

**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 | 25 |
| 1.3) Trình chỉnh sửa bố cục | 43 |
| 1.4) Văn bản và các chế độ cuộn | 43 |
| 1.5) Tài nguyên có sẵn | 43 |
| Bài 2) Activities..... | 43 |
| 2.1) Activity và Intent..... | 43 |
| 2.2) Vòng đời của Activity và trạng thái | 43 |
| 2.3) Intent ngầm định | 47 |
| Bài 3) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ | 47 |
| 3.1) Trình gỡ lỗi..... | 47 |
| 3.2) Kiểm thử đơn vị | 47 |
| 3.3) Thư viện hỗ trợ | 47 |
| CHƯƠNG 2. Trải nghiệm người dùng..... | 48 |
| Bài 1) Tương tác người dùng | 48 |
| 1.1) Hình ảnh có thể chọn..... | 48 |
| 1.2) Các điều khiển nhập liệu | 48 |
| 1.3) Menu và bộ chọn | 48 |
| 1.4) Điều hướng người dùng | 48 |
| 1.5) RecyclerView | 48 |
| Bài 2) Trải nghiệm người dùng thú vị..... | 48 |
| 2.1) Hình vẽ, định kiểu và chủ đề | 48 |
| 2.2) Thẻ và màu sắc | 48 |
| 2.3) Bố cục thích ứng..... | 48 |
| Bài 3) Kiểm thử giao diện người dùng | 48 |

| | |
|--|----|
| 3.1) Espresso cho việc kiểm tra UI | 48 |
| CHƯƠNG 3. Làm việc trong nền | 48 |
| Bài 1) Các tác vụ nền..... | 48 |
| 1.1) AsyncTask..... | 48 |
| 1.2) AsyncTask và AsyncTaskLoader | 48 |
| 1.3) Broadcast receivers | 48 |
| Bài 2) Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền..... | 48 |
| 2.1) Thông báo..... | 48 |
| 2.2) Trình quản lý cảnh báo | 48 |
| 2.3) JobScheduler | 48 |
| CHƯƠNG 4. Lưu dữ liệu người dùng | 49 |
| Bài 1) Tùy chọn và cài đặt..... | 49 |
| 1.1) Shared preferences | 49 |
| 1.2) Cài đặt ứng dụng | 49 |
| Bài 2) Lưu trữ dữ liệu với Room | 49 |
| 2.1) Room, LiveData và ViewModel..... | 49 |
| 2.2) Room, LiveData và ViewModel..... | 49 |
| 3.1) Trình gowx lỗi | |

CHƯƠNG 1. LÀM QUEN

Bài 1) Tạo ứng dụng đầu tiên

1.1) Android Studio và Hello World

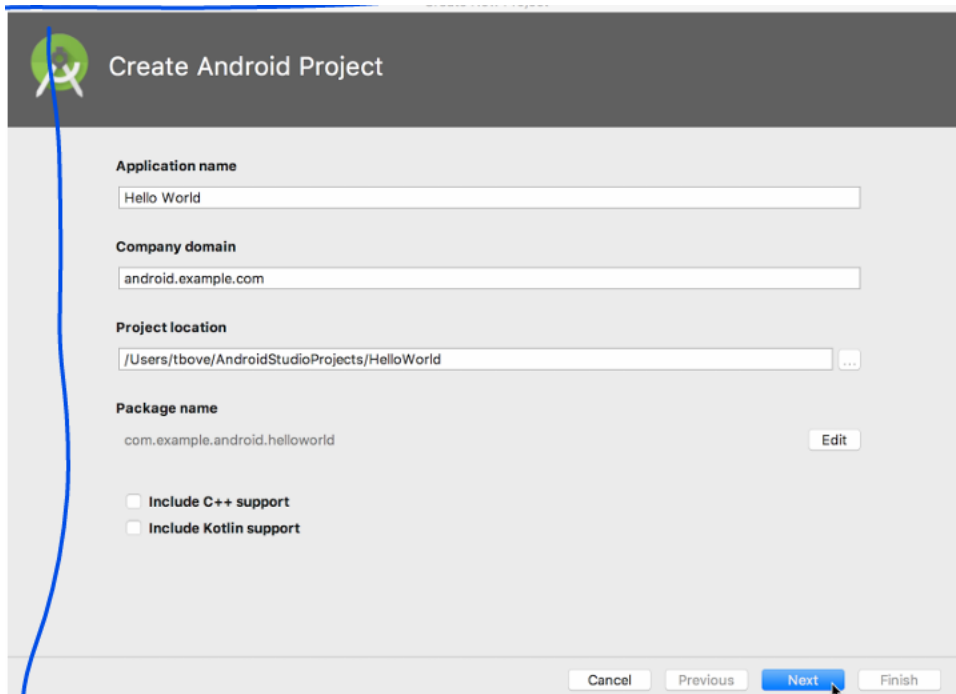
Giới thiệu

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

Sau đây là giao diện của ứng dụng đã hoàn thành:

7:40



Hello World!

Nhiệm vụ 1: Cài đặt Android Studio

Android Studio cung cấp một môi trường phát triển tích hợp (IDE) hoàn chỉnh bao gồm một trình soạn thảo mã nâng cao và một bộ mẫu ứng dụng. Ngoài ra, nó còn chứa các công cụ để phát triển, gỡ lỗi, thử nghiệm và hiệu suất giúp phát triển ứng dụng nhanh hơn và dễ dàng hơn. Bạn có thể thử nghiệm ứng dụng của mình bằng nhiều trình giả lập được cấu hình sẵn hoặc trên thiết bị di động của riêng bạn, xây dựng các ứng dụng sản xuất và phát hành trên cửa hàng Google Play.

Lưu ý: Android Studio liên tục được cải tiến. 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 tính chạy Windows hoặc Linux và cho máy Mac chạy macOS. OpenJDK (Java Development Kit) mới nhất được đóng gói cùng với Android Studio.

Để bắt đầu và chạy Android Studio, trước tiên hãy kiểm tra các yêu cầu hệ thống để đảm bảo rằng hệ thống của bạn đáp ứng các yêu cầu đó. Quá trình cài đặt tương tự nhau cho tất cả các nền tảng. Mọi khác biệt đều được ghi chú bên dưới.

1. Điều hướng đến trang web dành cho nhà phát triển Android và làm theo hướng dẫn để tải xuống và cài đặt Android Studio.

2. Chấp nhận cấu hình mặc định cho tất cả các bước và đảm bảo rằng tất cả các thành phần đều được chọn để cài đặt.

3. Sau khi hoàn tất quá trình cài đặt, Trình hướng dẫn thiết lập sẽ tải xuống và cài đặt 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 Internet của bạn và một số bước có vẻ thừa thãi.

4. Khi quá trình tải xuống hoàn tất, Android Studio sẽ khởi động và bạn đã sẵn sàng tạo dự án đầu tiên của mình.

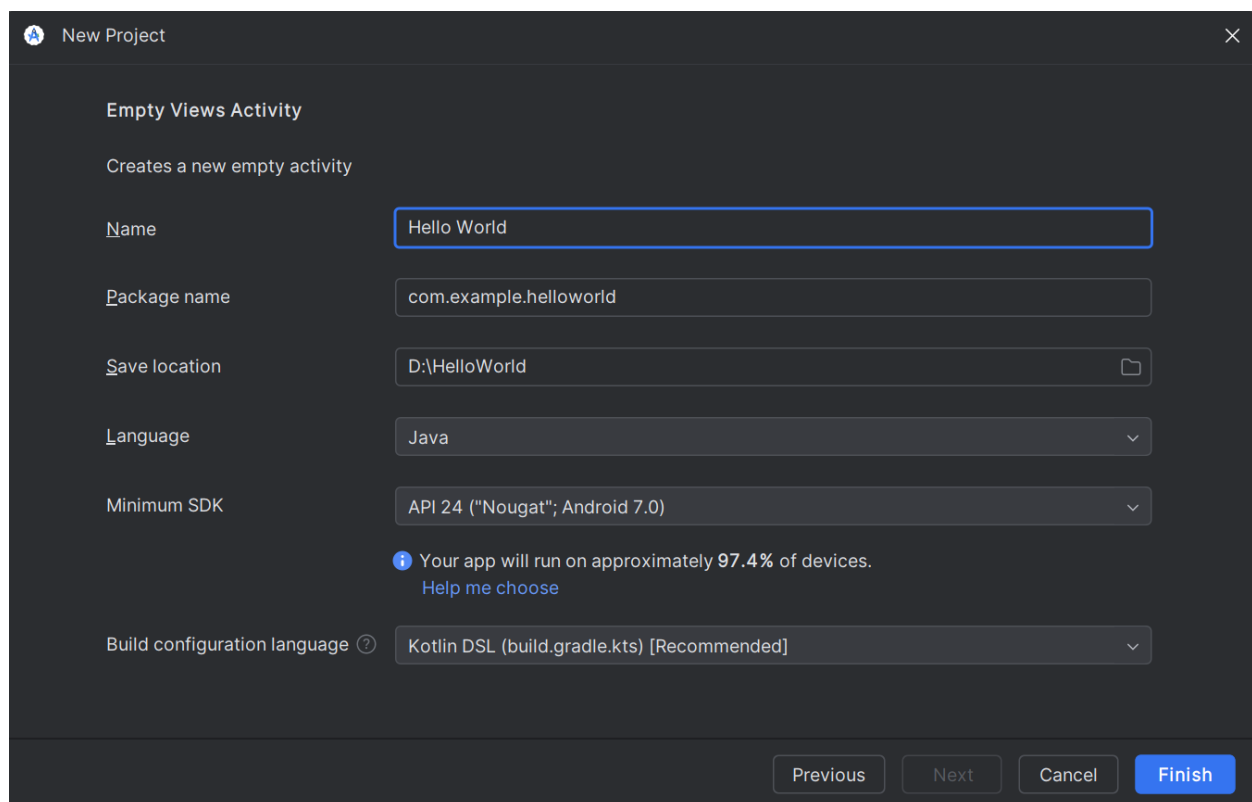
Khắc phục sự cố: Nếu bạn gặp sự cố với quá trình cài đặt, hãy kiểm tra ghi chú phát hành Android Studio hoặc nhờ người hướng dẫn trợ giúp.

Nhiệm vụ 2: Tạo ứng dụng Hello World

Trong nhiệm vụ này, bạn sẽ tạo một ứng dụng hiển thị "Hello World" để xác minh rằng Android Studio đã được cài đặt đúng cách và để tìm hiểu những điều cơ bản về phát triển với 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, hãy nhấp vào Start a new Android Studio project.
3. Trong cửa sổ Create Android Project, hãy nhập Hello World cho tên Ứng dụng



4. Xác minh rằng Project location mặc định là nơi bạn muốn lưu trữ ứng dụng Hello World và các dự án Android Studio khác hoặc thay đổi thành thư mục bạn muốn.

5. Chấp nhận android.example.com mặc định cho Tên miền công ty hoặc tạo một tên miền công ty duy nhất.

Nếu bạn 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 ứng dụng của bạn sau này là công việc bổ sung.

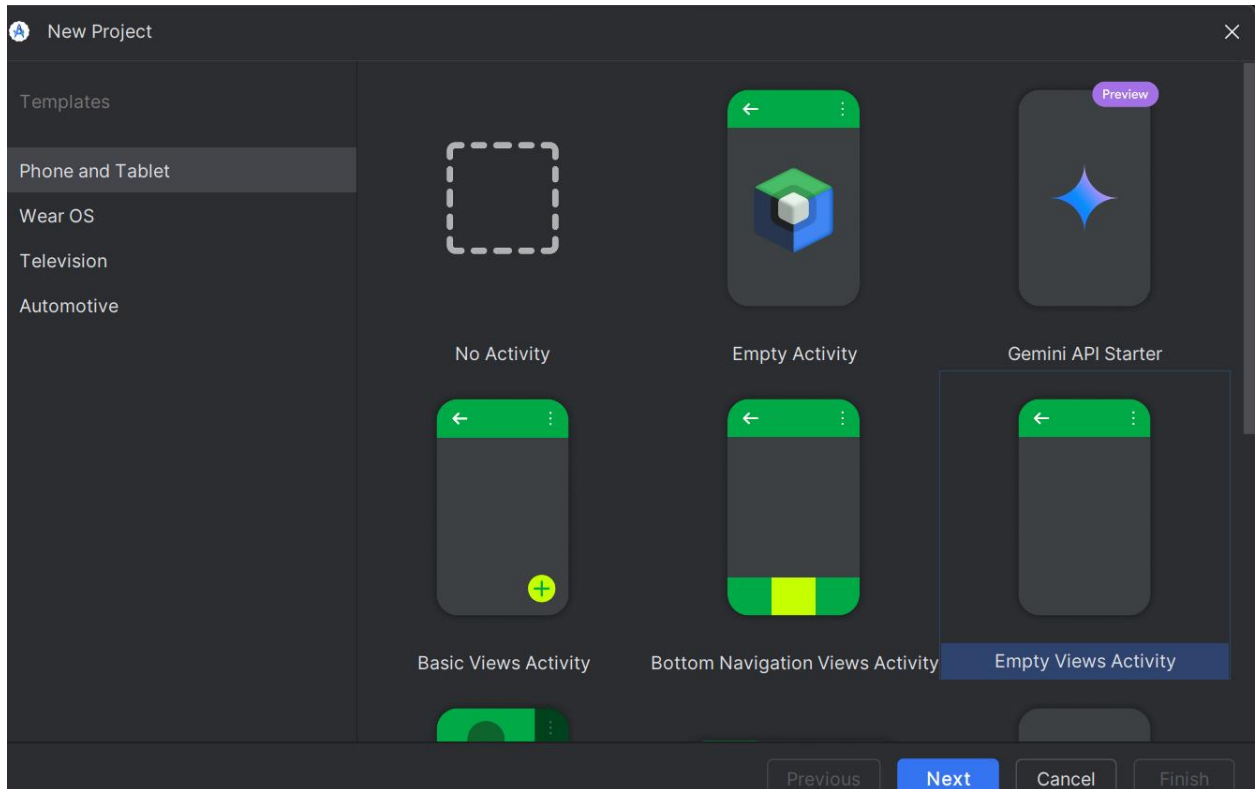
6. Bỏ chọn các tùy chọn Bao gồm hỗ trợ C++ và Bao gồm hỗ trợ Kotlin, rồi nhấp vào Next.

7. Trên màn hình Thiết bị Android đích, Điện thoại và Máy tính bảng phải được chọn. Đảm bảo rằng API 15: Android 4.0.3 IceCreamSandwich được đặt làm SDK tối thiểu; nếu không, hãy sử dụng menu bật lên để đặt.

Đây là các thiết lập được sử dụng bởi các ví dụ trong các bài học cho khóa học này. Tính đến thời điểm viết bài này, các thiết lập này giúp ứng dụng Hello World của bạn tương thích với 97% thiết bị Android đang hoạt động trên Cửa hàng Google Play.

8. Bỏ chọn mục Bao gồm hỗ trợ ứng dụng tức thì và tất cả các tùy chọn khác. Sau đó, nhấp vào Tiếp theo. Nếu dự án của bạn yêu cầu các thành phần bổ sung cho SDK mục tiêu đã chọn, Android Studio sẽ tự động cài đặt chúng.

9. Cửa sổ Thêm hoạt động xuất hiện. Activity là một điều duy nhất, tập trung mà người dùng có thể thực hiện. Đây là thành phần quan trọng của bất kỳ ứng dụng Android nào. Activity thường có bố cục liên quan đến nó để xác định cách các thành phần UI 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 Activity như được hiển thị bên dưới 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, Activity trống do mẫu cung cấp được đặt tên là MainActivity. Bạn có thể thay đổi tên 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 tùy chọn 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. Nhấp vào Finish.

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

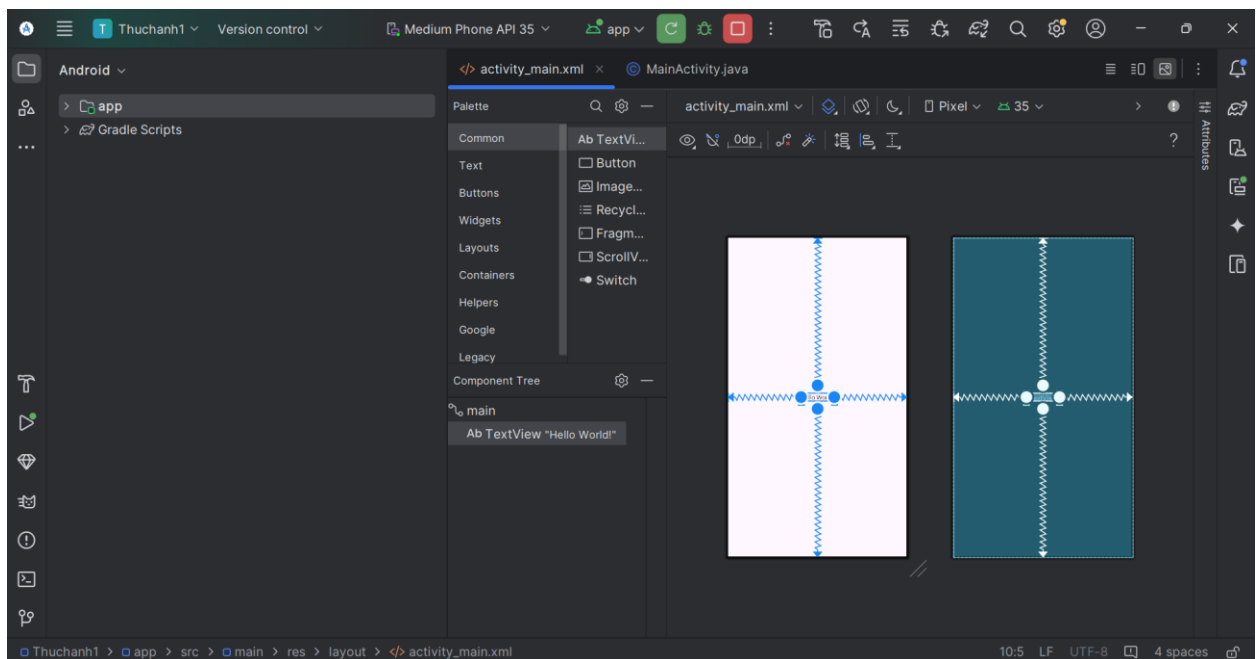
Mẹo: Xem trang Cấu hình nhà phát triển bản dựng của bạn để biết thông tin chi tiết.

Bạn cũng có thể thấy thông báo "Mẹo trong ngày" với các phím tắt và các mẹo hữu ích khác. Nhấp vào

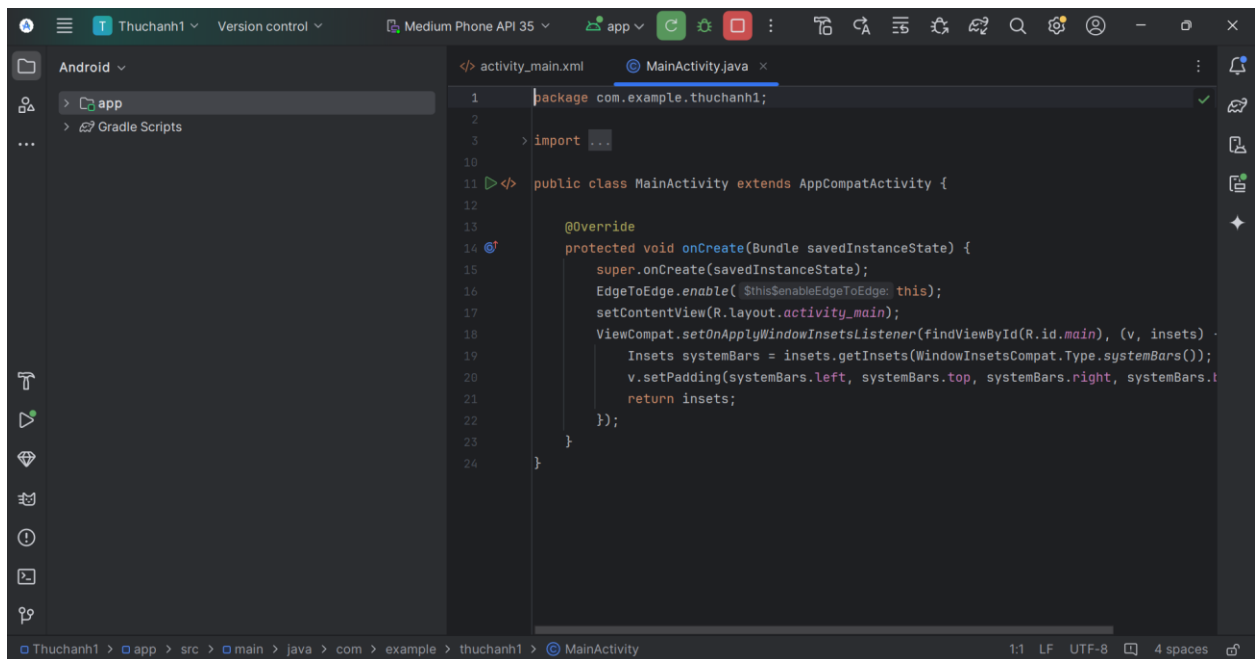
Đóng để đóng thông báo.

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

1. Nhấp vào tab `activity_main.xml` để xem trình chỉnh sửa bố cục.
2. Nhấp vào tab Thiết kế của trình chỉnh sửa bố cục, nếu chưa chọn, để hiển thị bản trình bày đồ họa của bố cục như được hiển thị bên dưới.



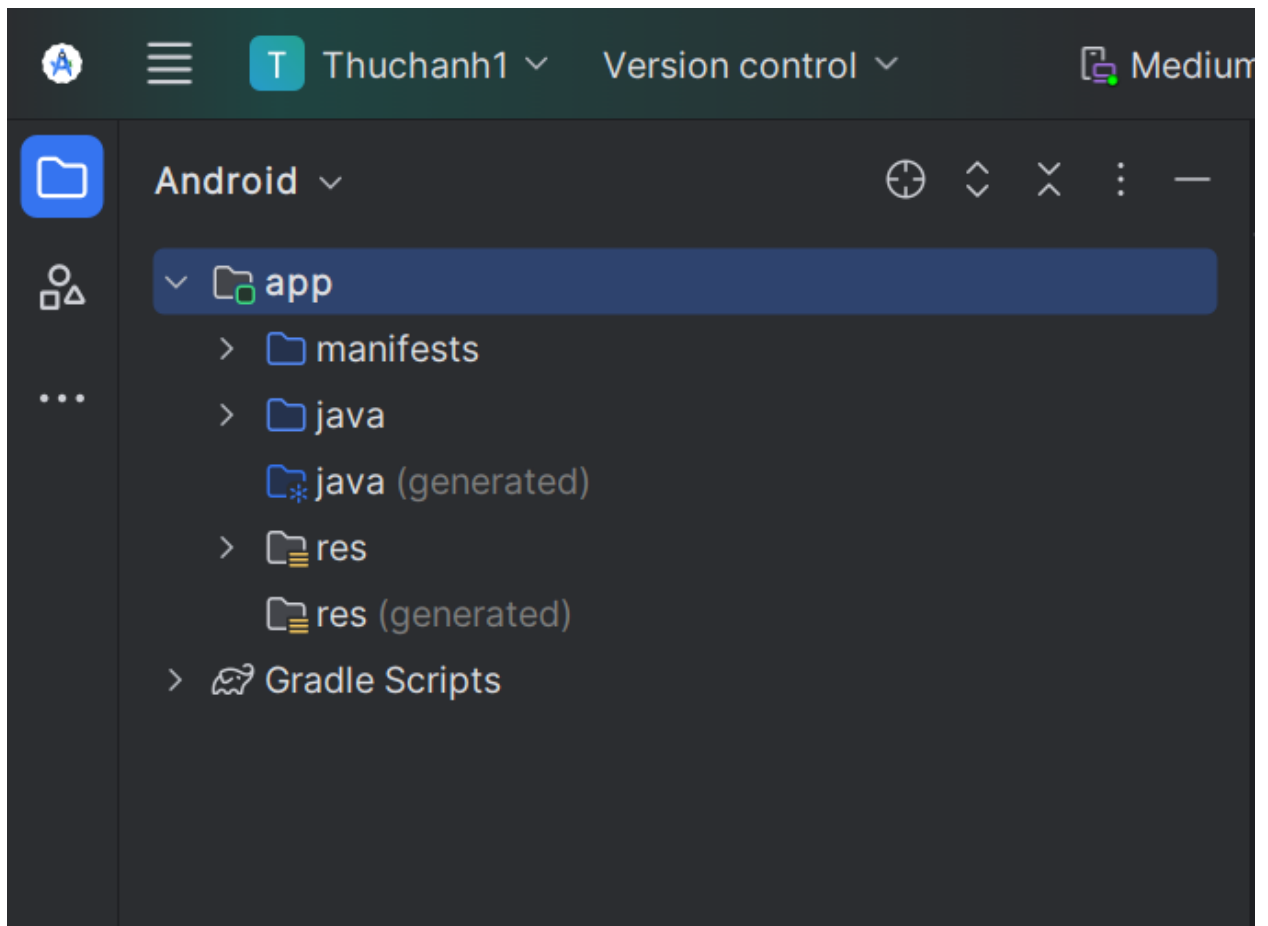
3. Nhấp vào tab `MainActivity.java` để xem trình soạn thảo mã như được hiển thị bên dưới



2.2 Khám phá ngăn Project > Android

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

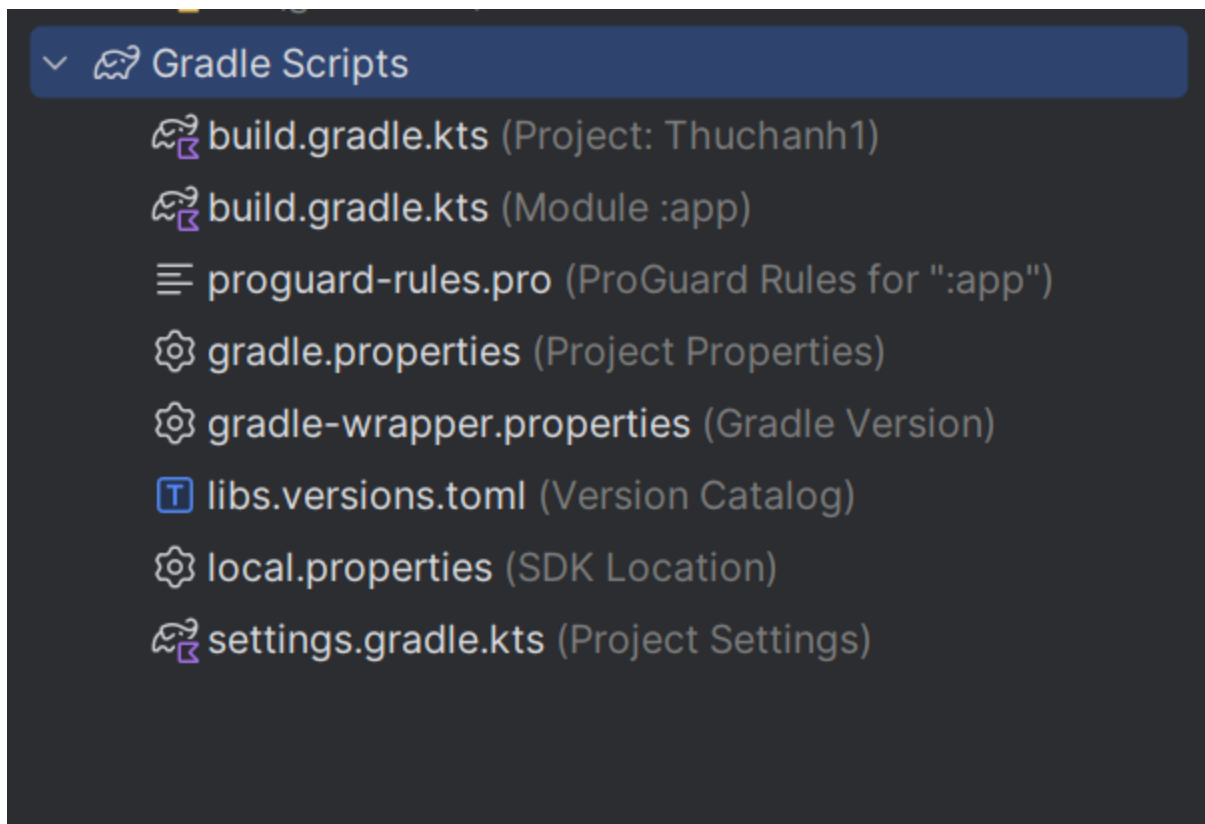
1. Nếu chưa chọn, hãy nhấp vào tab Project trong cột tab dọc ở bên trái của cửa sổ Android Studio. Ngăn Project sẽ xuất hiện.
2. Để xem dự án trong hệ thống phân cấp dự án Android chuẩn, hãy chọn Android từ menu bật lên ở đầu ngăn Project, như được hiển thị bên dưới.



2.3 Khám phá thư mục Gradle Scripts

Hệ thống 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 của mình dưới dạng 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 được hiển thị bên dưới.



Thực hiện theo các bước sau để khám phá hệ thống Gradle:

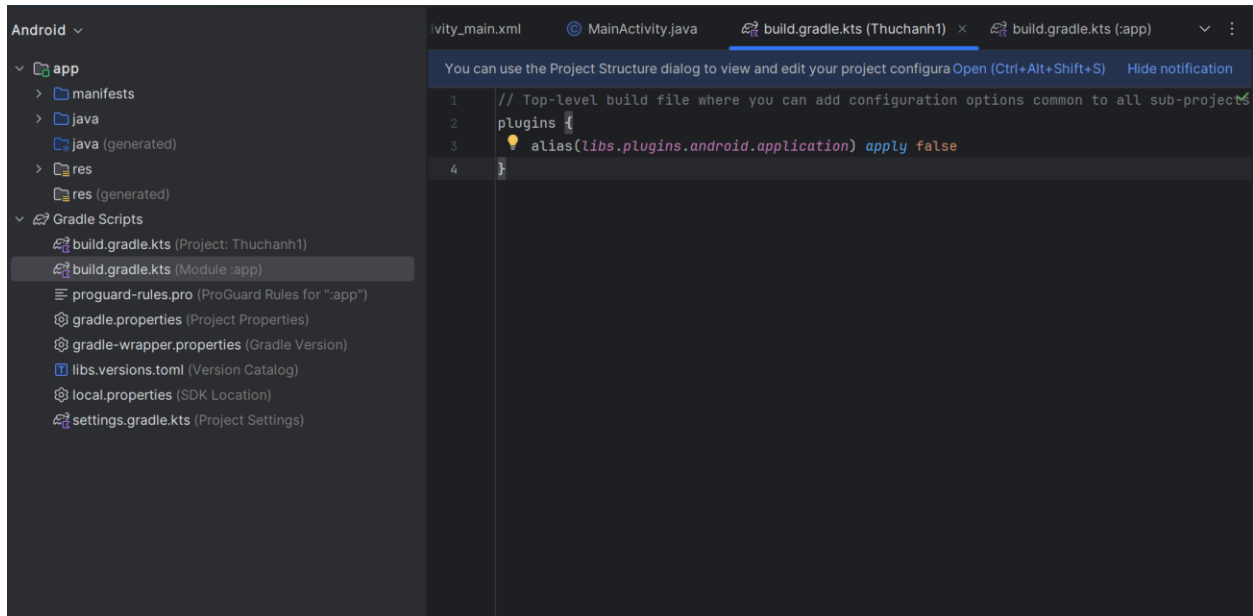
1. Nếu thư mục Gradle Scripts chưa được mở rộng, hãy nhấp vào hình tam giác để mở rộng thư mục.

Thư mục này chứa tất cả các tệp mà hệ thống dựng cần.

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 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 tệp.

Theo mặc định, tệp dựng cấp cao nhất sử dụng khối buildscript để xác định các kho lưu trữ Gradle và các phụ thuộc chung cho tất cả các mô-đun trong dự án. Khi phụ thuộc của bạn là thứ gì đó khác với 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 các vị trí kho lưu trữ:



3. Tìm tệp build.gradle(Module:app).

Ngoài tệp build.gradle cấp dự án, mỗi mô-đun đều có 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). Định cấu hình các 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 các cài đặt trong tệp AndroidManifest.xml hoặc tệp build.gradle cấp cao nhất.

Tệp này thường là tệp cần chỉnh sửa khi thay đổi cấu hình cấp ứng dụng, chẳng hạn như khai báo các phụ thuộc trong phần phụ thuộc. Bạn có thể khai báo phụ thuộc thư viện bằng một trong một số cấu hình phụ thuộc khác nhau. Mỗi cấu hình phụ thuộc cung cấp cho Gradle các hướng dẫn khác nhau về cách sử dụng thư viện. Ví dụ: câu lệnh triển khai `fileTree(dir: 'libs', include: ['*.jar'])` thêm một 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.thuchanh1"
7      compileSdk = 35
8
9      defaultConfig {
10         applicationId = "com.example.thuchanh1"
11         minSdk = 24
12         targetSdk = 35
13         versionCode = 1
14         versionName = "1.0"
15
16         testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
17     }

```

```

19     buildTypes {
20         release {
21             isMinifyEnabled = false
22             proguardFiles(
23                 getDefaultProguardFile( name: "proguard-android-optimize.txt"),
24                 "proguard-rules.pro"
25             )
26         }
27     }
28     compileOptions {
29         sourceCompatibility = JavaVersion.VERSION_11
30         targetCompatibility = JavaVersion.VERSION_11
31     }
32 }

```

```

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  }

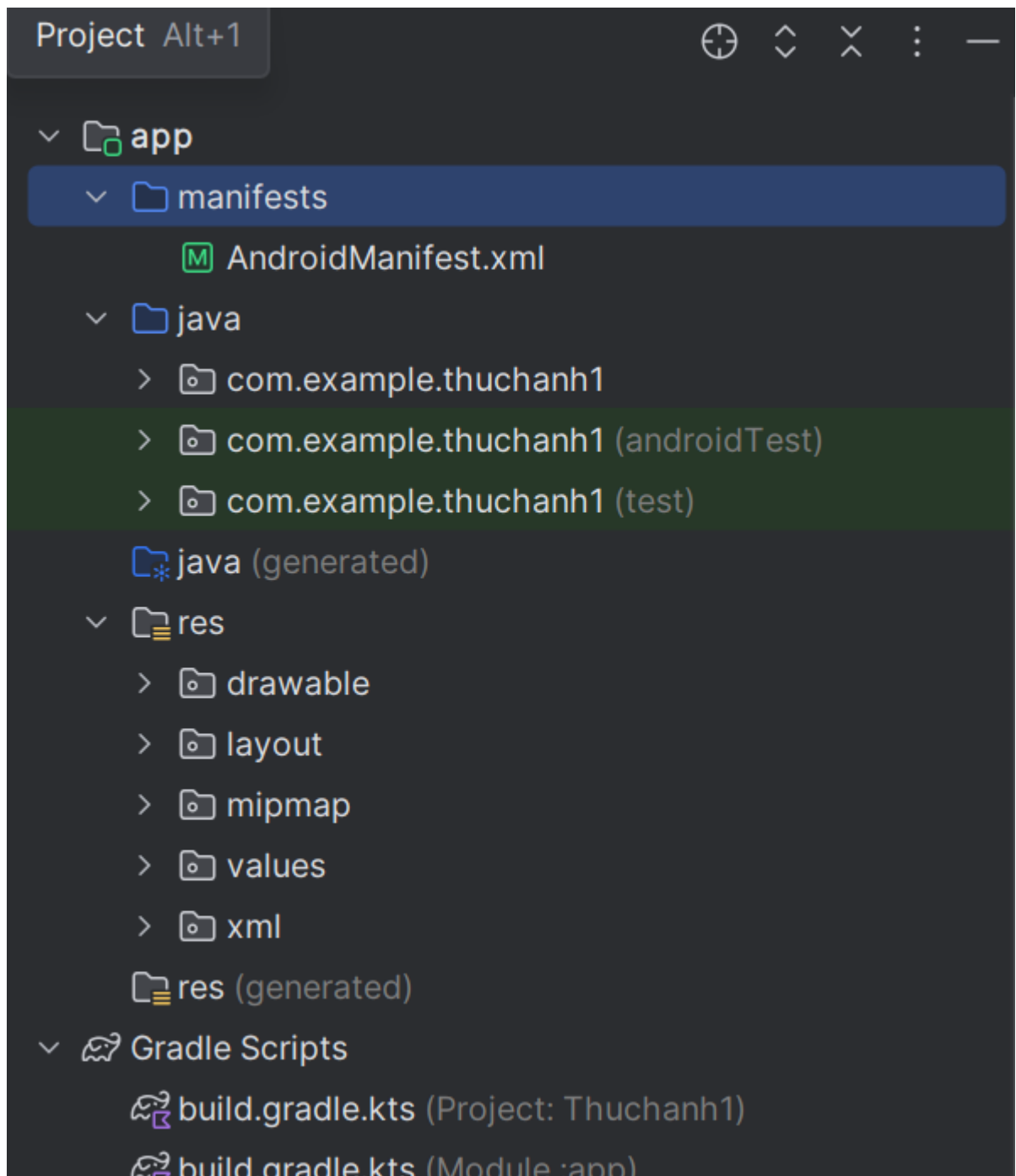
```

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

2.4 Khám phá các thư mục app và res

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

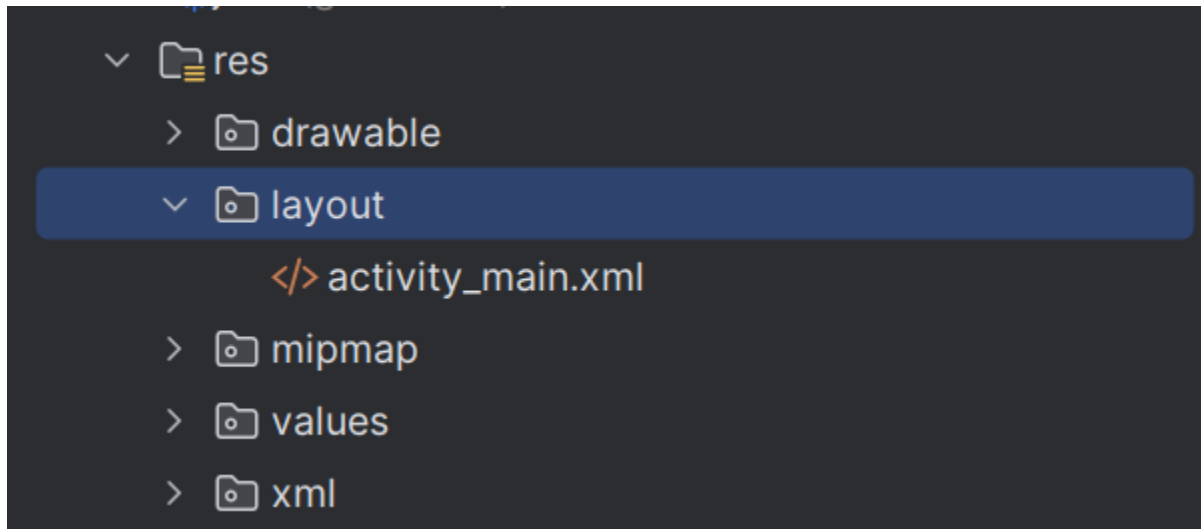
1. Mở rộng thư mục app, thư mục java và thư mục com.example.android.helloworld để xem tệp java MainActivity. Nhấp đúp vào tệp sẽ mở tệp đó trong trình chỉnh sửa mã.



Thư mục java bao gồm các tệp lớp Java trong ba thư mục con, như thể hiện trong hình trên. Thư mục com.example.hello.helloworld (hoặc tên miền bạn đã chỉ định) chứa tất cả các tệp cho một gói ứng dụng. Hai thư mục còn lại được sử dụng để thử nghiệm và được mô tả trong một bài học khác. Đối với ứng dụng Hello World, chỉ có một gói và nó chứa MainActivity.java. Tên của Hoạt động đầu tiên (màn hình) mà người dùng nhìn thấy, cũng khởi tạo các tài nguyên trên toàn ứng dụng,

thường được gọi là MainActivity (phần mở rộng tệp bị bỏ qua trong ngăn Dự án > Android).

2. Mở rộng thư mục res và thư mục layout, rồi nhấp đúp vào tệp activity_main.xml để mở tệp đó trong trình chỉnh sửa layout.



Thư mục res chứa các tài nguyên, chẳng hạn như layout, chuỗi và hình ảnh. Activity thường được liên kết với layout của các chế độ xem UI được định nghĩa là tệp XML. Tệp này thường được đặt tên theo Activity của nó.

2.5 Khám phá thư mục manifests

Thư mục manifests 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 AndroidManifest.xml.

Tệp AndroidManifest.xml mô tả tất cả các thành phần của ứng dụng Android của bạn. Tất cả các thành phần cho một ứng dụng, chẳng hạn như mỗi Activity, phải được khai báo trong tệp XML này. Trong các bài học khác của khóa học, bạn sẽ sửa đổi tệp này để thêm các tính năng và quyền tính năng. Để biết phần giới thiệu, hãy xem App Manifest tổng quan.

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

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

Khi sử dụng Trình quản lý AVD, bạn sẽ xác định các đặc điểm phần cứng của thiết bị, cấp độ API, bộ nhớ, giao diện và các thuộc tính khác của thiết bị và lưu dưới dạng thiết bị ảo. Với các thiết bị ảo, bạn có thể thử nghiệm ứng dụng trên các cấu hình thiết bị khác nhau (như máy tính bảng và điện thoại) với các cấp độ API khác nhau mà không cần sử dụng thiết bị vật lý.

3.1 Tạo thiết bị ảo Android (AVD)

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

1. Trong Android Studio, chọn Tools > Android > AVD Manager hoặc nhấp vào biểu tượng Trình quản lý AVD trên thanh công cụ. Màn hình Thiết bị ảo của bạn 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 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 cấu hình sẵn. Đối với mỗi thiết bị, bảng cung cấp một cột cho kích thước màn hình chéo (Size), độ phân giải màn hình tính bằng pixel (Resolution) và mật độ pixel (Density).

3. Chọn một thiết bị như Nexus 5x hoặc Pixel XL, rồi nhấp vào Next. Màn hình System Image xuất hiện

4. Nhấp vào tab Recommended nếu chưa chọn và chọn phiên bản hệ thống Android nào để chạy trên thiết bị ảo (như Oreo)

Có nhiều phiên bản hơn so với phiên bản hiển thị trong tab Recommended. Hãy xem các tab x86 Images và Other Images để xem chúng.

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

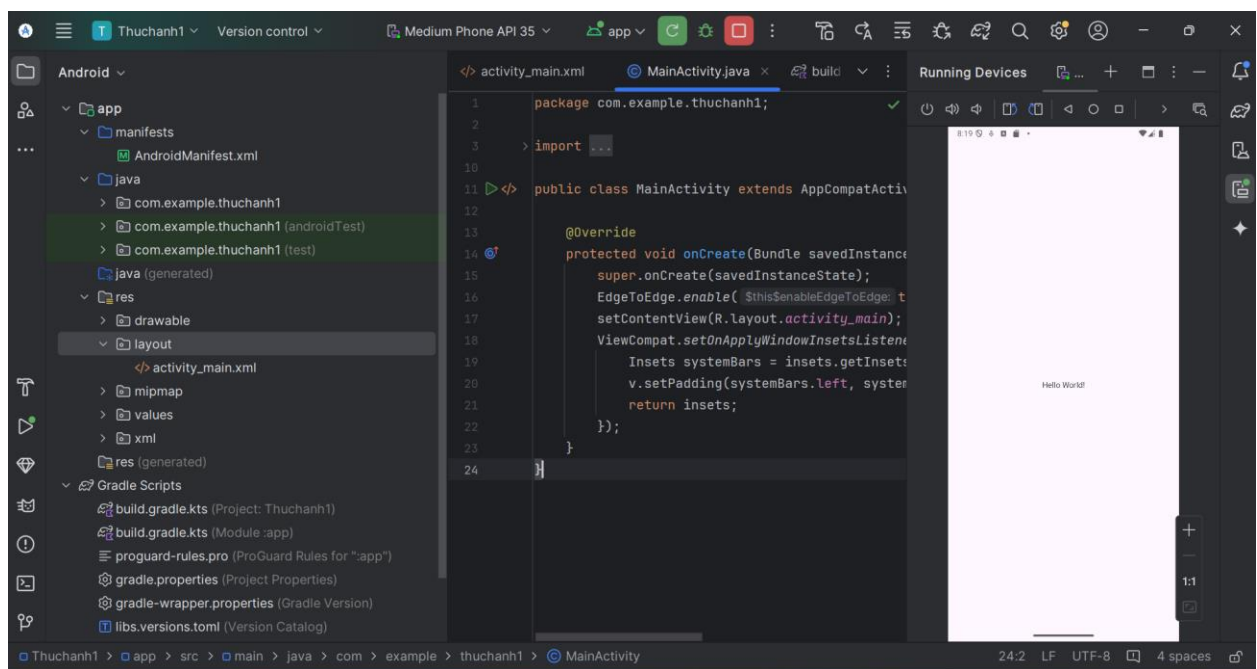
5. Sau khi chọn ảnh hệ thống, hãy nhấp vào Next. Cửa sổ Android Virtual Device (AVD) xuất hiện. Bạn cũng có thể thay đổi tên của AVD. Kiểm tra cấu hình của bạn và nhấp vào Hoàn tất.

3.2 Chạy ứng dụng trên thiết bị ảo

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

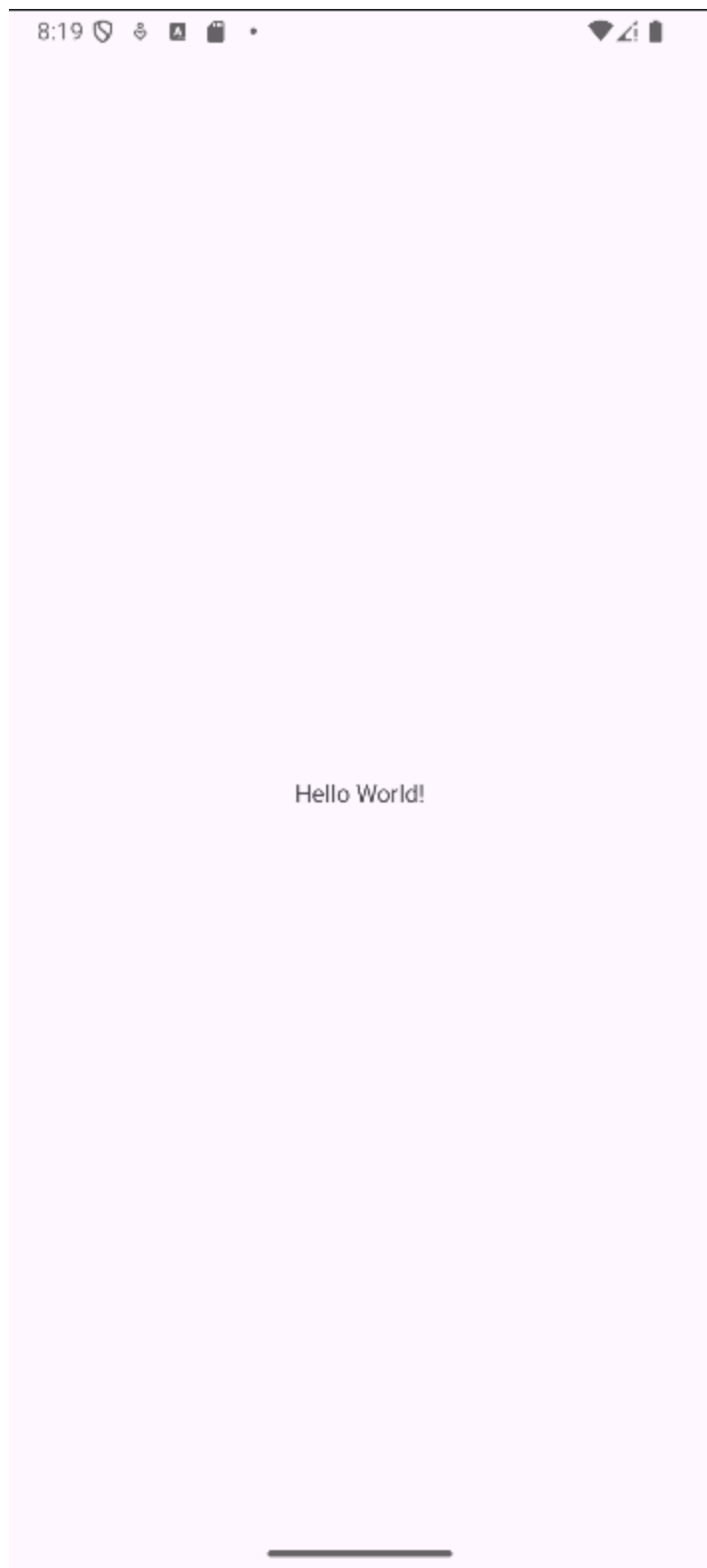
1. Trong Android Studio, chọn Chạy > Chạy ứng dụng hoặc nhấp vào biểu tượng Chạy trên thanh công cụ.

2. Cửa sổ Chọn mục tiêu triển khai, bên dưới Thiết bị ảo khả dụng, chọn thiết bị ảo mà bạn vừa tạo và nhấp vào OK.



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

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



Mẹo: Khi thử nghiệm trên thiết bị ảo, bạn nên khởi động thiết bị một lần, ngay khi bắt đầu phiên của mình. Bạn không nên đóng thiết bị cho đến khi hoàn tất thử

nghiệm ứng dụng, 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 Thoát 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 luôn phải kiểm tra ứng dụng của mình trên cả thiết bị ảo và 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 của bạn 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 phù hợp cho thiết bị của mình. Đối với trình điều khiển USB chạy trên Windows, hãy xem Trình điều khiển USB OEM.

4.1 Bật gỡ lỗi USB

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

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

1. Trên thiết bị của bạn, hãy mở Settings, tìm kiếm About phone, nhấp vào About phone và chạm vào Build bảy lần.

2. Quay lại màn hình trước đó (Cài đặt / Hệ thống). Developer options xuất hiện trong danh sách.

Chạm vào Developer options.

3. Chọn Gỡ lỗi USB.

4.2 Chạy ứng dụng của bạn trên thiết bị

Bây giờ bạn có thể kết nối thiết bị của mình và chạy ứng dụng từ Android Studio.

1. Kết nối thiết bị của bạn với máy phát triển bằng cáp USB.
2. Nhấp vào nút Chạy trên thanh công cụ. Cửa sổ Chọn Mục tiêu Triển khai mở ra với danh sách các trình giả lập và thiết bị được kết nối khả dụng.
3. Chọn thiết bị của bạn và nhấp vào OK.

Android Studio cài đặt và chạy ứng dụng trên thiết bị của bạn.

Xử lý sự cố

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

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

Nếu máy tính của bạn vẫn không tìm thấy thiết bị hoặc tuyên bố thiết bị "không được phép", hãy làm theo các bước sau:

1. Rút phích cắm thiết bị.
2. Trên thiết bị, hãy mở Tùy chọn nhà phát triển trong ứng dụng Cài đặt.
3. Nhấn vào Thu hồi quyền gỡ lỗi USB.
4. Kết nối lại thiết bị với máy tính của bạn.
5. Khi được nhắc, hãy cấp quyền.

Bạn có thể cần cài đặt trình điều khiển USB phù hợp cho thiết bị của mình. Xem tài liệu Sử dụng thiết bị phần cứng

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

Trong nhiệm vụ này, bạn sẽ thay đổi một số thông tin về cấu hình ứng dụng trong tệp build.gradle(Module:app) để tìm hiểu cách thực hiện 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

Thực hiện theo các bước sau:

1. Mở rộng thư mục Gradle Scripts nếu thư mục này chưa mở và nhấp đúp vào tệp build.gradle(Module:app).

Nội dung của tệp sẽ xuất hiện trong trình chỉnh sửa mã.

2. Trong khối defaultConfig, hãy thay đổi giá trị của minSdkVersion thành 17 như hiển thị bên dưới (ban đầu được đặt thành 15).

Trình chỉnh sửa mã sẽ hiển thị thanh thông báo ở trên cùng với liên kết Đồng bộ hóa ngay.

5.2 Đồng bộ hóa cấu hình Gradle mới

Khi bạn thực hiện 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 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 nhấp vào Đồng bộ hóa ngay trên thanh thông báo xuất hiện khi thực hiện thay đổi (như được hiển thị trong hình trước) hoặc nhấp vào biểu tượng Đồng bộ hóa dự án với tệp Gradle trên thanh công cụ.

Khi quá trình đồng bộ hóa Gradle hoàn tất, thông báo Gradle build finished sẽ xuất hiện ở góc dưới bên trái của cửa sổ Android Studio.

Để tìm hiểu sâu hơn về Gradle, hãy xem tài liệu Tổng quan về hệ thống bản dựng và Cấu hình bản dựng Gradle

.

1.2) Giao diện người dùng tương tác đầu tiên

Giới thiệu

Giao diện người dùng(UI) xuất hiện trên màn hình của 1 thiết bị Android bao gồm 1 hệ thống phân cấp các đối tượng được gọi là views --- mỗi thành phần trên màn hình đều là 1 view. Lớp view đại diện cho khối xây dựng cơ bản của tất cả các

thành phần UI, và là lớp cơ sở cho các lớp cung cấp các thành phần UI có tính tương tác như là các nút bấm, các hộp kiểm, và các trường nhập văn bản. Các lớp con thường được sử dụng của View, được mô tả trong nhiều 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 thành phần có thể nhấp khác (chẳng hạn như **RadioButton**, **CheckBox**, và **Spinner**) để cung cấp hành vi tương tác.
- ☐ **ScrollView** và **RecyclerView** để hiển thị danh sách có thể cuộn.
- ☐ **ImageView** để hiển thị hình ảnh.
- ☐ **ConstraintLayout** và **LinearLayout** để chứa các phần tử **View** khác và định vị chúng trên màn hình.

Mã Java điều khiển giao diện người dùng (UI) được nằm trong một lớp mở rộng từ Activity. Một Activity thường được liên kết với một bố cục UI được định nghĩa trong một tệp XML. Tệp XML này thường được đặt tên theo Activity tương ứng và xác định cách sắp xếp các phần tử View trên màn hình.

Ví dụ, trong ứng dụng Hello World, lớp MainActivity sẽ hiển thị một bố cục được định nghĩa trong tệp activity_main.xml, bao gồm một TextView có nội dung "Hello World".

Trong các ứng dụng phức tạp hơn, một Activity triển khai các hoạt động để phản hồi thao tác nhấn của người dùng, vẽ nội dung đồ họa, hoặc yêu cầu dữ liệu của 1 cơ sở dữ liệu hoặc internet. Bạn học thêm về lớp hoạt động trong 1 bài học khác.

Trong phần thực hành này, bạn sẽ tìm hiểu cách tạo ứng dụng tương tác đầu tiên của mình—một ứng dụng cho phép người dùng sự tương tác. Bạn tạo một ứng dụng bằng 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 bạn có thể hoàn thành các bài thực hành khác trong khóa học này.

Những gì bạn nên biết

Bạn nên làm quen với:

- Cách cài đặt và mở Android Studio
- Cách tạo ứng dụng HelloWorld

- Cách chạy ứng dụng HelloWorld

Những gì bạn sẽ học

- Tạo 1 ứng dụng với giao diện hành vi
- Cách sử dụng trình chỉnh sửa bố cục để thiết kế bố cục
- Cách chỉnh sửa bố cục trong XML
- Rất nhiều thuật ngữ mới. Kiểm tra bảng thuật ngữ từ vựng và khái niệm cho thân thiện định nghĩa. Cách tạo 1 ứng dụng với giao diện hành vi

Những gì bạn sẽ làm

- Tạo một ứng dụng và thêm hai phần tử Button cùng một TextView vào bố cục.
- Điều chỉnh từng phần tử trong ConstraintLayout để ràng buộc chúng với lề và các phần tử khác.
- Thay đổi các thuộc tính của phần tử giao diện người dùng (UI).
- Chỉnh sửa bố cục của ứng dụng trong XML.
- Tách các chuỗi cố định thành tài nguyên chuỗi (string resources).
- Triển khai phương thức xử lý sự kiện nhấn 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, ứng dụng sẽ hiển thị một thông báo ngắn (Toast) trên màn hình. Khi nhấn vào nút thứ hai, bộ đếm số lần nhấn sẽ tăng lên và hiển thị trong TextView, bắt đầu từ số 0.

Dưới đây là giao diện của ứng dụng sau khi hoàn thành:

Nhiệm vụ 1: Tạo và khám phá một dự án mới

Trong bài thực hành này, bạn sẽ thiết kế và triển khai một dự án cho ứng dụng HelloToast. Một liên kết đến mã giải pháp được cung cấp ở cuối.

1.1 Tạo dự án Android Studio

14. Khởi động Android Studio và tạo một dự án mới với các tham số sau:

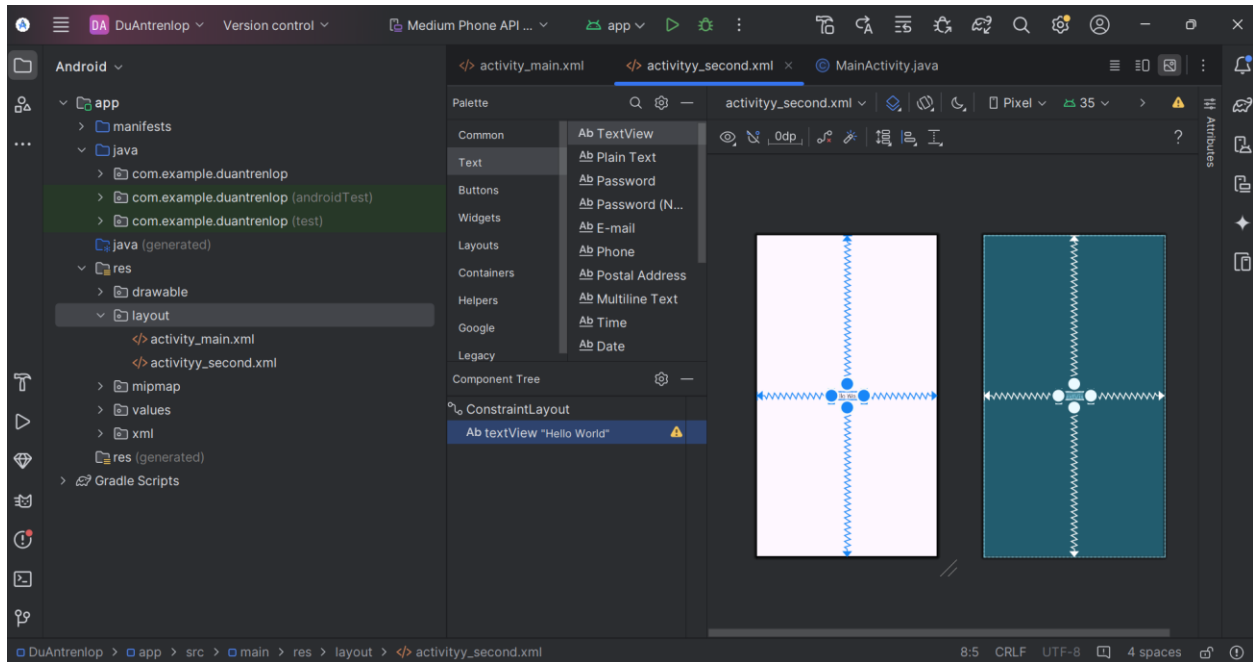
| Thuộc tính | Giá trị |
|------------------------------------|---|
| Tên ứng dụng | Hello Toast |
| Tên công ty | com.example.android (hoặc miền của bạn) |
| Thiết bị hỗ trợ | Điện thoại và máy tính bảng |
| SDK tối thiểu | API15: Android 4.0.3 IceCreamSanwich |
| Mẫu (Template) | Hoạt động trống |
| Chọn hộp "Generate Layout file" | Đã chọn |
| Chọn hộp "Backwards Compatibility" | Đã chọn |

15. Chọn run > run app hoặc nhấp vào biểu tượng chạy trên thanh công cụ để xây dựng và thực thi ứng dụng trên trình giả lập hoặc thiết bị của bạn.

1.2 Khám phá trình chỉnh sửa bố cục

Android Studio cung cấp trình chỉnh sửa bố cục để nhanh chóng xây dựng bố cục của các thành phần giao diện người dùng (UI) của ứng dụng. Trình chỉnh sửa này cho phép bạn kéo các thành phần vào chế độ xem thiết kế trực quan và bản thiết kế, định vị chúng trong bố cục, thêm ràng buộc và đặt thuộc tính. Ràng buộc xác định vị trí của thành phần UI trong bố cục. Ràng buộc biểu thị kết nối hoặc căn chỉnh với chế độ xem khác, bố cục cha hoặc hướng dẫn vô hình.

Khám phá trình chỉnh sửa bố cục và tham khảo hình bên dưới khi bạn làm theo các bước được đánh số:




1. Trong thư mục **app > res > layout** trong ngăn Project > Android, hãy nhấp đúp vào tệp `activity_main.xml` để mở tệp đó, nếu tệp đó chưa được mở.
2. Nhấp vào tab **Design** nếu tệp đó chưa được chọn. Bạn sử dụng tab **Design** để thao tác các thành phần và bố cục, và tab **Text** để chỉnh sửa mã XML cho bố cục.
3. Ngăn **Palettes** hiển thị các thành phần UI mà bạn có thể sử dụng trong bố cục của ứng dụng.
4. Ngăn **Component tree** hiển thị phân cấp chế độ xem của các thành phần UI. Các phần tử View được sắp xếp theo hệ thống phân cấp cây gồm các phần tử cha và con, trong đó phần tử con kế thừa các thuộc tính của phần tử cha. Trong hình trên, TextView là phần tử con của ConstraintLayout. Bạn sẽ tìm hiểu về các phần tử này sau trong bài học này.
5. Các ngăn thiết kế và bản thiết kế của trình chỉnh sửa bố cục hiển thị các phần tử UI trong bố cục. Trong hình trên, bố cục chỉ hiển thị một phần tử: TextView hiển thị "Hello World".
6. Tab **Attribute** hiển thị ngăn **Attribute** để thiết lập thuộc tính cho phần tử UI.


Mẹo: Xem tài liệu [Building a UI with Layout Editor](#) để biết thêm chi tiết về cách sử dụng trình chỉnh sửa bố cục, hoặc tham khảo [Meet Android Studio](#) để hiểu rõ hơn về Android Studio.

2.1 Kiểm tra các ràng buộc của phần tử

Thực hiện theo các bước sau:

1. Mở `activity_main.xml` từ ngăn Project > Android nếu nó chưa mở. Nếu tab Design chưa được chọn, hãy nhấp vào tab đó.

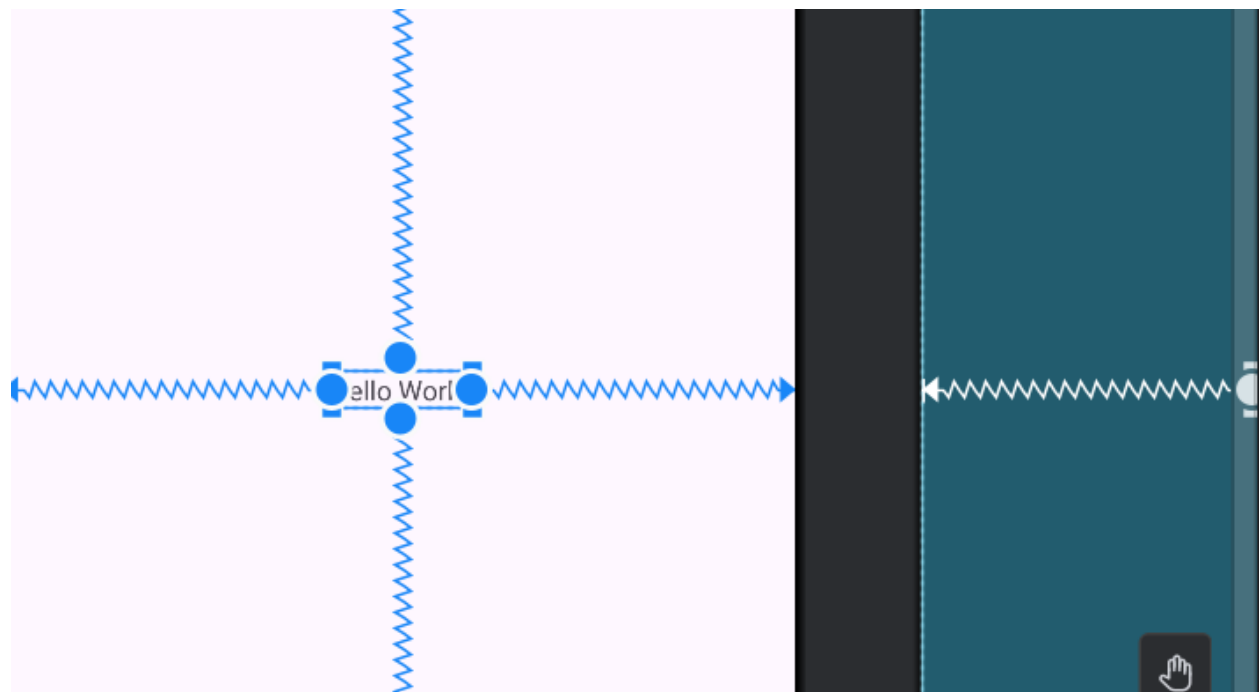
Nếu không có bản thiết kế, hãy nhấp vào nút Select Design  Surface trên thanh công cụ và chọn Design + Blueprint.

2. Công cụ Autoconnect  cũng nằm trong thanh công cụ. Theo mặc định, công cụ này được bật. Đối với bước này, hãy đảm bảo rằng công cụ không bị tắt.

3. Nhấp vào nút phóng to  90%  để phóng to các ngăn thiết kế và bản thiết kế để xem cận cảnh.

4. Chọn TextView trong ngăn Component Tree. TextView "Hello World" được tô sáng trong các ngăn thiết kế và bản thiết kế và các ràng buộc cho phần tử sẽ hiển thị.

5. Tham khảo hình ảnh động bên dưới để biết bước này. Nhấp vào tay cầm hình tròn ở bên phải của TextView để xóa ràng buộc theo chiều ngang liên kết chế độ xem với bên phải của bố cục. TextView nhảy sang bên trái vì nó không còn bị giới hạn ở bên phải nữa. Để thêm lại ràng buộc theo chiều ngang, hãy nhấp vào cùng một tay cầm và kéo một đường sang bên phải của bố cục.



Trong các ngăn thiết kế hoặc bản thiết kế, các tay cầm sau sẽ xuất hiện trên phần tử TextView:

- **Constraint handle:** Để tạo ràng buộc như trong hình động ở trên, hãy nhấp vào một tay cầm ràng buộc, được hiển thị dưới dạng hình tròn ở bên cạnh một phần tử. Sau đó, kéo tay cầm đến một tay cầm ràng buộc khác hoặc đến ranh giới cha. Đường ngoằn ngoèo biểu thị ràng buộc.



- Tay cầm thay đổi kích thước: Để thay đổi kích thước phần tử, hãy kéo các tay cầm thay đổi kích thước hình vuông. Tay cầm sẽ chuyển thành góc nghiêng khi bạn kéo nó.



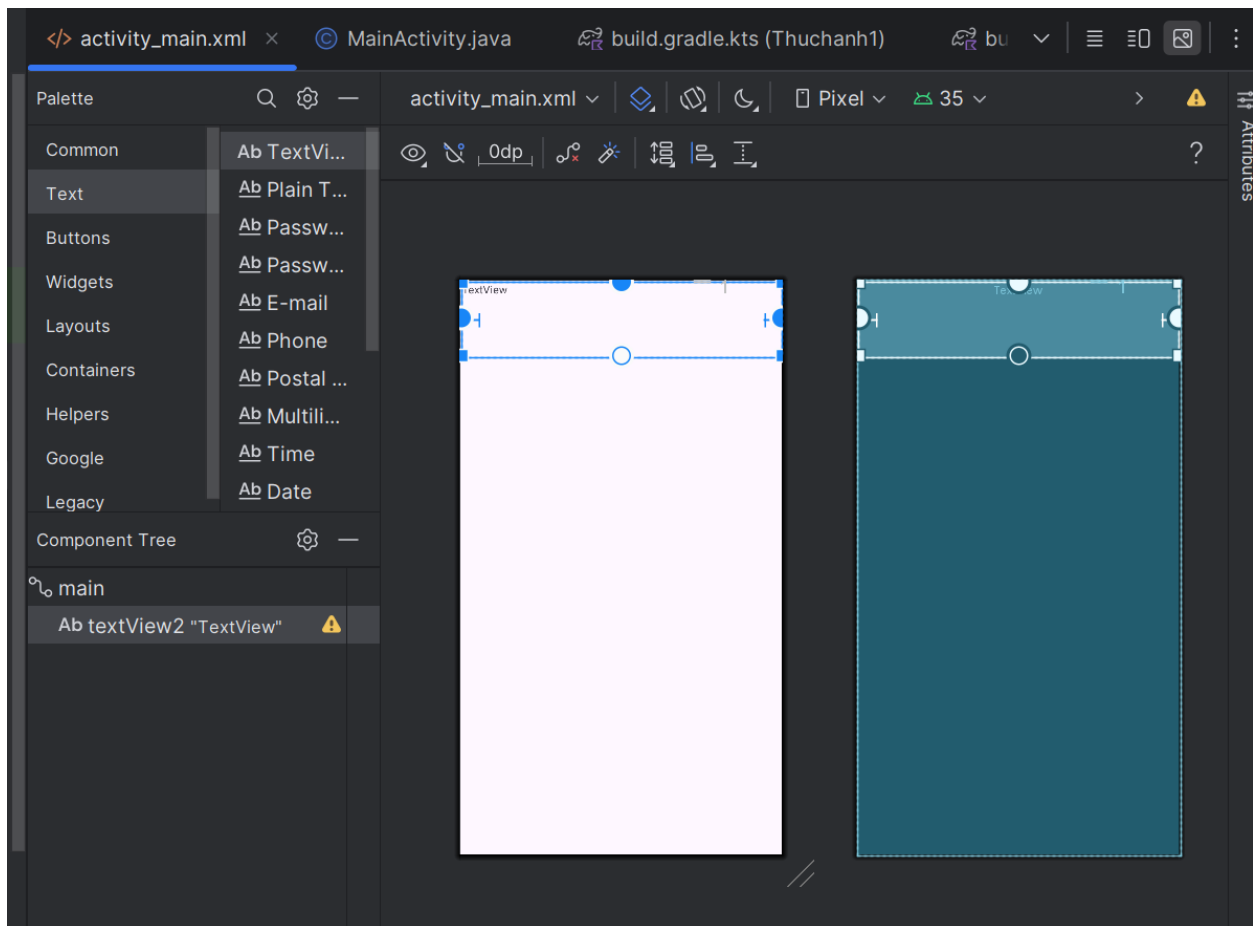
2.2 Thêm Button vào bố cục

Khi được bật, the Autoconnect sẽ tự động tạo hai hoặc nhiều ràng buộc cho một thành phần UI vào bố cục cha. Sau khi bạn kéo thành phần vào bố cục, công cụ này sẽ tạo ra các ràng buộc dựa trên vị trí của thành phần.

Thực hiện theo các bước sau để thêm Nút:

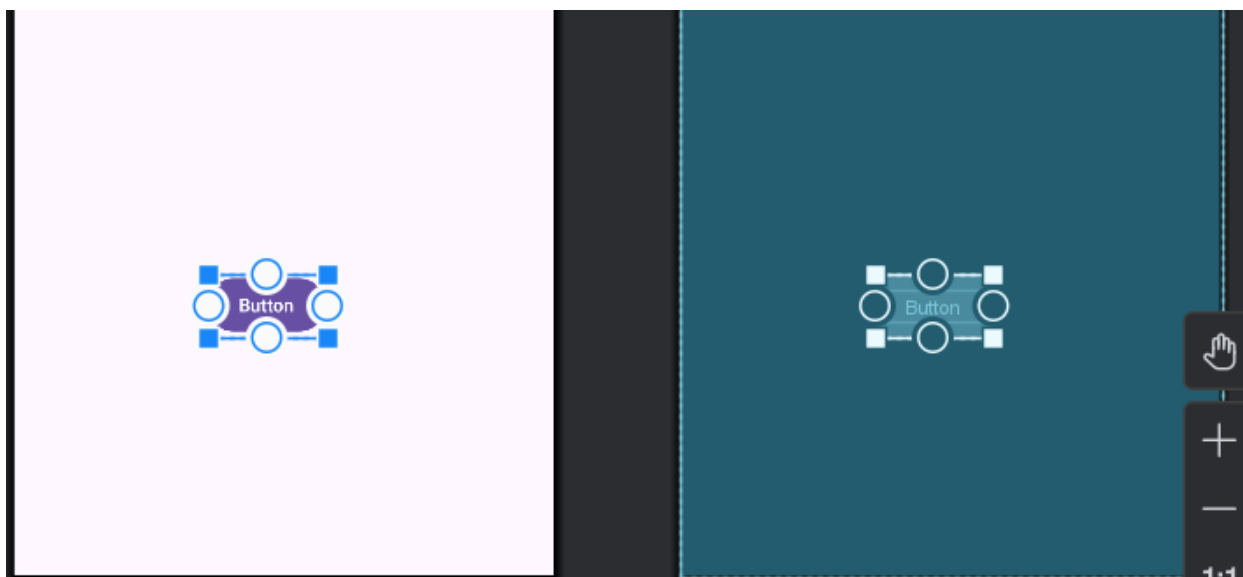
1. Bắt đầu với một bảng trắng. Thành phần TextView không cần thiết, vì vậy khi thành phần này vẫn được chọn, hãy nhấn phím Delete hoặc chọn Edit > Delete. Bây giờ bạn có một bố cục hoàn toàn trống.

2. Kéo Button từ ngăn Palette đến bất kỳ vị trí nào trong bố cục. Nếu bạn thả Button vào khu vực giữa trên cùng của bố cục, các ràng buộc có thể tự động xuất hiện. Nếu không, bạn có thể kéo các ràng buộc lên trên cùng, bên trái và bên phải của bố cục như minh họa trong hình động bên dưới.



2.3 Thêm Button thứ hai vào bố cục

1. Kéo một Button khác từ ngăn Bảng màu vào giữa bố cục như thể hiện trong hình động bên dưới. Autoconnect có thể cung cấp các ràng buộc theo chiều ngang cho bạn (nếu không, bạn có thể tự kéo chúng).
2. Kéo một ràng buộc theo chiều dọc xuống dưới cùng của bố cục (tham khảo hình bên dưới).



Bạn có thể xóa các ràng buộc khỏi một phần tử bằng cách chọn phần tử đó và di con trỏ lên trên để hiển thị nút **Clear Constraints**. Nhấp vào button này để xóa tất cả các ràng buộc trên phần tử đã chọn. Để xóa một ràng buộc duy nhất, hãy nhấp vào trình xử lý cụ thể đặt ràng buộc đó.

Để xóa tất cả các ràng buộc trong toàn bộ bố cục, hãy nhấp vào công cụ **Clear All Constraints** trên thanh công cụ. Công cụ này hữu ích nếu bạn muốn làm lại tất cả các ràng buộc trong bố cục của mình.

Nhiệm vụ 3: Thay đổi các thuộc tính của phần tử UI

Ngăn Attributes cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một phần tử UI. Bạn có thể tìm thấy các attributes (được gọi là properties) chung cho tất cả các chế độ xem trong View class documentation

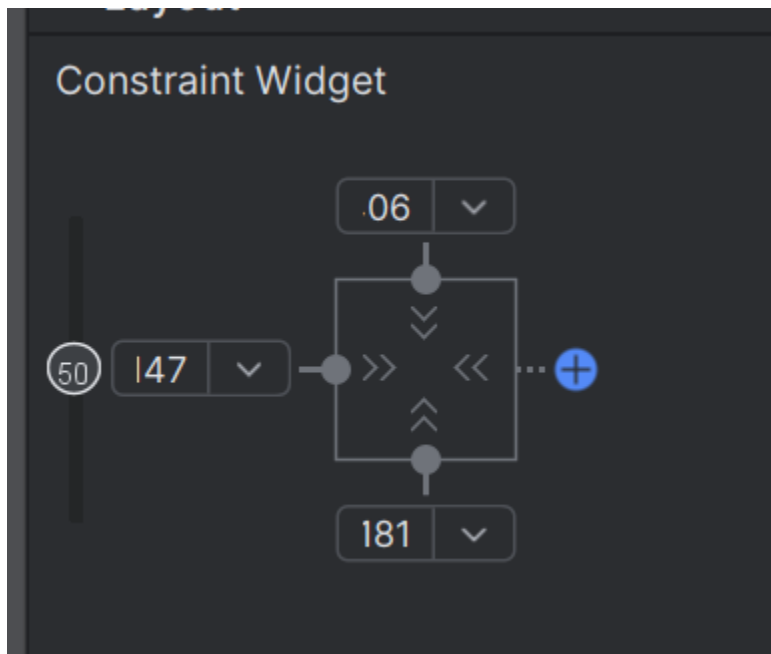
Trong nhiệm vụ này, bạn nhập các giá trị mới và thay đổi các giá trị cho các thuộc tính Nút quan trọng, có thể áp dụng cho hầu hết các loại Chế độ xem.

3.1 Thay đổi kích thước Nút

Trình chỉnh sửa bố cục cung cấp các nút điều chỉnh kích thước ở cả bốn góc của Chế độ xem để bạn có thể thay đổi kích thước Chế độ xem một cách nhanh chóng. Bạn có thể kéo các nút điều chỉnh ở mỗi góc của Chế độ xem để thay đổi kích thước, nhưng làm như vậy sẽ mã hóa cứng các kích thước

chiều rộng và chiều cao. Tránh mã hóa cứng các kích thước cho hầu hết các thành phần Chế độ xem, vì các kích thước được mã hóa cứng không thể thích ứng với các kích thước nội dung và màn hình khác nhau.

Thay vào đó, hãy sử dụng ngăn Attributes ở bên phải của trình chỉnh sửa bố cục để chọn chế độ định cỡ không sử dụng các kích thước được mã hóa cứng. Ngăn Attributes bao gồm một bảng điều khiển định cỡ hình vuông được gọi là trình kiểm tra chế độ xem ở trên cùng. Các ký hiệu bên trong hình vuông biểu thị các thiết lập chiều cao và chiều rộng như sau:



Trong hình trên:

1.Height control . Kiểm soát này chỉ định thuộc tính `layout_height` và xuất hiện trong hai phân đoạn ở phía trên và phía dưới của hình vuông. Các góc cho biết rằng kiểm soát này được đặt thành `wrap_content`, nghĩa là View sẽ mở rộng theo chiều dọc khi cần để vừa với nội dung của nó. "8" cho biết lề chuẩn được đặt thành 8dp.

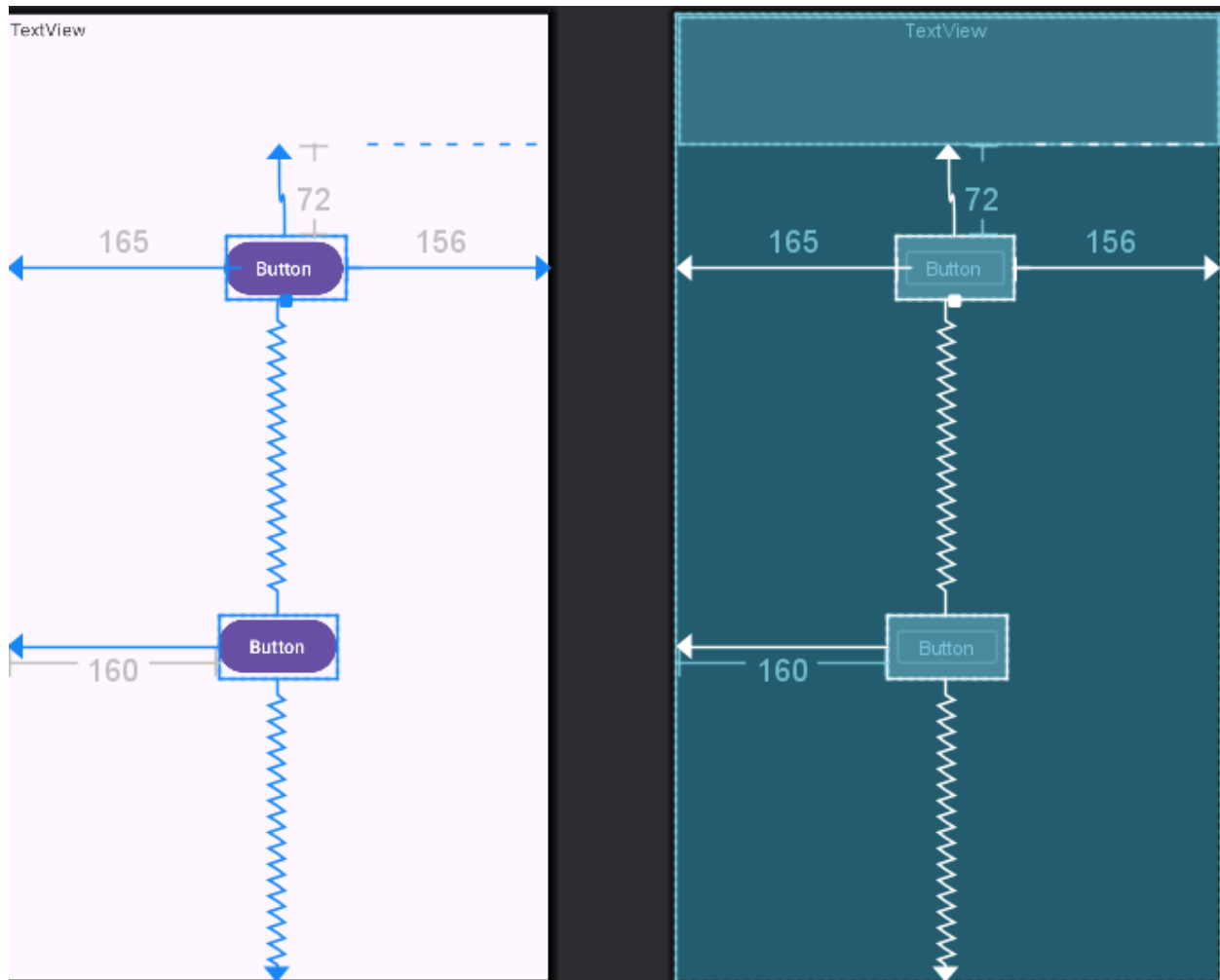
2.Width . Kiểm soát này chỉ định `layout_width` và xuất hiện trong hai phân đoạn ở phía

trái và phải của hình vuông. Các góc cho biết rằng kiểm soát này được đặt thành `wrap_content`, có nghĩa là View sẽ mở rộng theo chiều ngang khi cần để vừa với nội dung của nó, lên đến biên độ 8dp.

3. Nút đóng ngăn Attributes. Nhấp để đóng ngăn.

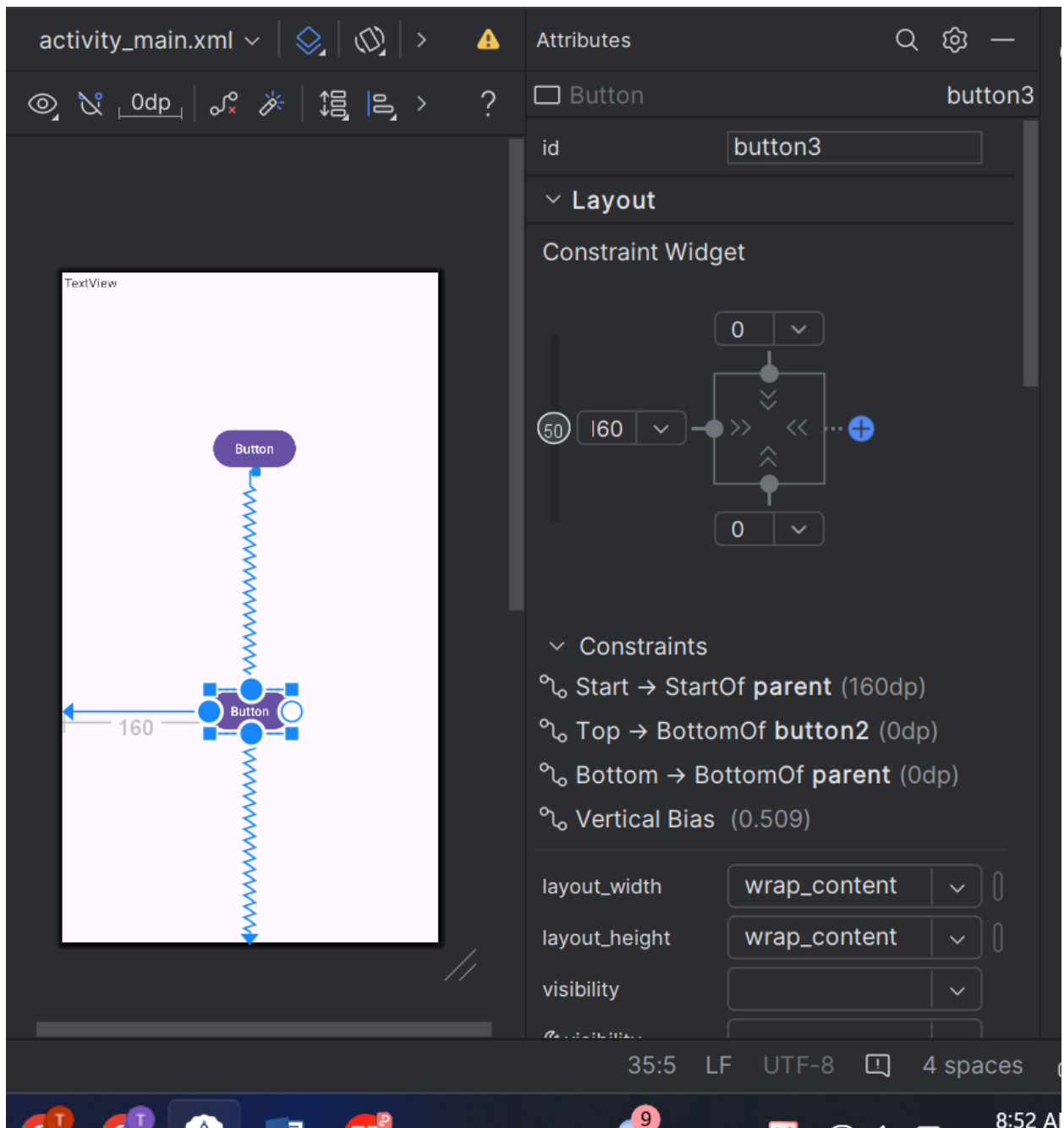
Thực hiện theo các bước sau:

1. Chọn Nút trên cùng trong ngăn Cây thành phần.
2. Nhấp vào tab Thuộc tính ở bên phải cửa sổ trình chỉnh sửa bố cục.
3. Nhấp vào nút điều khiển chiều rộng hai lần—lần nhấp đầu tiên sẽ thay đổi thành Fixed với các đường thẳng và lần nhấp thứ hai sẽ thay đổi thành Phù hợp với các ràng buộc với cuộn lò xo, như thể hiện trong hình động bên dưới.



Kết quả của việc thay đổi điều khiển chiều rộng, thuộc tính `layout_width` trong ngăn Attributes hiển thị giá trị `match_constraint` và phần tử Button kéo dài theo chiều ngang để lấp đầy khoảng trống giữa bên trái và bên phải của bố cục.

4. Chọn Button thứ hai và thực hiện các thay đổi tương tự đối với `layout_width` như trong bước trước, như thể hiện trong hình bên dưới.



Như đã trình bày trong các bước trước, các thuộc tính `layout_width` và `layout_height` trong ngăn Thuộc tính thay đổi khi bạn thay đổi các điều khiển chiều cao và chiều rộng trong trình kiểm tra. Các thuộc tính này có thể lấy một trong ba giá trị cho bố cục, đó là `ConstraintLayout`:

- Thiết lập `match_constraint` mở rộng phần tử View để lấp đầy phần tử cha theo chiều rộng hoặc chiều cao—lên đến một lề, nếu có. Phần tử cha trong trường hợp này là `ConstraintLayout`. Bạn tìm hiểu thêm về `ConstraintLayout` trong tác vụ tiếp theo.

- Thiết lập `wrap_content` thu nhỏ kích thước của phần tử View để nó chỉ đủ lớn để bao quanh nội dung của nó. Nếu không có nội dung, phần tử View sẽ trở nên vô hình.

- Để chỉ định kích thước cố định điều chỉnh theo kích thước màn hình của thiết bị, hãy sử dụng số lượng cố định pixel không phụ thuộc vào mật độ (đơn vị dp). Ví dụ: 16dp nghĩa là 16 pixel không phụ thuộc vào mật độ.

Mẹo: Nếu bạn thay đổi thuộc tính `layout_width` bằng menu bật lên của nó, thuộc tính `layout_width` được đặt thành 0 vì không có kích thước nào được đặt. Thiết lập này giống như `match_constraint`—view có thể mở rộng tối đa có thể để đáp ứng các ràng buộc và thiết lập lề.

3.2 Thay đổi các thuộc tính của Button

Để xác định duy nhất từng View trong bố cục Activity, mỗi View hoặc lớp con View (chẳng hạn như Button) cần có một ID duy nhất. Và để có thể sử dụng, các phần tử Button cần có văn bản. Các phần tử View cũng có thể có nền có thể là màu hoặc hình ảnh.

Ngăn Thuộc tính cung cấp quyền truy cập vào tất cả các thuộc tính mà bạn có thể gán cho một phần tử View. Bạn có thể nhập giá trị cho từng thuộc tính, chẳng hạn như các thuộc tính `android:id`, `background`, `textColor` và `text`

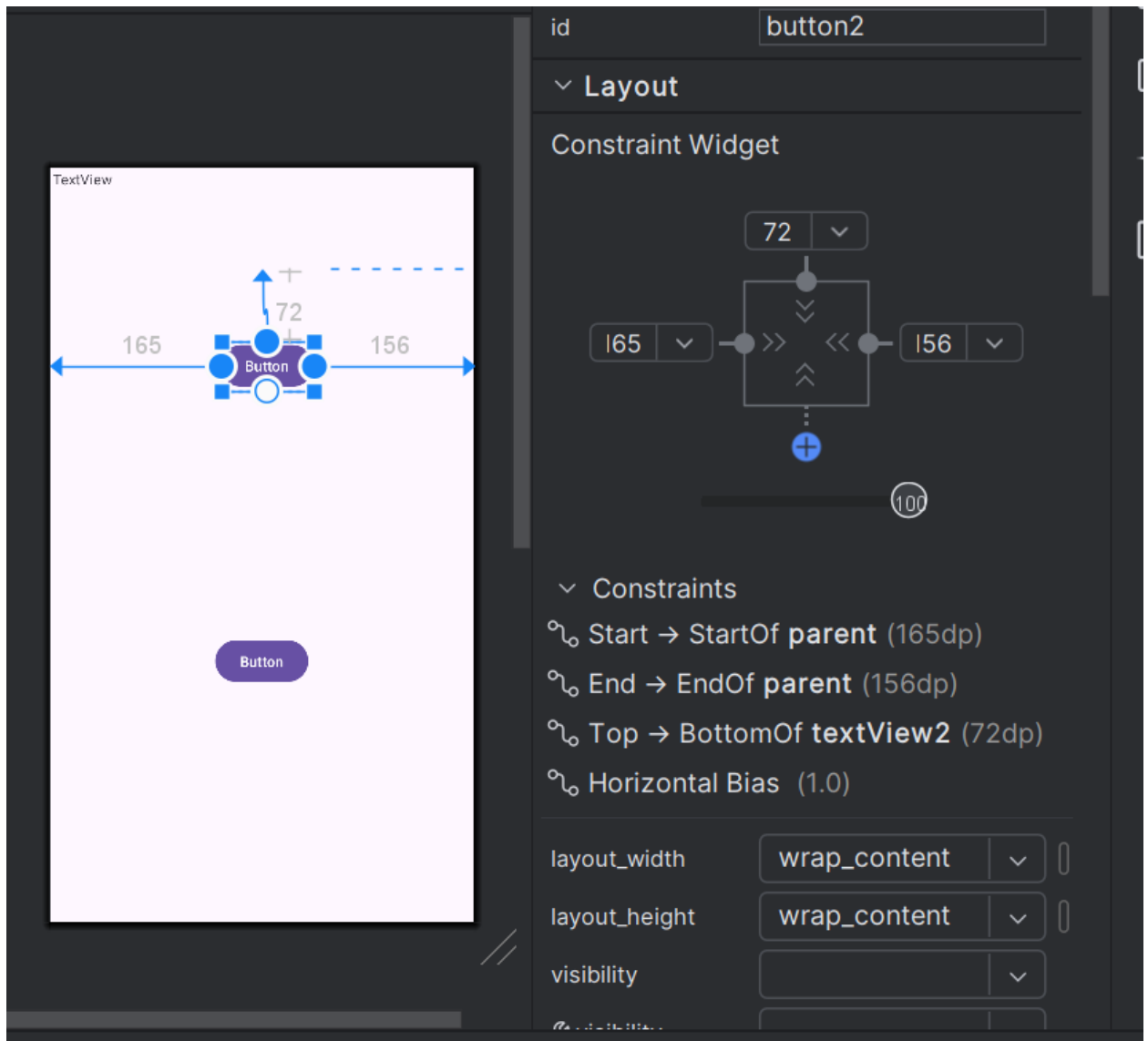
Hình ảnh động sau đây minh họa cách thực hiện các bước này:

1. Sau khi chọn Button đầu tiên, hãy chỉnh sửa trường ID ở đầu ngăn Thuộc tính thành `button_toast` cho thuộc tính `android:id`, được sử dụng để xác định phần tử trong bố cục.

2. Đặt thuộc tính `background` thành `@color/colorPrimary`. (Khi bạn nhập `@c`, các lựa chọn sẽ xuất hiện để dễ dàng lựa chọn.)

3. Đặt thuộc tính `textColor` thành `@android:color/white`.

4. Chỉnh sửa thuộc tính `text` thành Toast.



5. Thực hiện các thay đổi thuộc tính tương tự cho Button thứ hai, sử dụng `button_count` làm ID, Count cho thuộc tính text và cùng màu cho background và text như các bước trước.

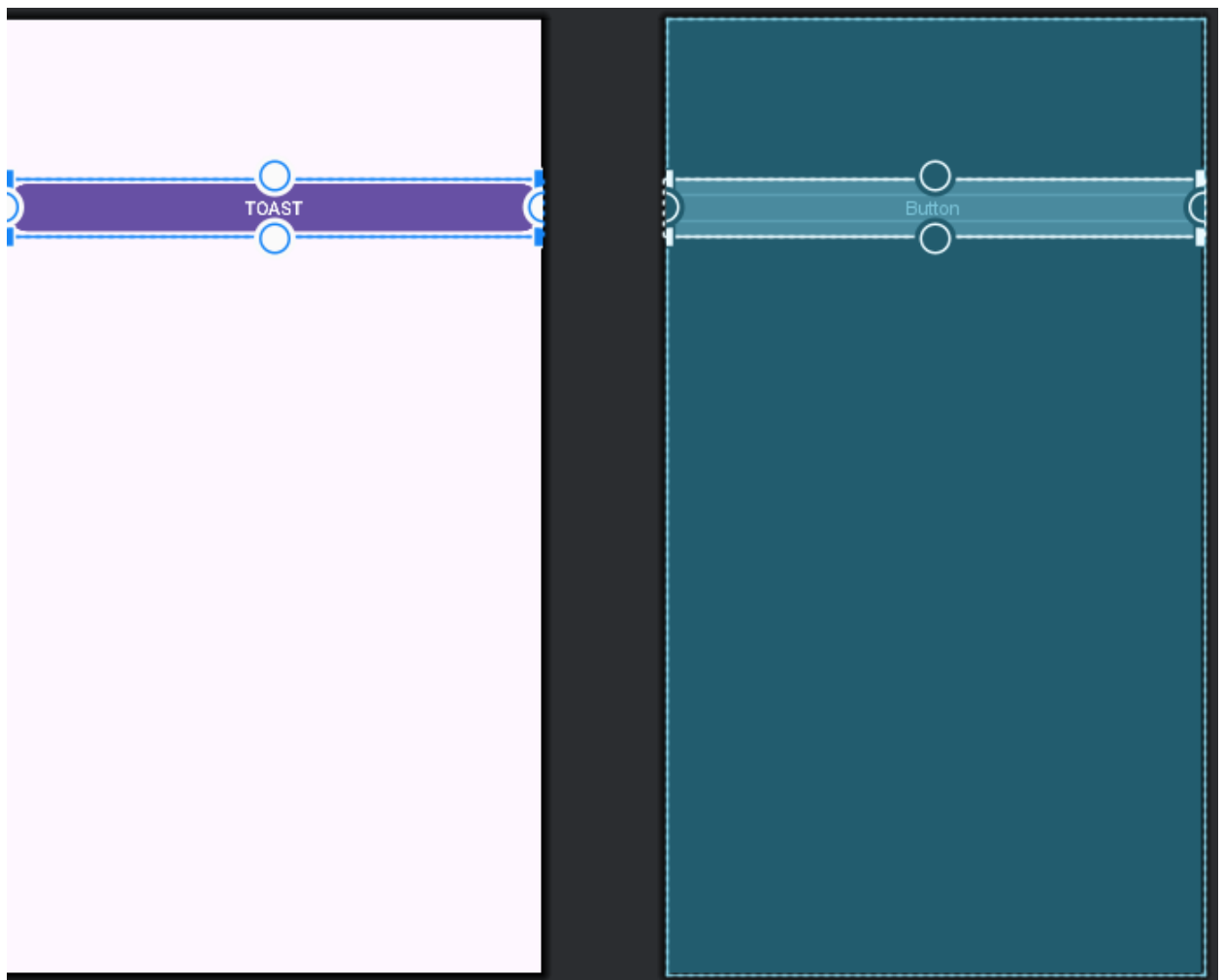
`ColorPrimary` là màu chính của chủ đề, một trong những màu cơ sở chủ đề được xác định trước được xác định trong tệp tài nguyên `colors.xml`. Nó được sử dụng cho thanh ứng dụng. Sử dụng màu cơ sở cho các thành phần UI khác sẽ tạo ra một UI thống nhất. Bạn sẽ tìm hiểu thêm về chủ đề ứng dụng và Material Design trong bài học khác.

Nhiệm vụ 4: Thêm `TextEdit` và đặt thuộc tính của nó

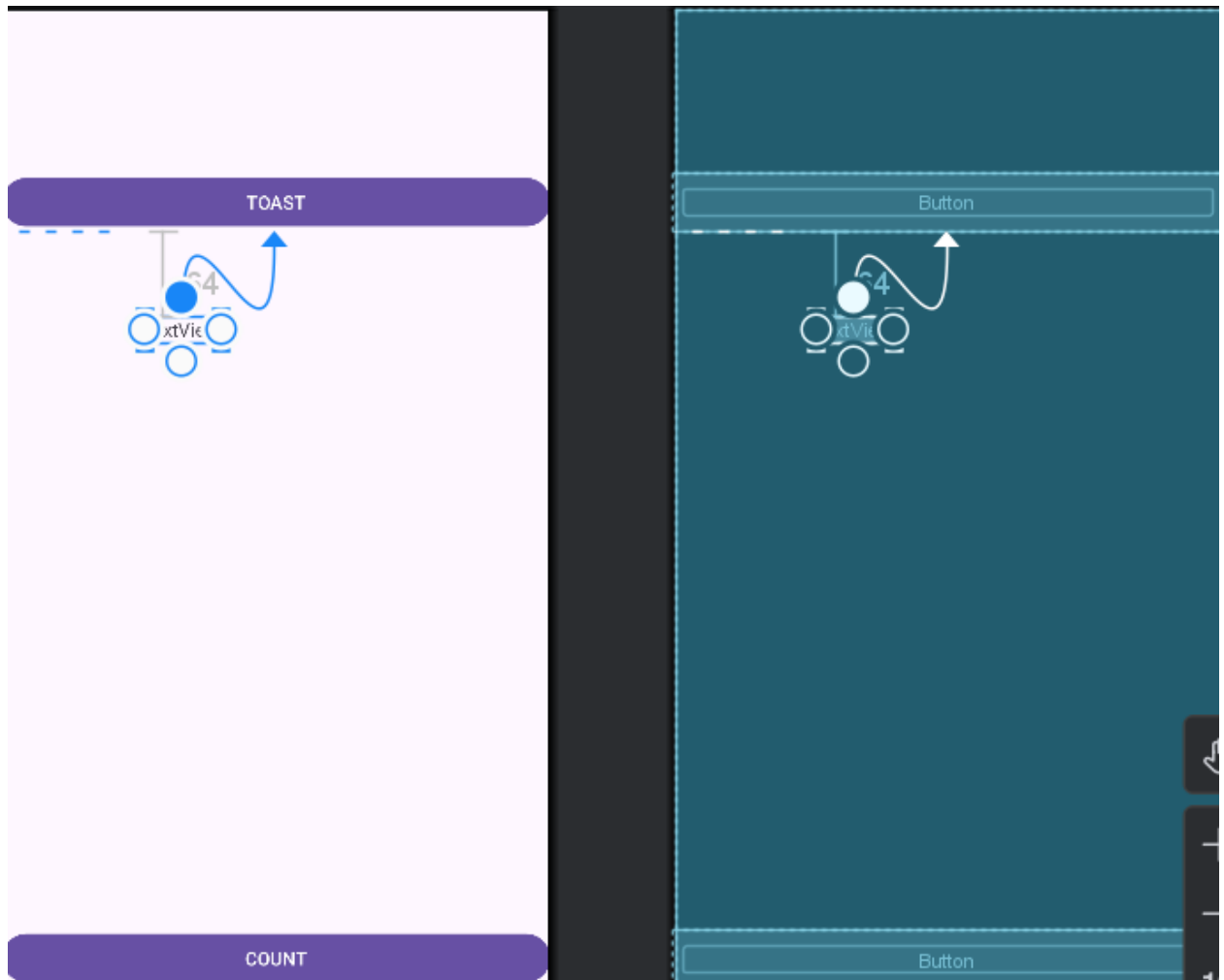
Một trong những lợi ích của ConstraintLayout là khả năng căn chỉnh hoặc hạn chế các thành phần so với các thành phần khác. Trong tác vụ này, bạn sẽ thêm một TextView vào giữa bố cục và giới hạn theo chiều ngang với lề và theo chiều dọc với hai phần tử Button. Sau đó, bạn sẽ thay đổi các thuộc tính cho TextView trong ngăn Thuộc tính.

4.1 Thêm TextView và các ràng buộc

1. Như được hiển thị trong hình động bên dưới, hãy kéo một TextView từ ngăn Palette lên phần trên của bố cục và kéo một ràng buộc từ đầu TextView đến tay cầm ở phía dưới của Toast Button. Thao tác này giới hạn TextView ở bên dưới Button.



2. Như được hiển thị trong hình động bên dưới, hãy kéo một ràng buộc từ dưới cùng của TextView đến tay cầm ở trên cùng của Nút đếm và từ các cạnh của TextView đến các cạnh của bố cục. Thao tác này ràng buộc TextView ở giữa bố cục giữa hai phần tử Nút.



4.2 Đặt các thuộc tính của TextView

Với TextView được chọn, hãy mở ngăn Thuộc tính, nếu ngăn này chưa mở. Đặt các thuộc tính cho TextView như được hiển thị trong hình động bên dưới. Các thuộc tính bạn chưa gặp phải được giải thích sau hình:

1. Đặt ID thành `show_count`.
2. Đặt văn bản thành 0.
3. Đặt `textSize` thành 160sp.

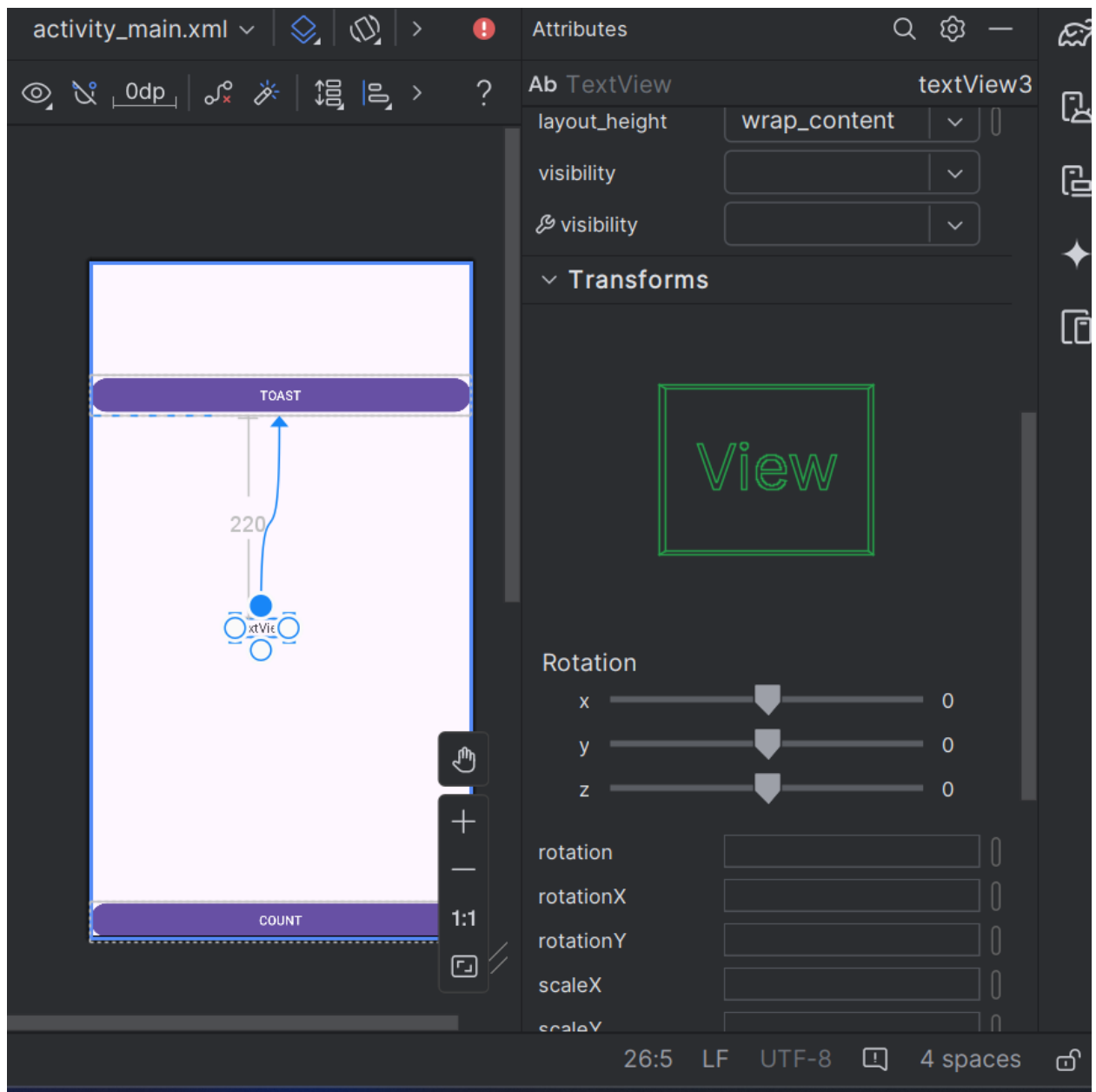
4. Đặt `textStyle` thành **B** (in đậm) và `textAlignment` thành `ALIGNCENTER` (căn giữa đoạn văn).

5. Thay đổi các điều khiển kích thước chế độ xem theo chiều ngang và chiều dọc (`layout_width` và `layout_height`) thành `match_constraint`.

6. Đặt `textColor` thành `@color/colorPrimary`

7. Cuộn xuống gần và nhấp vào Xem tất cả các thuộc tính, cuộn xuống trang thứ hai của thuộc tính để làm nền, sau đó nhập `#FFF00` để có màu vàng.

8. Cuộn xuống trọng lực, mở rộng trọng lực và chọn `center_ver` (để căn giữa theo chiều dọc).



- **textSize:** Kích thước văn bản của TextView. Đối với bài học này, kích thước được đặt thành 160sp. Sp là viết tắt của pixel không phụ thuộc tỷ lệ và giống như dp, là đơn vị tỷ lệ với mật độ màn hình và tùy chọn kích thước phông chữ của người dùng. Sử dụng đơn vị dp khi bạn chỉ định kích thước phông chữ để kích thước được điều chỉnh cho cả mật độ màn hình và tùy chọn của người dùng.
- **textStyle** và **textAlignment:** Kiểu văn bản, được đặt thành B (in đậm) trong bài học này và căn chỉnh văn bản, được đặt thành ALIGNCENTER (căn giữa đoạn văn).

- gravity: Thuộc tính trọng lực chỉ định cách căn chỉnh View trong View hoặc ViewGroup cha của nó. Trong bước này, bạn căn giữa TextView để căn giữa theo chiều dọc trong ConstraintLayout cha.

Bạn có thể nhận thấy rằng thuộc tính nền nằm trên trang đầu tiên của ngăn Thuộc tính cho một Nút, nhưng lại nằm trên trang thứ hai của ngăn Thuộc tính cho một TextView. Ngăn Thuộc tính thay đổi cho từng loại Chế độ xem: Các thuộc tính phổ biến nhất cho loại Chế độ xem xuất hiện trên trang đầu tiên, và các thuộc tính còn lại được liệt kê trên trang thứ hai. Để quay lại trang đầu tiên của ngăn Thuộc tính, hãy nhấp vào biểu tượng trên thanh công cụ ở đầu ngăn.

1.3) Trình chỉnh sửa bố cục

1.4) Văn bản và các chế độ cuộn

1.5) Tài nguyên có sẵn

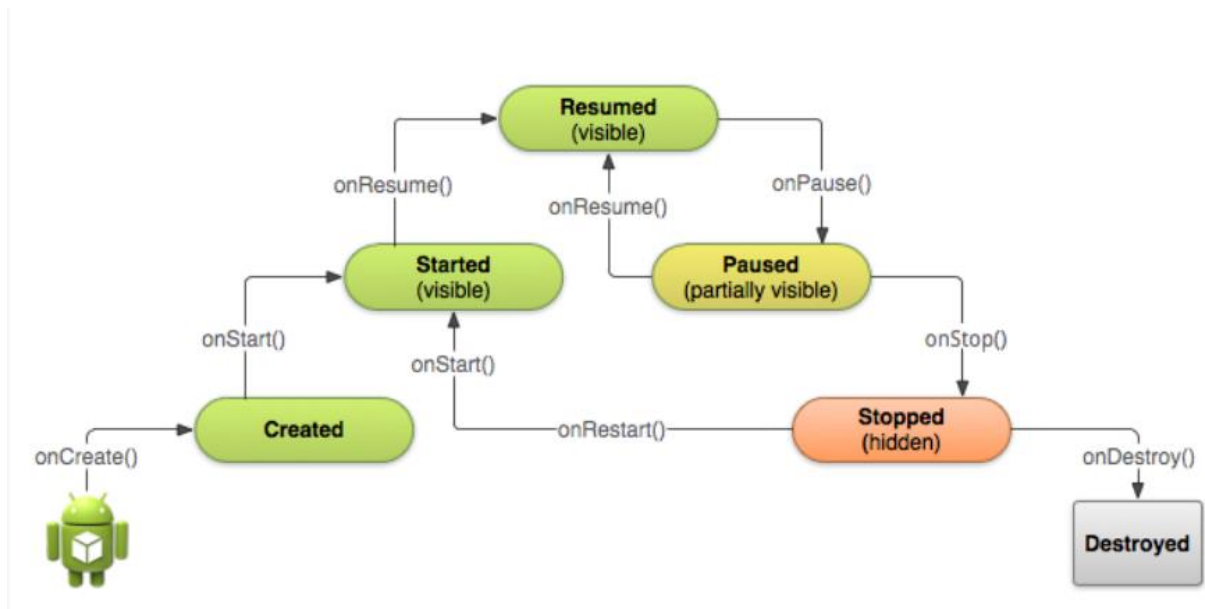
Bài 2) Activities

2.1) Activity và Intent

2.2) Vòng đời của Activity và trạng thái

Giới thiệu

Trong bài thực hành này, bạn sẽ tìm hiểu thêm về *vòng đời của hoạt động*. Vòng đời là tập hợp các trạng thái mà một hoạt động có thể có trong toàn bộ vòng đời của nó, từ khi nó được tạo ra cho đến khi nó bị hủy và hệ thống thu hồi lại tài nguyên của nó. Khi người dùng điều hướng giữa các hoạt động trong ứng dụng của bạn (cũng như vào và ra khỏi ứng dụng của bạn), các hoạt động sẽ chuyển đổi giữa các trạng thái khác nhau trong vòng đời của chúng.



Mỗi giai đoạn trong vòng đời của một hoạt động có một phương thức gọi lại tương ứng: `onCreate()`, `onStart()`, `onPause()`, v.v. Khi một hoạt động thay đổi trạng thái, phương thức gọi lại liên quan sẽ được gọi. Bạn đã thấy một trong những phương thức này: `onCreate()`. Bằng cách ghi đè bất kỳ phương thức gọi lại vòng đời nào trong các lớp Activity của bạn, bạn có thể thay đổi hành vi mặc định của hoạt động để phản hồi lại các hành động của người dùng hoặc hệ thống.

Trạng thái hoạt động cũng có thể thay đổi để phản hồi các thay đổi về cấu hình thiết bị, ví dụ khi người dùng xoay thiết bị từ chế độ dọc sang chế độ ngang. Khi những thay đổi về cấu hình này xảy ra, hoạt động sẽ bị hủy và được tạo lại ở trạng thái mặc định, và người dùng có thể mất thông tin mà họ đã nhập vào hoạt động. Để tránh gây nhầm lẫn cho người dùng, điều quan trọng là bạn phải phát triển ứng dụng của mình để ngăn ngừa mất dữ liệu bất ngờ. Sau đó trong phần thực hành này, bạn sẽ thử nghiệm với các thay đổi về cấu hình và tìm hiểu cách duy trì trạng thái của hoạt động để phản hồi các thay đổi về cấu hình thiết bị và các sự kiện vòng đời hoạt động khác.

Trong bài thực hành này, bạn thêm các câu lệnh ghi nhật ký vào ứng dụng TwoActivities và quan sát các thay đổi trong vòng đời hoạt động khi bạn sử dụng ứng dụng. Sau đó, bạn bắt đầu làm việc với những thay đổi này và khám phá cách xử lý đầu vào của người dùng trong những điều kiện này.

Những điều bạn cần biết

Bạn cần có thể:

- Tạo và chạy một dự án ứng dụng trong Android Studio.
- Thêm các câu lệnh nhật ký vào ứng dụng của bạn và xem các nhật ký đó trong ngăn Logcat.
- Hiểu và làm việc với một Activity và một Intent, và thoải mái tương tác với chúng.

Những điều bạn sẽ học

- Vòng đời của Activity hoạt động như thế nào.
- Khi một Activity bắt đầu, tạm dừng, dừng hẳn và bị hủy.
- Về các phương thức gọi lại vòng đời liên quan đến các thay đổi của Activity.
- Tác động của các hành động (chẳng hạn như thay đổi cấu hình) có thể dẫn đến các sự kiện vòng đời của Activity.
- Cách duy trì trạng thái của Activity trong các sự kiện vòng đời.

Những gì bạn sẽ làm

- Thêm mã vào ứng dụng TwoActivities từ bài thực hành trước để triển khai các cuộc gọi lại vòng đời Hoạt động khác nhau để bao gồm các câu lệnh ghi nhật ký.
- Quan sát các thay đổi trạng thái khi ứng dụng của bạn chạy và khi bạn tương tác với từng Hoạt động trong ứng dụng của mình.
- Sửa đổi ứng dụng của bạn để giữ nguyên trạng thái phiên bản của một Hoạt động được tạo lại bất ngờ để phản hồi hành vi của người dùng hoặc thay đổi cấu hình trên thiết bị.

Tổng quan về ứng dụng

Trong bài thực hành này, bạn sẽ thêm vào ứng dụng TwoActivities. Ứng dụng trông và hoạt động gần giống như trong codelab trước. Ứng dụng chứa hai triển khai Activity và cung cấp cho người dùng khả năng gửi giữa chúng. Những thay

đổi bạn thực hiện đối với ứng dụng trong bài thực hành này sẽ không ảnh hưởng đến hành vi người dùng có thể nhìn thấy của ứng dụng.

Nhiệm vụ 1: Thêm các lệnh gọi lại vòng đời vào TwoActivities

Trong nhiệm vụ này, bạn sẽ triển khai tất cả các phương thức gọi lại vòng đời Activity để in thông báo vào logcat khi các phương thức đó được gọi. Các thông báo nhật ký này sẽ cho phép bạn xem khi nào Trạng thái vòng đời Activity thay đổi và cách những thay đổi trạng thái vòng đời đó ảnh hưởng đến ứng dụng của bạn khi ứng dụng chạy.

1.1 (Tùy chọn) Sao chép dự án TwoActivities

Đối với các nhiệm vụ trong bài thực hành này, bạn sẽ sửa đổi dự án TwoActivities hiện tại mà bạn đã xây dựng trong bài thực hành trước. Nếu bạn muốn giữ nguyên dự án TwoActivities trước đó, hãy làm theo các bước trong Phụ lục: Tiềm ích để tạo bản sao của dự án.

1.2 Triển khai lệnh gọi lại vào MainActivity

1. Mở dự án TwoActivities trong Android Studio và mở MainActivity trong ngăn Project > Android.
2. Trong phương thức onCreate(), hãy thêm các câu lệnh nhật ký sau:
3. Thêm ghi đề cho lệnh gọi lại onStart(), với một câu lệnh vào nhật ký cho sự kiện đó:

1.3 Triển khai các hàm gọi lại vòng đời trong SecondActivity

Bây giờ bạn đã triển khai các phương thức gọi lại vòng đời cho MainActivity, hãy thực hiện tương tự cho SecondActivity.

1. Mở SecondActivity.
2. Ở đầu lớp, thêm một hằng số cho biến LOG_TAG:

3. Thêm các lệnh gọi lại vòng đời và các câu lệnh nhật ký vào Activity thứ hai. (Bạn có thể Sao chép và Dán các phương thức gọi lại từ MainActivity.)
4. Thêm một câu lệnh nhật ký vào phương thức `returnReply()` ngay trước phương thức `finish()`:

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