        It contains <b>bold</b> and <i>italic</i> formatting. You can also combine <b><i>both</i></b>.\n\n
        If you have an apostrophe, escape it like this: It's important.\n
        Double-quotes must also be escaped: \"Android Development\".\n\n
        Visit <a>www.google.com</a> for more information.
    </string>
</resources>
```

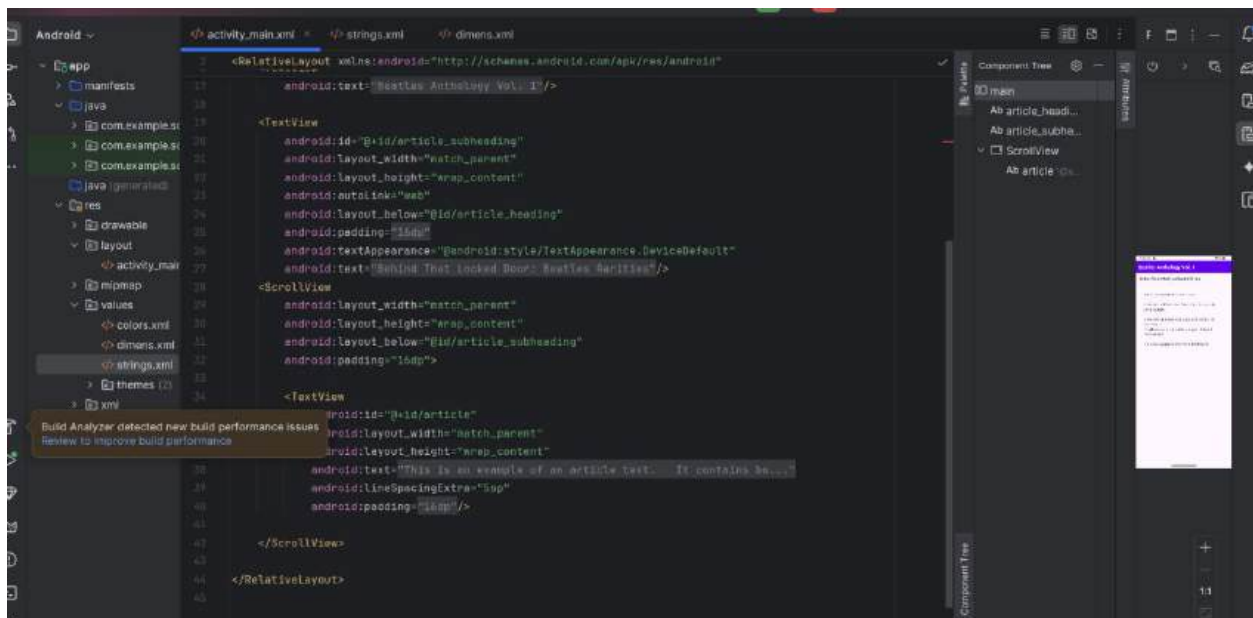
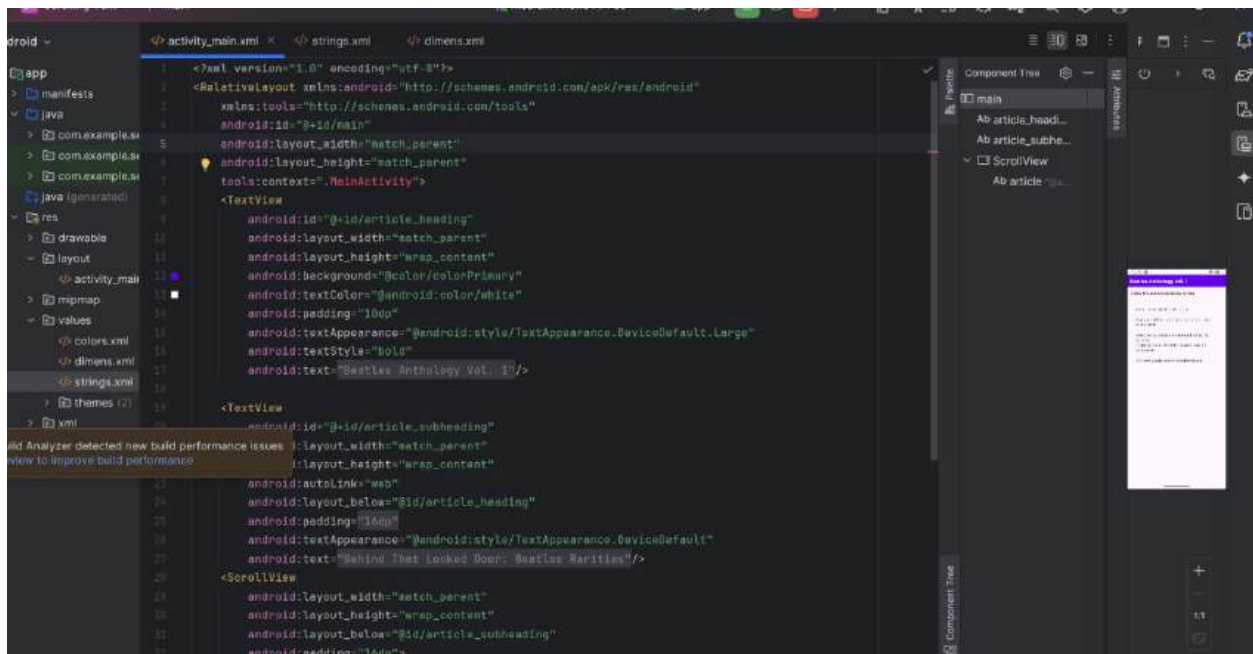
1.3 Chạy chương trình

Chạy ứng dụng. Bài viết xuất hiện, nhưng người dùng không thể cuộn bài viết vì bạn chưa bao gồm ScrollView (bạn sẽ thực hiện trong tác vụ tiếp theo). Cũng lưu ý rằng nhấn vào liên kết web hiện không làm được gì. Bạn cũng sẽ khắc phục điều đó trong nhiệm vụ tiếp theo.



Giải mã tác vụ 1

Tập bố cục activity_main.xml trông như sau:

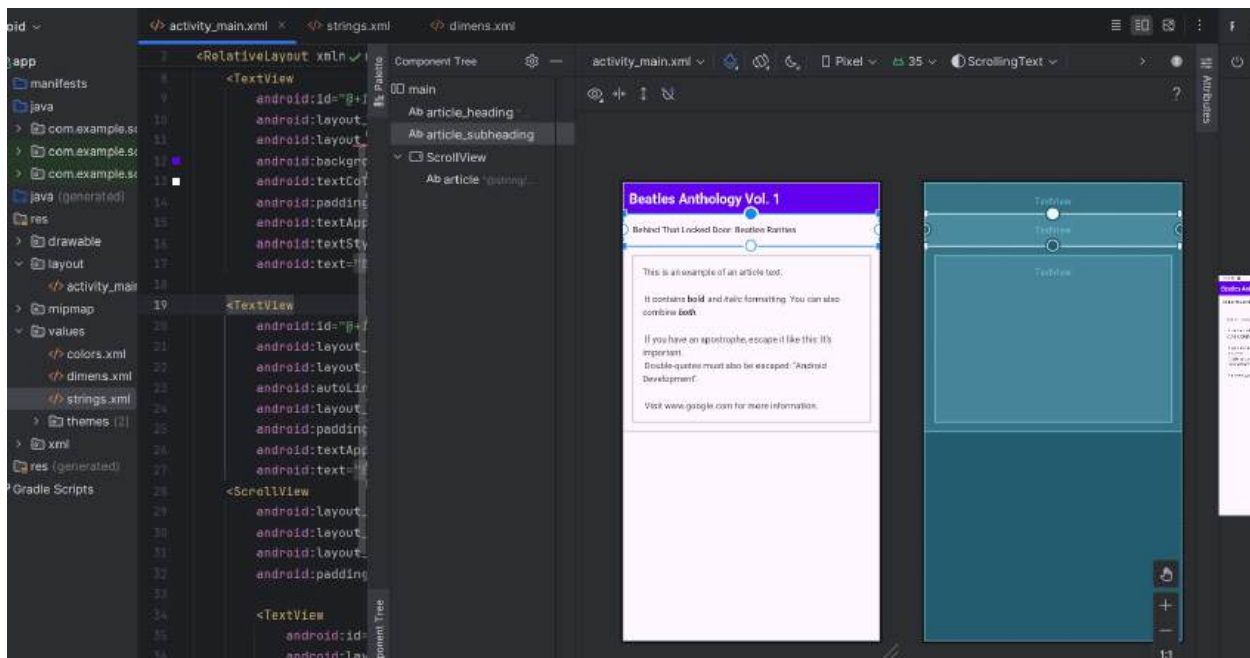
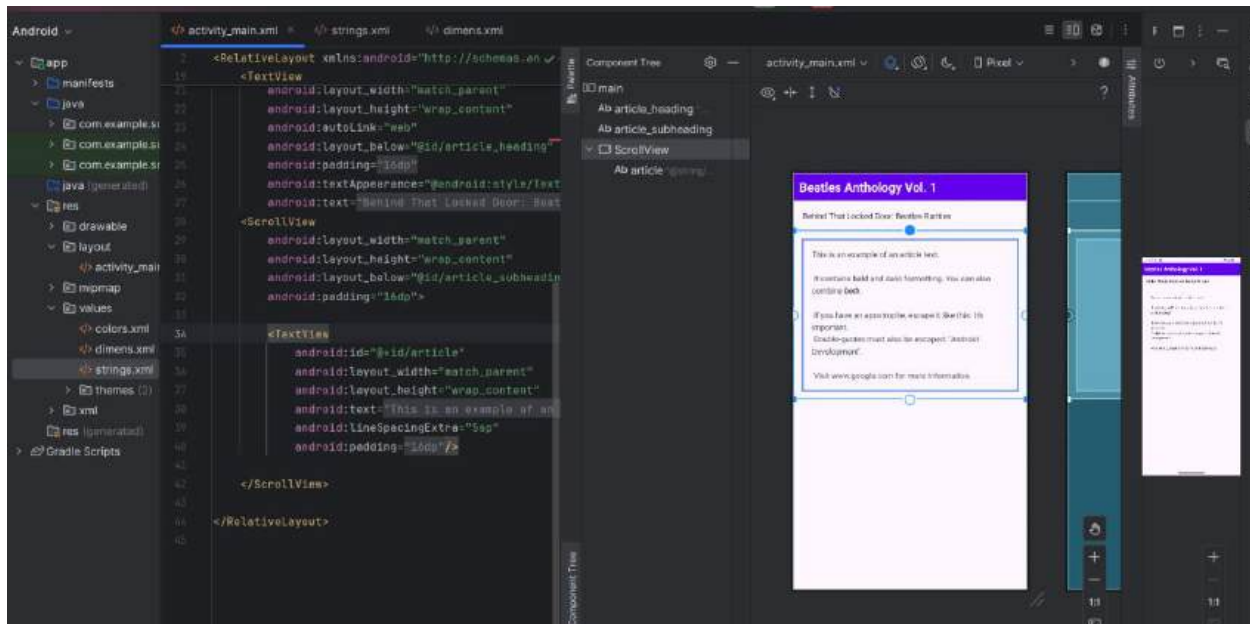


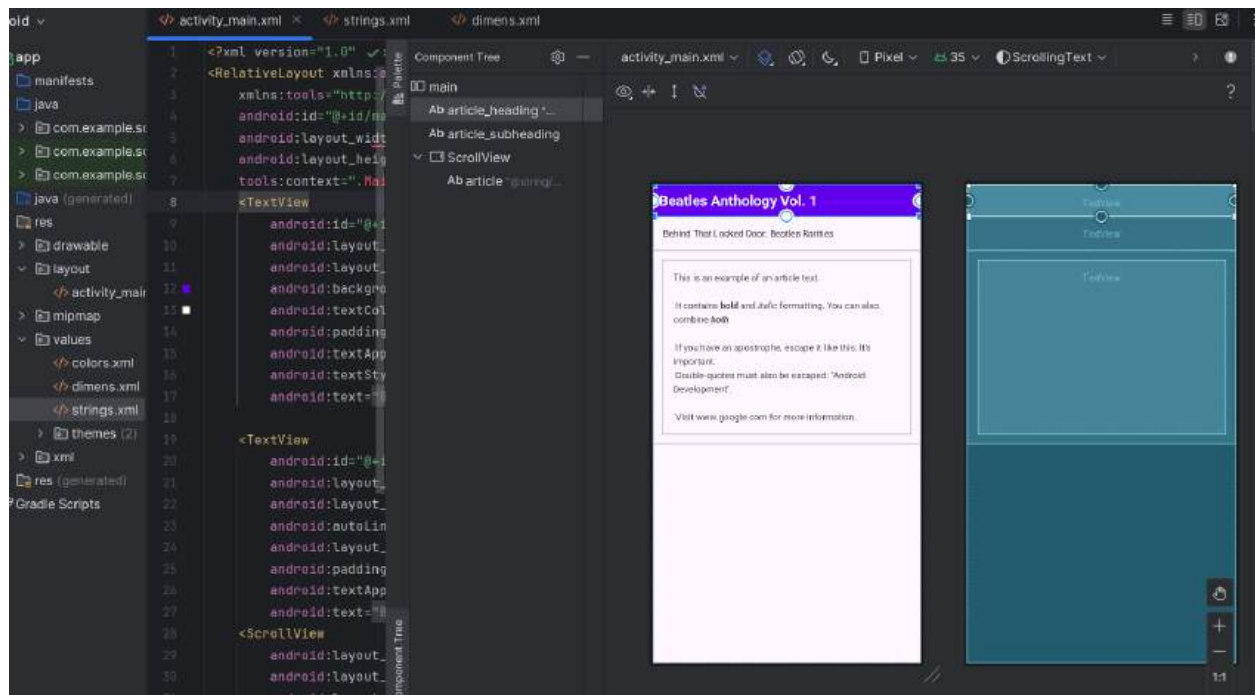
Tác vụ 2: Thêm Scroview và liên kết web đang hoạt động

```

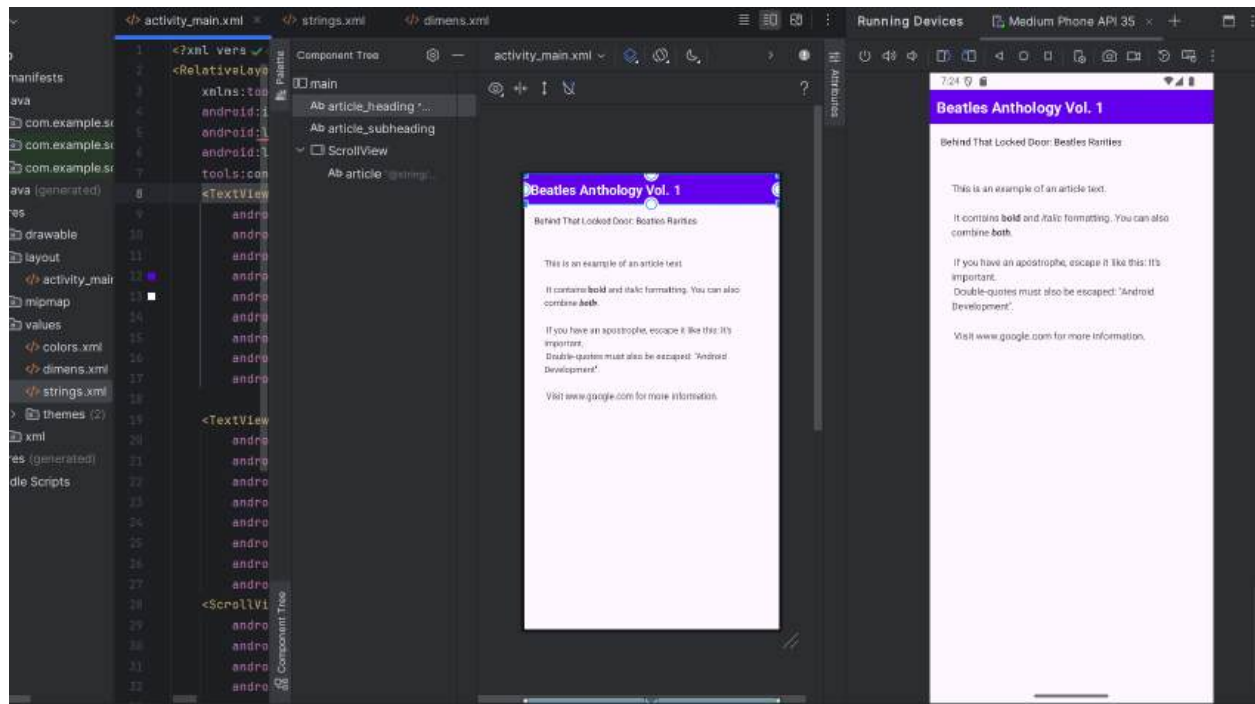
<TextView
    android:id="@+id/article_subheading"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:autoLink="web"
    android:layout_below="@id/article_heading"
    android:padding="16dp"
    android:textAppearance="@android:style/TextAppearance.DeviceDefault"
    android:text="Behind That Locked Door: Beatles Rarities"/>

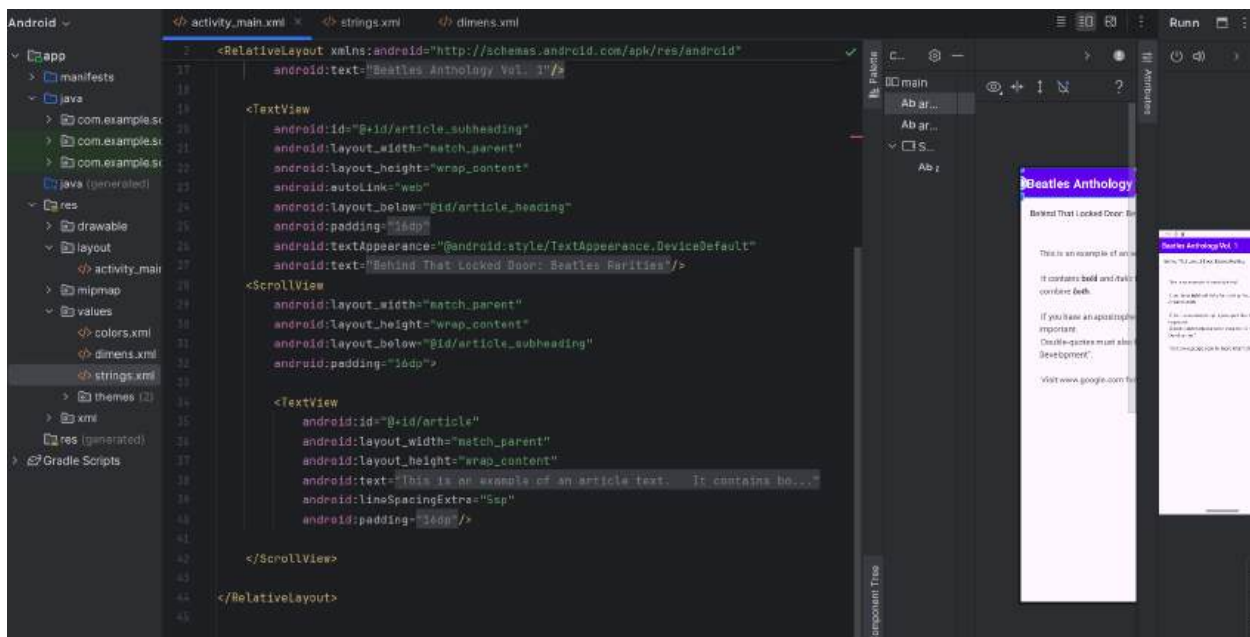
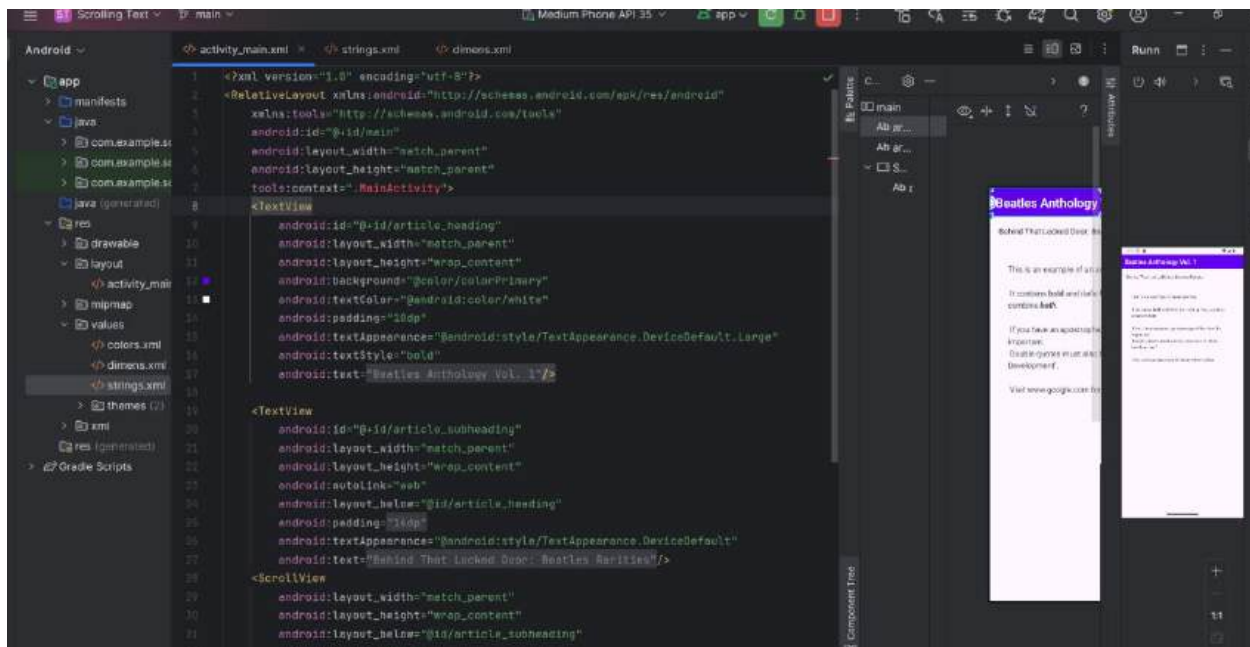
```





2.3 Chạy chương trình





1.5) Tài nguyên có sẵn

Bài 2) Activities

2.1) Activity và Intent

2.2) Vòng đời của Activity và trạng thái

2.3) Intent ngầm định

Bài 3) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ

3.1) Trình gỡ lỗi

3.2) Kiểm thử đơn vị

3.3) Thư viện hỗ trợ

CHƯƠNG 2. TRẢI NGHIỆM NGƯỜI DÙNG

Bài 1) Tương tác người dùng

- 1.1) Hình ảnh có thể chọn**
- 1.2) Các điều khiển nhập liệu**
- 1.3) Menu và bộ chọn**
- 1.4) Điều hướng người dùng**
- 1.5) RecyclerView**

Bài 2) Trải nghiệm người dùng thú vị

- 2.1) Hình vẽ, định kiểu và chủ đề**
- 2.2) Thẻ và màu sắc**
- 2.3) Bố cục thích ứng**

Bài 3) Kiểm thử giao diện người dùng

- 3.1) Espresso cho việc kiểm tra UI**

CHƯƠNG 3. LÀM VIỆC TRONG NỀN

Bài 1) Các tác vụ nền

- 1.1) AsyncTask**
- 1.2) AsyncTask và AsyncTaskLoader**
- 1.3) Broadcast receivers**

Bài 2) Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền

- 2.1) Thông báo**
- 2.2) Trình quản lý cảnh báo**
- 2.3) JobScheduler**

CHƯƠNG 4. LƯU DỮ LIỆU NGƯỜI DÙNG

Bài 1) Tùy chọn và cài đặt

1.1) Shared preferences

1.2) Cài đặt ứng dụng

Bài 2) Lưu trữ dữ liệu với Room

2.1) Room, LiveData và ViewModel

2.2) Room, LiveData và ViewModel