

**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

<b>CHƯƠNG 1. Làm quen.....</b>	<b>4</b>
Bài 1) Tạo ứng dụng đầu tiên .....	4
1.1) Android Studio và Hello World .....	4
1.2) Giao diện người dùng tương tác đầu tiên.....	5
1.3) Trình chỉnh sửa bố cục.....	5
1.4) Văn bản và các chế độ cuộn.....	5
1.5) Tài nguyên có sẵn.....	5
Bài 2) Hoạt động.....	5
2.1) Hoạt động và Ý định.....	5
2.2) Vòng đời của Activity và trạng thái.....	5
2.3) Ý định ngầm định.....	5
Bài 3) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ.....	5
3.1) Trình gỡ lỗi.....	5
3.2) Kiểm thử đơn vị .....	5
3.3) Thư viện hỗ trợ .....	5
<b>CHƯƠNG 2. Trải nghiệm người dùng.....</b>	<b>6</b>
Bài 1) Tương tác người dùng.....	6
1.1) Hình ảnh có thể chọn.....	6
1.2) Các điều khiển nhập liệu.....	6
1.3) Menu và bộ chọn.....	6
1.4) Điều hướng người dùng .....	6
1.5) Chế độ xem tái chế.....	6
Bài 2) Trải nghiệm người dùng thú vị .....	6
2.1) Hình vẽ, định kiểu và chủ đề.....	6

2.2) Thẻ và màu sắc.....	6
2.3) Bố cục thích ứng.....	6
Bài 3) Kiểm thử giao diện người dùng.....	6
3.1) Espresso cho việc kiểm tra UI.....	6
<b>CHƯƠNG 3. Làm việc trong nền.....</b>	<b>6</b>
Bài 1) Các tác vụ nền.....	6
1.1) Nhiệm vụ không đồng bộ.....	6
1.2) AsyncTask và AsyncTaskLoader .....	6
1.3) Máy thu phát sóng .....	6
Bài 2) Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền.....	6
2.1) Thông báo .....	6
2.2) Trình quản lý cảnh báo .....	6
2.3) JobScheduler .....	6
<b>CHƯƠNG 4. Lưu trữ dữ liệu người dùng.....</b>	<b>7</b>
Bài 1) Tùy chọn và cài đặt.....	7
1.1) Tùy chọn được chia sẻ.....	7
1.2) Cài đặt ứng dụng.....	7
Bài 2) Lưu trữ dữ liệu với Room.....	7
2.1) Room, LiveData và ViewModel.....	7
2.2) Room, LiveData và ViewModel.....	7
<b>3.1) Trinhf gowx loi .....</b>	

## **CHƯƠNG 1. LÀM QUEN**

### **Bài 1) Tạo ứng dụng đầu tiên**

#### **1.1) Android Studio và Hello World**

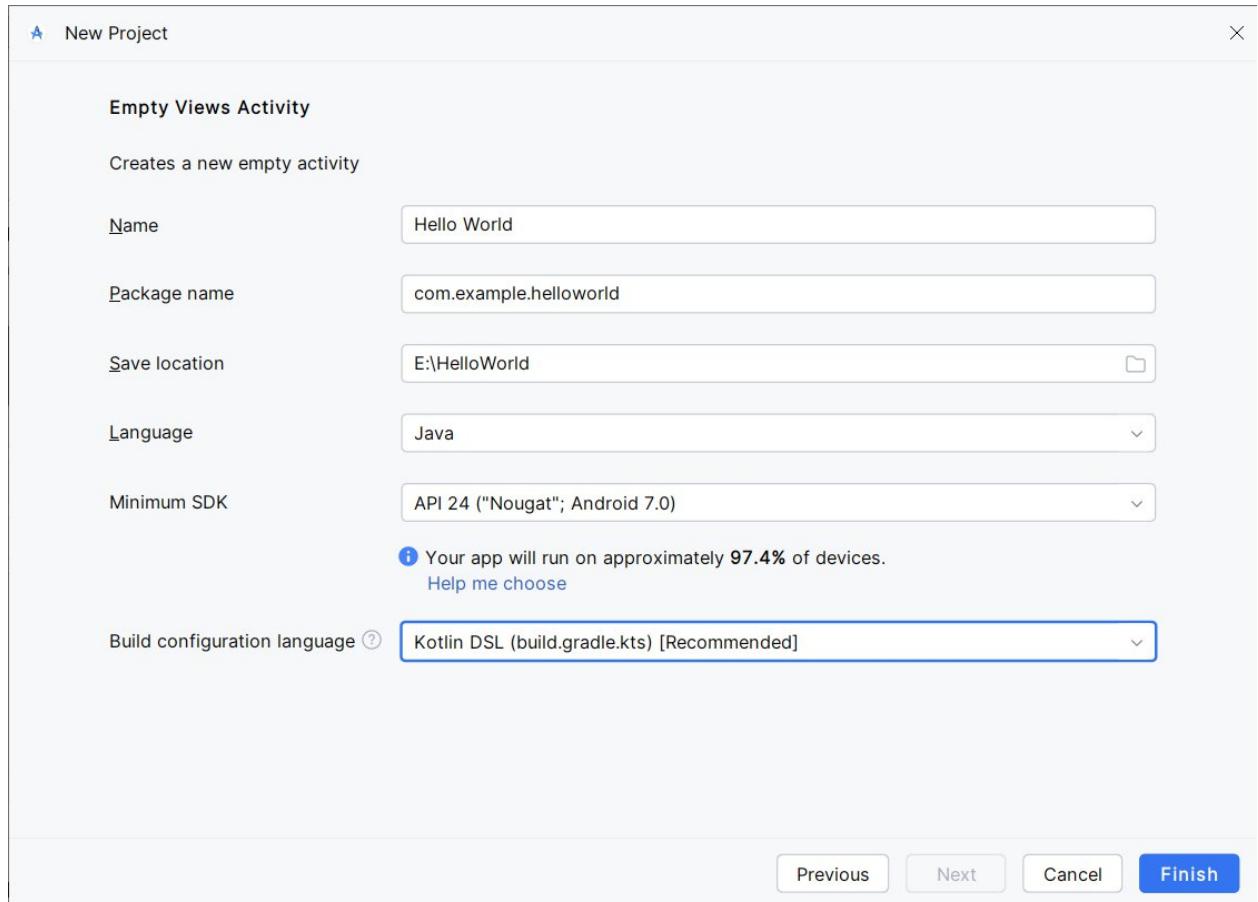
##### **Giới thiệu**

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 một dự án mới cho ứng dụng Hello World bằng cách sử dụng mẫu có sẵn. Ứng dụng đơn giản này sẽ hiển thị dòng chữ "Hello World" trên màn hình của thiết bị Android, dù là máy ảo hay thiết bị thật.

Đây là giao diện của ứng dụng sau khi hoàn thành:



## Bước 1: Cài đặt Android Studio

Android Studio là một môi trường phát triển tích hợp (IDE) hoàn chỉnh, cung cấp trình chỉnh sửa mã nâng cao cùng với các mẫu ứng dụng có sẵn. Ngoài ra, nó còn đi kèm với các công cụ hỗ trợ lập trình, gỡ lỗi, kiểm thử và tối ưu hiệu suất, giúp việc phát triển ứng dụng trở nên nhanh chóng và thuận tiện hơn.

Bạn có thể kiểm thử ứng dụng trên nhiều trình giả lập được cấu hình sẵn hoặc trực tiếp trên thiết bị di động của mình, sau đó xây dựng ứng dụng hoàn chỉnh và phát hành trên cửa hàng Google Play.

Lưu ý: [Android Studio liên tục được cập nhật và cải tiến](#). Để biết thông tin mới nhất về yêu cầu hệ thống cũng như hướng dẫn cài đặt, hãy truy cập [trang chính thức](#) của Android Studio.

Android Studio có thể được cài đặt trên máy tính chạy Windows, Linux và macOS. Phiên bản mới nhất của OpenJDK (Java Development Kit) đã được tích hợp sẵn trong Android Studio.

Trước khi bắt đầu, bạn cần kiểm tra yêu cầu hệ thống để đảm bảo thiết bị của mình đáp ứng đủ điều kiện. Quá trình cài đặt Android Studio trên các hệ điều hành nhìn chung là giống nhau, chỉ có một vài điểm khác biệt nhỏ sẽ được ghi chú riêng.

1. Truy cập [trang web](#) dành cho nhà phát triển Android và làm theo hướng dẫn để tải xuống, cài đặt Android Studio.
2. Chấp nhận các thiết lập mặc định trong suốt quá trình cài đặt và đảm bảo tất cả các thành phần cần thiết đều được chọn.
3. Sau khi cài đặt xong, trình hướng dẫn thiết lập (Setup Wizard) sẽ tải xuống và cài đặt thêm một số thành phần bổ sung, bao gồm Android SDK. Quá trình này có thể mất thời gian tùy thuộc vào tốc độ mạng của bạn, và có thể có một số bước lặp lại.
4. Khi hoàn tất, Android Studio sẽ tự động khởi động và bạn có thể bắt đầu tạo dự án đầu tiên của mình.

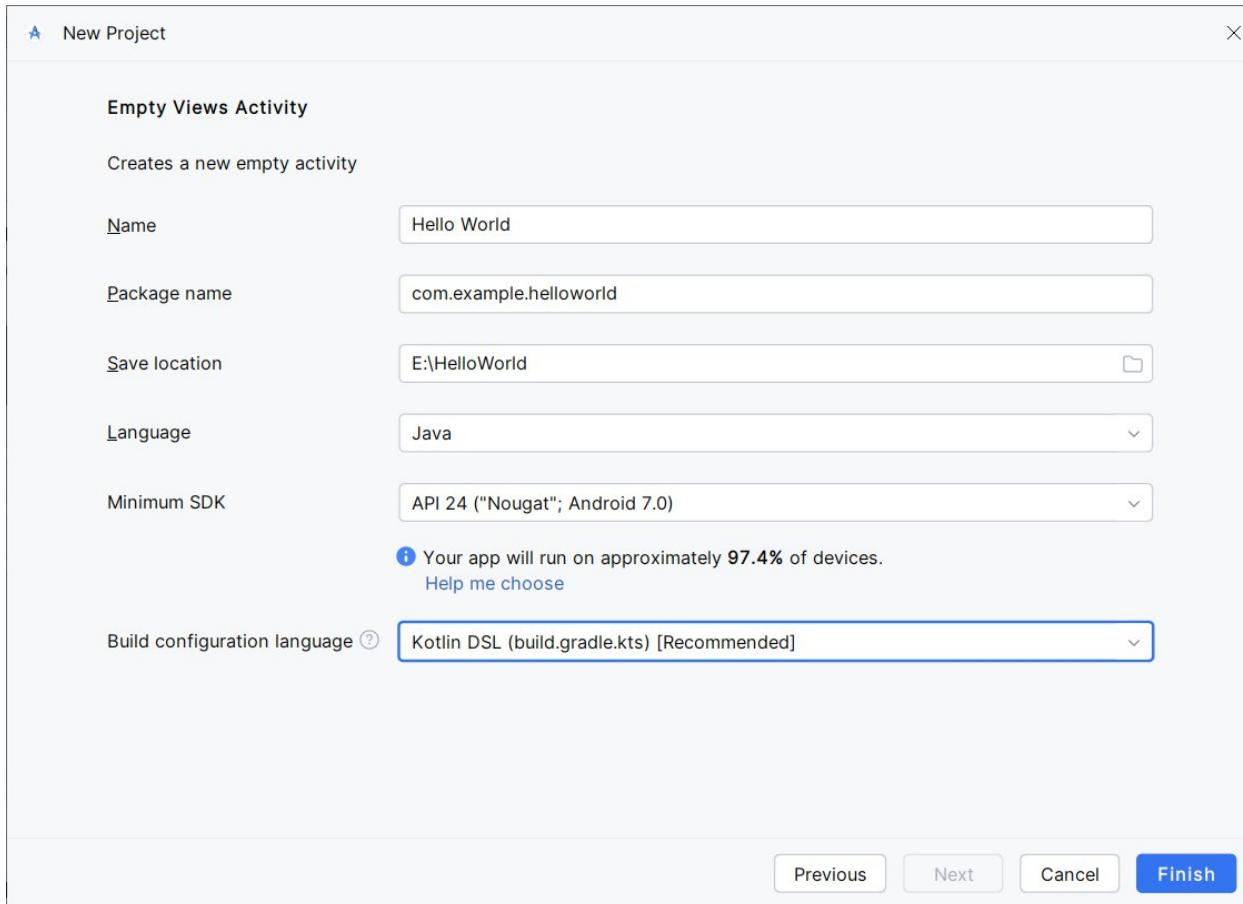
**Xử lý sự cố:** Nếu gặp vấn đề trong quá trình cài đặt, bạn có thể kiểm tra ghi chú phát hành (release notes) của Android Studio hoặc nhờ sự trợ giúp từ giảng viên/hướng dẫn viên của mình.

## Bước 2: Tạo ứng dụng Hello World

Trong bước này, bạn sẽ tạo một ứng dụng hiển thị dòng chữ "Hello World". Việc này giúp bạn kiểm tra xem Android Studio đã được cài đặt đúng chưa, đồng thời làm quen với những thao tác cơ bản trong quá trình phát triển ứng dụng Android.

### 2.1 Tạo dự án ứng dụng

1. Mở Android Studio (nếu chưa mở).
2. Trong màn hình chào mừng Welcome to Android Studio, nhấp vào Start a new Android Studio project (Bắt đầu một dự án Android Studio mới).
3. Trong cửa sổ Create Android Project, nhập Hello World vào ô Application name (Tên ứng dụng).

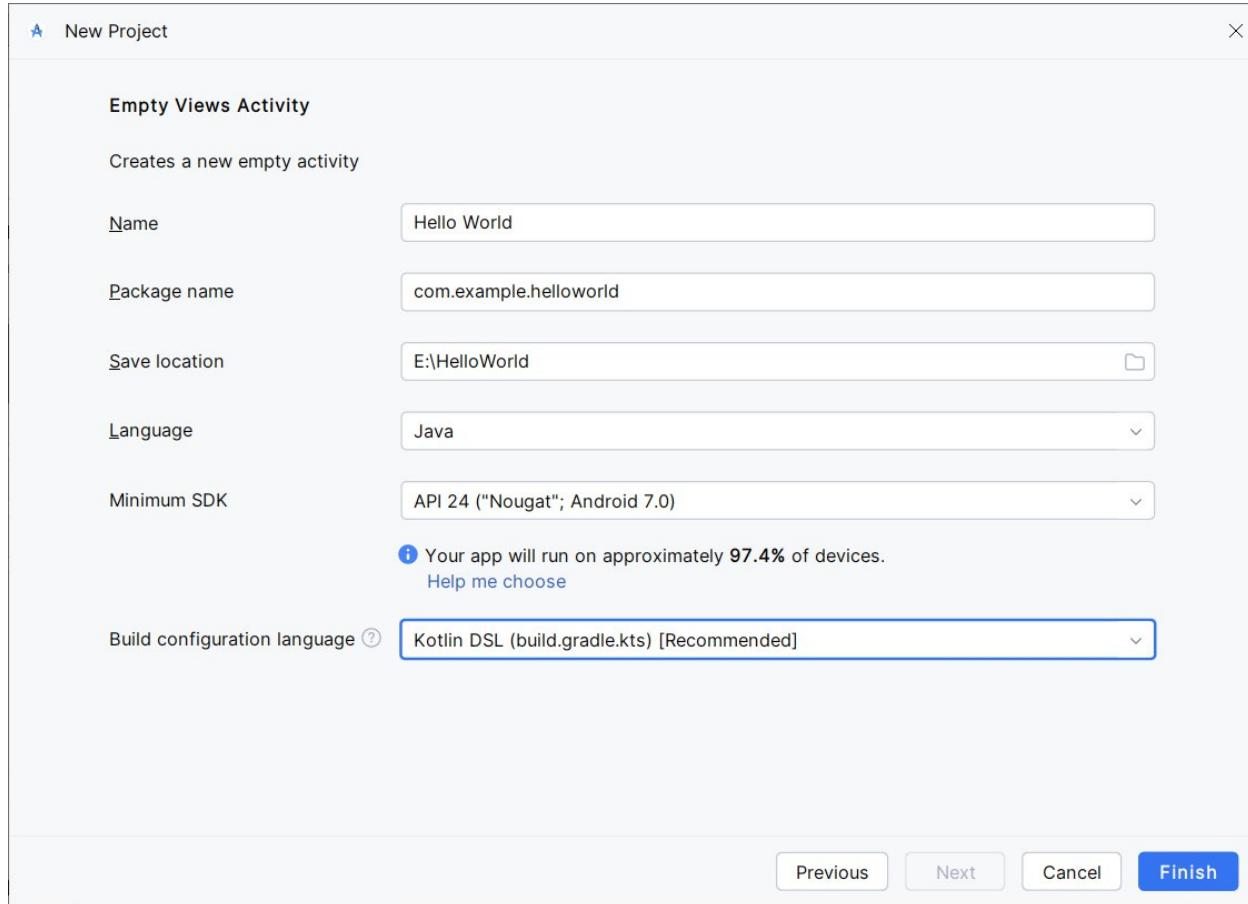


4. Kiểm tra xem Project location (vị trí lưu trữ dự án) có đúng với thư mục bạn muốn lưu ứng dụng Hello World và các dự án Android Studio khác không. Nếu cần, bạn có thể thay đổi thành thư mục mong muốn.
5. Chấp nhận giá trị mặc định android.example.com cho Company Domain (tên miền công ty), hoặc nhập một tên miền riêng nếu bạn muốn.

Nếu bạn không có ý định xuất bản ứng dụng, bạn có thể giữ nguyên giá trị mặc định. Lưu ý: Việc thay đổi package name (tên gói) sau này có thể tốn nhiều công sức.

6. Bỏ chọn hai tùy chọn Include C++ support và Include Kotlin support, sau đó nhấn Next.
7. Trong màn hình Target Android Devices (Thiết bị Android mục tiêu), đảm bảo tùy chọn Phone and Tablet (Điện thoại và máy tính bảng) được chọn.

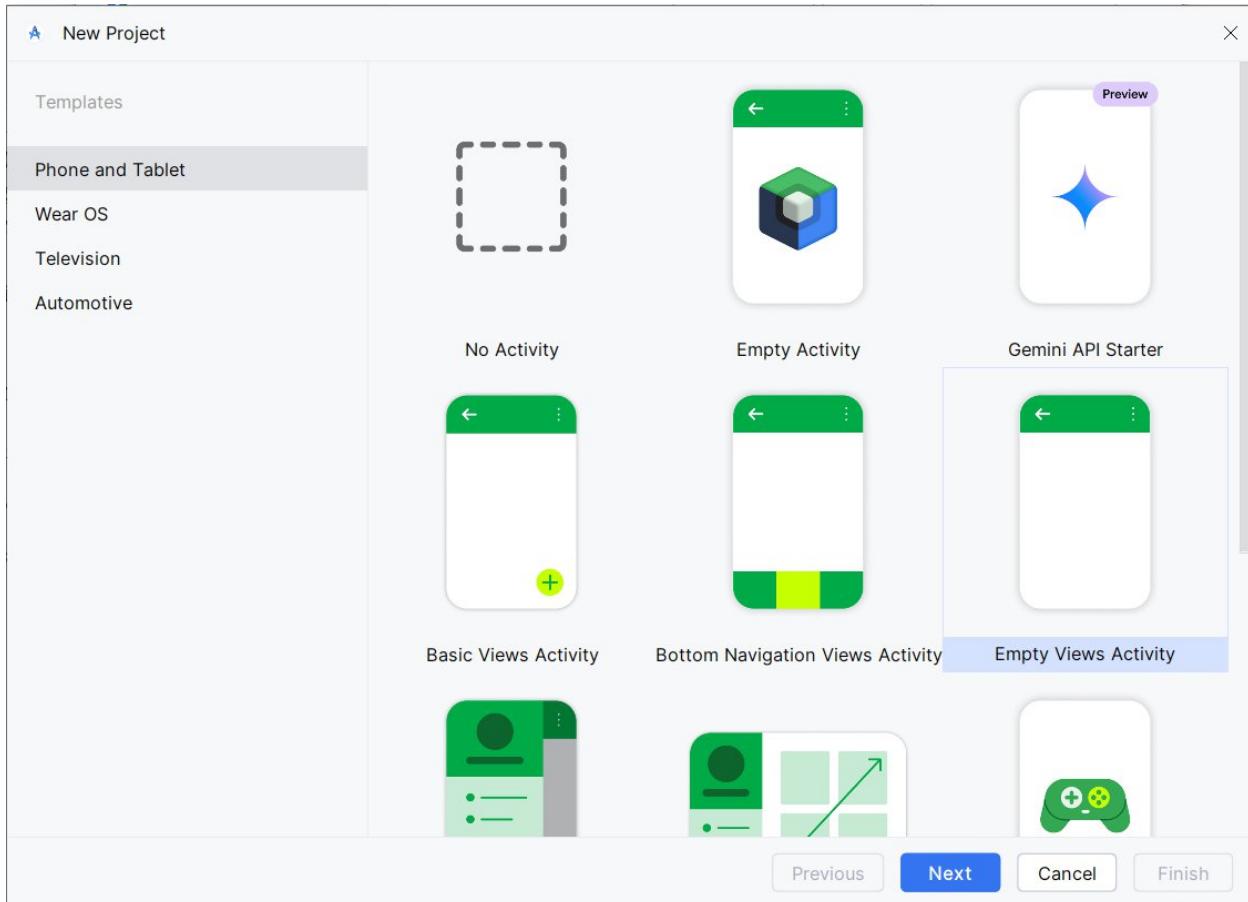
**Kiểm tra Minimum SDK (SDK tối thiểu) có được đặt là API 15: Android 4.0.3 Ice Cream Sandwich không. Nếu không, hãy sử dụng menu thả xuống để chọn đúng phiên bản.**



Các thiết lập này được sử dụng trong các ví dụ của khóa học này. Tại thời điểm viết hướng dẫn, những thiết lập này giúp ứng dụng Hello World tương thích với 97% thiết bị Android đang hoạt động trên Cửa hàng Google Play.

8. Bỏ chọn tùy chọn Instant App support (Hỗ trợ ứng dụng tức thì) và tất cả các tùy chọn khác, sau đó nhấn Next.
  - o Nếu dự án của bạn cần thêm các thành phần bổ sung cho SDK mục tiêu, Android Studio sẽ tự động tải xuống và cài đặt.
9. Cửa sổ Add an Activity (Thêm một Activity) sẽ xuất hiện. Activity là một màn hình hoặc một thao tác cụ thể mà người dùng có thể thực hiện trong ứng dụng. Đây là một thành phần quan trọng của bất kỳ ứng dụng

**Android nào. Mỗi Activity thường đi kèm với một layout để xác định giao diện người dùng trên màn hình. Android Studio cung cấp nhiều mẫu Activity để bạn dễ dàng bắt đầu chọn Empty Activity (Hoạt động trống) như hình bên dưới, rồi nhấn Next.**



**10. Cửa sổ Configure Activity (Cấu hình Activity) sẽ xuất hiện. Giao diện này có thể khác nhau tùy vào mẫu Activity bạn đã chọn ở bước trước. Mặc định, Empty Activity sẽ được đặt tên là MainActivity. Bạn có thể thay đổi tên nếu muốn, nhưng bài hướng dẫn này sẽ sử dụng MainActivity.**

**11. Đảm bảo tùy chọn "Generate Layout file" (Tạo tệp giao diện) được chọn. Tên tệp layout mặc định là activity\_main, bạn có thể đổi nếu muốn, nhưng trong bài hướng dẫn này, chúng ta sẽ sử dụng activity\_main.**

**12. Đảm bảo tùy chọn "Backwards Compatibility (App Compat)" được chọn. Tùy chọn này giúp ứng dụng của bạn tương thích với các phiên bản Android cũ hơn.**

### 13.Nhấn Finish để hoàn tất quá trình tạo dự án.

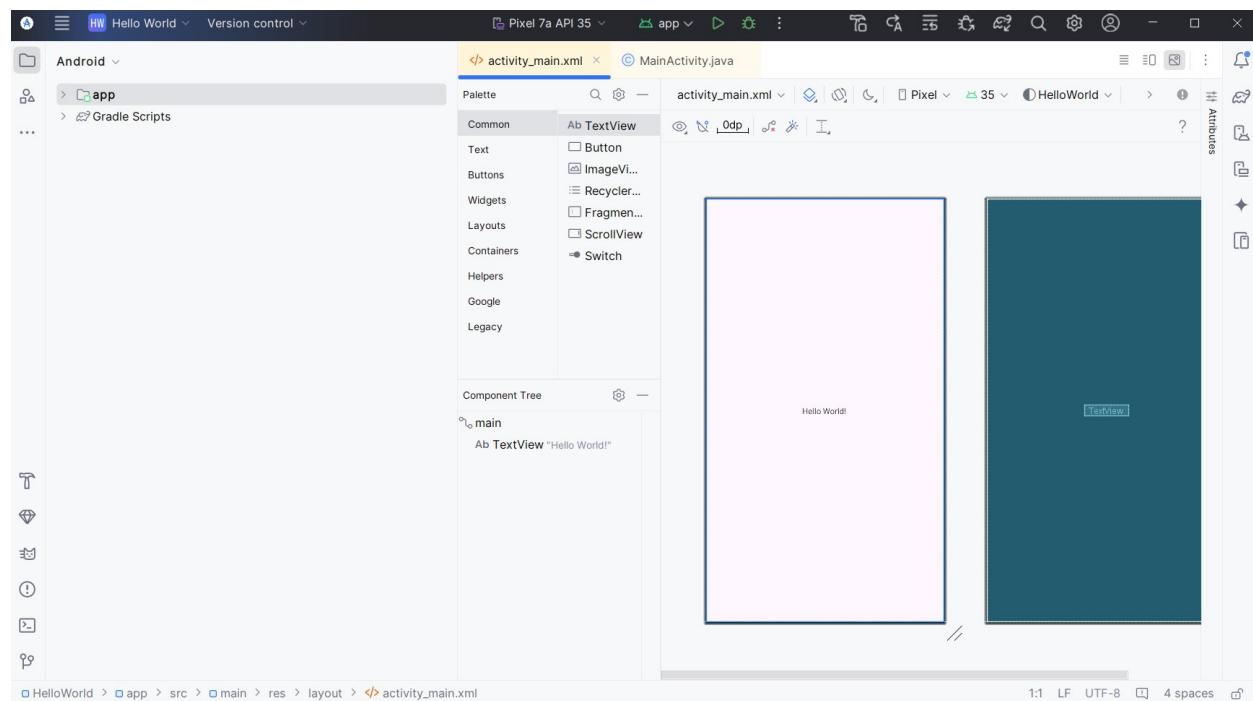
Sau khi hoàn tất, Android Studio sẽ tạo một thư mục chứa dự án của bạn và tiến hành xây dựng dự án bằng Gradle. Quá trình này có thể mất một chút thời gian.

**Mẹo:** Bạn có thể truy cập trang Configure your build (Cấu hình bản dựng) dành cho nhà phát triển để tìm hiểu chi tiết hơn về quá trình này.

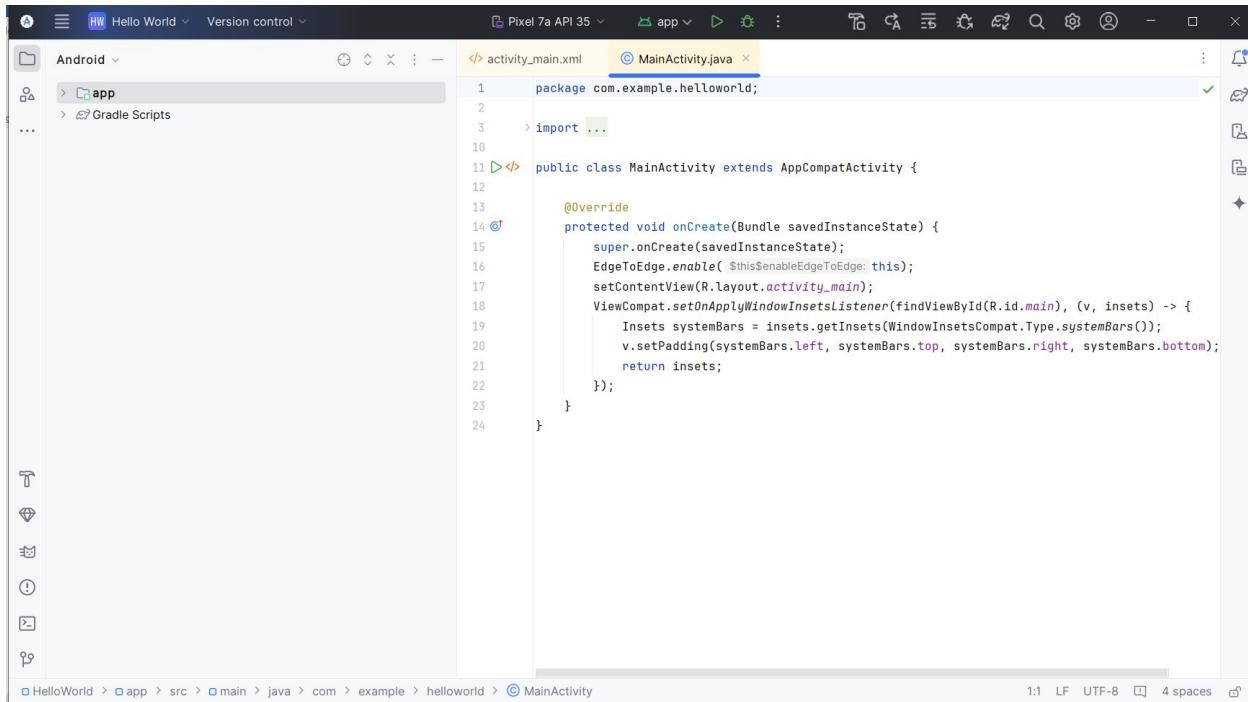
Ngoài ra, bạn có thể thấy cửa sổ "Tip of the day" (Mẹo trong ngày), hiển thị các phím tắt và mẹo hữu ích khác. Nhấn Close để đóng cửa sổ này.

Sau khi Android Studio hoàn tất, trình chỉnh sửa mã sẽ xuất hiện. Hãy thực hiện các bước sau:

1. Nhấp vào tab activity\_main.xml để mở trình chỉnh sửa giao diện.
2. Nếu chưa được chọn, hãy nhấp vào tab Design trong trình chỉnh sửa để hiển thị giao diện đồ họa của layout, giống như hình minh họa bên dưới.



3. Nhấp vào tab MainActivity.java để mở trình chỉnh sửa mã, nơi bạn có thể xem và chỉnh sửa mã nguồn của ứng dụng, như hình minh họa bên dưới.



The screenshot shows the Android Studio interface with the project 'Hello World' selected. The left sidebar shows the project structure with 'app' selected. The main editor window displays the Java code for 'MainActivity.java'. The code is as follows:

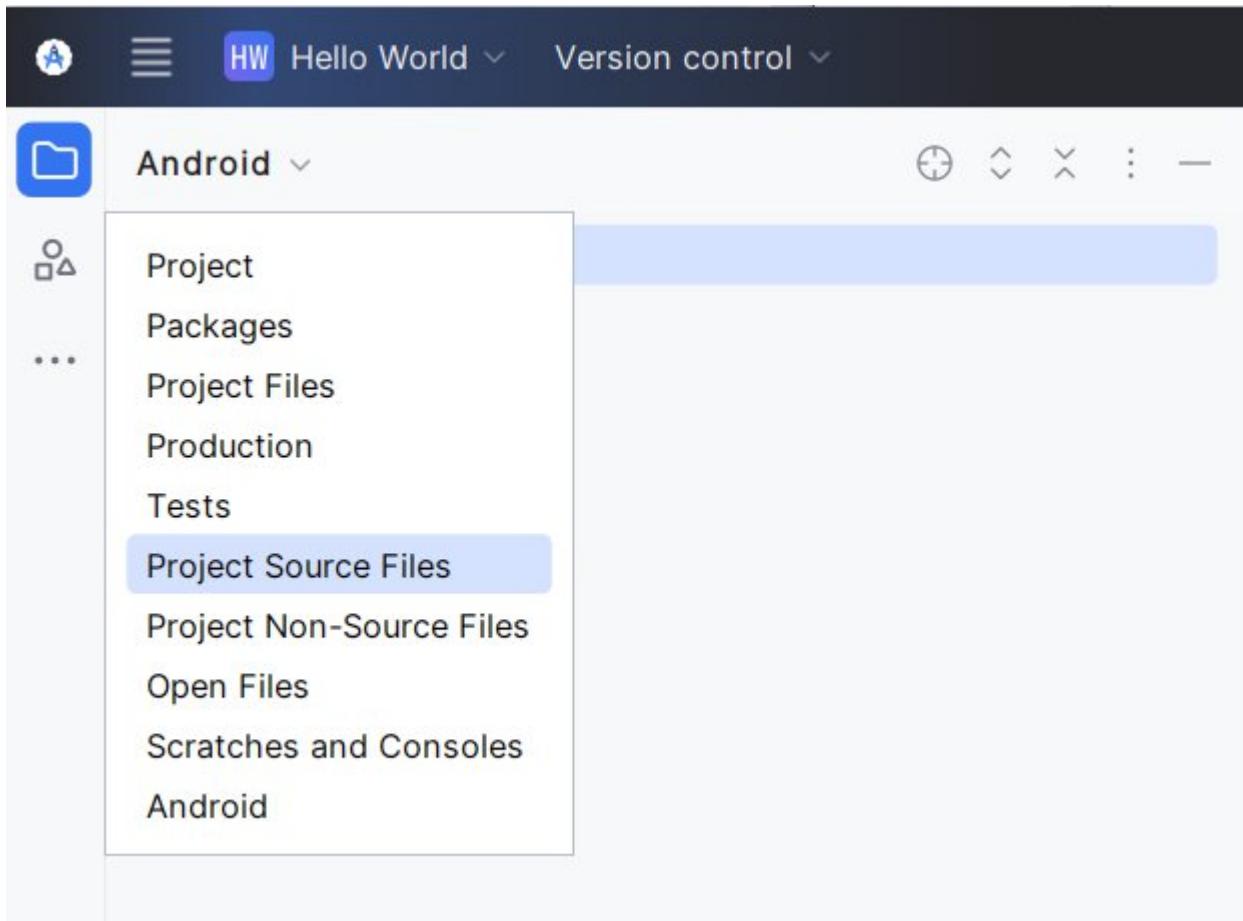
```
1 package com.example.helloworld;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
11    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

The status bar at the bottom indicates the file path: HelloWorld > app > src > main > java > com > example > helloworld > MainActivity. It also shows the encoding as UTF-8 and 4 spaces.

## 2.2 Khám phá bảng điều hướng Project > Android

Trong phần này, bạn sẽ tìm hiểu cách tổ chức dự án trong Android Studio.

1. Nếu chưa được chọn, hãy nhấp vào tab Project trong cột tab dọc bên trái cửa sổ Android Studio. Khi đó, bảng Project sẽ xuất hiện.
2. Để xem dự án theo cấu trúc thư mục tiêu chuẩn của Android, hãy chọn Android từ menu thả xuống ở đầu bảng Project, như hình minh họa bên dưới.

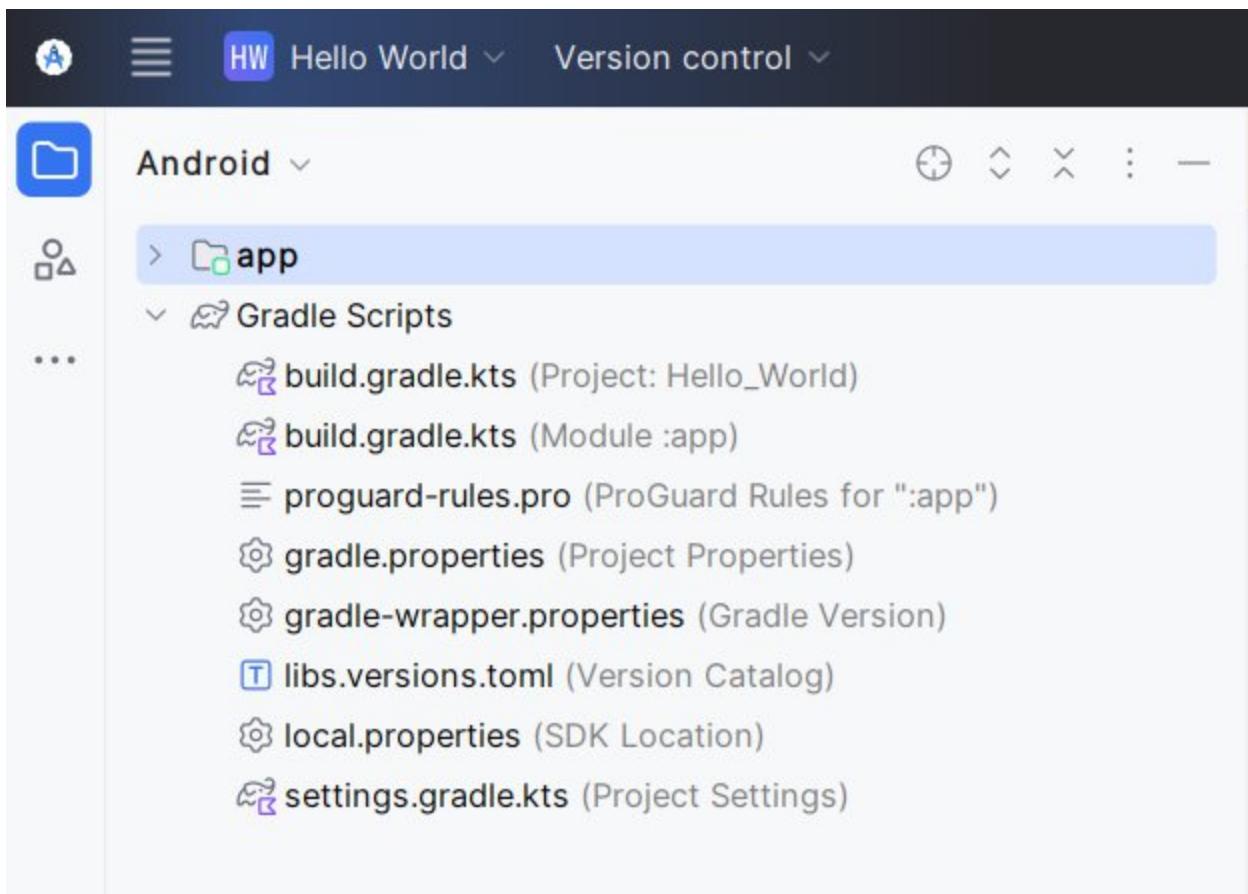


Lưu ý: Chương này và các chương khác sẽ gọi Project pane (bảng điều hướng dự án) khi được đặt ở chế độ Android là Project > Android pane.

### 2.3 Khám phá thư mục Gradle Scripts

Hệ thống Gradle trong Android Studio giúp bạn dễ dàng thêm các thư viện bên ngoài hoặc các mô-đun thư viện khác vào dự án dưới dạng dependencies (phụ thuộc).

Khi bạn tạo một dự án ứng dụng mới, bảng điều hướng Project > Android sẽ xuất hiện với thư mục Gradle Scripts được mở rộng, như hình minh họa bên dưới.



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

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

Thư mục này chứa tất cả các tệp cần thiết cho hệ thống build của dự án.

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

Đây là nơi chứa các tùy chọn cấu hình chung cho tất cả các mô-đun trong dự án của bạn. Mỗi dự án trong Android Studio đều có một tệp Gradle build cấp cao nhất. Hầu hết thời gian, bạn không cần thay đổi tệp này, nhưng việc hiểu nội dung của nó vẫn rất hữu ích.

Mặc định, tệp build.gradle cấp cao nhất sử dụng khôi buildscript để xác định kho lưu trữ và các dependencies chung cho toàn bộ dự án. Khi ứng dụng của

bạn sử dụng dependencies bên ngoài (không phải thư viện nội bộ hoặc file tree), Gradle sẽ tìm kiếm chúng trong các kho lưu trữ trực tuyến được khai báo trong khôi repositories của tệp này. Mặc định, các dự án mới trong Android Studio sẽ khai báo hai kho lưu trữ chính: Jcenter và Google (bao gồm Google Maven repository)



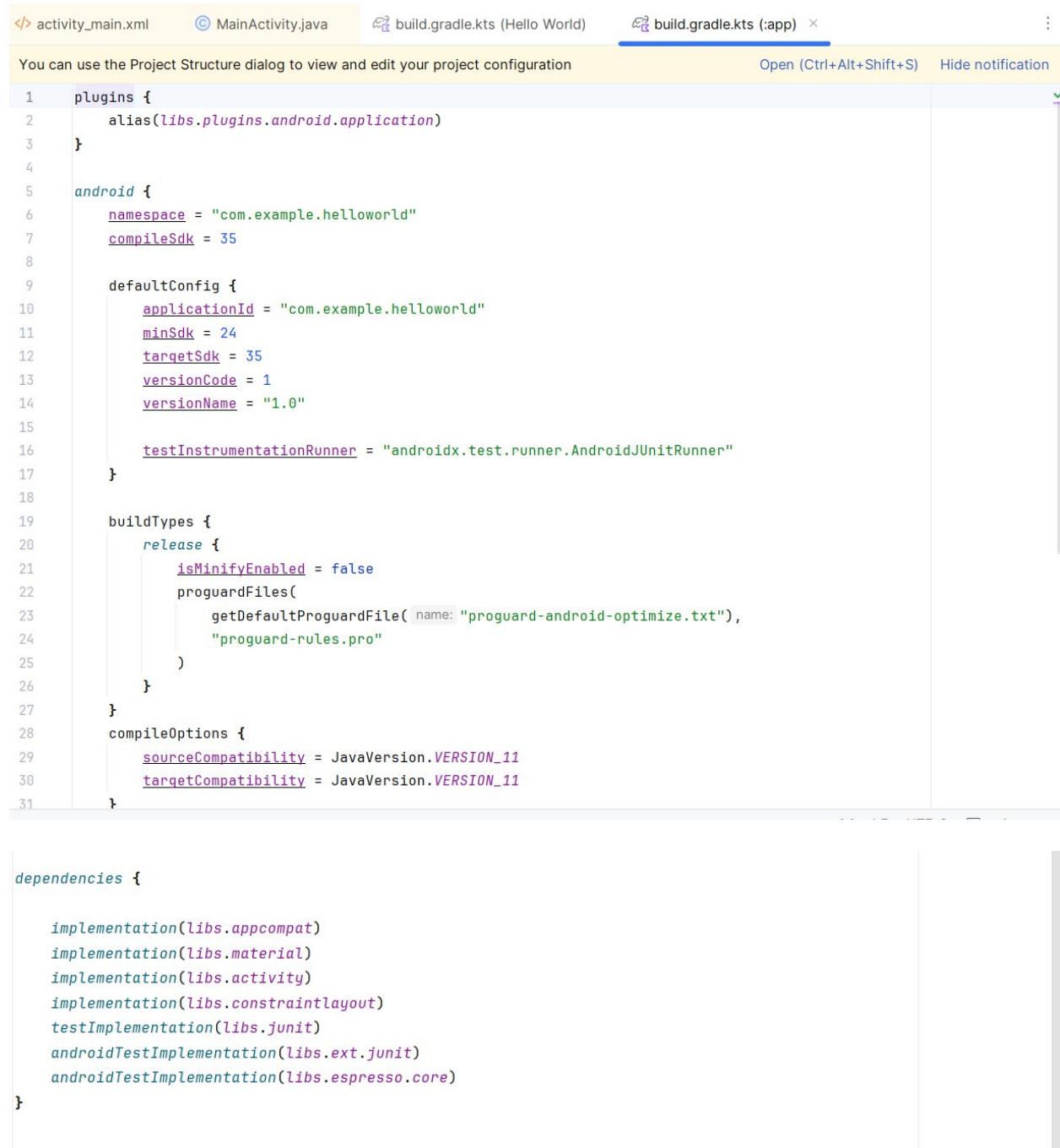
The screenshot shows the Android Studio interface with the 'build.gradle.kts (Hello World)' tab selected. The code editor displays the following Kotlin code:

```
// Top-level build file where you can add configuration options common to all sub-projects/modules
plugins {
    alias(libs.plugins.android.application) apply false
}
```

3. Tìm tệp build.gradle (Module: app). Ngoài tệp build.gradle ở cấp dự án, mỗi module cũng có tệp build.gradle riêng, cho phép bạn cấu hình các cài đặt build cho từng module cụ thể (ứng dụng HelloWorld chỉ có một module). Việc cấu hình các cài đặt build này cho phép bạn cung cấp các tùy chọn đóng gói tùy chỉnh, chẳng hạn như các loại build bổ sung và các product flavor. 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 được chỉnh sửa nhiều nhất khi thay đổi cấu hình ở cấp ứng dụng, chẳng hạn như khai báo các dependency trong phần dependencies. Bạn có thể khai báo một thư viện phụ thuộc bằng một trong nhiều cấu hình dependency khác nhau. Mỗi cấu hình dependency cung cấp cho Gradle các hướng dẫn khác nhau về cách sử dụng thư viện.

## Dưới đây là tệp build.gradle (Module: app) cho ứng dụng HelloWord:



```
plugins {
    alias(libs.plugins.android.application)
}

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

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

        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            isMinifyEnabled = false
            proguardFiles(
                getDefaultProguardFile("proguard-android-optimize.txt"),
                "proguard-rules.pro"
            )
        }
    }
    compileOptions {
        sourceCompatibility = JavaVersion.VERSION_11
        targetCompatibility = JavaVersion.VERSION_11
    }
}

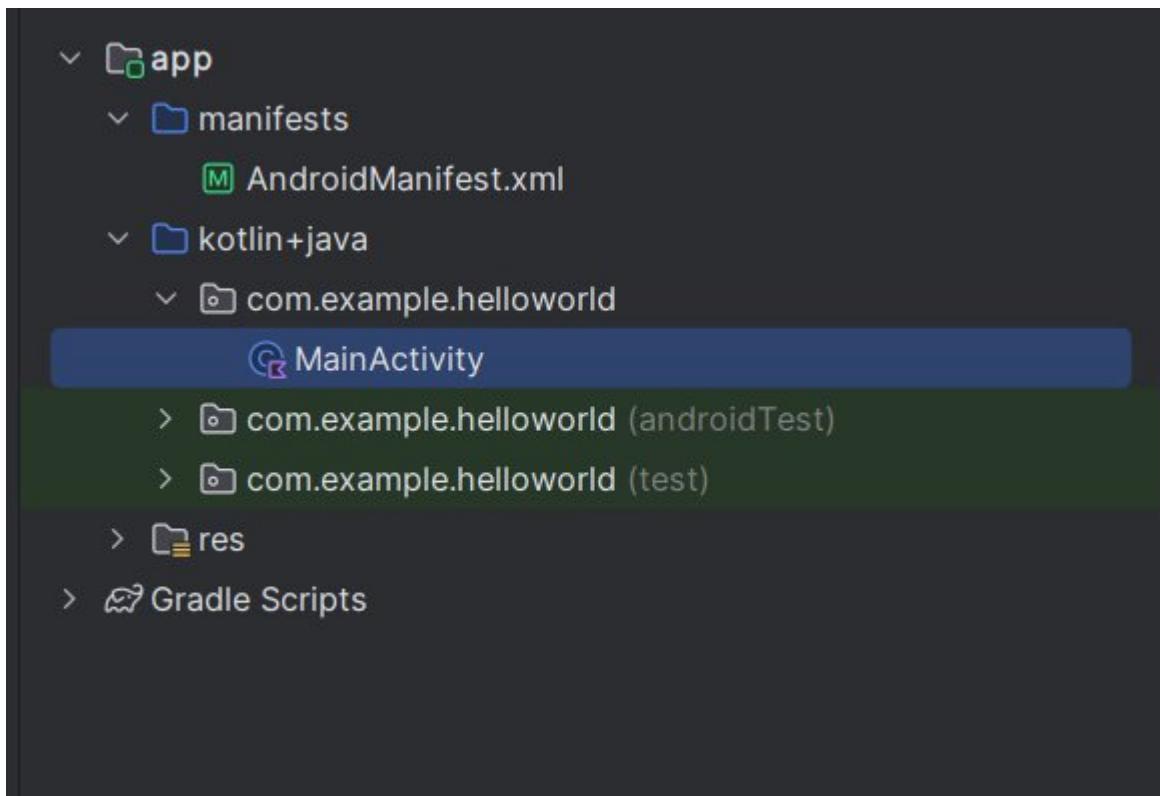
dependencies {
    implementation(libs.appcompat)
    implementation(libs.material)
    implementation(libs.activity)
    implementation(libs.constraintlayout)
    testImplementation(libs.junit)
    androidTestImplementation(libs.ext.junit)
    androidTestImplementation(libs.espresso.core)
}
```

4. Nhấn vào biểu tượng tam giác bên cạnh Gradle Scripts để thu gọn danh sách các tệp đó

## 2.4 Khám phá các thư mục app và res

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

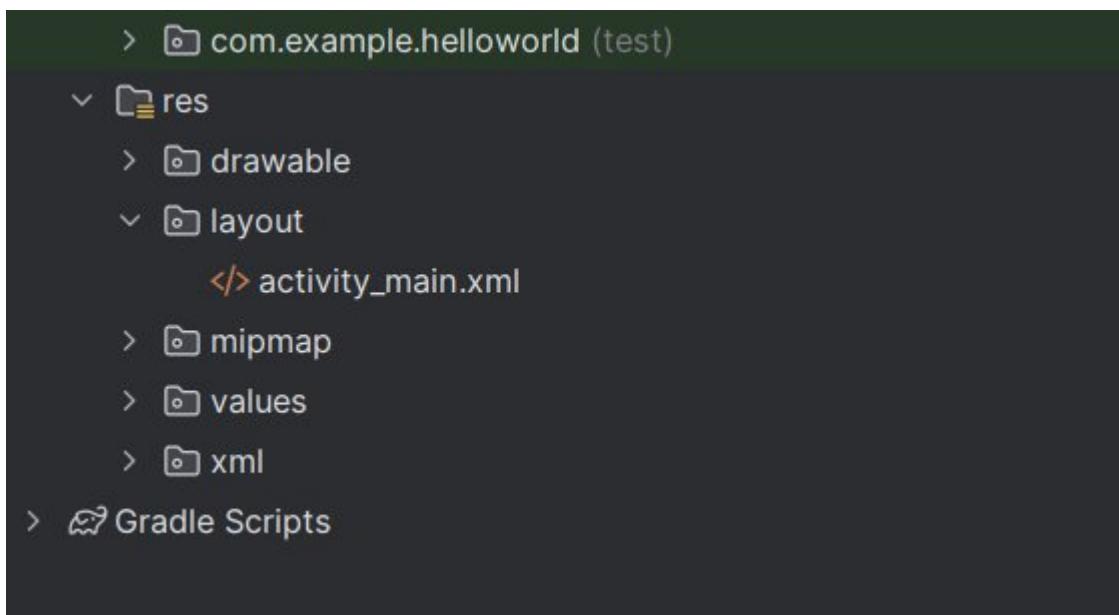
1. Mở rộng thư mục app, thư mục java, và thư mục com.example.android.helloworld để xem tệp MainActivity.java. Nhấp đúp vào tệp để mở nó trong trình chỉnh sửa mã.



Thư mục java bao gồm các tệp lớp Java trong ba thư mục con, như được hiển thị trong hình trên. Thư mục com.example.hello.helloworld (hoặc tên miền mà bạn đã chỉ định) chứa tất cả các tệp cho một gói ứng dụng. Hai thư mục còn lại được sử dụng để kiểm thử và sẽ được mô tả trong một bài học khác.

Đối với ứng dụng Hello World, chỉ có một gói duy nhất và nó chứa `MainActivity.java`. Tên của Activity đầu tiên (màn hình đầu tiên) mà người dùng nhìn thấy, đồng thời khởi tạo các tài nguyên chung của ứng dụng, theo thông lệ sẽ được gọi là `MainActivity` (phần mở rộng tệp được ẩn trong chế độ hiển thị `Project > Android`).

**2.Mở rộng thư mục res và thư mục layout, sau đó nhấp đúp vào tệp activity\_main.xml để mở nó trong trình chỉnh sửa giao diện.**



**Thư mục res chứa các tài nguyên như bố cục (layouts), chuỗi ký tự (strings) và hình ảnh (images).**

Một Activity thường được liên kết với một bố cục giao diện người dùng (UI views), được định nghĩa trong một tệp XML.

Tệp này thường được đặt tên theo Activity tương ứng của nó.

## 2.5 Khám phá thư mục manifests

**Thư mục manifests chứa các tệp cung cấp thông tin quan trọng về ứng dụng của bạn cho hệ thống Android.**

Hệ thống Android 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. Tất cả các thành phần của ứng dụng, chẳng hạn như mỗi Activity, đều phải được khai báo trong tệp XML này. Trong các bài học khác, bạn sẽ sửa đổi tệp này để thêm tính năng và quyền truy cập tính năng.

Để tìm hiểu tổng quan, hãy xem phần Giới thiệu về App Manifest (App Manifest Overview).

### Bài 3: Sử dụng thiết bị ảo (trình giả lập)

Trong bài này, bạn sẽ sử dụng Android Virtual Device (AVD) Manager để tạo một thiết bị ảo (hay còn gọi là trình giả lập) mô phỏng cấu hình của một thiết bị Android cụ thể và chạy ứng dụng trên đó.

Lưu ý rằng Android Emulator có các yêu cầu bổ sung ngoài yêu cầu hệ thống cơ bản của Android Studio.

Bằng cách sử dụng AVD Manager, bạn có thể:

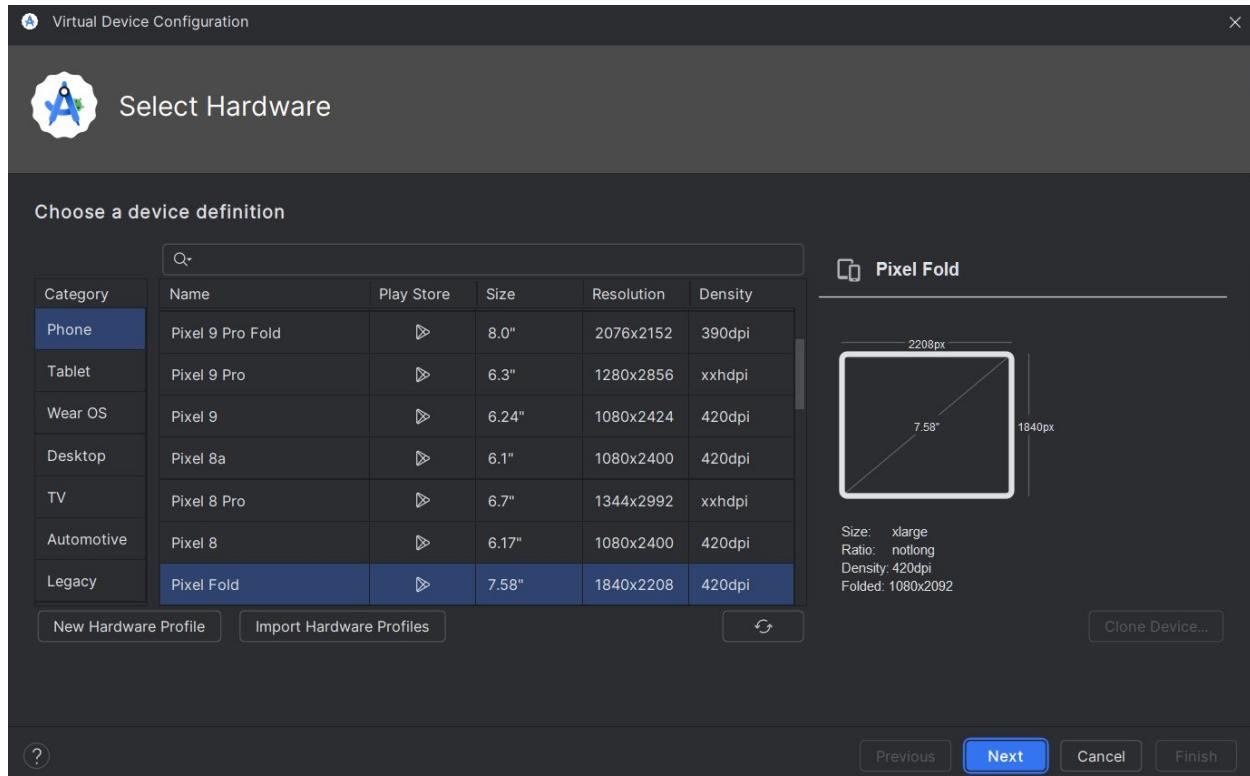
- Xác định các đặc điểm phần cứng của thiết bị,
- Cấu hình API level, bộ nhớ, giao diện và các thuộc tính khác,
- Lưu nó dưới dạng một thiết bị ảo.

Với thiết bị ảo, bạn có thể kiểm thử ứng dụng trên nhiều cấu hình thiết bị khác nhau (chẳng hạn như điện thoại hoặc máy tính bảng với các API level khác nhau) mà không cần sử dụng thiết bị vật lý.

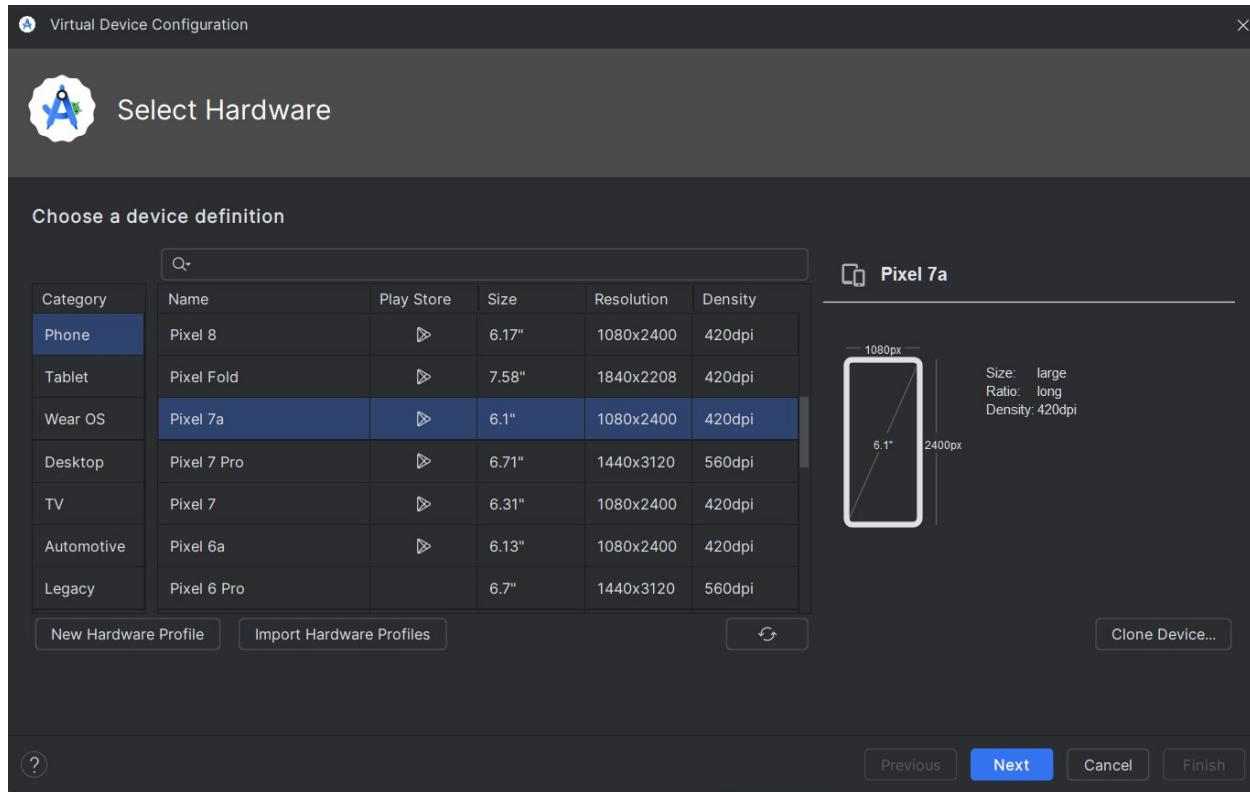
#### 3.1 Tạo một thiết bị ảo Android (AVD)

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

1. Trong Android Studio, chọn Tools > Android > AVD Manager, hoặc nhấp vào biểu tượng AVD Manager trên thanh công cụ.
2. Màn hình Your Virtual Devices xuất hiện.
  - Nếu bạn đã tạo thiết bị ảo trước đó, màn hình sẽ hiển thị danh sách các thiết bị.
  - Nếu chưa có thiết bị nào, danh sách sẽ trống (như hình minh họa bên dưới).

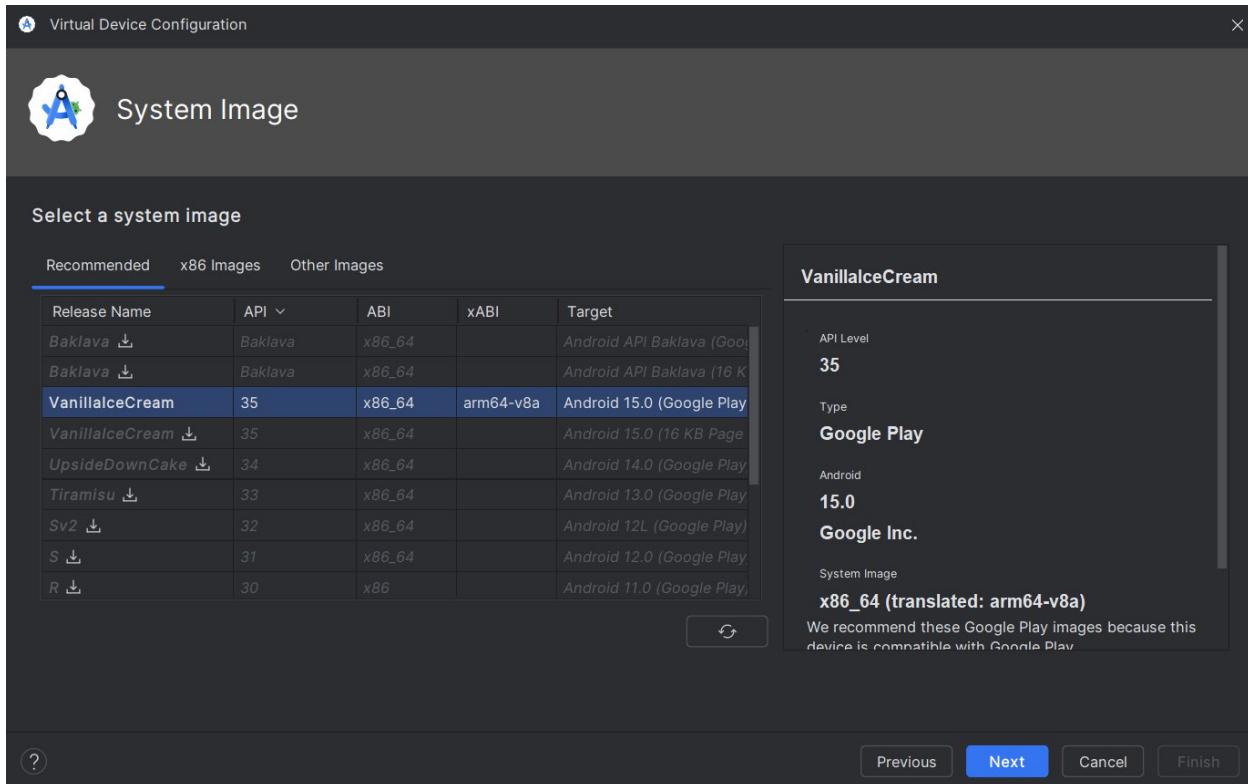


2. Nhấp vào +Create Virtual Device. Cửa sổ Select Hardware xuất hiện, hiển thị danh sách các thiết bị phần cứng được cấu hình sẵn. Đối với mỗi thiết bị, bảng cung cấp một cột cho kích thước màn hình đường chéo (Size), độ phân giải màn hình tính bằng pixel (Resolution) và mật độ điểm ảnh (Density).



**3. Chọn một thiết bị, chẳng hạn như Nexus 5X hoặc Pixel XL, rồi nhấp vào Next. Màn hình System Image sẽ xuất hiện.**

**4. Nhấp vào tab Recommended nếu nó chưa được chọn, sau đó chọn phiên bản hệ điều hành Android để chạy trên thiết bị ảo (chẳng hạn như Oreo).**



Có nhiều phiên bản khác ngoài những phiên bản hiển thị trong tab Recommended. Hãy xem thêm trong các tab x86 Images và Other Images để tìm các phiên bản khác.

Nếu có liên kết Download bên cạnh một hệ điều hành mà bạn muốn sử dụng, điều đó có nghĩa là nó chưa được cài đặt. Nhấp vào liên kết để bắt đầu tải xuống, sau đó nhấp vào Finish khi quá trình tải xuống hoàn tất.

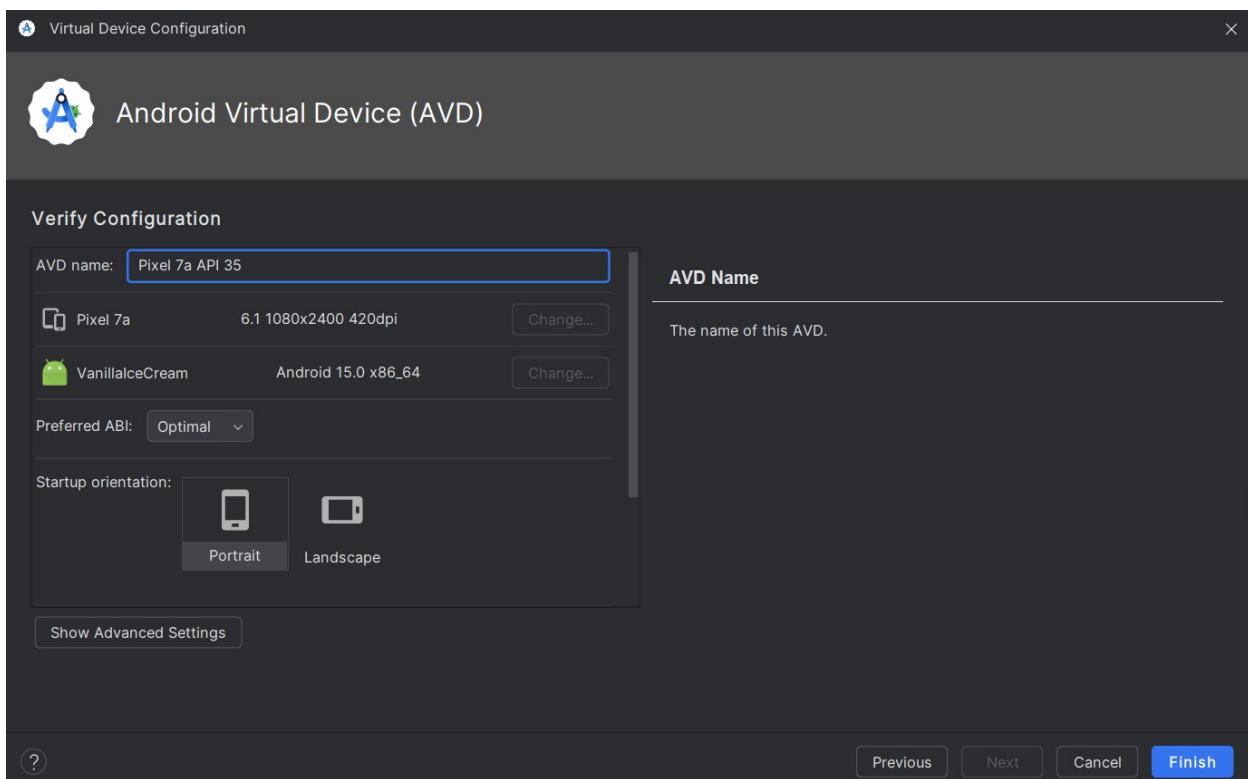
5.Sau khi chọn xong hệ điều hành, nhấp vào Next. Cửa sổ Android Virtual Device (AVD) sẽ xuất hiện. Bạn có thể đổi tên AVD nếu muốn. Kiểm tra lại cấu hình của bạn và nhấp vào Finish.

### 3.2 Chạy ứng dụng trên thiết bị ảo

Trong bước này, bạn sẽ chạy ứng dụng Hello World của mình.

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

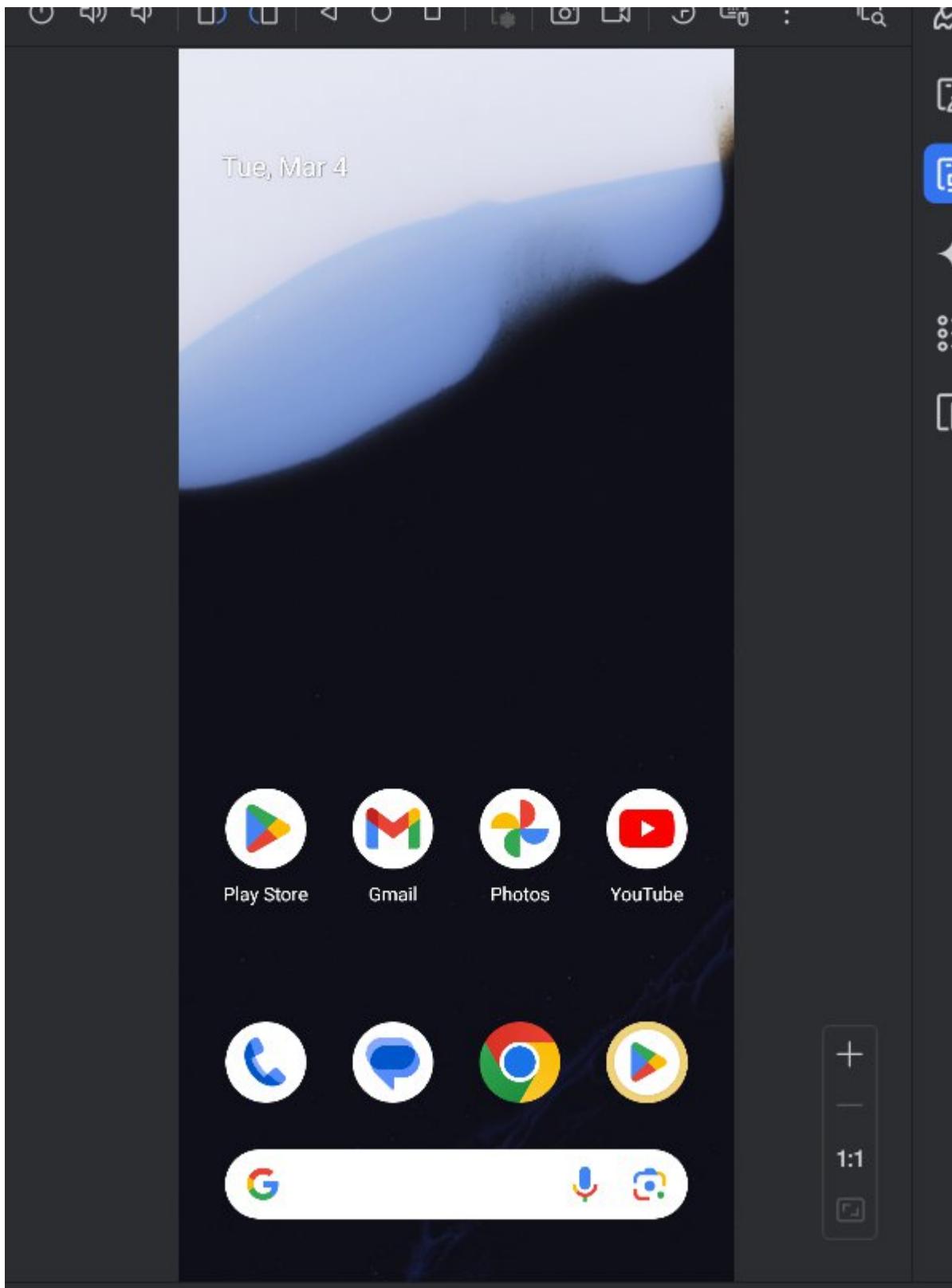
**2.Trong cửa sổ Select Deployment Target, dưới mục Available Virtual Devices, chọn thiết bị ảo mà bạn vừa tạo, sau đó nhấn vào OK.**



**Trình giả lập sẽ khởi động giống như một thiết bị vật lý. Tùy vào tốc độ của máy tính, quá trình này có thể mất một chút thời gian.**

**Ứng dụng của bạn sẽ được biên dịch (build), và khi trình giả lập sẵn sàng, Android Studio sẽ tải ứng dụng lên trình giả lập và chạy nó.**

**Bạn sẽ thấy ứng dụng Hello World hiển thị như trong hình minh họa sau.**



**Mẹo:** Khi kiểm thử trên thiết bị ảo, bạn nên khởi động nó một lần ngay từ đầu phiên làm việc của mình. Bạn không nên đóng nó cho đến khi bạn hoàn thành việc kiểm thử ứng dụng của mình, để ứng dụng không phải trải qua quá trình khởi động thiết bị nữa. Để đóng thiết bị ảo, hãy nhấp vào nút X ở trên cùng của 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 một thiết bị di động vật lý như điện thoại hoặc máy tính bảng. Bạn nên luôn kiểm tra ứng dụng của mình trên cả thiết bị ảo và thiết bị vật lý.

**Bạn cần gì:**

- Một thiết bị Android như điện thoại hoặc máy tính bảng.
- Một 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. Hãy xem 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 dựa 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. Chức năng này được bật trong cài đặt "Tùy chọn nhà phát triển" của thiết bị.

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

1. Trên thiết bị của bạn, mở Cài đặt (Settings), tìm kiếm "Thông tin điện thoại" (About phone), nhấp vào "Thông tin điện thoại" và chạm vào "Số bản dựng" (Build number) bảy lần.
2. Quay lại màn hình trước (Cài đặt / Hệ thống). "Tùy chọn nhà phát triển" (Developer options) xuất hiện trong danh sách. Chạm vào "Tùy chọn nhà phát triển".

3. Chọn "Gỡ lỗi USB" (USB Debugging).

#### 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 của bạn bằng cáp USB.
2. Nhấp vào nút Chạy (Run) trên thanh công cụ. Cửa sổ Chọn Mục tiêu Triển khai (Select Deployment Target) sẽ mở ra với danh sách các trình giả lập khả dụng và các thiết bị đã kết nối.
3. Chọn thiết bị của bạn và nhấp vào OK.

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

#### Khắc phục sự cố

Nếu Android Studio không nhận ra thiết bị của bạn, hãy thử những cách 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 báo thiết bị là "không được ủy quyền", hãy làm theo các bước sau:

1. Rút phích cắm của thiết bị.
2. Trên thiết bị, mở "Tùy chọn nhà phát triển" (Developer Options) trong ứng dụng Cài đặt (Settings).
3. Chạm vào "Thu hồi ủy quyền gỡ lỗi USB" (Thu hồi ủy 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 thích hợp cho thiết bị của mình. Xem tài liệu "Sử dụng Thiết bị Phàn cứng" (Sử dụng thiết bị phần cứng).

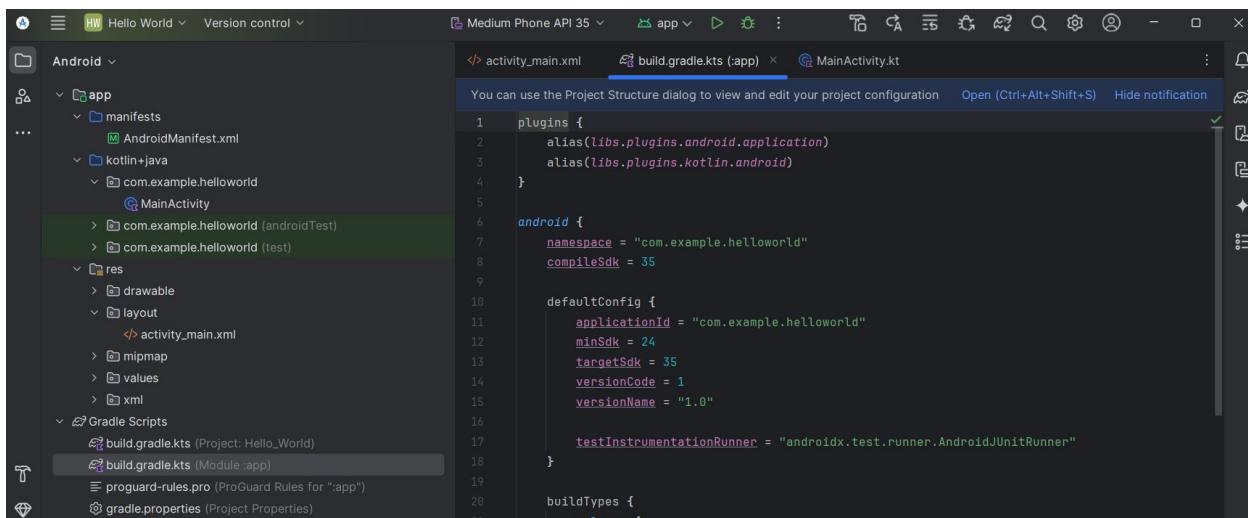
#### Nhiệm vụ 5: Thay đổi cấu hình Gradle của ứng dụng

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

Thực hiện theo các bước sau:

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



The screenshot shows the Android Studio interface with the project 'Hello World' open. The left sidebar shows the project structure with a expanded 'app' module. The right side shows the code editor with the 'build.gradle.kts (app)' file open. The code in the file is as follows:

```
plugins {
    alias(libs.plugins.android.application)
    alias(libs.plugins.kotlin.android)
}

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

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

        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
```

Trình soạn thảo mã hiển thị một thanh thông báo ở trên cùng với liên kết "Sync Now".

## 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 nó 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 "Sync Now" trong thanh thông báo xuất hiện khi thực hiện thay đổi (như được hiển thị trong hình trước), hoặc nhấp vào biểu tượng "Sync Project with Gradle Files" 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.

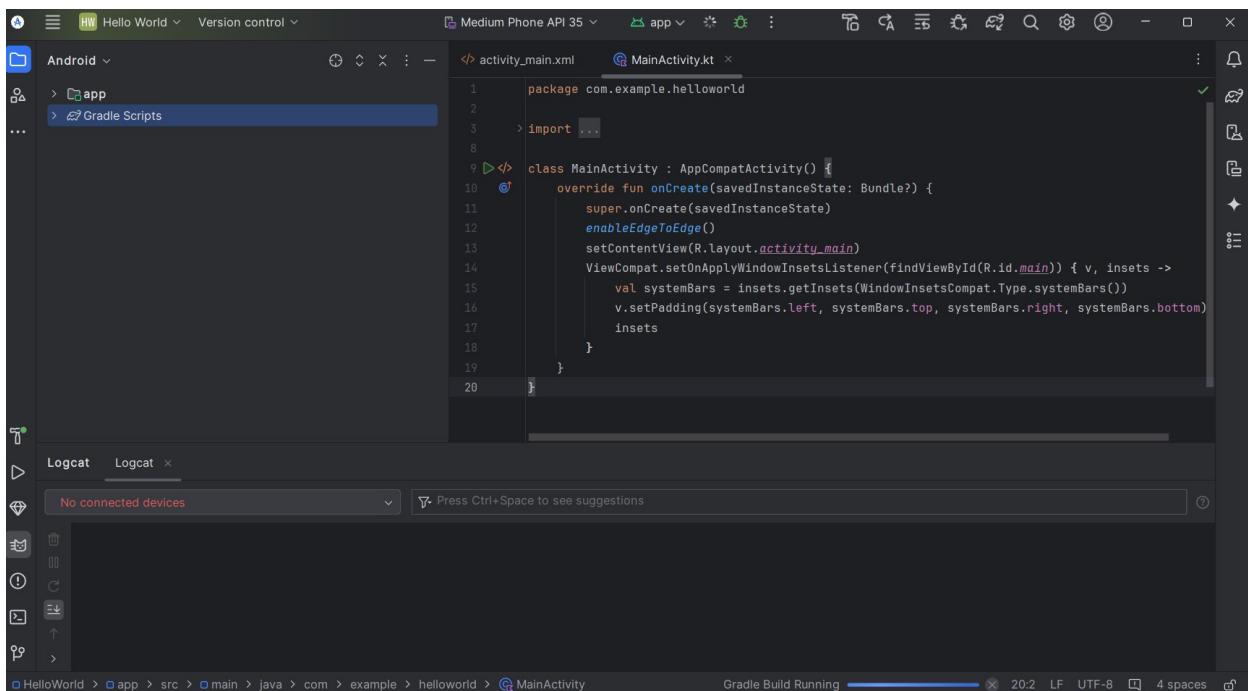
Để tìm hiểu sâu hơn về Gradle, hãy xem tài liệu "Tổng quan về Hệ thống Bản dựng" (Build System Overview) và "Cấu hình Bản dựng Gradle" (Configuring Gradle Builds).

**Nhiệm vụ 6: Thêm các câu lệnh ghi nhật ký (log statements) vào ứng dụng của bạn**

Trong nhiệm vụ này, bạn sẽ thêm các câu lệnh ghi nhật ký vào ứng dụng của mình, câu lệnh này hiển thị các thông báo trong ngăn Logcat. Các 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 các ngoại lệ.

## 6.1 Xem ngăn Logcat

Để xem ngăn Logcat, hãy nhấp vào tab Logcat ở cuối cửa sổ Android Studio như hình dưới đây.



Trong hình trên:

1. Tab Logcat để mở và đóng ngăn Logcat, nơi hiển thị thông tin về ứng dụng của bạn khi nó đang chạy. Nếu bạn thêm các câu lệnh Log vào ứng dụng của mình, các thông báo Log sẽ xuất hiện ở đây.

2. Menu Mức độ nhật ký (Log level) được đặt thành Verbose (mặc định), hiển thị tất cả các thông báo Log. Các cài đặt khác bao gồm Debug, Error, Info và Warn.

## 6.2 Thêm các câu lệnh ghi nhật ký vào ứng dụng của bạn

Các câu lệnh ghi nhật ký trong mã ứng dụng của bạn hiển thị các thông báo trong ngăn Logcat. Ví dụ:

```
// Log message
Log.d(LOG_TAG, msg: "Hello World")
```

Các thành phần của thông báo là:

- Log: Lớp Log để gửi các thông báo nhật ký đến ngăn Logcat.
- d: Mức độ nhật ký Gỡ lỗi (Debug Log level), dùng để lọc hiển thị thông báo nhật ký trong ngăn Logcat. Các mức độ nhật ký khác là e cho Lỗi (Error), w cho Cảnh báo (Warn) và i cho Thông tin (Info).
- "MainActivity": Đối số đầu tiên là một thẻ (tag) có thể được sử dụng để lọc các thông báo trong ngăn Logcat. Thông thường, đây là tên của Activity nơi thông báo bắt nguồn. Tuy nhiên, bạn có thể đặt bất cứ điều gì hữu ích cho bạn để gỡ lỗi. Theo quy ước, các thẻ nhật ký được định nghĩa là các hằng số cho Activity:

```
class MainActivity : AppCompatActivity() {

    companion object {
        private const val LOG_TAG = "MainActivity"
    }
}
```

- "Hello world": Đối số thứ hai là thông báo thực tế.

Thực hiện theo các bước sau:

1. Mở ứng dụng Hello World của bạn trong Android Studio và mở MainActivity.
2. Để tự động thêm các câu lệnh nhập (imports) rõ ràng vào dự án của bạn (chẳng hạn như android.util.Log cần thiết để sử dụng Log), hãy chọn

"File > Settings" trong Windows hoặc "Android Studio > Preferences" trong macOS.

3. Chọn "Editor > General > Auto Import". Chọn tất cả các hộp kiểm và đặt "Insert imports on paste" thành "All".
4. Nhấp vào "Apply" rồi nhấp vào "OK".
5. Trong phương thức onCreate() của MainActivity, hãy thêm câu lệnh sau:

```
    Log.d("MainActivity", "Hello World");
```

Phương thức onCreate() bây giờ sẽ trông giống như đoạn mã sau:

```
override fun onCreate(savedInstanceState: Bundle?) {  
    super.onCreate(savedInstanceState)  
    enableEdgeToEdge()  
    setContentView(R.layout.activity_main)  
  
    // Log message  
    Log.d(LOG_TAG, msg: "Hello World")
```

6.Nếu ngăn Logcat chưa mở, hãy nhấp vào tab Logcat ở cuối Android Studio để mở nó.

7.Kiểm tra xem tên mục tiêu (target) và tên gói (package name) của ứng dụng có chính xác hay không.

8.Thay đổi mức độ nhật ký (Log level) trong ngăn Logcat thành Gỡ lỗi (Debug) (hoặc giữ nguyên là Chi tiết (Verbose) vì có rất ít thông báo nhật ký).

9.Chạy ứng dụng của bạn.

```
    Log.d("MainActivity", "Hello World");
```

## Thử thách lập trình

Lưu ý: Tất cả các thử thách lập trình là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

**Thử thách:** Bây giờ bạn đã thiết lập và quen thuộc với quy trình phát triển cơ bản, hãy thực hiện những điều sau:

1. Tạo một dự án mới trong Android Studio.
2. Thay đổi lời chào "Hello World" thành "Chúc mừng sinh nhật đến [tên người có sinh nhật gần đây]" và tên của người có sinh nhật gần đây.
3. (Tùy chọn) Chụp ảnh màn hình ứng dụng đã hoàn thành của bạn và gửi email cho người mà bạn đã quên sinh nhật.
4. Một cách sử dụng phổ biến của lớp Log là ghi lại các ngoại lệ Java khi chúng xảy ra trong chương trình của bạn. Có một số phương thức hữu ích, chẳng hạn như Log.e(), bạn có thể sử dụng cho mục đích này. Tìm hiểu các phương thức bạn có thể sử dụng để bao gồm một ngoại lệ với một thông báo Log. Sau đó, viết mã trong ứng dụng của bạn để kích hoạt và ghi lại một ngoại lệ.

## Tóm tắt

- Để cài đặt Android Studio, hãy truy cập Android Studio và làm theo hướng dẫn để tải xuống và cài đặt.
- Khi tạo một ứng dụng mới, hãy đảm bảo rằng API 15: Android 4.0.3 IceCreamSandwich được đặt làm SDK tối thiểu.
- Để xem cấu trúc phân cấp Android của ứng dụng trong ngăn Project, hãy nhấp vào tab Project trong cột tab dọc, sau đó chọn Android trong menu bật lên ở trên cùng.
- Chính sửa tệp build.gradle(Module:app) khi bạn cần thêm thư viện mới vào dự án của mình hoặc thay đổi phiên bản thư viện.
- Tất cả mã và tài nguyên cho ứng dụng đều nằm trong các thư mục app và res. Thư mục java bao gồm các hoạt động, thử nghiệm và các thành phần khác trong mã nguồn Java. Thư mục res chứa các tài nguyên, chẳng hạn như bố cục, chuỗi và hình ảnh.
- Chính sửa tệp AndroidManifest.xml để thêm các thành phần tính năng và quyền vào ứng dụng Android của bạn. Tất cả các thành phần cho một ứng dụng, chẳng hạn như nhiều hoạt động, phải được khai báo trong tệp XML này.
- Sử dụng trình quản lý Thiết bị Ảo Android (AVD) để tạo một thiết bị ảo (còn được gọi là trình giả lập) để chạy ứng dụng của bạn.
- Thêm các câu lệnh Log vào ứng dụng của bạn, câu lệnh này hiển thị các thông báo trong ngăn Logcat như một công cụ cơ bản để gỡ lỗi.

- Để chạy ứng dụng của bạn trên thiết bị Android vật lý bằng Android Studio, hãy bật Gỡ lỗi USB trên thiết bị. Mở "Cài đặt" > "Thông tin điện thoại" và chạm vào "Số bản dựng" bảy lần. Quay lại màn hình trước ("Cài đặt") và chạm vào "Tùy chọn nhà phát triển". Chọn "Gỡ lỗi USB".

## Các khái niệm liên quan

Tài liệu về các khái niệm liên quan nằm trong 1.0: Giới thiệu về Android và 1.1 Ứng dụng Android đầu tiên của bạn.

## Tìm hiểu thêm

### Tài liệu Android Studio:

- Trang tải xuống Android Studio
- Ghi chú phát hành Android Studio
- Làm quen với Android Studio
- Công cụ dòng lệnh Logcat
- Trình quản lý Thiết bị Ảo Android (AVD)
- Tổng quan về Tệp Kê khai Ứng dụng (App Manifest Overview)
- Cấu hình bản dựng của bạn
- Lớp Log
- Tạo và quản lý Thiết bị Ảo

### Khác:

- Làm cách nào để cài đặt Java?
- Cài đặt Phần mềm JDK và Đặt JAVA\_HOME
- Trang web Gradle
- Cú pháp Apache Groovy
- Trang Wikipedia Gradle

## Bài tập về nhà

### Xây dựng và chạy một ứng dụng

- Tạo một dự án Android mới từ Mẫu Trống (Empty Template).
- Thêm các câu lệnh ghi nhật ký cho các mức độ nhật ký khác nhau trong onCreate() trong hoạt động chính.

- Tạo một trình giả lập cho một thiết bị, nhắm mục tiêu bất kỳ phiên bản Android nào bạn thích và chạy ứng dụng.
- Sử dụng bộ lọc trong Logcat để tìm các câu lệnh ghi nhật ký của bạn và điều chỉnh các mức để chỉ hiển thị các câu lệnh ghi nhật ký gỡ lỗi hoặc lỗi.

Trả lời các câu hỏi sau

Câu hỏi 1

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

- MainActivity.java
- AndroidManifest.xml
- activity\_main.xml
- xây dựng.gradle

Câu hỏi 2

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

- app\_name
- xmlns:ứng dụng
- android:tên
- applicationId

Câu hỏi 3

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

- Trình Giám sát Thiết bị Android (Android Device Monitor)
- Trình quản lý AVD (AVD Manager)
- Trình quản lý SDK (SDK Manager)
- Trình Chính sửa Chủ đề (Theme Editor)

Câu hỏi 4

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

Log.i("MainActivity", "MainActivity layout is complete"); Bạn thấy câu lệnh "MainActivity layout is complete" trong ngăn Logcat nếu menu Mức độ nhật ký được đặt thành mục nào sau đây? (Gợi ý: có thể có nhiều câu trả lời đúng.)

- Dài dòng
- Gỡ lỗi
- Thông tin
- Cảnh báo
- Lỗi
- Khẳng định

Gửi ứng dụng của bạn để chấm điểm

Kiểm tra để đảm bảo ứng dụng có những điều sau:

- Một Activity hiển thị "Hello World" trên màn hình.
- Các câu lệnh Log trong onCreate() trong hoạt động chính.
- Mức độ nhật ký trong ngăn Logcat chỉ hiển thị các câu lệnh ghi nhật ký gỡ lỗi hoặc lỗi.

## 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 view - mọi thành phần của màn hình đều là một View. Lớp View đại diện cho khối xây dựng cơ bản cho tất cả các thành phần giao diện người dùng và là 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 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 các mục có thể cuộn.

- **ImageView** để hiển thị hình ảnh.
- **ConstraintLayout** và **LinearLayout** để chứa các thành phần View khác và định vị chúng.

Mã Java hiển thị và điều khiển giao diện người dùng nằm trong một lớp mở rộng Activity. Một Activity thường được liên kết với một bố cục của các view giao diện người dùng được định nghĩa là một tệp XML (eXtended Markup Language). Tệp XML này thường được đặt tên theo Activity của nó và xác định bố cục của các thành phần View trên màn hình.

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

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

Trong bài thực hành này, bạn sẽ học cách tạo ứng dụng tương tác đầu tiên của mình - một ứng dụng cho phép tương tác người dùng. Bạn tạo một ứng dụng bằng cách sử dụng mẫu Empty Activity. Bạn cũng học cách sử dụng trình chỉnh sửa bố cục để thiết kế bố cục và cách chỉnh sửa bố cục trong XML. Bạn cần phát triển các kỹ năng này để có thể hoàn thành các bài thực hành khác trong khóa học này.

### Những gì bạn sẽ học

- Cách tạo một ứng dụng có hành vi tương tác.
- 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 chú giải các từ vựng và khái niệm để có các định nghĩa thân thiện.

### Những gì bạn sẽ làm

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

### Tổng quan về ứng dụng

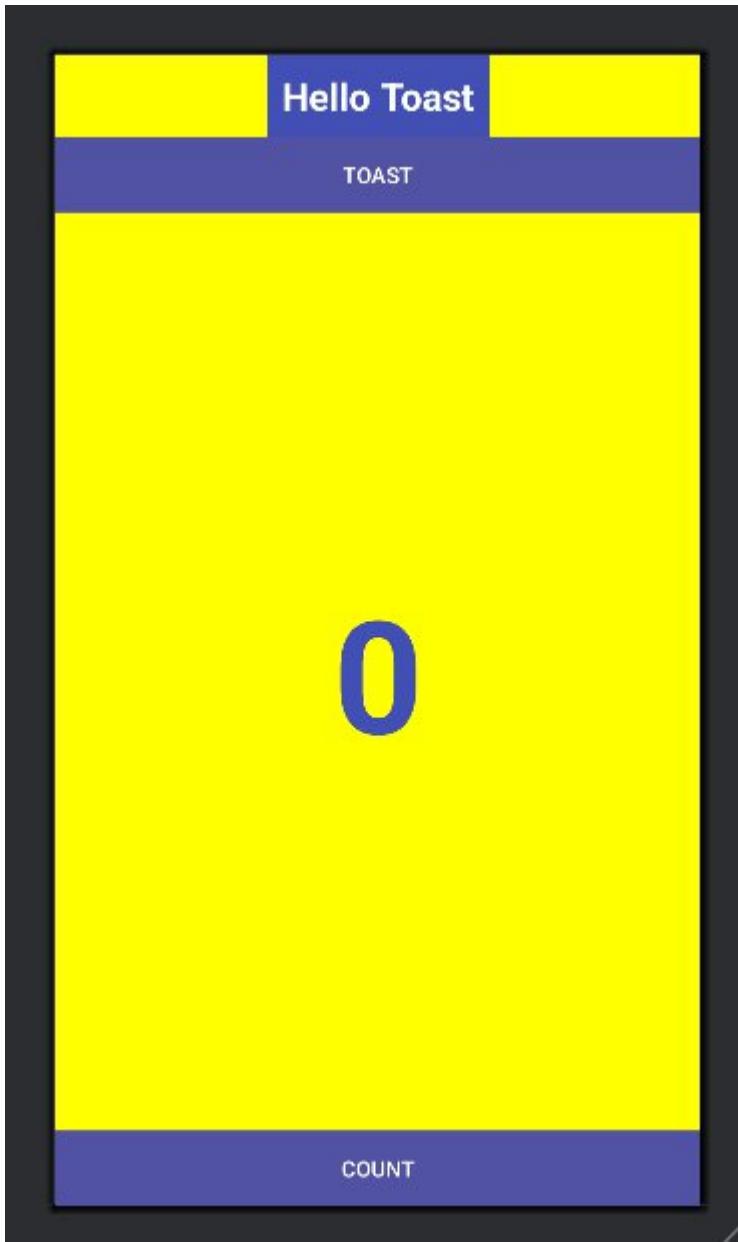
Ứng dụng HelloToast bao gồm hai phần tử Button và một TextView. Khi người dùng chạm vào Button đầu tiên, nó sẽ hiển thị một thông báo ngắn (một Toast) trên màn hình. Chạm vào Button thứ hai sẽ tăng bộ đếm "click" được hiển thị trong TextView, bắt đầu từ số không.

Đây là hình ảnh ứng dụng đã hoàn thành trông như thế nào:

Những gì bạn đã nên biết

Bạn nên quen thuộc với:

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



## 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 nguồn giải pháp được cung cấp ở phần cuối.

### 1.1 Tạo dự án Android Studio

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

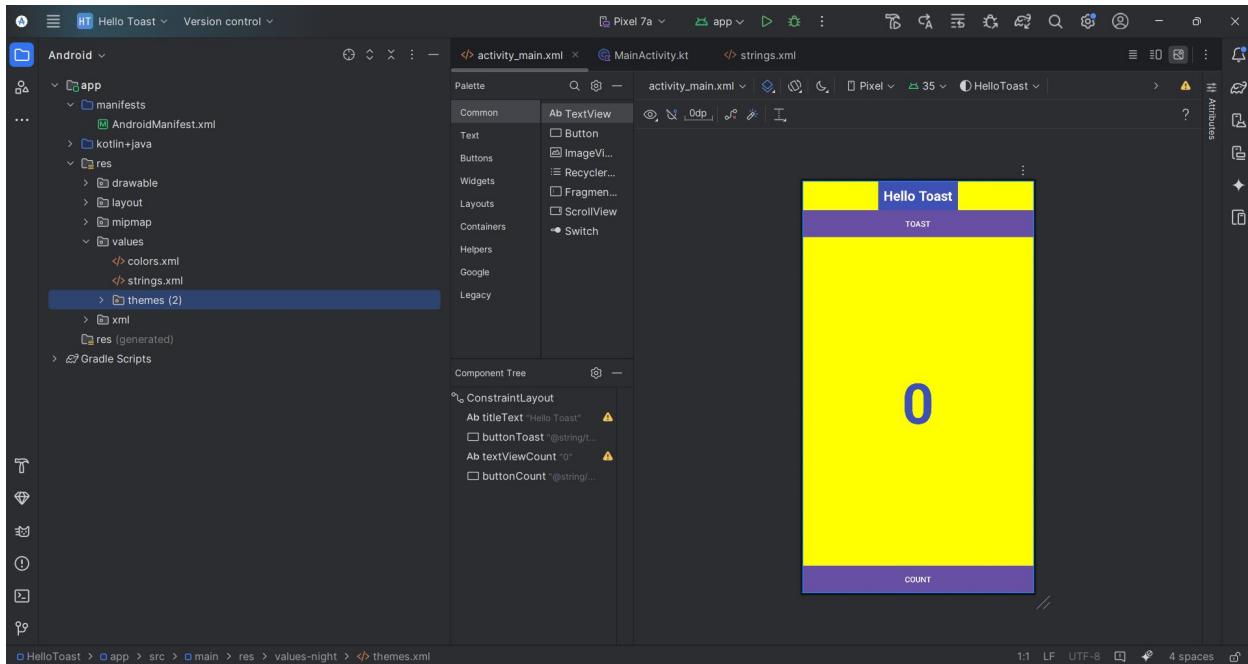
<b>Thuộc tính</b>	<b>Giá trị</b>
<b>Tên ứng dụng</b>	<b>Xin chào Toast</b>
<b>Tên nhà phát triển</b>	<b>com.example.android (hoặc tên miền của riêng bạn)</b>
<b>SDK tối thiểu cho điện thoại và máy tính bảng</b>	<b>API15: Android 4.0.3 IceCreamSandwich</b>
<b>Mẫu</b>	<b>Hoạt động trống (Hoạt động trống)</b>
<b>Hộp tạo tệp bối cục:</b>	<b>Được chọn</b>
<b>Hộp tương thích ngược:</b>	<b>Được chọn</b>

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

### **1.2 Khám phá trình chỉnh sửa bối cục**

Android Studio cung cấp trình chỉnh sửa bối cục để xây dựng nhanh chóng bối cục giao diện người dùng (UI) của ứng dụng. Nó cho phép bạn kéo các phần tử vào giao diện thiết kế trực quan và chế độ xem bản thiết kế, định vị chúng trong bối cục, thêm các ràng buộc và đặt thuộc tính. Các ràng buộc xác định vị trí của một phần tử UI trong bối cục. Một ràng buộc thể hiện sự kết nối hoặc căn chỉnh với một chế độ xem khác, bối cục cha, hoặc một đường hướng dẫn vô hình.

**Khám phá trình chỉnh sửa bối cục và tham khảo hình dưới đây khi bạn thực hiện theo các bước được đánh số:**



1.Trong thư mục app > res > layout trong khung Project > Android, nhấp đúp vào tệp activity\_main.xml để mở nếu tệp chưa được mở.

2.Nhấp vào tab Design nếu nó chưa được chọn. Bạn sử dụng tab Design để thao tác với các phần tử và bố cục, và tab Text để chỉnh sửa mã XML cho bố cục.

3.Khung Palettes hiển thị các phần tử giao diện người dùng (UI) mà bạn có thể sử dụng trong bố cục của ứng dụng.

4.Khung Component Tree hiển thị thứ tự phân cấp của các phần tử giao diện người dùng. Các phần tử giao diện được sắp xếp theo dạng cây phân cấp gồm cha và con, trong đó phần tử con kế thừa các thuộc tính của phần tử cha. Trong hình trên, TextView là con của ConstraintLayout. Bạn sẽ tìm hiểu thêm về các phần tử này sau trong bài học này.

5. Các khung thiết kế và bản thiết kế của trình chỉnh sửa cục hiển thị các phần tử UI trong bố cục. Trong hình trên, bố cục chỉ hiển thị một phần tử: một TextView hiển thị dòng chữ "Hello World".

**6. Tab Attributes hiển thị khung thuộc tính để thiết lập các thuộc tính cho một phần tử UI.**

**Mẹo:** Xem tài liệu Building a UI with Layout Editor để biết chi tiết về cách sử dụng trình chỉnh sửa bố cục, và Meet Android Studio để xem toàn bộ tài liệu về Android Studio.

### **Nhiệm vụ 2: Thêm các phần tử View trong trình chỉnh sửa bố cục**

Trong nhiệm vụ này, bạn tạo bố cục giao diện người dùng cho ứng dụng HelloToast trong trình chỉnh sửa bố cục bằng cách sử dụng các tính năng của ConstraintLayout. Bạn có thể tạo các ràng buộc thủ công như được trình bày sau này, hoặc tự động bằng công cụ Autoconnect.

#### **2.1 Kiểm tra các ràng buộc của phần tử**

**Thực hiện theo các bước sau:**

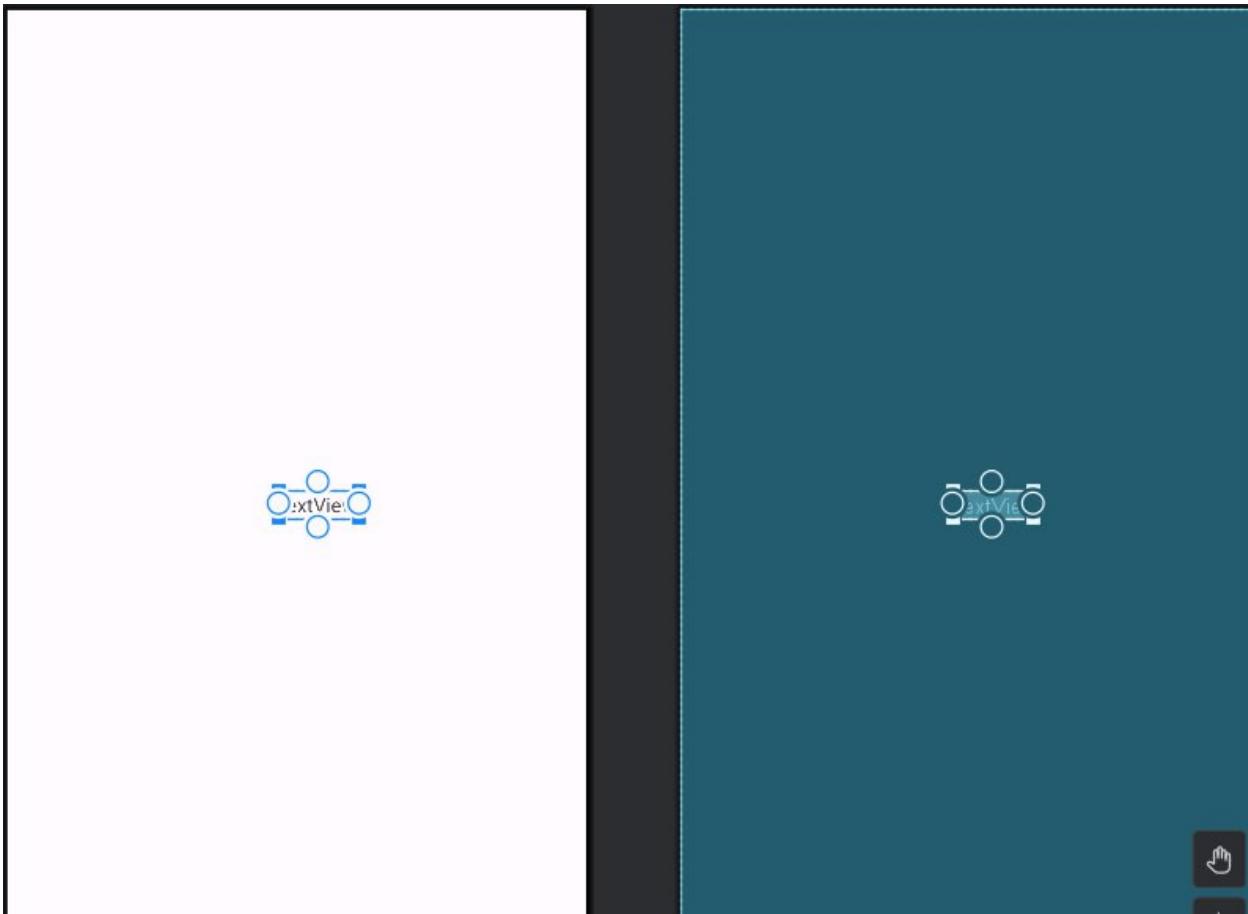
**1. Mở activity\_main.xml từ khung Project > Android nếu tệp chưa được mở. Nếu tab Design chưa được chọn, hãy nhấp vào nó. Nếu không có bản thiết kế, nhấp vào nút Select Design Surface trên thanh công cụ và chọn Design + Blueprint.**

**2. Công cụ Autoconnect cũng nằm trên thanh công cụ và được bật theo mặc định. Ở bước này, đảm bảo rằng công cụ này không bị tắt.**

**3. Nhấp vào nút phóng to để phóng to khung thiết kế và bản thiết kế để xem cận cảnh.**

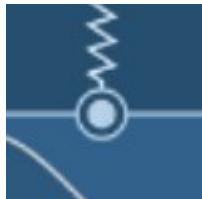
4. Chọn TextView trong khung Component Tree. TextView "Hello World" sẽ được đánh dấu trong khung thiết kế và bản thiết kế, và các ràng buộc của phần tử này sẽ hiển thị.

5. Tham khảo hình động dưới đây cho bước này. Nhấp vào tay cầm hình tròn ở phía bên phải của TextView để xóa ràng buộc ngang liên kết phần tử với phía bên phải của bố cục. TextView sẽ nhảy sang phía bên trái vì nó không còn bị ràng buộc với phía bên phải nữa. Để thêm lại ràng buộc ngang, nhấp vào cùng tay cầm đó và kéo một đường sang phía bên phải của bố cục.



Trong khung bản thiết kế hoặc khung thiết kế, các tay cầm sau xuất hiện trên phần tử TextView:  
● Tay cầm ràng buộc: Để tạo một ràng buộc như được hiển thị trong

hình động phía trên, nhấp vào tay cầm ràng buộc, được biểu thị bằng một hình tròn ở cạnh của phần tử. Sau đó, kéo tay cầm này đến một tay cầm ràng buộc khác hoặc đến đường biên của bố cục cha. Một đường zigzag thể hiện ràng buộc.



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



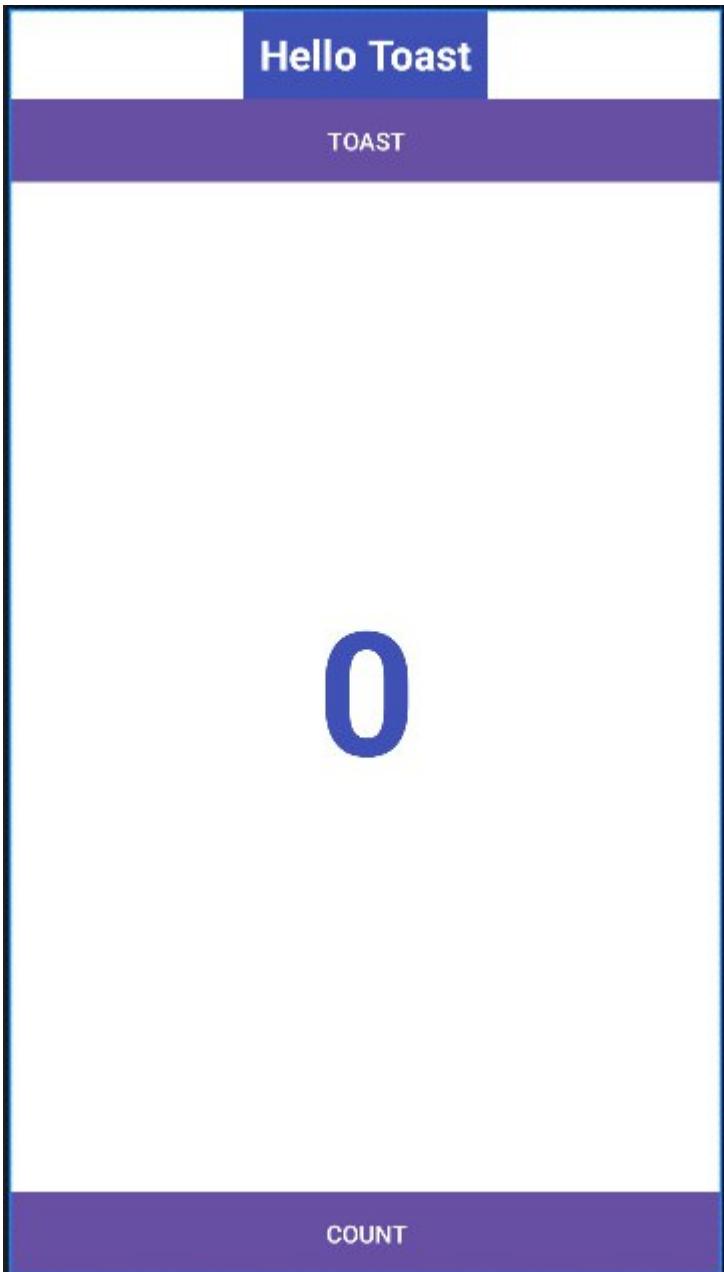
## 2.2 Thêm một Button vào bố cục

Khi được bật, công cụ Autoconnect sẽ tự động tạo hai hoặc nhiều ràng buộc cho một phần tử giao diện người dùng (UI) với bố cục cha. Sau khi bạn kéo phần tử vào bố cục, nó sẽ tạo các ràng buộc dựa trên vị trí của phần tử đó.

Thực hiện theo các bước sau để thêm một Button:

1. **Bắt đầu với một bố cục trống.** Phần tử TextView không cần thiết, vì vậy trong khi nó vẫn đang được chọn, nhấn phím Delete hoặc chọn Edit > Delete. Jetzt haben Sie einen leeren Layout-Bereich.

2. **Kéo một Button từ khung Palette đến bất kỳ vị trí nào trong bố cục.** Nếu bạn thả Button vào khu vực giữa phía trên của bố cục, các ràng buộc có thể tự động xuất hiện. Nếu không, bạn có thể kéo các ràng buộc đến phía trên, bên trái và bên phải của bố cục như được hiển thị trong hình động dưới đây.



### 2.3 Thêm một Button thứ hai vào bối cảnh

1. Kéo một Button khác từ khung Palette vào giữa bối cảnh như được hiển thị trong hình động dưới đây. Autoconnect có thể tự động cung cấp các ràng buộc ngang cho bạn (nếu không, bạn có thể tự kéo chúng).
2. Kéo một ràng buộc dọc đến phía dưới của bối cảnh (tham khảo hình dưới đây).



Bạn có thể xóa các ràng buộc khỏi một phần tử bằng cách chọn phần tử đó và di chuột qua nó để hiển thị nút Clear Constraints. Nhấp vào nút này để xóa tất cả các ràng buộc trên phần tử đã chọn. Để xóa một ràng buộc riêng lẻ, nhấp vào tay cầm cụ thể đã đặt ràng buộc đó. Để xóa tất cả các ràng buộc trong toàn bộ bố cục, 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 thuộc tính của phần tử UI

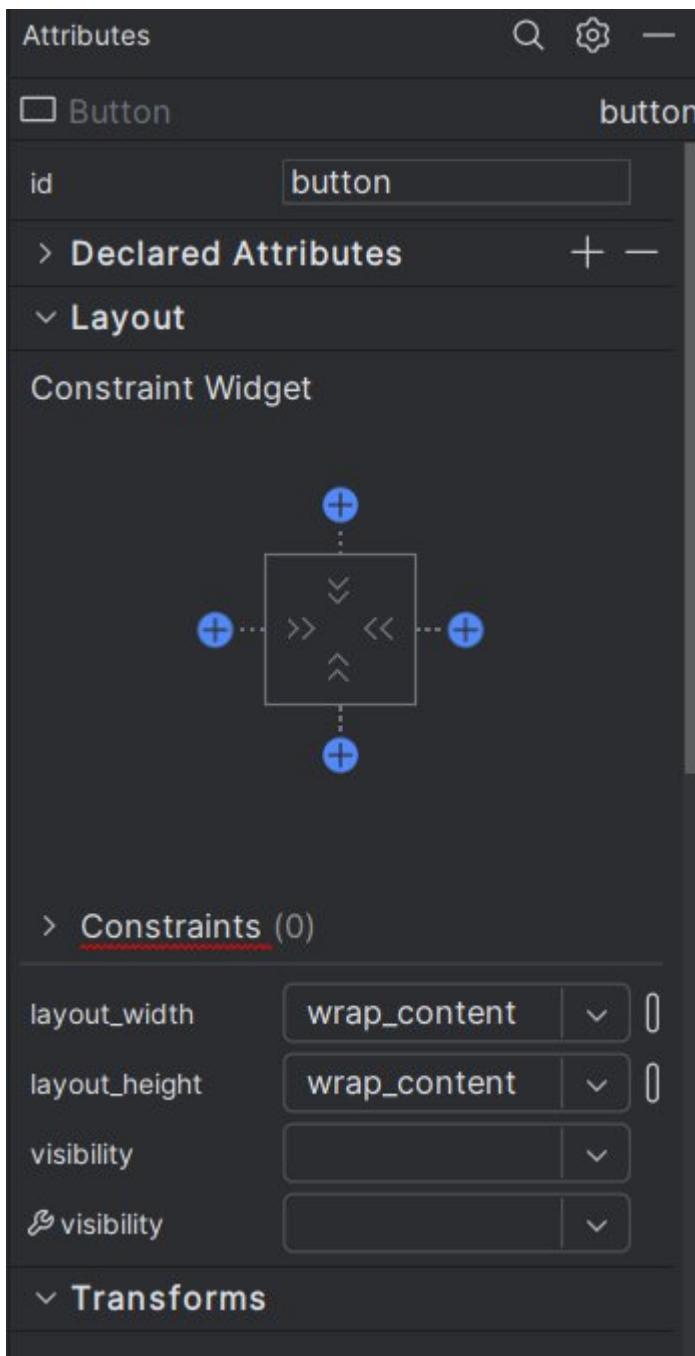
Khung Attributes cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một phần tử giao diện người dùng (UI). Bạn có thể tìm thấy các thuộc tính (được gọi là thuộc tính) chung cho tất cả các chế độ xem trong tài liệu lớp View. Trong nhiệm vụ này, bạn sẽ nhập các giá trị mới và thay đổi các giá trị cho các thuộc tính quan trọng của Button, những thuộc tính này áp dụng cho hầu hết các loại View.

#### 3.1 Thay đổi kích thước của Button

Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước ở cả bốn góc của một View để bạn có thể nhanh chóng thay đổi kích thước của View. Bạn có thể kéo các tay cầm ở mỗi góc của View để thay đổi kích thước, nhưng việc này sẽ cố định chiều rộng và chiều cao (hardcode). Tránh cố định kích thước cho hầu hết

các phần tử View, vì các kích thước cố định không thể thích nghi với nội dung và kích thước màn hình khác nhau.

Thay vào đó, hãy sử dụng khung Attributes ở phía bên phải của trình chỉnh sửa bối cảnh để chọn chế độ kích thước không sử dụng kích thước cố định. Khung Attributes bao gồm một bảng điều khiển kích thước hình vuông được gọi là view inspector ở phía trên cùng. Các biểu tượng bên trong hình vuông thể hiện cài đặt chiều cao và chiều rộng như sau:



Trong hình trên:

- Điều khiển chiều cao. Điều khiển này xác định thuộc tính `layout_height` và xuất hiện ở hai đoạn trên cạnh trên và dưới của hình vuông. Các góc nghiêng cho biết điều khiển này được đặt thành `wrap_content`, có nghĩa là View sẽ mở rộng theo

chiều dọc nếu cần để vừa với nội dung của nó. Số "8" biểu thị lề tiêu chuẩn được đặt thành 8dp.

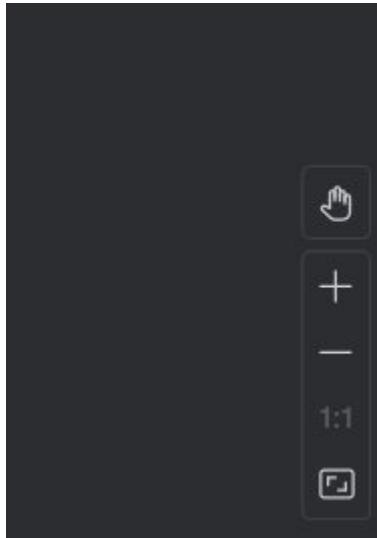
2. Điều khiển chiều rộng. Điều khiển này xác định thuộc tính layout\_width và xuất hiện ở hai đoạn trên cạnh trái và phải của hình vuông. Các góc nghiêng cho biết điều khiển này được đặt thành wrap\_content, có nghĩa là View sẽ mở rộng theo chiều ngang nếu cần để vừa với nội dung của nó, tối đa đến lề 8dp.

3. Nút đóng khung Attributes. Nhấp để đóng khung này.

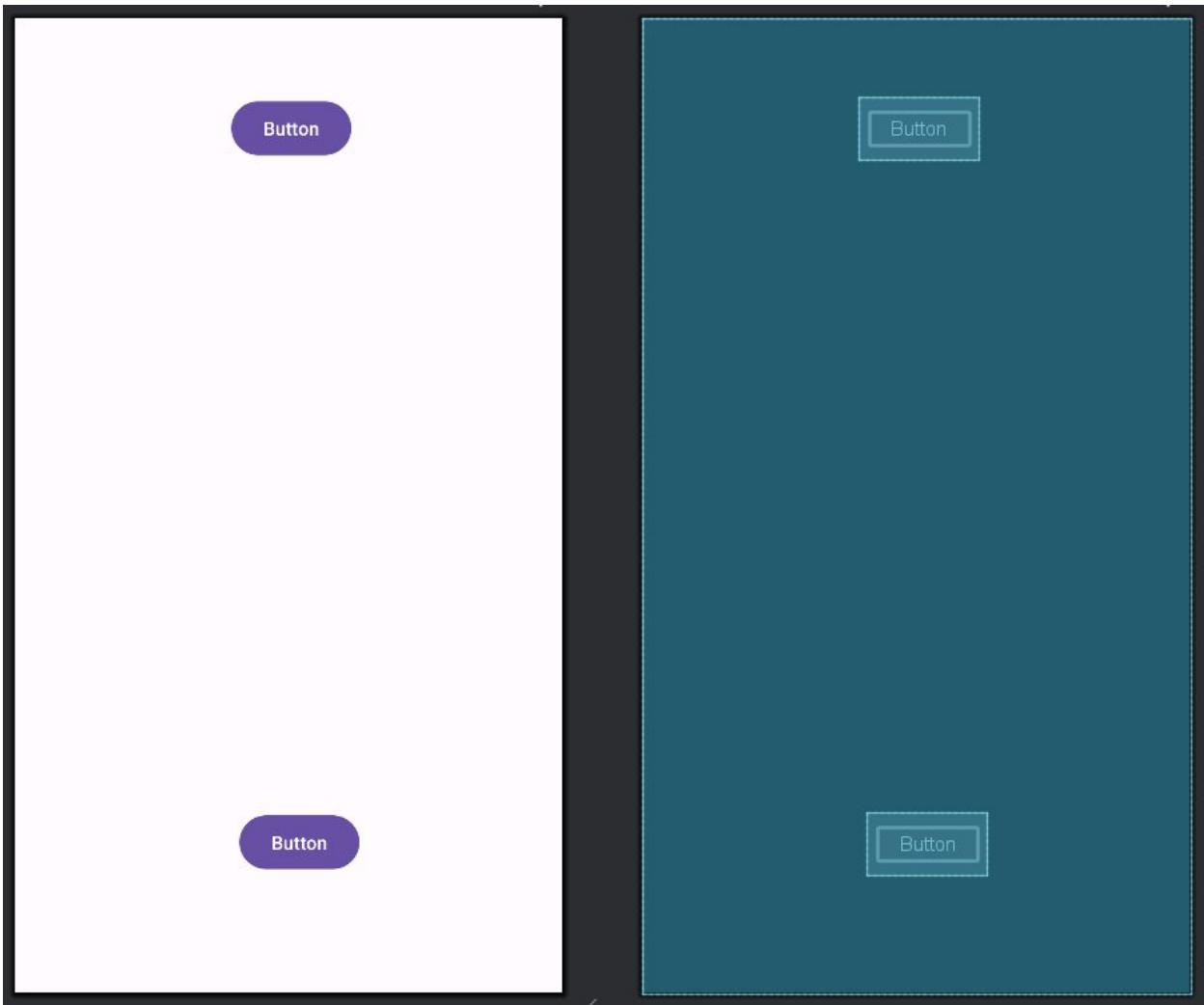
Thực hiện theo các bước sau:

1. Chọn Button trên cùng trong khung Component Tree.

2. Nhấp vào tab Attributes ở phía bên phải của cửa sổ trình chỉnh sửa bố cục.

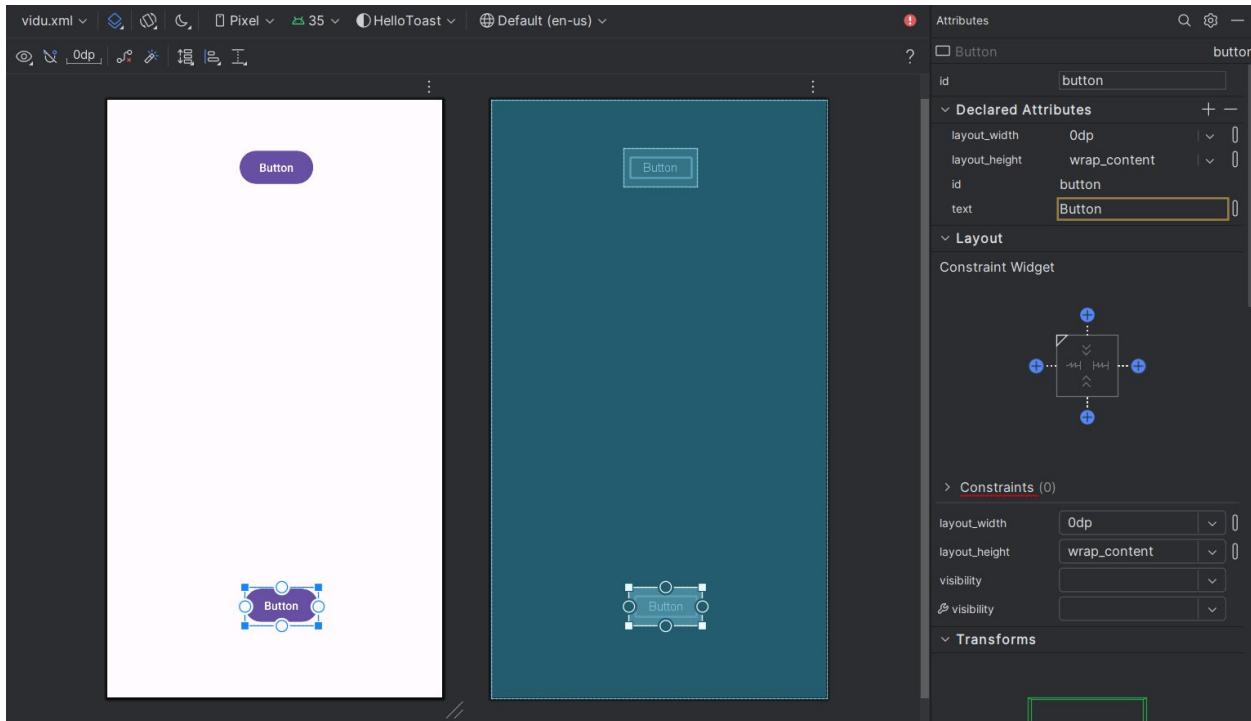


3. Nhấp vào điều khiển chiều rộng hai lần—lần nhấp đầu tiên sẽ thay đổi thành Fixed với các đường thẳng, và lần nhấp thứ hai sẽ thay đổi thành Match Constraints với các lò xo, như được hiển thị trong hình động dưới đây.



Kết quả của việc thay đổi điều khiển chiều rộng, thuộc tính `layout_width` trong khung Attributes hiển thị giá trị `match_constraint` và phần tử Button giãn ra theo chiều ngang để lấp đầy khoảng trống giữa các cạnh trái và 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 được hiển thị trong hình dưới đây.



Như đã hiển thị trong các bước trước, các thuộc tính `layout_width` và `layout_height` trong khung `Attributes` sẽ thay đổi khi bạn điều chỉnh các điều khiển chiều cao và chiều rộng trong `inspector`. Các thuộc tính này có thể nhận một trong ba giá trị cho bố cục, trong trường hợp này là `ConstraintLayout`:

- Thiết lập `match_constraint` sẽ mở rộng phần tử View theo chiều rộng hoặc chiều cao để lấp đầy bố cục cha—tối đa đến lề, nếu có lề được đặt. Trong trường hợp này, bố cục cha là `ConstraintLayout`. Bạn sẽ tìm hiểu thêm về `ConstraintLayout` trong nhiệm vụ tiếp theo.
- Thiết lập `wrap_content` sẽ thu nhỏ kích thước của phần tử View để vừa đủ để bao quanh nội dung của nó. Nếu không có nội dung, phần tử View sẽ trở nên vô hình.
- Để chỉ định một kích thước cố định phù hợp với kích thước màn hình của thiết

bị, hãy sử dụng một số lượng pixel độc lập với mật độ (dp). Ví dụ, 16dp có nghĩa là 16 pixel độc lập với mật độ.

Mẹo: Nếu bạn thay đổi thuộc tính layout\_width bằng menu bật lên, thuộc tính layout\_width sẽ được đặt thành 0 vì không có kích thước nào được đặt. Thiết lập này tương đương với match\_constraint—phần tử có thể mở rộng tối đa để đáp ứng các ràng buộc và cài đặt lề.

### 3.2 Thay đổi thuộc tính của Button

Để nhận diện từng View một cách duy nhất trong bố cục của Activity, mỗi View hoặc lớp con của View (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 sắc hoặc hình ảnh.

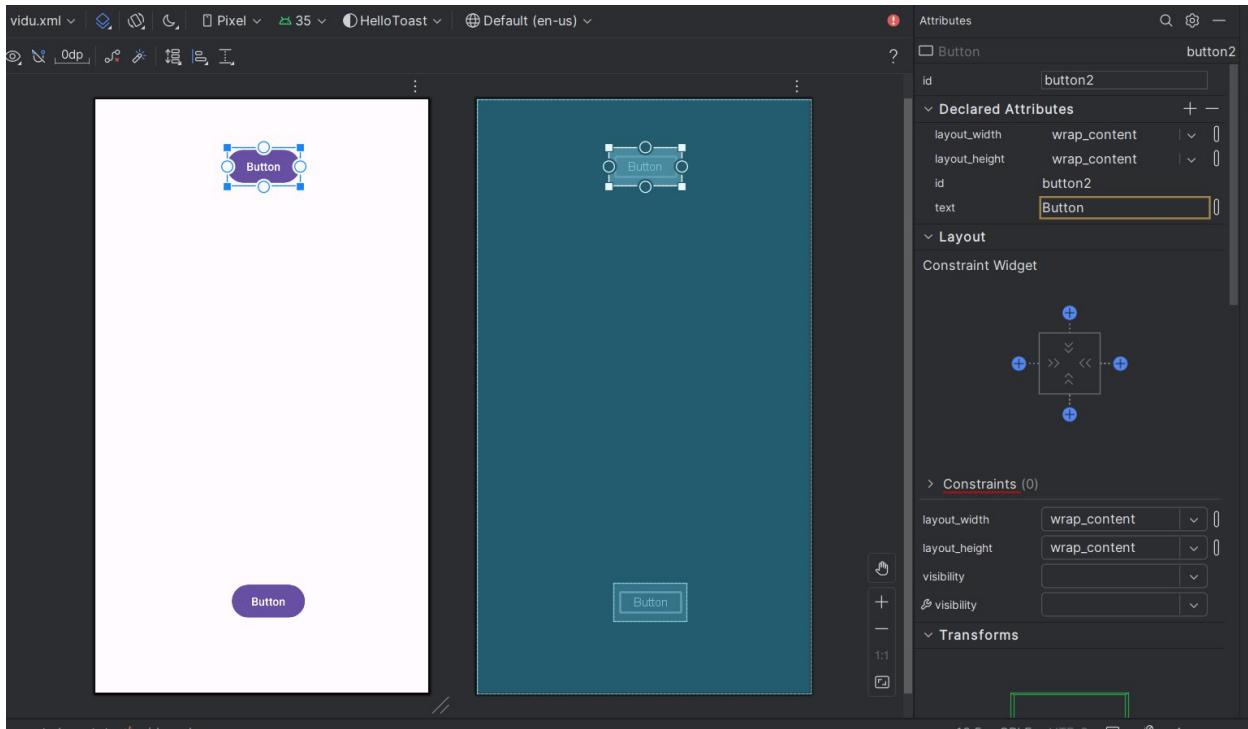
Khung Attributes cung cấp quyền truy cập vào tất cả các thuộc tính mà bạn có thể gán cho một phần tử View. Bạn có thể nhập giá trị cho từng thuộc tính, chẳng hạn như android:id, background, textColor, và text.

Hình động sau đây minh họa cách thực hiện các bước này: 1. Sau khi chọn Button đầu tiên, chỉnh sửa trường ID ở đầu khung Attributes thành button\_toast cho thuộc tính android:id, được sử dụng để nhận diện 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 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 bản được định nghĩa sẵn trong tệp tài nguyên colors.xml. Nó được sử dụng cho thanh ứng dụng (app bar). Sử dụng các màu cơ bản cho các phần tử giao diện người dùng khác tạo ra một giao diện đồng nhất. Bạn sẽ tìm hiểu thêm về chủ đề ứng dụng và Material Design trong một bài học khác.

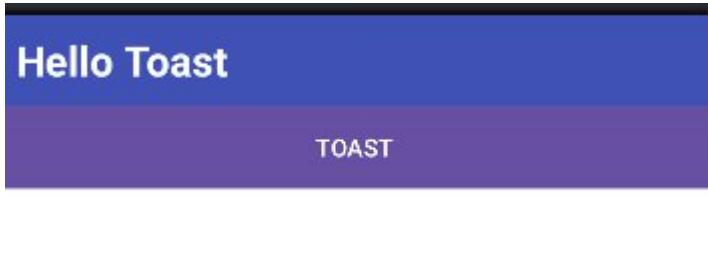
#### Nhiệm vụ 4: Thêm một TextView và đặt thuộc tính của nó

Một trong những lợi ích của ConstraintLayout là khả năng căn chỉnh hoặc ràng buộc các phần tử theo mối quan hệ với các phần tử khác. Trong nhiệm vụ này, bạn sẽ thêm một TextView vào giữa bố cục và ràng buộc nó theo chiều ngang với

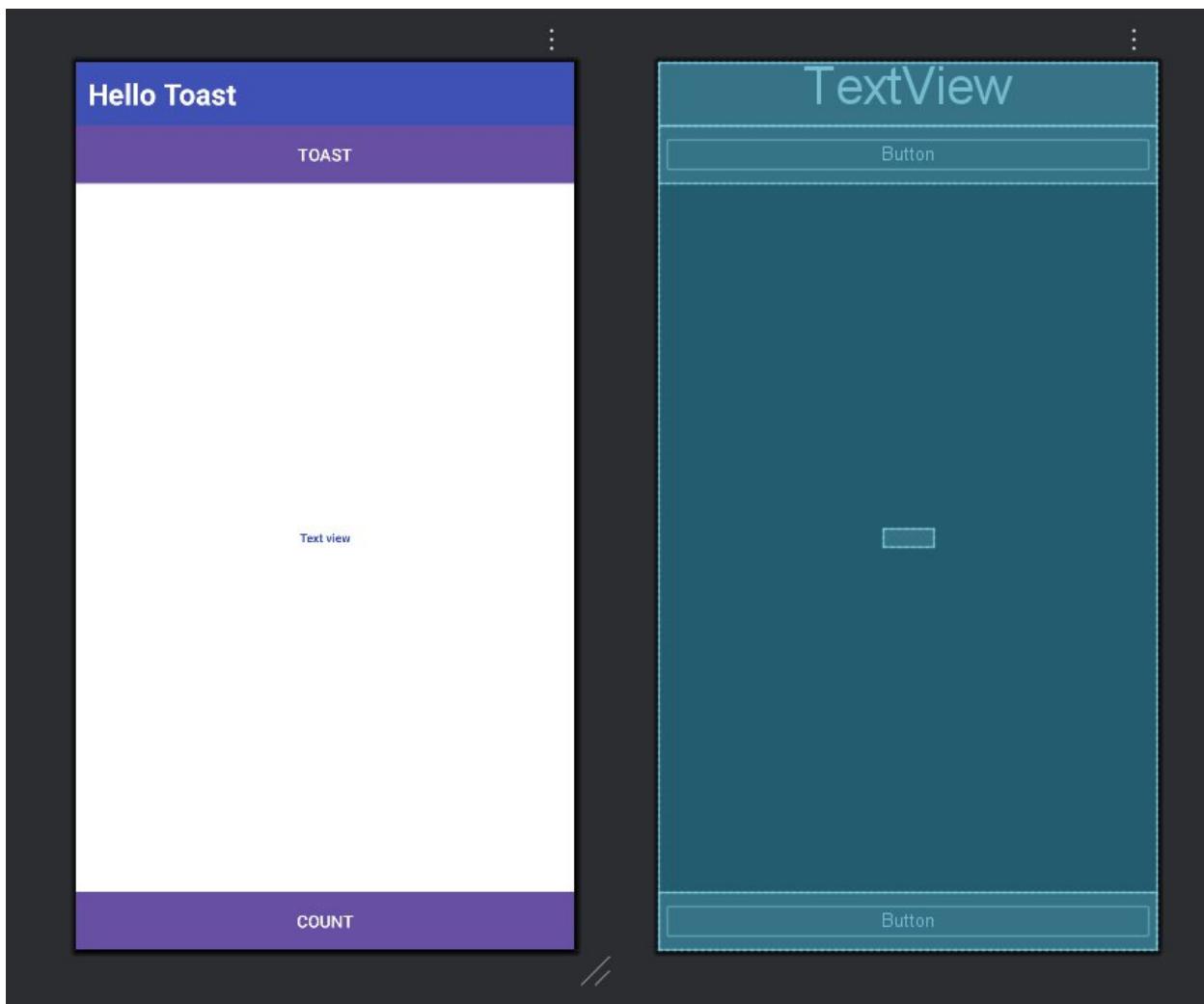
các lề và theo chiều dọc với hai phần tử Button. Sau đó, bạn sẽ thay đổi các thuộc tính của TextView trong khung Attributes.

#### 4.1 Thêm một TextView và các ràng buộc

1. Như được hiển thị trong hình động dưới đây, kéo một TextView từ khung Palette đến phần trên của bố cục, và kéo một ràng buộc từ cạnh trên của TextView đến tay cầm ở phía dưới của Button Toast. Điều này ràng buộc TextView để nằm bên dưới Button.



2. Như được hiển thị trong hình động dưới đây, kéo một ràng buộc từ cạnh dưới của TextView đến tay cầm ở phía trên của Button Count, và từ các cạnh bên của TextView đến các cạnh của bố cục. Điều này ràng buộc



4.2 Thiết lập thuộc tính cho TextView  
Với TextView đã được chọn, mở khung Attributes nếu nó chưa được mở. Thiết lập các thuộc tính cho TextView như được hiển thị trong hình động dưới đây. Các thuộc tính mà bạn chưa gặp sẽ được giải thích sau hình:

1. Đặt ID thành `show_count`.

2. Đặt `text` thành `0`.

3. Đặt `textSize` thành `160sp`.

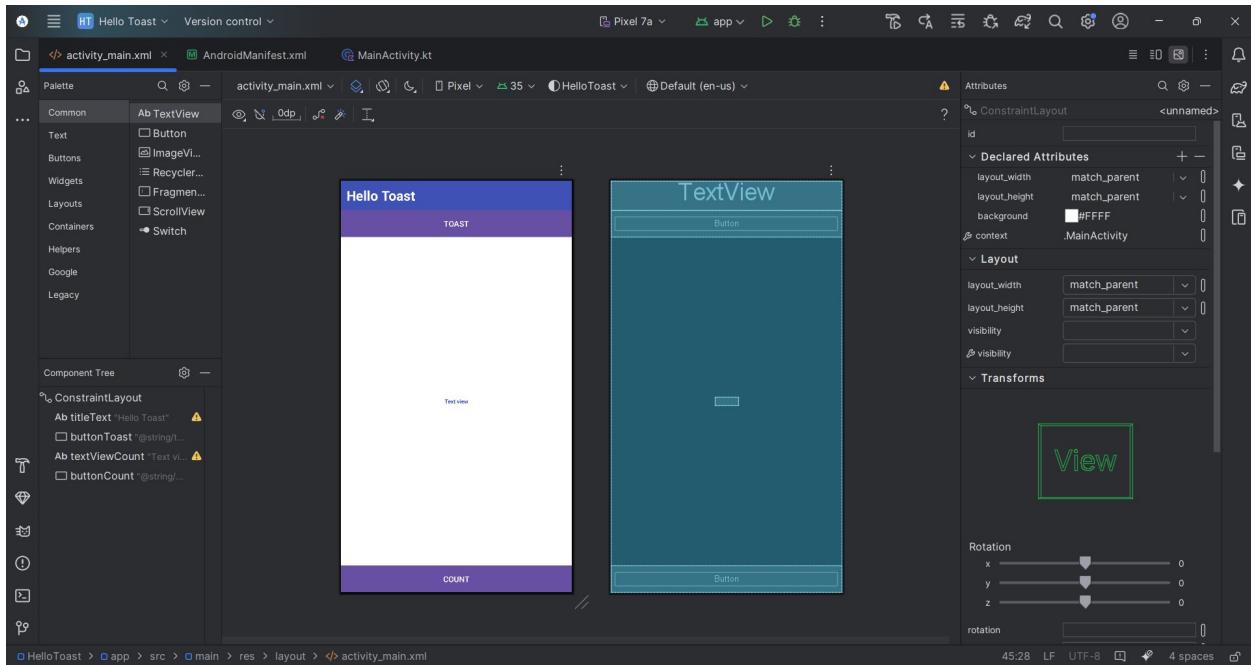
**4. Đặt textStyle thành B (in đậm) và textAlign thành ALIGNCENTER (căn giữa đoạn văn).**

**5. Thay đổi các điều khiển kích thước ngang và dọc của chế độ xem (layout\_width và layout\_height) thành match\_constraint.**

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

**7. Cuộn xuống khung và nhấp vào View all attributes, cuộn xuống trang thứ hai của các thuộc tính đến background, sau đó nhập #FFF00 để chọn một sắc thái màu vàng.**

**8. Cuộn xuống đến gravity, mở rộng gravity, và chọn center\_ver (để căn giữa theo chiều dọc).**



**● textSize: Kích thước văn bản của TextView. Trong bài học này, kích thước được đặt thành 160sp. sp là viết tắt của scale-independent pixel (pixel độc lập với tỷ**

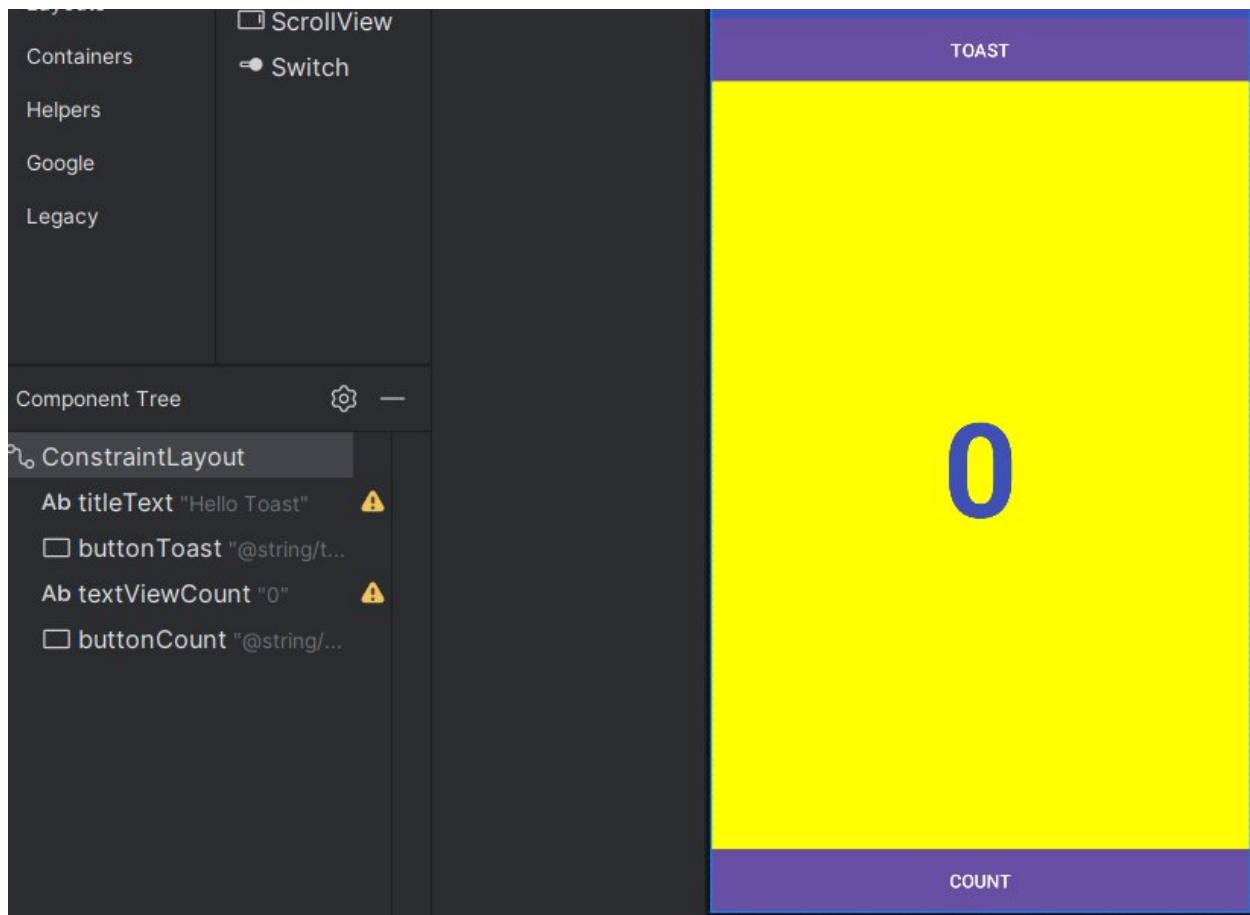
lệ), và tương tự như dp, đây là đơn vị thay đổi theo mật độ màn hình và tùy chọn kích thước phông chữ của người dùng. Sử dụng đơn vị dp khi bạn chỉ định kích thước phông chữ để kích thước được điều chỉnh phù hợp với cả mật độ màn hình và tùy chọn của người dùng.

- **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 gravity xác định cách một View được căn chỉnh trong View cha hoặc ViewGroup của nó. Trong bước này, bạn căn giữa TextView theo chiều dọc trong ConstraintLayout cha.

Bạn có thể nhận thấy rằng thuộc tính background nằm ở trang đầu tiên của khung Attributes đối với một Button, nhưng ở trang thứ hai của khung Attributes đối với một TextView. Khung Attributes thay đổi tùy theo từng loại View: Các thuộc tính phổ biến nhất cho loại View đó sẽ xuất hiện ở trang đầu tiên, và các thuộc tính còn lại được liệt kê ở trang thứ hai. Để quay lại trang đầu tiên của khung Attributes, nhấp vào biểu tượng trên thanh công cụ ở đầu khung.

#### Nhiệm vụ 5: Chính sửa bố cục trong XML

Bố cục của ứng dụng Hello Toast gần như đã hoàn thành! Tuy nhiên, một dấu chấm than xuất hiện bên cạnh mỗi phần tử giao diện người dùng trong Component Tree. Di chuột qua các dấu chấm than này để xem thông báo cảnh báo, như được hiển thị bên dưới. Cảnh báo giống nhau xuất hiện cho cả ba phần tử: các chuỗi cố định (hardcoded strings) nên sử dụng tài nguyên.



Cách dễ nhất để khắc phục các vấn đề bối cục là chỉnh sửa bối cục trong XML. Trong khi trình chỉnh sửa bối cục là một công cụ mạnh mẽ, một số thay đổi sẽ dễ thực hiện hơn khi chỉnh sửa trực tiếp trong mã nguồn XML.

## 5.1 Mở mã XML cho bối cục

Trong nhiệm vụ này, mở tệp `activity_main.xml` nếu nó chưa được mở, và nhấp vào tab `Text` ở phía dưới của trình chỉnh sửa bối cục.

**5.2 Trích xuất tài nguyên chuỗi** Thay vì mã hóa cứng (hard-coding) các chuỗi, đây là một cách làm tốt nhất để sử dụng tài nguyên chuỗi, đại diện cho các chuỗi đó. Việc đặt các chuỗi trong một tệp riêng biệt giúp dễ quản lý hơn, đặc biệt nếu bạn sử dụng các chuỗi này nhiều lần. Ngoài ra, tài nguyên chuỗi là bắt buộc để dịch và bản địa hóa ứng dụng của bạn, vì bạn cần tạo một tệp tài nguyên chuỗi cho từng ngôn ngữ.

```
<!-- Nút Toast -->
<Button
    android:id="@+id/buttonToast"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:text="@string/toast_button"
    android:background="#3F51B5"
    android:textColor="#FFFFFF"
    app:layout_constraintTop_toBottomOf="@+id/titleText"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"/>

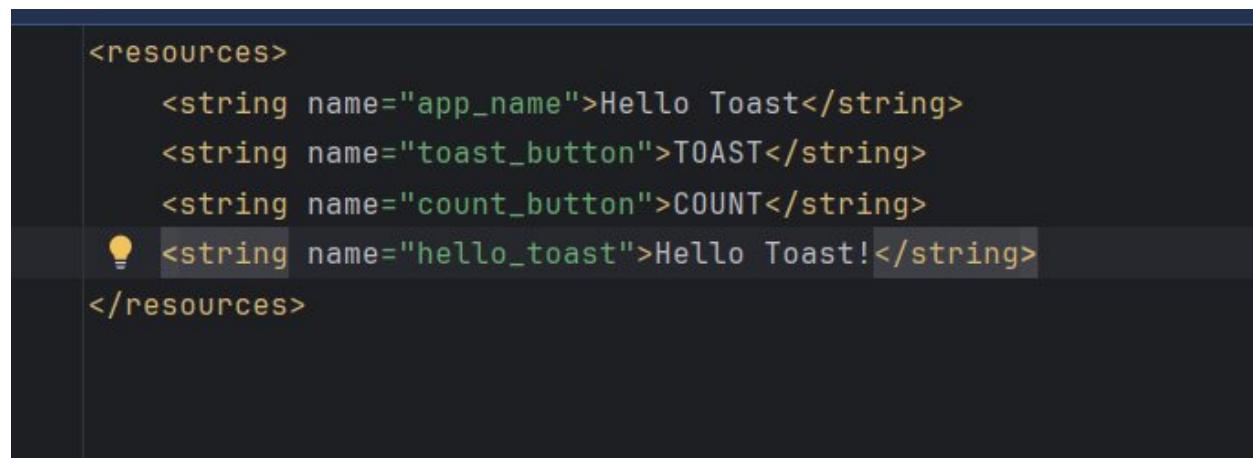
<!-- Hiển thị số đếm -->
<TextView
    android:id="@+id/textViewCount"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="0"
    android:textSize="100sp"
    android:textStyle="bold"
    android:textColor="#3F51B5"
```

- 1.Nhấp một lần vào từ "Toast" (cảnh báo được làm nổi bật đầu tiên).
2. Nhấn Alt-Enter trên Windows hoặc Option-Enter trên macOS và chọn Extract string resource từ menu bật lên.
3. Nhập button\_label\_toast cho tên Tài nguyên.
4. Nhấp OK. Một tài nguyên chuỗi sẽ được tạo trong tệp values/res/string.xml,

và chuỗi trong mã của bạn sẽ được thay thế bằng tham chiếu đến tài nguyên:@string/button\_label\_toast

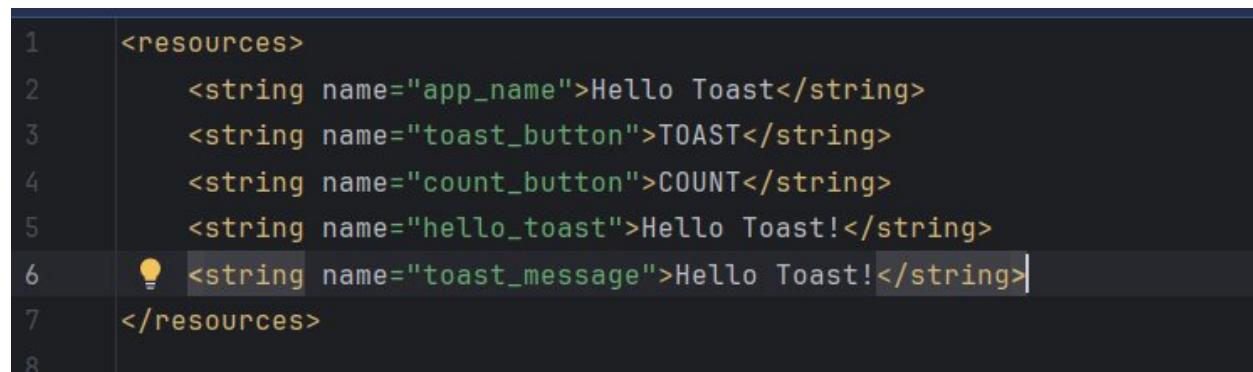
5. Trích xuất các chuỗi còn lại: button\_label\_count cho "Count", và count\_initial\_value cho "0".

6. Trong khung Project > Android, mở rộng mục values trong res, sau đó nhấp đúp vào strings.xml để xem các tài nguyên chuỗi của bạn trong tệp strings.xml.



```
<resources>
    <string name="app_name">Hello Toast</string>
    <string name="toast_button">TOAST</string>
    <string name="count_button">COUNT</string>
    💡 <string name="hello_toast">Hello Toast!</string>
</resources>
```

7. Bạn cần thêm một chuỗi khác để sử dụng trong nhiệm vụ tiếp theo, chuỗi này sẽ hiển thị một thông điệp. Thêm vào tệp strings.xml một tài nguyên chuỗi khác có tên toast\_message cho cụm từ "Hello Toast!":



```
1 <resources>
2     <string name="app_name">Hello Toast</string>
3     <string name="toast_button">TOAST</string>
4     <string name="count_button">COUNT</string>
5     <string name="hello_toast">Hello Toast!</string>
6     💡 <string name="toast_message">Hello Toast!</string>
7 </resources>
8
```

Mẹo: Các tài nguyên chuỗi bao gồm tên ứng dụng, xuất hiện trên thanh ứng dụng ở đầu màn hình nếu bạn bắt đầu dự án ứng dụng của mình bằng cách sử dụng

**Mẫu Trống.** Bạn có thể thay đổi tên ứng dụng bằng cách chỉnh sửa tài nguyên `app_name`.

### Nhiệm vụ 6: Thêm trình xử lý onClick cho các nút

Trong nhiệm vụ này, bạn thêm một phương thức Java cho mỗi Nút trong `MainActivity`, phương thức này sẽ thực thi khi người dùng nhấn vào Nút.

#### 6.1 Thêm thuộc tính onClick và trình xử lý cho mỗi Nút

Trình xử lý nhấp chuột là một phương thức được gọi khi người dùng nhấp hoặc nhấn vào một phần tử UI có thể nhấp. Trong Android Studio, bạn có thể chỉ định tên của phương thức trong trường `onClick` trong ngăn Thuộc tính của tab Thiết kế. Bạn cũng có thể chỉ định tên của phương thức trình xử lý trong trình soạn thảo XML bằng cách thêm thuộc tính `android:onClick` vào Nút. Bạn sẽ sử dụng phương pháp thứ hai vì bạn chưa tạo các phương thức trình xử lý và trình soạn thảo XML cung cấp một cách tự động để tạo các phương thức đó.

1. Với trình soạn thảo XML đang mở (tab Văn bản), hãy tìm Nút có `android:id` được đặt thành `button_toast`
2. Thêm thuộc tính `android:onClick` vào cuối phần tử `button_toast` sau thuộc tính cuối cùng và trước dấu kết thúc `>`:
3. Nhấp vào biểu tượng bóng đèn màu đỏ xuất hiện bên cạnh thuộc tính. Chọn **Create click handler** (**Tạo trình xử lý nhấp chuột**), chọn `MainActivity` và nhấp vào OK.

Nếu biểu tượng bóng đèn màu đỏ không xuất hiện, hãy nhấp vào tên phương thức ("`showToast`"). Nhấn Alt-Enter (Option-Enter trên máy Mac), chọn **Create 'showToast(view)' in MainActivity** (**Tạo 'showToast(view)' trong MainActivity**) và nhấp vào OK.

Hành động này tạo một phần giữ chỗ cho phương thức `showToast()` trong `MainActivity`, như được hiển thị ở cuối các bước này.

4. Lặp lại hai bước cuối cùng với Button `button_count`: Thêm thuộc tính `android:onClick` vào cuối và thêm trình xử lý nhấp chuột:

Giao diện code XML sau khi thêm các thành phần sẽ hiển thị như sau:

```
<!-- Nút Toast -->
<Button
    android:id="@+id/buttonToast"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:text="@string/toast_button"
    android:background="#3F51B5"
    android:textColor="#FFFFFF"
    android:onClick="showToast"
    app:layout_constraintTop_toBottomOf="@+id/titleText"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />

<!-- Hiển thị số đếm -->
<TextView
    android:id="@+id/textViewCount"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="0"
    android:textSize="100sp"
    android:textStyle="bold"
    android:textColor="#3F51B5"
    app:layout_constraintTop_toBottomOf="@+id/buttonToast"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintBottom_toTopOf="@+id/buttonCount" />

<!-- Nút Count -->
<Button
    android:id="@+id/buttonCount"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:text="@string/count_button"
    android:background="#3F51B5"
    android:textColor="#FFFFFF"
    android:onClick="countUp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

5. Nếu MainActivity.java chưa mở, hãy mở rộng java trong Project > Android, mở rộng com.example.android.hellotoast, sau đó nhấp đúp vào MainActivity. Trình soạn thảo mã sẽ xuất hiện với mã trong MainActivity.

```
package com.example.hellotoast

import android.os.Bundle
import android.widget.Button
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

    private var count = 0
    private lateinit var textViewCount: TextView

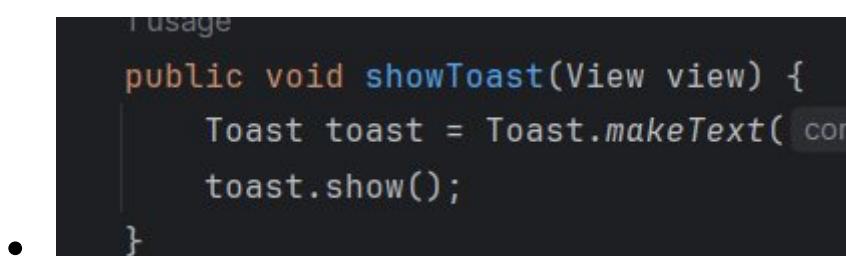
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
```

## 6.2 Chỉnh sửa trình xử lý Nút Toast

Bây giờ bạn sẽ chỉnh sửa phương thức showToast()—trình xử lý nhấp chuột của Nút Toast trong MainActivity—để nó hiển thị một thông báo. Toast cung cấp một cách để hiển thị một thông báo đơn giản trong một cửa sổ bật lên nhỏ. Nó chỉ chiếm lượng không gian cần thiết cho thông báo. Hoạt động hiện tại vẫn hiển thị và tương tác được. Toast có thể hữu ích để kiểm tra tính tương tác trong ứng dụng của bạn—thêm một thông báo Toast để hiển thị kết quả của việc nhấn vào Nút hoặc thực hiện một hành động.

Làm theo các bước sau để chỉnh sửa trình xử lý nhấp chuột của Nút Toast:

Xác định vị trí phương thức showToast() vừa được tạo.



Để tạo một thể hiện của Toast, hãy gọi phương thức factory `makeText()` trên lớp `Toast`. Câu lệnh này chưa hoàn chỉnh cho đến khi bạn hoàn thành tất cả các bước.

```
public void showToast(View view) {  
    Toast toast = Toast.makeText(context: this,
```

Cung cấp ngữ cảnh (`context`) của Activity ứng dụng. Vì `Toast` hiển thị trên đầu giao diện người dùng Activity, hệ thống cần thông tin về Activity hiện tại. Khi bạn đã ở trong ngữ cảnh của Activity mà bạn cần ngữ cảnh, hãy sử dụng `this` như một lối tắt.

```
    toast.show();
```

Cung cấp thông báo để hiển thị, chẳng hạn như một tài nguyên chuỗi (`toast_message` mà bạn đã tạo ở bước trước). Tài nguyên chuỗi `toast_message` được xác định bởi `R.string`.

```
    Toast toast = Toast.makeText(context: this, R.string.toast_message,
```

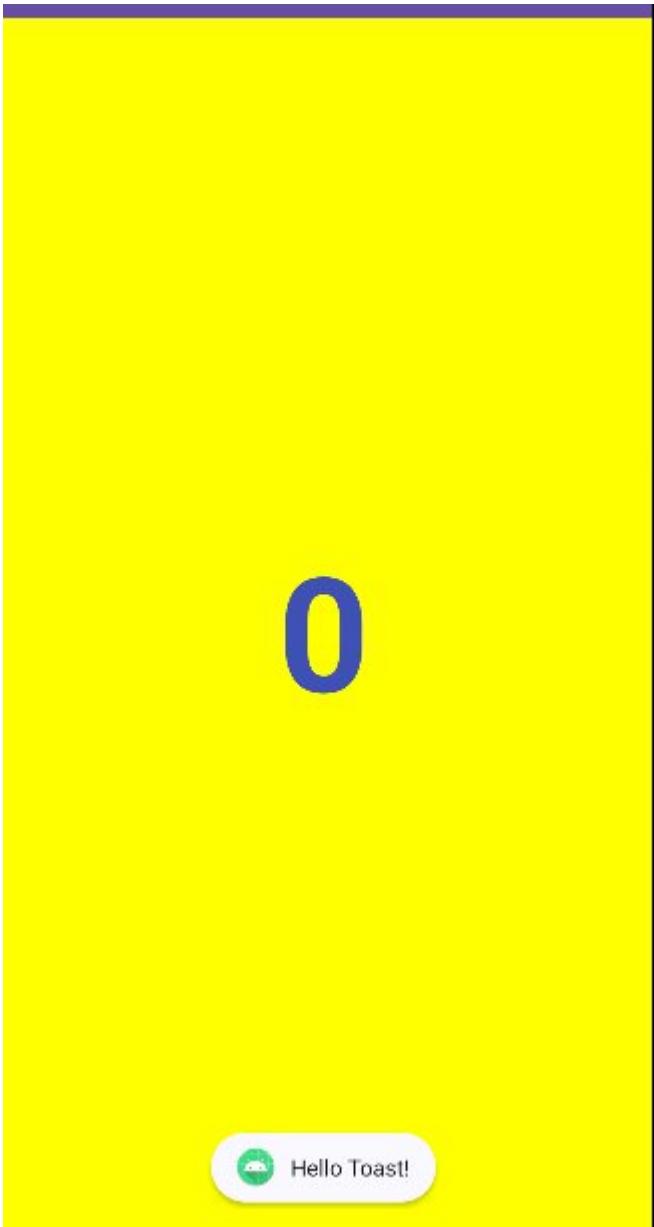
Cung cấp thời lượng hiển thị. Ví dụ: `Toast.LENGTH_SHORT` hiển thị `toast` trong một thời gian tương đối ngắn. Thời lượng hiển thị của `Toast` có thể là `Toast.LENGTH_LONG` hoặc `Toast.LENGTH_SHORT`. Độ dài thực tế là khoảng 3,5 giây cho `Toast` dài và 2 giây cho `Toast` ngắn.

```
public void showToast(View view) {  
    Toast toast = Toast.makeText(context: this, R.string.toast_message, Toast.LENGTH_SHORT);  
    toast.show();  
}
```

Hiển thị `Toast` bằng cách gọi `show()`. Sau đây là toàn bộ phương thức `showToast()`:

```
1 usage  
public void showToast(View view) {  
    Toast toast = Toast.makeText(context: this, R.string.toast_message, Toast.LENGTH_SHORT);  
    toast.show();  
}
```

**Chạy ứng dụng và xác minh rằng thông báo Toast xuất hiện khi nhấn vào nút Toast.**



### **6.3 Chỉnh sửa trình xử lý Nút Đếm**

Bây giờ bạn sẽ chỉnh sửa phương thức `countUp()`—trình xử lý nhấp chuột của Nút Đếm trong `MainActivity`—để nó hiển thị số đếm hiện tại sau khi Đếm được nhấn. Mỗi lần nhấn sẽ tăng số đếm thêm một.

Mã cho trình xử lý phải:

- Theo dõi số đếm khi nó thay đổi.
- Gửi số đếm đã cập nhật đến TextView để hiển thị.

Làm theo các bước sau để chỉnh sửa trình xử lý nhấp chuột của Nút Đếm:

1. Xác định vị trí phương thức countUp() vừa được tạo.

```
| usage  
public void countUp(View view) {  
    ...
```

2. Để theo dõi số đếm, bạn cần một biến thành viên riêng tư. Mỗi lần nhấn vào nút Đếm sẽ tăng giá trị của biến này. Nhập đoạn sau, đoạn này sẽ được tô sáng màu đỏ và hiển thị biểu tượng bóng đèn màu đỏ:

```
| usage  
public void countUp(View view) {  
    mCount++;
```

Nếu biểu tượng bóng đèn màu đỏ không xuất hiện, hãy chọn biểu thức mCount++. Bóng đèn màu đỏ sẽ xuất hiện.

3. Nhấp vào biểu tượng bóng đèn màu đỏ và chọn Create field 'mCount' (Tạo trường 'mCount') từ menu bật lên. Điều này tạo ra một biến thành viên riêng tư ở đầu MainActivity và Android Studio giả định rằng bạn muốn nó là một số nguyên (int).

```
public class MainActivity extends AppCompatActivity {  
  
    2 usages  
    private int mCount = 0; // Biến đếm
```

4. Thay đổi câu lệnh biến thành viên riêng tư để khởi tạo biến thành không:

```
public class MainActivity extends AppCompatActivity {  
  
    2 usages  
    private int mCount = 0; // Biến đếm
```

5. Cùng với biến trên, bạn cũng cần một biến thành viên riêng tư cho tham chiếu của TextView `show_count`, mà bạn sẽ thêm vào trình xử lý nhấp chuột. Gọi biến này là `mShowCount`:

```
public class MainActivity extends AppCompatActivity {  
  
    2 usages  
    private int mCount = 0; // Biến đếm  
    3 usages  
    private TextView mShowCount; // TextView để hiển thị số đếm
```

6. Bây giờ bạn đã có `mShowCount`, bạn có thể lấy tham chiếu đến TextView bằng ID bạn đã đặt trong tệp bố cục. Để chỉ lấy tham chiếu này một lần, hãy chỉ định nó trong phương thức `onCreate()`. Như bạn học trong một bài học khác, phương thức `onCreate()` được sử dụng để khởi tạo bố cục, có nghĩa là đặt chế độ xem nội dung của màn hình thành bố cục XML. Bạn cũng có thể sử dụng nó để lấy tham chiếu đến các phần tử UI khác trong bố cục, chẳng hạn như TextView. Xác định vị trí phương thức `onCreate()` trong `MainActivity`.

```
public class MainActivity extends AppCompatActivity {

    2 usages
    private int mCount = 0; // Biến đếm
    3 usages
    private TextView mShowCount; // TextView để hiển thị số đếm

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

        // Gán TextView hiển thị số đếm từ layout
        mShowCount = findViewById(R.id.textViewCount);
    }
}
```

## 7. Thêm câu lệnh findViewById vào cuối phương thức

```
public class MainActivity extends AppCompatActivity {

    2 usages
    private int mCount = 0; // Biến đếm
    3 usages
    private TextView mShowCount; // TextView để hiển thị số đếm

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

        // Gán TextView hiển thị số đếm từ layout
        mShowCount = findViewById(R.id.textViewCount);
    }
}
```

## 8.

9. Một View, giống như một chuỗi, là một tài nguyên có thể có một ID. Lệnh gọi `findViewById` lấy ID của một view làm tham số và trả về View đó. Vì phương thức trả về một View, bạn phải ép kiểu kết quả thành loại view bạn mong đợi, trong trường hợp này là (`TextView`).

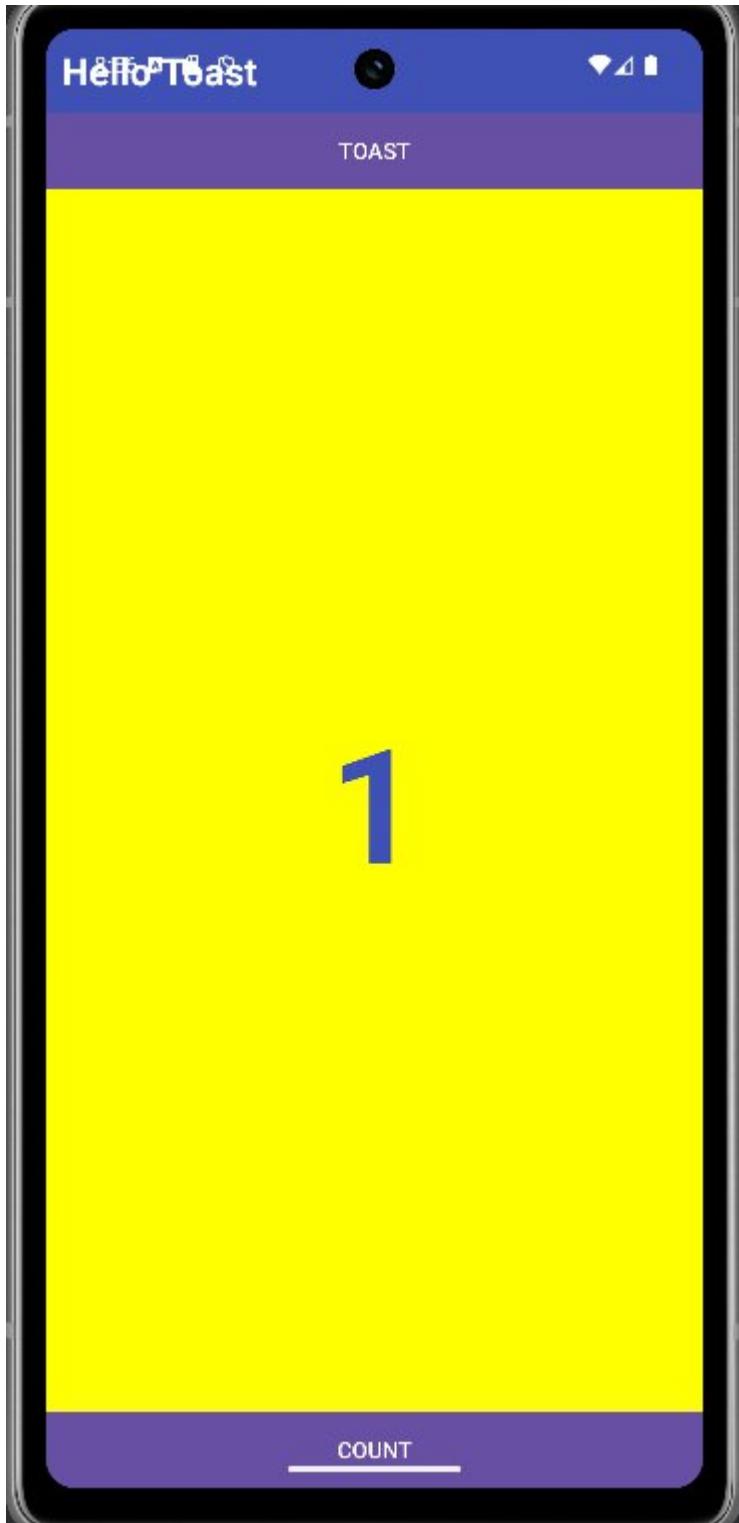
```
    mShowCount;
    if (mShowCount != null) {
        mShowCount.setText(String.valueOf(mCount));
```

10.Bây giờ bạn đã gán TextView cho mShowCount, bạn có thể sử dụng biến này để đặt văn bản trong TextView thành giá trị của biến mCount. Thêm đoạn sau vào phương thức countUp():

Toàn bộ phương thức countUp() bây giờ trông như thế này:

```
1 usage
public void countUp(View view) {
    mCount++;
    if (mShowCount != null) {
        mShowCount.setText(String.valueOf(mCount));
    }
}
```

Chạy ứng dụng để xác minh rằng số đếm tăng lên khi bạn nhấn vào nút Đếm.



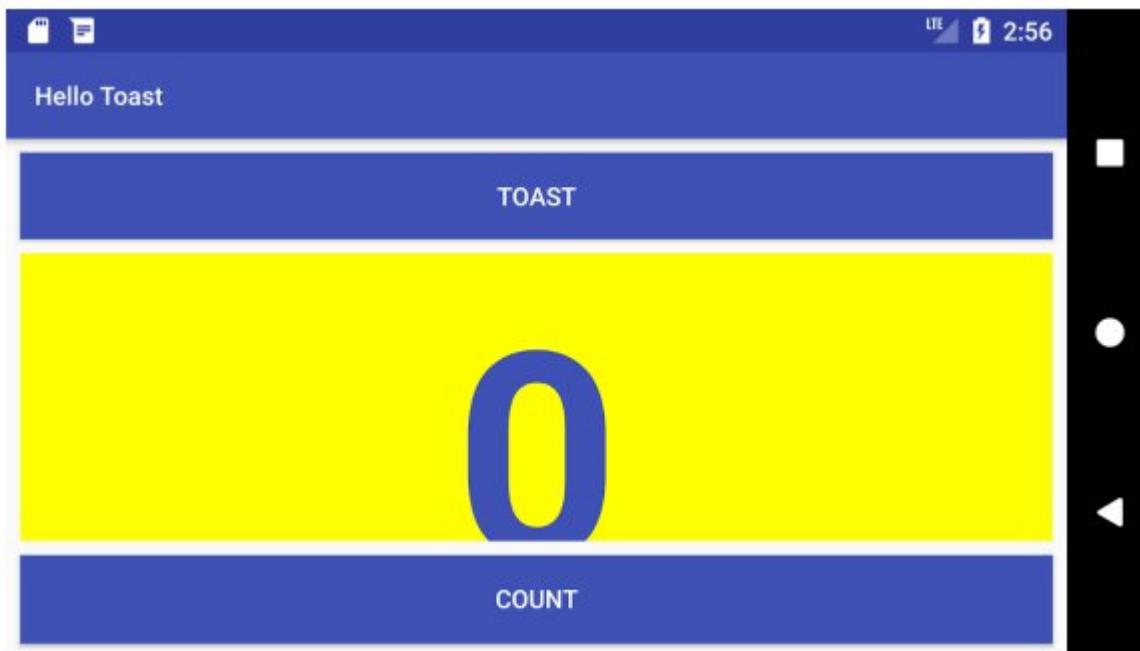
Mẹo: Để có một hướng dẫn chi tiết về cách sử dụng ConstraintLayout, hãy xem Codelab Sử dụng ConstraintLayout để thiết kế các chế độ xem của bạn.

## Mã giải pháp

### Dự án Android Studio: HelloToastChallengeCoding

Lưu ý: Tất cả các thử thách mã hóa là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau này.

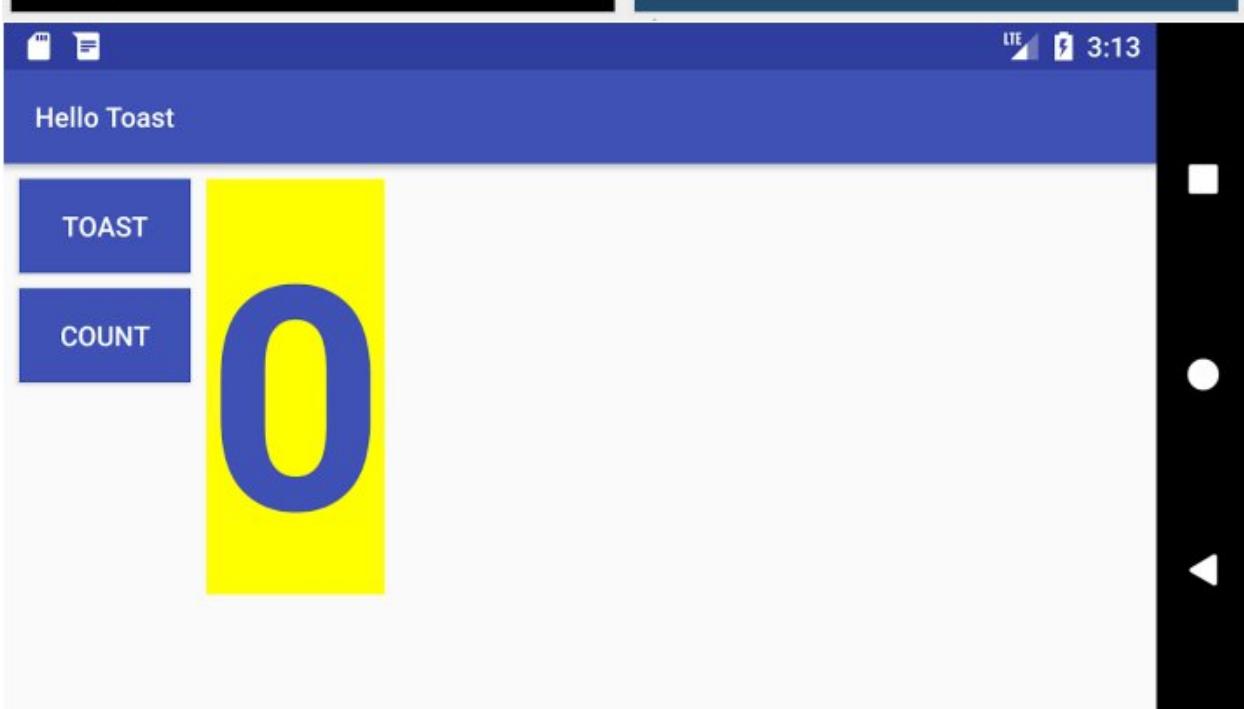
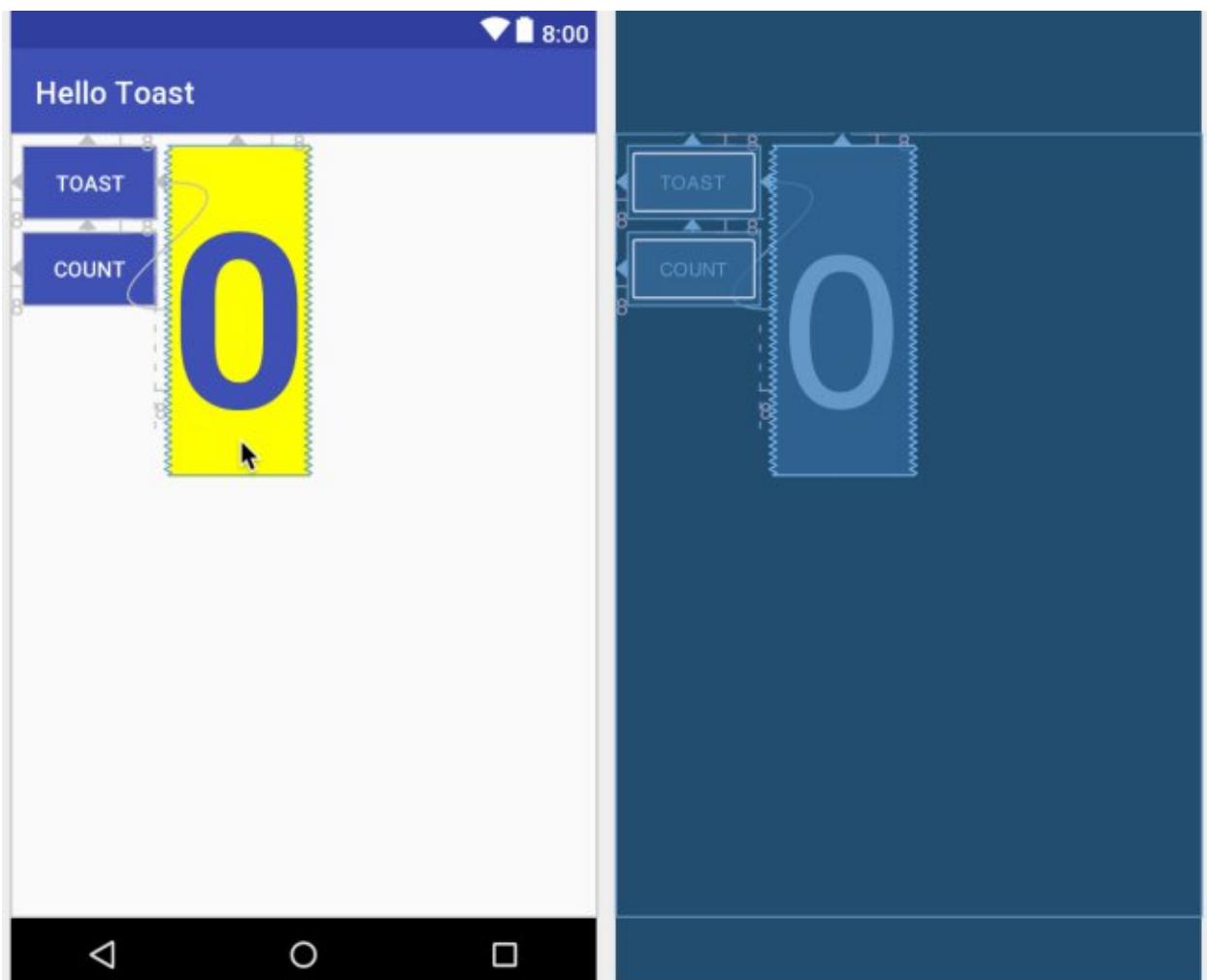
Ứng dụng HelloToast trông ổn khi thiết bị hoặc trình giả lập được định hướng theo chiều dọc. Tuy nhiên, nếu bạn chuyển thiết bị hoặc trình giả lập sang hướng ngang, Nút Đếm có thể chồng lên TextView dọc theo đáy như hình dưới đây.



Thử thách: Thay đổi bố cục để nó trông đẹp ở cả hướng ngang và hướng dọc:

1. Trên máy tính của bạn, hãy tạo một bản sao của thư mục dự án HelloToast và đổi tên nó thành HelloToastChallenge.
2. Mở HelloToastChallenge trong Android Studio và tái cấu trúc nó. (Xem Phụ lục: Tiện ích để biết hướng dẫn về cách sao chép và tái cấu trúc một dự án.)

3. Thay đổi bố cục để Nút Toast và Nút Đếm xuất hiện ở phía bên trái, như trong hình bên dưới. TextView xuất hiện bên cạnh chúng, nhưng chỉ đủ rộng để hiển thị nội dung của nó. (Gợi ý: Sử dụng wrap\_content.)
4. Chạy ứng dụng ở cả hướng ngang và hướng dọc.



## Bài 2) Tóm tắt mã giải pháp thử thách

Dự án Android Studio: HelloToastChallenge

### Tóm tắt

#### View, ViewGroup và bố cục:

- Tất cả các phần tử giao diện người dùng (UI) là các lớp con của lớp View và do đó kế thừa nhiều thuộc tính của lớp cha View.
- Các phần tử View có thể được nhóm bên trong một ViewGroup, hoạt động như một container. Mỗi quan hệ là cha-con, trong đó cha là ViewGroup và con là View hoặc một ViewGroup khác.
- Phương thức `onCreate()` được sử dụng để bơm (inflate) bố cục, có nghĩa là đặt view nội dung của màn hình thành bố cục XML. Bạn cũng có thể sử dụng nó để lấy tham chiếu đến các phần tử UI khác trong bố cục.
- Một View, giống như một chuỗi, là một tài nguyên có thể có ID. Lệnh gọi `findViewById` lấy ID của một view làm tham số và trả về View.

#### Sử dụng trình chỉnh sửa bố cục:

- Nhấp vào tab **Design** để thao tác các phần tử và bố cục, và tab **Text** để chỉnh sửa mã XML cho bố cục.
- Trong tab **Design**, ngăn **Palettes** hiển thị các phần tử UI mà bạn có thể sử dụng trong bố cục ứng dụng của mình và ngăn **Component tree** hiển thị hệ thống phân cấp view của các phần tử UI.
- Các ngăn thiết kế và bản thiết kế của trình chỉnh sửa bố cục hiển thị các phần tử UI trong bố cục.
- Tab **Attributes** hiển thị ngăn **Attributes** để đặt các thuộc tính cho một phần tử UI.
- **Tay cầm ràng buộc (Constraint handle):** Nhấp vào tay cầm ràng buộc, được hiển thị dưới dạng một vòng tròn ở mỗi bên của một phần tử, sau đó kéo đến một tay cầm ràng buộc khác hoặc đến ranh giới cha để tạo một ràng buộc. Ràng buộc được biểu thị bằng đường zigzag.
- **Tay cầm thay đổi kích thước (Resizing handle):** Bạn có thể kéo tay cầm thay đổi kích thước hình vuông để thay đổi kích thước phần tử. Trong khi kéo, tay cầm sẽ thay đổi thành một góc xiên.
- Khi được bật, công cụ **Autoconnect** tự động tạo hai hoặc nhiều ràng buộc cho một phần tử UI đối với bố cục cha. Sau khi bạn kéo phần tử vào bố cục, nó sẽ tạo các ràng buộc dựa trên vị trí của phần tử.
- Bạn có thể xóa các ràng buộc khỏi một phần tử bằng cách chọn phần tử đó và di chuột qua nó để hiển thị nút **Clear Constraints**. Nhấp vào nút này để xóa tất cả

các ràng buộc trên phần tử đã chọn. Để xóa một ràng buộc duy nhất, hãy nhấp vào tay cầm cụ thể đặt ràng buộc đó.

- Ngăn **Attributes** cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một phần tử UI. Nó cũng bao gồm một bảng điều khiển kích thước hình vuông gọi là **view inspector** ở trên cùng. Các ký hiệu bên trong hình vuông đại diện cho cài đặt chiều cao và chiều rộng.

### Đặt chiều rộng và chiều cao bố cục:

Thuộc tính `layout_width` và `layout_height` thay đổi khi bạn thay đổi các điều khiển kích thước chiều cao và chiều rộng trong view inspector. Các thuộc tính này có thể nhận một trong ba giá trị cho `ConstraintLayout`:

- Cài đặt `match_constraint` mở rộng view để lấp đầy phần cha của nó theo chiều rộng hoặc chiều cao — tối đa là một margin, nếu có.
- Cài đặt `wrap_content` thu nhỏ kích thước view để view vừa đủ lớn để chứa nội dung của nó. Nếu không có nội dung, view sẽ trở nên vô hình.
- Sử dụng một số cố định dp (pixel độc lập với mật độ) để chỉ định một kích thước cố định, được điều chỉnh cho kích thước màn hình của thiết bị.

### Trích xuất tài nguyên chuỗi:

Thay vì mã hóa cứng chuỗi, thực tiễn tốt nhất là sử dụng tài nguyên chuỗi, đại diện cho chuỗi. Thực hiện theo các bước sau:

1. Nhấp một lần vào chuỗi được mã hóa cứng để trích xuất, nhấn **Alt-Enter** (Option-Enter trên máy Mac) và chọn **Extract string resources** từ menu bật lên.
2. Đặt **Tên tài nguyên**.
3. Nhấp vào OK . Điều này tạo ra một tài nguyên chuỗi trong tệp `values/res/string.xml`, và chuỗi trong mã của bạn được thay thế bằng một tham chiếu đến tài nguyên: `@string/button_label_toast`

### Xử lý các lần nhấp:

- Trình xử lý nhấp là một phương thức được gọi khi người dùng nhấp hoặc nhấn vào một phần tử UI.
- Chỉ định trình xử lý nhấp cho một phần tử UI, chẳng hạn như một Button, bằng cách nhập tên của nó vào trường **onClick** trong ngăn **Attributes** của tab **Design**, hoặc trong trình chỉnh sửa XML bằng cách thêm thuộc tính `android:onClick` vào một phần tử UI, chẳng hạn như một Button.

- Tạo trình xử lý nhấp trong Activity chính bằng tham số View. Ví dụ: public void showToast(View view) {/...}.
- Bạn có thể tìm thấy thông tin về tất cả các thuộc tính của Button trong tài liệu lớp Button và tất cả các thuộc tính TextView trong tài liệu lớp TextView.

### **Hiển thị thông báo Toast:**

Toast cung cấp một cách để hiển thị một thông báo đơn giản trong một cửa sổ bật lên nhỏ. Nó chỉ chiếm không gian cần thiết cho thông báo. Để tạo một phiên bản của Toast, hãy làm theo các bước sau:

1. Gọi phương thức factory makeText() trên lớp Toast.
2. Cung cấp ngữ cảnh của ứng dụng Activity và thông báo để hiển thị (chẳng hạn như một tài nguyên chuỗi).
3. Cung cấp thời lượng hiển thị, ví dụ: Toast.LENGTH\_SHORT cho một khoảng thời gian ngắn. Thời lượng có thể là Toast.LENGTH\_LONG hoặc Toast.LENGTH\_SHORT.
4. Hiển thị Toast bằng cách gọi show().

### **Khái niệm liên quan:**

Tài liệu khái niệm liên quan có trong 1.2: Bố cục và tài nguyên cho giao diện người dùng.

### **Tìm hiểu thêm:**

Tài liệu dành cho nhà phát triển Android:

- Android Studio
- Xây dựng giao diện người dùng bằng Layout Editor
- Xây dựng giao diện người dùng đáp ứng với ConstraintLayout
- Bố trí
- Cảnh
- Nút
- Chế độ xem văn bản
- Tài nguyên Android
- Tài nguyên R.color chuẩn Android
- Hỗ trợ các mật độ khác nhau
- Sự kiện đầu vào Android
- Ngữ cảnh

### **Khác:**

- Lớp học lập trình: Sử dụng ConstraintLayout để thiết kế chế độ xem

- Thuật ngữ từ vựng và khái niệm

Codelab tiếp theo là Android fundamentals 1.2 Phần B: Trình chỉnh sửa bố cục.

## Bài học 1.2 Phần B: Trình chỉnh sửa bố cục

### Giới thiệu

Như bạn đã học trong 1.2 Phần A: Giao diện người dùng đầu tiên của bạn, bạn có thể xây dựng giao diện người dùng (UI) bằng ConstraintLayout trong trình chỉnh sửa bố cục, đặt các phần tử UI trong bố cục bằng cách sử dụng các kết nối ràng buộc với các phần tử khác và với các cạnh bố cục. ConstraintLayout được thiết kế để dễ dàng kéo các phần tử UI vào trình chỉnh sửa bố cục.

ConstraintLayout là một ViewGroup, là một View đặc biệt có thể chứa các đối tượng View khác (gọi là con hoặc view con). Phần thực hành này trình bày thêm các tính năng của ConstraintLayout và trình chỉnh sửa bố cục.

Phần thực hành này cũng giới thiệu hai lớp con ViewGroup khác:

- LinearLayout: Một nhóm căn chỉnh các phần tử View con bên trong nó theo chiều ngang hoặc chiều dọc.
- RelativeLayout: Một nhóm các phần tử View con, trong đó mỗi phần tử View được định vị và căn chỉnh tương đối với các phần tử View khác trong ViewGroup. Vị trí của các phần tử View con được mô tả liên quan đến nhau hoặc đến ViewGroup cha.

### Những gì bạn nên đã biết

Bạn nên có thể:

- Tạo một ứng dụng Hello World bằng Android Studio.
- Chạy một ứng dụng trên trình giả lập hoặc thiết bị.
- Tạo một bố cục đơn giản cho một ứng dụng với ConstraintLayout.
- Trích xuất và sử dụng tài nguyên chuỗi.

### Những gì bạn sẽ học

- Cách tạo một biến thể bố cục cho hướng ngang (landscape).
- Cách tạo một biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Cách sử dụng ràng buộc baseline để căn chỉnh các phần tử UI với văn bản.
- Cách sử dụng các nút pack và align để căn chỉnh các phần tử trong bố cục.

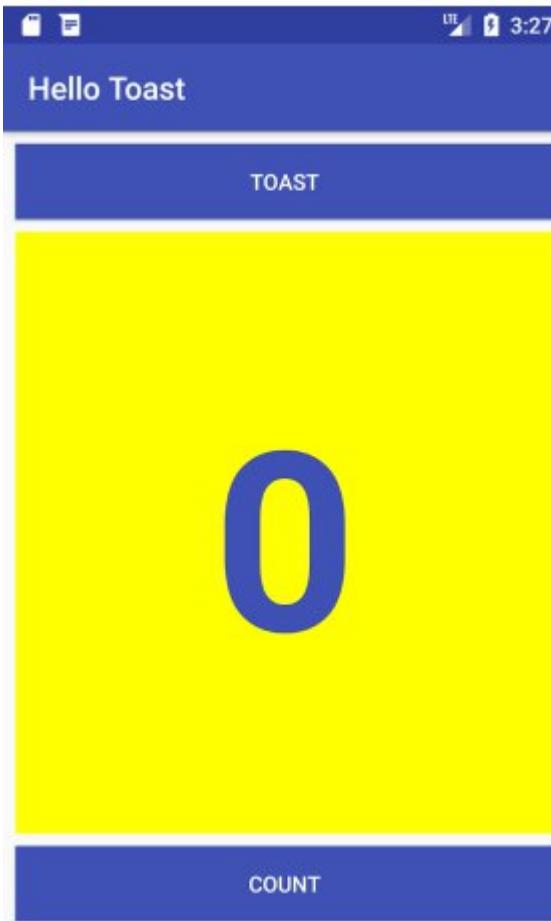
- Cách định vị các view trong một LinearLayout.
- Cách định vị các view trong một RelativeLayout.

## Những gì bạn sẽ làm

- Tạo một biến thể bố cục cho hướng màn hình ngang.
- Tạo một biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Sửa đổi bố cục để thêm các ràng buộc vào các phần tử UI.
- Sử dụng các ràng buộc baseline ConstraintLayout để căn chỉnh các phần tử với văn bản.
- Sử dụng các nút pack và align ConstraintLayout.
- Thay đổi bố cục để sử dụng LinearLayout.
- Định vị các phần tử trong một LinearLayout.
- Thay đổi bố cục để sử dụng RelativeLayout.
- Sắp xếp lại các view trong bố cục chính để tương đối với nhau.

## Tổng quan về ứng dụng

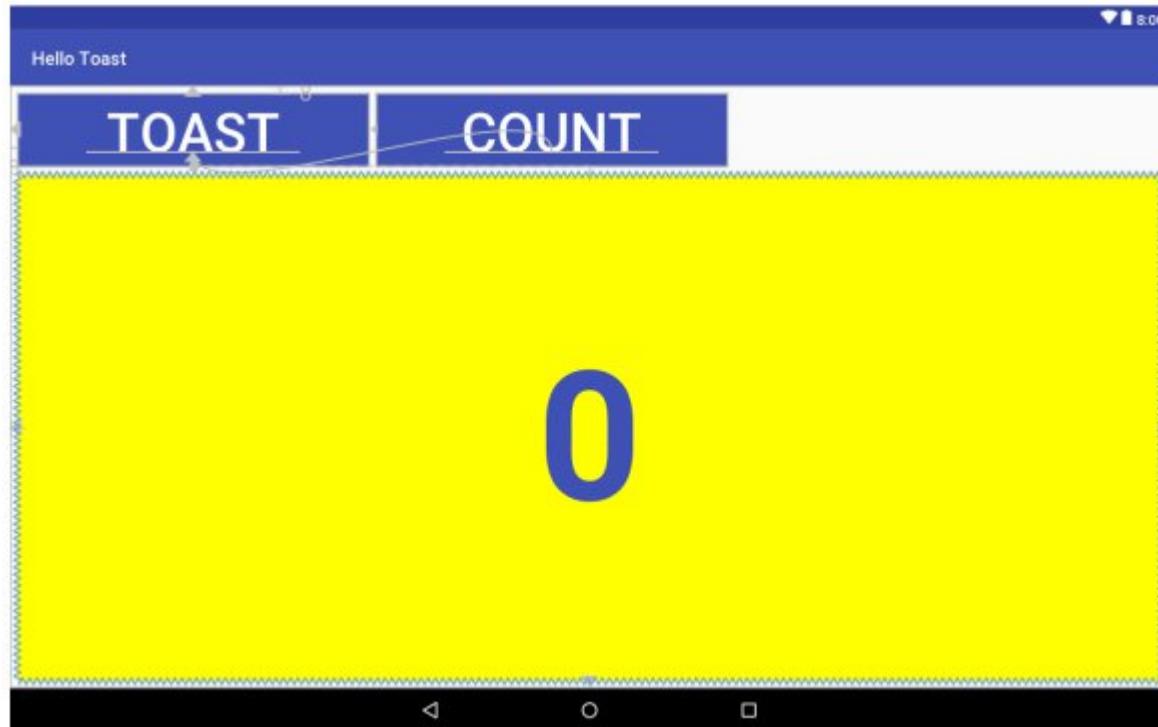
Ứng dụng Hello Toast trong một bài học trước sử dụng ConstraintLayout để sắp xếp các phần tử UI trong bố cục Activity, như trong hình bên dưới.



Để có thêm kinh nghiệm thực hành với ConstraintLayout, bạn sẽ tạo một biến thể của bố cục này cho hướng ngang như hình dưới đây.



Bạn cũng sẽ học cách sử dụng các ràng buộc baseline và một số tính năng căn chỉnh của ConstraintLayout bằng cách tạo một biến thể bố cục khác cho màn hình máy tính bảng.

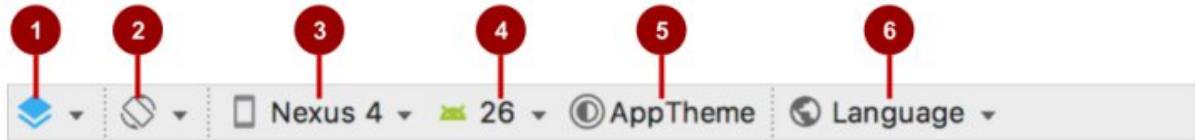


Bạn cũng sẽ tìm hiểu về các lớp con ViewGroup khác như LinearLayout và RelativeLayout, và thay đổi bố cục ứng dụng Hello Toast để sử dụng chúng.

### Nhiệm vụ 1: Tạo các biến thể bố cục

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

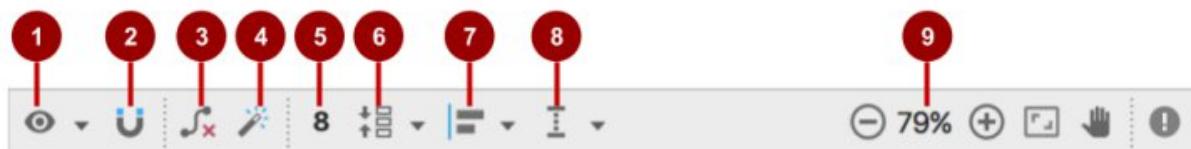
Trong nhiệm vụ này, bạn sẽ sử dụng một số nút trong hai thanh công cụ trên cùng của trình chỉnh sửa bố cục. Thanh công cụ trên cùng cho phép bạn định cấu hình giao diện của bản xem trước bố cục trong trình chỉnh sửa bố cục:



Trong hình trên:

1. **Chọn Bề mặt Thiết kế (Select Design Surface):** Chọn **Design** để hiển thị bản xem trước màu của bố cục của bạn hoặc **Blueprint** để chỉ hiển thị các đường viền cho từng phần tử UI. Để xem cả hai ngăn cạnh nhau, hãy chọn **Design + Blueprint**.
2. **Hướng trong Trình chỉnh sửa (Orientation in Editor):** Chọn **Portrait** hoặc **Landscape** để hiển thị bản xem trước theo hướng dọc hoặc ngang. Điều này rất hữu ích để xem trước bố cục mà không cần phải chạy ứng dụng trên trình giả lập hoặc thiết bị. Để tạo bố cục thay thế, hãy chọn **Create Landscape Variation** hoặc các biến thể khác.
3. **Thiết bị trong Trình chỉnh sửa (Device in Editor):** Chọn loại thiết bị (điện thoại/máy tính bảng, Android TV hoặc Android Wear).
4. **Phiên bản API trong Trình chỉnh sửa (API Version in Editor):** Chọn phiên bản Android để sử dụng để hiển thị bản xem trước.
5. **Chủ đề trong Trình chỉnh sửa (Theme in Editor):** Chọn một chủ đề (chẳng hạn như AppTheme) để áp dụng cho bản xem trước.
6. **Ngôn ngữ trong Trình chỉnh sửa (Locale in Editor):** Chọn ngôn ngữ và ngôn ngữ địa phương cho bản xem trước. Danh sách này chỉ hiển thị các ngôn ngữ có sẵn trong tài nguyên chuỗi (xem bài học về bản địa hóa để biết chi tiết về cách thêm ngôn ngữ). Bạn cũng có thể chọn **Preview as Right To Left** để xem bố cục như thể một ngôn ngữ RTL đã được chọn.

Thanh công cụ thứ hai cho phép bạn định cấu hình giao diện của các phần tử UI trong ConstraintLayout và thu phóng và xoay bản xem trước:



Trong hình trên:

1. **Hiển thị (Show):** Chọn **Show Constraints** và **Show Margins** để hiển thị chúng trong bản xem trước, hoặc để ngừng hiển thị chúng.
2. **Tự động kết nối (Autoconnect):** Bật hoặc tắt **Autoconnect**. Với **Autoconnect** được bật, bạn có thể kéo bất kỳ phần tử nào (chẳng hạn như một Button) đến bất kỳ phần nào của bố cục để tạo các ràng buộc đối với bố cục cha.

3. **Xóa tất cả các ràng buộc (Clear All Constraints):** Xóa tất cả các ràng buộc trong toàn bộ bố cục.
4. **Suy luận ràng buộc (Infer Constraints):** Tạo các ràng buộc bằng suy luận.
5. **Margin mặc định (Default Margins):** Đặt margin mặc định.
6. **Đóng gói (Pack):** Đóng gói hoặc mở rộng các phần tử đã chọn.
7. **Căn chỉnh (Align):** Căn chỉnh các phần tử đã chọn.
8. **Đường dẫn (Guidelines):** Thêm đường dẫn dọc hoặc ngang.
9. **Điều khiển thu phóng/xoay (Điều khiển thu phóng/xoay):** Phóng to hoặc thu nhỏ.

**Mẹo:** Để tìm hiểu thêm về cách sử dụng trình chỉnh sửa bố cục, hãy xem Build a UI with Layout Editor. Để tìm hiểu thêm về cách xây dựng bố cục bằng ConstraintLayout, hãy xem Build a Responsive UI with ConstraintLayout.

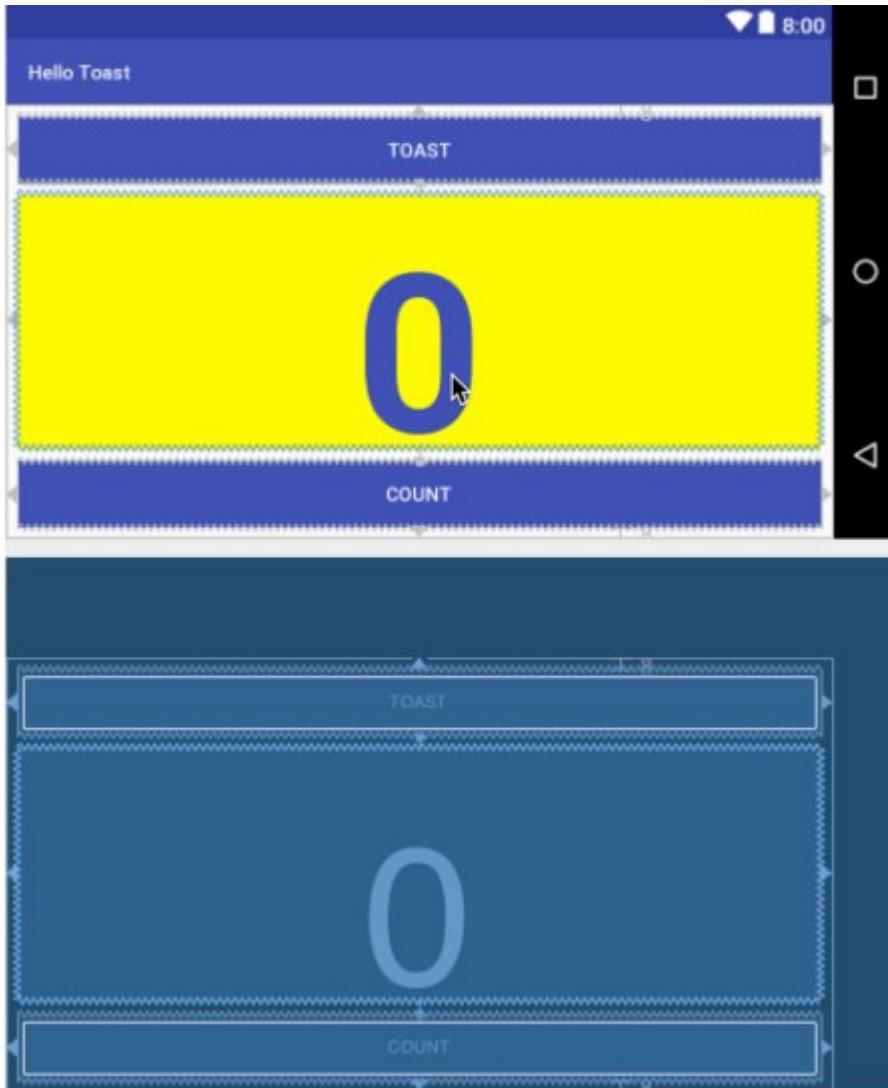
### 1.1 Xem trước bố cục theo hướng ngang

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

1. Mở ứng dụng Hello Toast từ bài học trước.

**Lưu ý:** Nếu bạn đã tải xuống mã giải pháp cho HelloToast, bạn cần xóa các bố cục landscape và extra-large đã hoàn thành mà bạn sẽ tạo trong nhiệm vụ này. Chuyển từ **Project > Android** sang **Project > Project Files** trong ngăn Project, mở rộng **app > app > src/main > res**, chọn cá thư mục **layout-land** và thư mục **layout-xlarge**, và chọn **Edit > Delete**. Sau đó, chuyển ngăn **Project** trở lại **Project > Android**.

1. Mở tệp bố cục **activity\_main.xml**. Nhấp vào tab **Design** nếu nó chưa được chọn.
2. Nhấp vào nút **Orientation in Editor** trên thanh công cụ trên cùng.
3. Chọn **Switch to Landscape** trong menu thả xuống. Bố cục xuất hiện theo hướng ngang như hình dưới đây. Để trở về hướng dọc, hãy chọn **Switch to Portrait**.

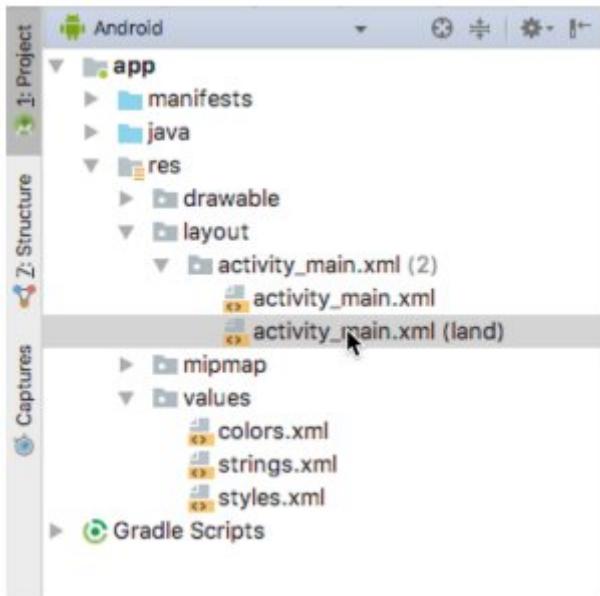


## 1.2 Tạo một biến thể bố cục cho hướng ngang

Sự khác biệt trực quan giữa hướng dọc và hướng ngang cho bố cục này là chữ số (0) trong phần tử `show_count` `TextView` quá thấp đối với hướng ngang — quá gần nút `Count`. Tùy thuộc vào thiết bị hoặc trình giả lập bạn sử dụng, phần tử `TextView` có thể xuất hiện quá lớn hoặc không được căn giữa vì kích thước văn bản được cố định thành `160sp`.

Để khắc phục điều này cho hướng ngang mà vẫn giữ nguyên hướng dọc, bạn có thể tạo biến thể của bố cục ứng dụng `Hello Toast` khác cho hướng ngang. Thực hiện theo các bước sau:

1. Nhấp vào nút **Orientation** in **Editor** trên thanh công cụ trên cùng.
2. Chọn **Create Landscape Variation**. Một cửa sổ trình chỉnh sửa mới sẽ mở ra với tab `land/activity_main.xml` hiển thị bố cục cho hướng landscape (ngang). Bạn có thể thay đổi bố cục này, dành riêng cho hướng ngang, mà không thay đổi hướng portrait (dọc) ban đầu.



- Trong ngăn Project > **Android**, hãy xem bên trong thư mục res > layout, và bạn sẽ thấy rằng Android Studio đã tự động tạo biến thể cho bạn, có tên là activity\_main.xml (land)

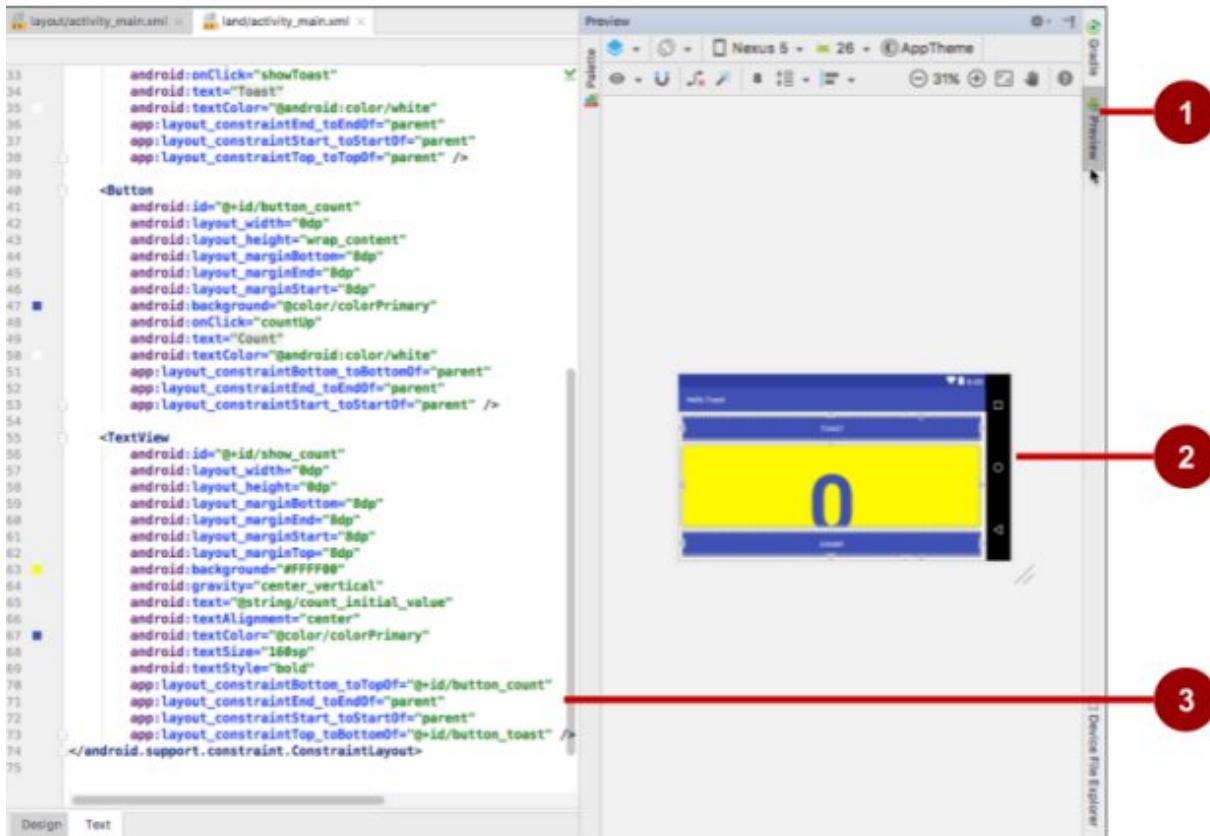
### 1.3 Xem trước bố cục cho các thiết bị khác nhau

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 giả lập. Thực hiện theo các bước sau:

- Tab land/activity\_main.xml vẫn phải mở trong trình chỉnh sửa bố cục; nếu không, hãy nhấp đúp vào tệp activity\_main.xml (land) trong thư mục bố cục.
- Nhấp vào nút **Device in Editor** trên thanh công cụ trên cùng.
- Chọn một thiết bị khác trong menu thả xuống. Ví dụ: chọn Nexus 4, Nexus 5, và sau đó Pixel để xem sự khác biệt trong các bản xem trước. Những khác biệt này là do kích thước văn bản cố định cho TextView.

### 1.4 Thay đổi bố cục cho hướng ngang

Bạn có thể sử dụng ngăn Attributes trong tab **Design** để đặt hoặc thay đổi các thuộc tính, nhưng đôi khi có thể nhanh hơn khi sử dụng tab **Text** để chỉnh sửa trực tiếp mã XML. 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 hình dưới đây

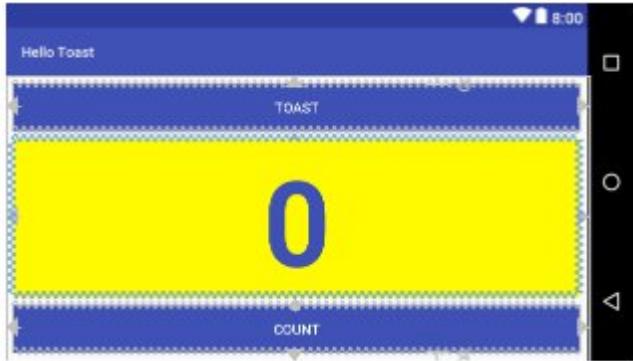


Hình trên cho thấy những điều sau:

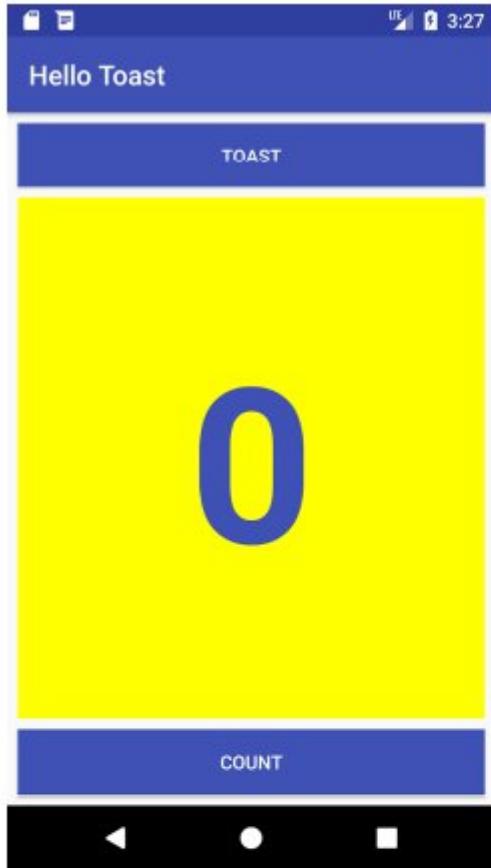
1. Tab **Preview** mà bạn sử dụng để hiển thị ngay xem trước
2. Ngay xem trước
3. Mã XML

Để thay đổi bố cục, hãy thực hiện theo các bước sau:

1. Tab land/activity\_main.xml vẫn phải mở trong trình chỉnh sửa bố cục; nếu không, hãy nhấp đúp vào tệp activity\_main.xml (land) trong thư mục bố cục.
2. Nhập vào tab **Text** và tab **Preview** (nếu chưa được chọn).
3. Tìm phần tử TextView trong mã XML.
4. Thay đổi thuộc tính android:textSize="160sp" thành android:textSize="120sp". Bản xem trước bố cục hiển thị kết quả:

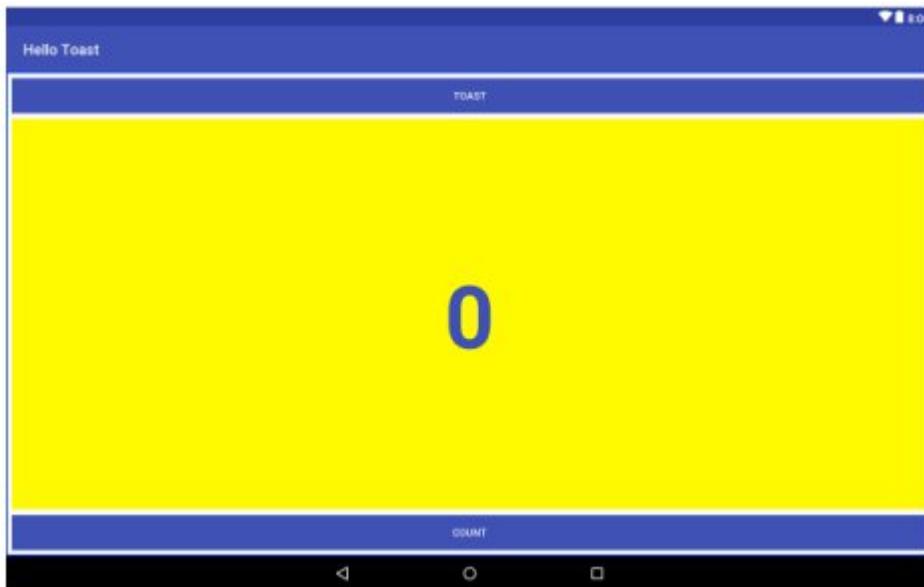


5. Chọn các thiết bị khác nhau trong menu thả xuống **Device in Editor** để xem bố cục trông như thế nào trên các thiết bị khác nhau ở hướng ngang. Trong ngăn trình chỉnh sửa, tab land/activity\_main.xml hiển thị bố cục cho hướng ngang. Tab activity\_main.xml hiển thị bố cục không thay đổi cho hướng dọc. Bạn có thể chuyển đổi qua lại bằng cách nhấp vào các tab.
1. Chạy ứng dụng trên trình giả lập hoặc thiết bị và chuyển đổi hướng từ dọc sang ngang để xem cả hai bố cục.



## 1.5 Tạo biến thể bố cục cho máy tính bảng

Như bạn đã học trước đó, bạn có thể xem trước bố cục cho các thiết bị khác nhau bằng cách nhấp vào nút **Device in Editor** trên thanh công cụ trên cùng. Nếu bạn chọn một thiết bị như Nexus 10 (một máy tính bảng) từ menu, bạn có thể thấy rằng bố cục không lý tưởng cho màn hình máy tính bảng — văn bản của mỗi Button quá nhỏ và sự sắp xếp của các phần tử Button ở trên cùng và dưới cùng không lý tưởng cho một máy tính bảng màn hình lớn.



Để khắc phục điều này cho máy tính bảng mà vẫn giữ nguyên hướng ngang và dọc kích thước điện thoại, bạn có thể tạo biến thể của bố cục hoàn toàn khác cho máy tính bảng. Thực hiện theo các bước sau:

1. Nhấp vào tab **Design** (nếu chưa được chọn) để hiển thị các ngăn thiết kế và bản thiết kế.
2. Nhấp vào nút **Orientation in Editor** trên thanh công cụ trên cùng.
3. Chọn **Create layout x-large Variation**.

Một cửa sổ trình chỉnh sửa mới sẽ mở ra với tab `xlarge/activity_main.xml` hiển thị bố cục cho một thiết bị có kích thước máy tính bảng. Trình chỉnh sửa cũng chọn một thiết bị máy tính bảng, chẳng hạn như Nexus 9 hoặc Nexus 10, cho bản xem trước. Bạn có thể thay đổi bố cục này, dành riêng cho máy tính bảng, mà không thay đổi các bố cục khác.

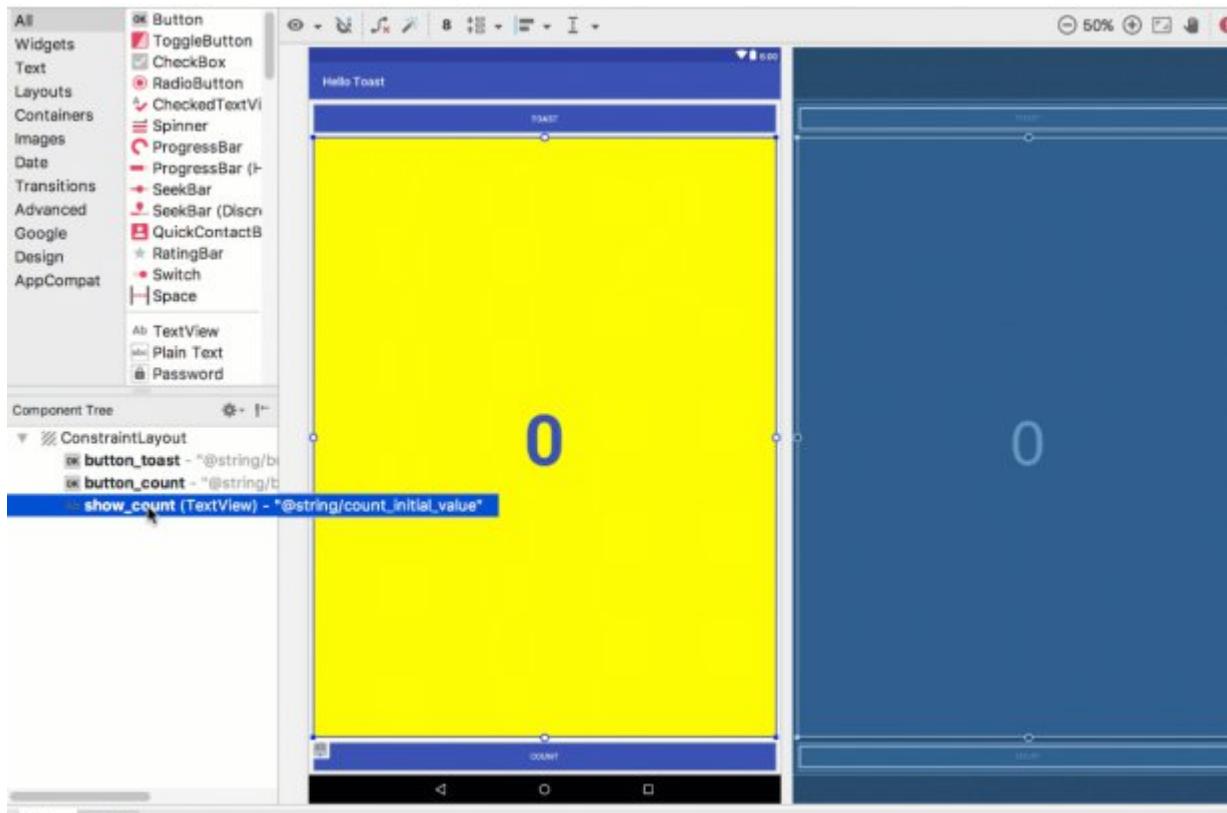
## 1.6 Thay đổi biến thể bố cục cho máy tính bảng

Bạn có thể sử dụng ngăn Attributes trong tab **Design** để thay đổi các thuộc tính cho bố cục này.

1. Tắt công cụ **Autoconnect** trên thanh công cụ. Đối với bước này, hãy đảm bảo rằng công cụ này bị tắt:
2. Xóa tất cả các ràng buộc trong bố cục bằng cách nhấp vào nút **Clear All Constraints** trên thanh công cụ.

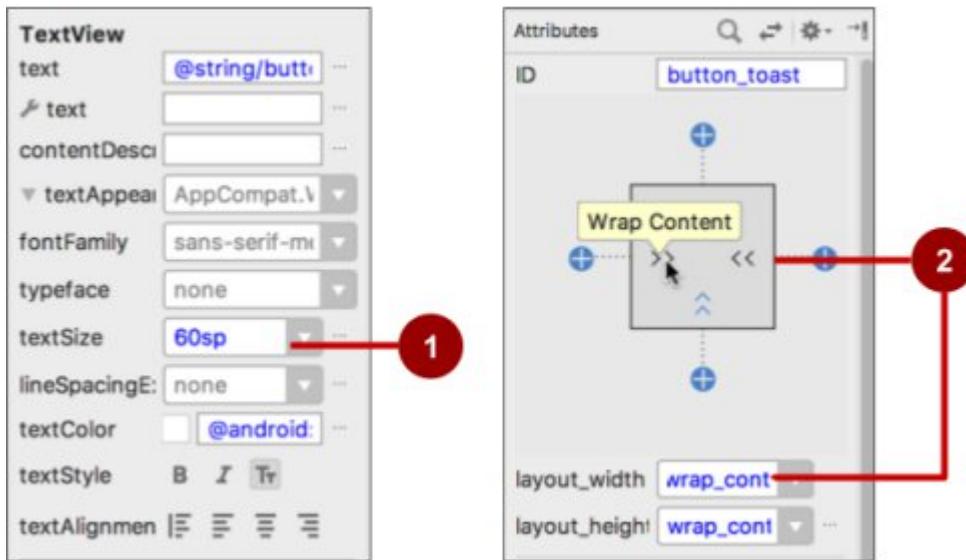
Với các ràng buộc đã bị xóa, bạn có thể di chuyển và thay đổi kích thước các phần tử trên bố cục một cách tự do.

1. Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước ở cả bốn góc của một phần tử để thay đổi kích thước của nó. Trong **Component Tree**, chọn TextView có tên là show\_count. Để đưa TextView ra khỏi đường đi để bạn có thể tự do kéo các phần tử Button, hãy kéo một góc của nó để thay đổi kích thước, như hình động bên dưới.



Thay đổi kích thước một phần tử sẽ mã hóa cứng các kích thước chiều rộng và chiều cao. Tránh mã hóa cứng các kích thước cho hầu hết các phần tử, vì bạn không thể dự đoán kích thước được mã hóa cứng sẽ trông như thế nào trên các 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 đi và bạn sẽ thay đổi kích thước trong một bước khác.

1. Chọn Button button\_toast trong **Component Tree**, nhấp vào tab **Attributes** để mở ngăn **Attributes** và thay đổi textSize thành 60sp (số 1 trong hình bên dưới) và layout\_width thành wrap\_content (số 2 trong hình bên dưới).



Như được hiển thị ở phía bên phải của hình trên (2), bạn có thể nhấp vào điều khiển chiều rộng của view inspector, xuất hiện trong hai phân đoạn ở bên trái và bên phải của hình vuông, cho đến khi nó hiển thị **Wrap Content**. Ngoài ra, bạn có thể chọn `wrap_content` từ menu `layout_width`.

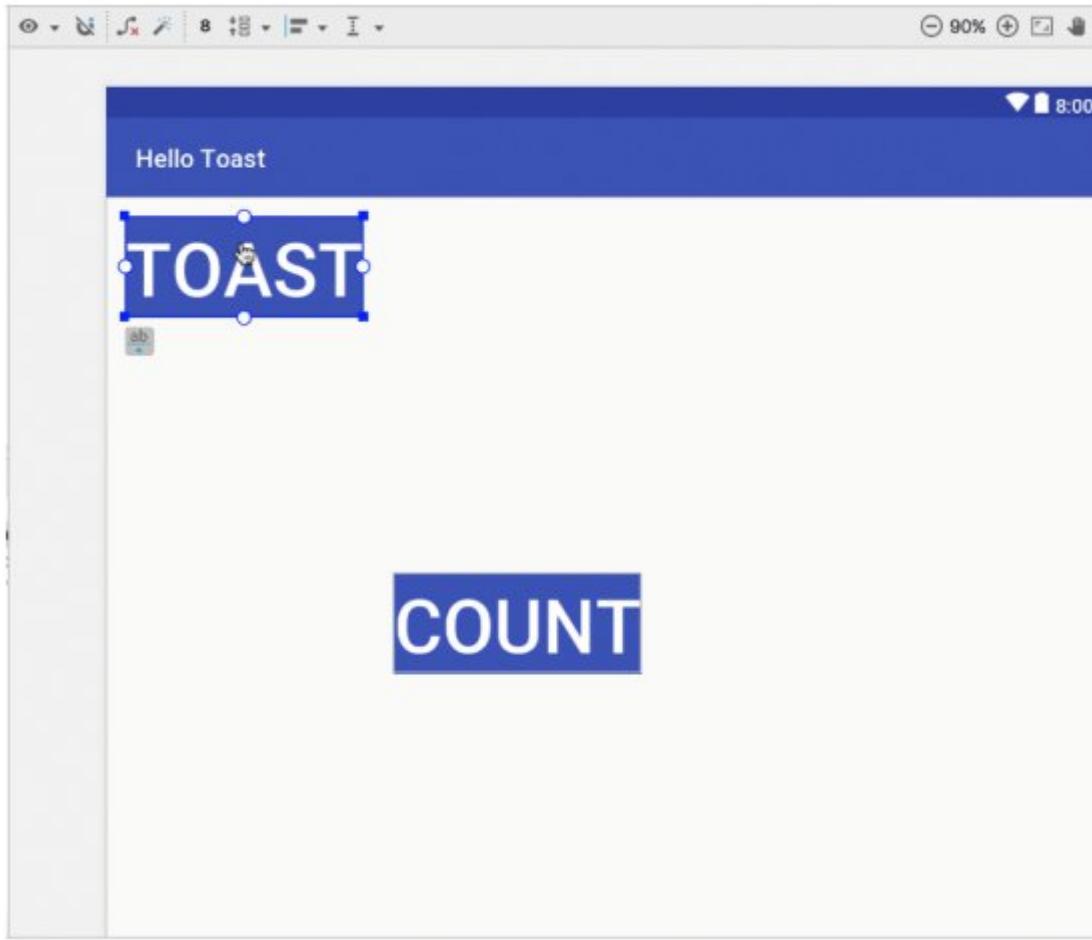
Bạn sử dụng `wrap_content` để nếu văn bản Button được bản địa hóa sang một ngôn ngữ khác, Button sẽ xuất hiện rộng hơn hoặc hẹp hơn để phù hợp với từ đó trong ngôn ngữ khác.

- Chọn Button `button_count` trong **Component Tree**, thay đổi `textSize` thành `60sp` và `layout_width` thành `wrap_content`, và kéo Button phía trên `TextView` đến một khoảng trống trong bố cục.

### 1.7 Sử dụng ràng buộc baseline

Bạn có thể căn chỉnh một phần tử UI chứa văn bản, chẳng hạn như `TextView` hoặc `Button`, với một phần tử UI khác chứa văn bản. Ràng buộc `baseline` cho phép bạn ràng buộc các phần tử sao cho các `baseline` văn bản khớp nhau.

- Ràng buộc `Button button_toast` vào phía trên và bên trái của bố cục, kéo `Button button_count` đến một khoảng trống gần `Button button_toast` và ràng buộc `Button button_count` vào phía bên trái của `Button button_toast`, như hình động dưới đây:

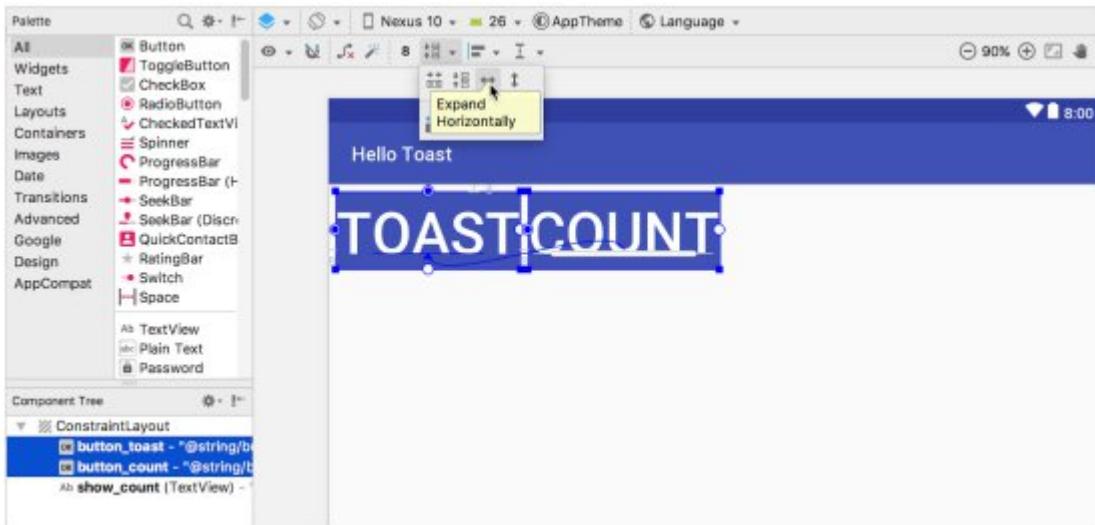


1. Sử dụng ràng buộc baseline, bạn có thể ràng buộc Button button\_count sao cho baseline văn bản của nó khớp với baseline văn bản của Button button\_toast. Chọn phần tử button\_count, sau đó di chuột qua phần tử cho đến khi nút ràng buộc baseline xuất hiện bên dưới phần tử.
2. Nhấp vào nút ràng buộc baseline. Tay cầm baseline xuất hiện, nhấp nháy màu xanh lục như hình động. Nhấp và kéo một đường ràng buộc baseline đến baseline của phần tử button\_toast.

### 1.8 Mở rộng các nút theo chiều ngang

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

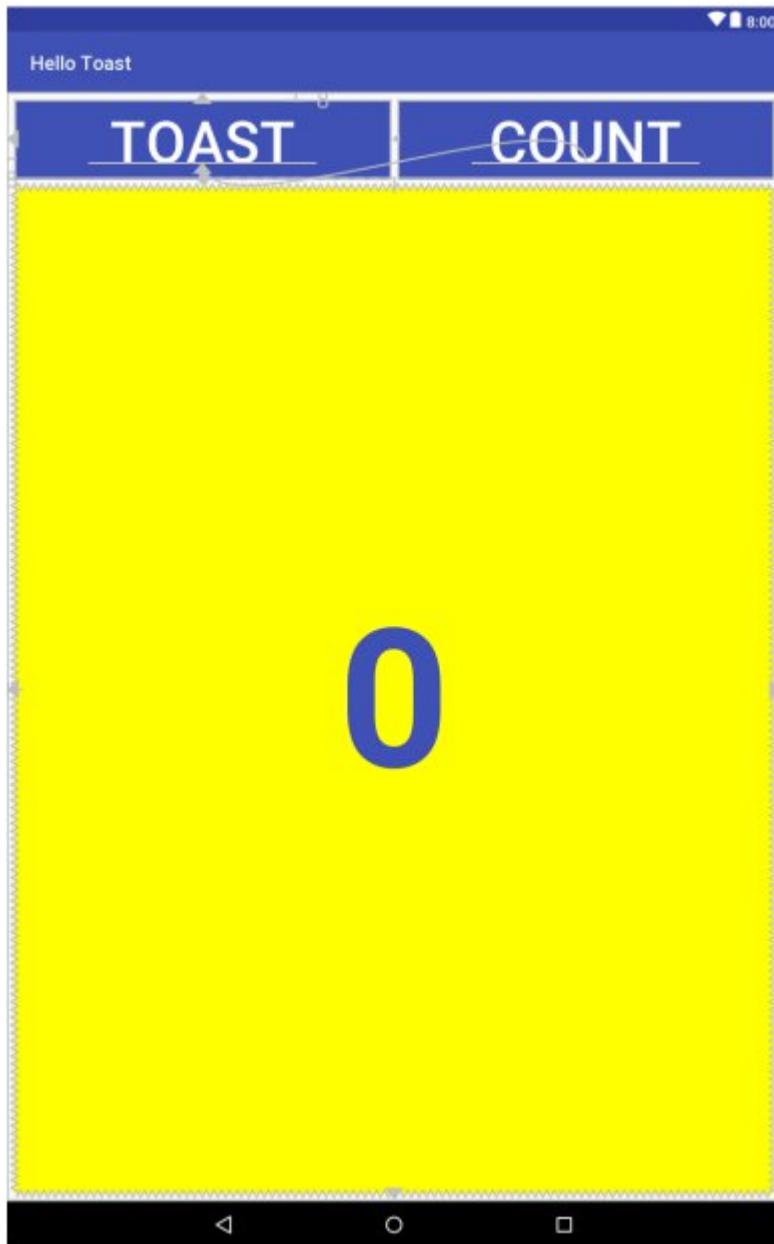
1. Chọn Button button\_count trong Component Tree và nhấn Shift để chọn Button button\_toast sao cho cả hai đều được chọn.
2. Nhấp vào nút pack trên thanh công cụ và chọn Expand Horizontally như trong hình dưới đây.



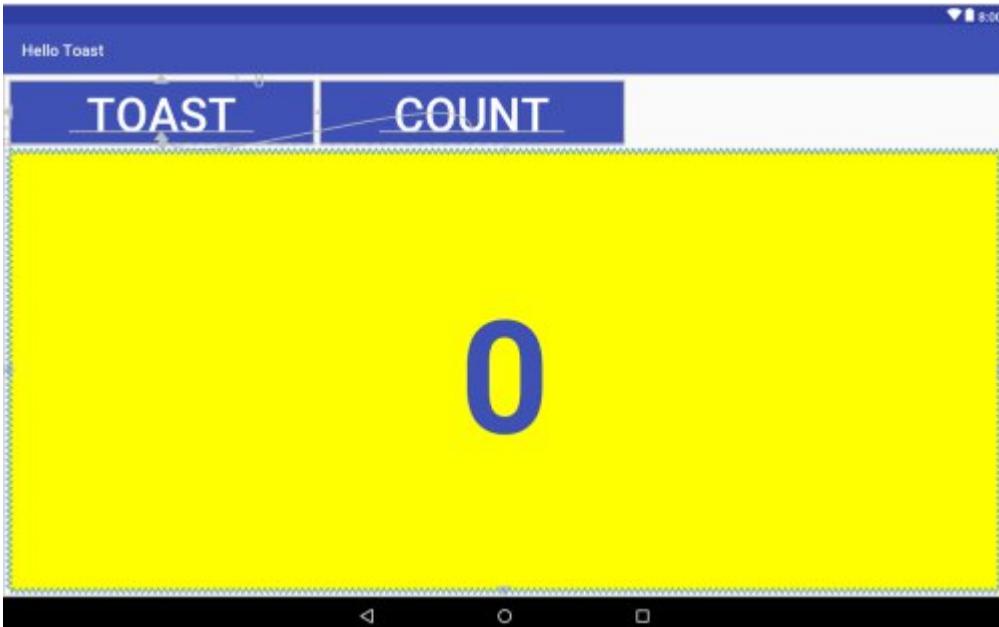
1. Để hoàn thành bố cục, hãy ràng buộc TextView show\_count vào phía dưới của Button button\_toast và vào các cạnh và phía dưới của bố cục, như trong hình động dưới đây.



2. Các bước cuối cùng là thay đổi layout\_width và layout\_height của TextView show\_count thành Match Constraints và textSize thành 200sp. Bố cục cuối cùng trông giống như hình dưới đây.



1. Nhấp vào nút **Orientation in Editor** trên thanh công cụ trên cùng và chọn **Switch to Landscape**. Bố cục máy tính bảng xuất hiện ở hướng ngang như hình dưới đây. (Bạn có thể chọn **Chuyển sang Chân dung** để quay lại hướng dọc.).



- Chạy ứng dụng trên các trình giả lập khác nhau và thay đổi hướng sau khi chạy ứng dụng để xem nó trông như thế nào trên các loại thiết bị khác nhau. Bạn đã tạo thành công một ứng dụng có thể chạy với giao diện người dùng phù hợp trên điện thoại và máy tính bảng có kích thước và mật độ màn hình khác nhau.

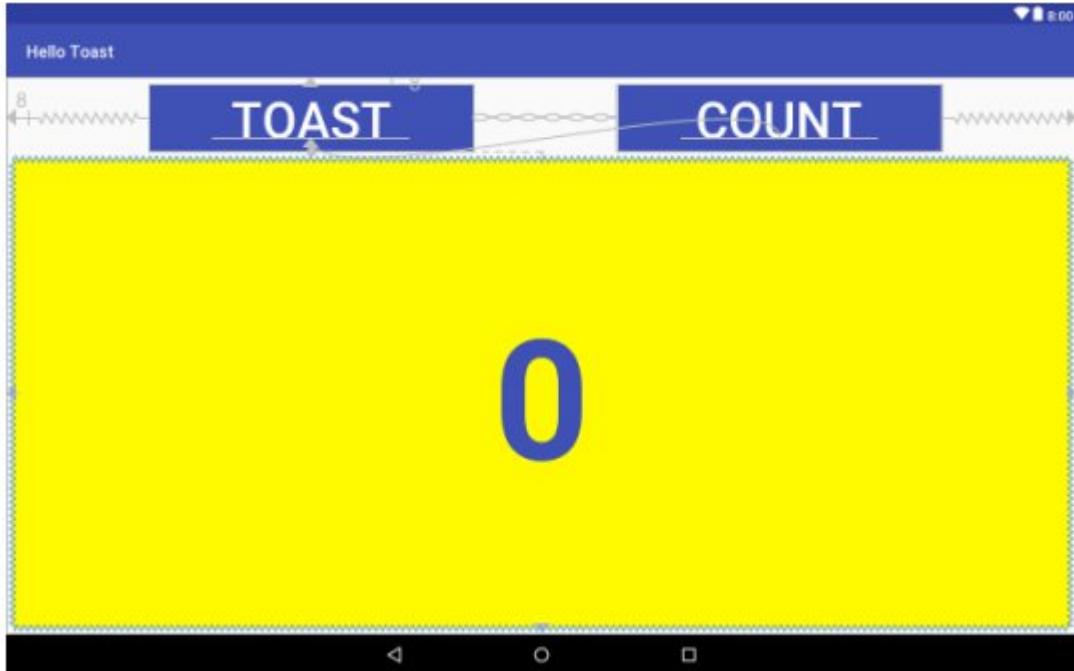
**Mẹo:** Để có hướng dẫn chi tiết về cách sử dụng ConstraintLayout, hãy xem [Sử dụng ConstraintLayout để thiết kế chế độ xem](#) của bạn.

### **Mã giải pháp cho Nhiệm vụ 1**

Dự án Android Studio: [HelloToast](#)

Thử thách mã hóa 1.

Lưu ý: Tất cả các thử thách mã hóa đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau này. Thử thách: Để phù hợp với hướng ngang (ngang) cho máy tính bảng, bạn có thể căn giữa các phần tử Button trong `activity_main.xml (xlarge)` để chúng xuất hiện như trong hình bên dưới. Gợi ý: Chọn các phần tử, bấm vào nút căn chỉnh trên thanh công cụ và chọn Căn giữa theo chiều ngang.



Mã giải pháp thử thách 1 dự án Android Studio: HelloToastChallenge2

### Nhiệm vụ 2: Thay đổi bố cục thành LinearLayout

LinearLayout là một ViewGroup sắp xếp bộ sưu tập các chế độ xem của nó theo hàng ngang hoặc dọc. Một

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 là bắt buộc phải có các thuộc tính sau:

- **layout\_width**
- **layout\_height**
- **định hướng**

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 theo chiều rộng hoặc chiều cao. Khi LinearLayout

là chế độ xem gốc, nó 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. 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):** Chỉ định kích thước cố định, được điều chỉnh cho màn hình

mật độ của thiết bị. Ví dụ: 16dp có nghĩa là 16 pixel không phụ thuộc vào mật độ.

Định hướng có thể là:

- **ngang:** Chế độ xem được sắp xếp từ trái sang phải.
- **dọc:** Chế độ xem 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 Hello Toast từ tác vụ trước đó.

2. Mở tệp bố cục activity\_main.xml (nếu nó chưa được mở) và nhấp vào Văn bản tab tại ở dưới cùng của ngăn chỉnh sửa để xem mã XML. Ở trên cùng của mã XML là Dòng thẻ sau:

```
<android.support.constraint.ConstraintLayout xmlns:android="http://...>
```

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

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android" android:layout_width="match_parent" android:layout_height="match_parent" android:orientation="vertical">
```

#### 4. Đảm bảo rằng thẻ đóng ở cuối mã đã thay đổi thành </LinearLayout> (Android)

Studio tự động thay đổi thẻ kết thúc nếu bạn thay đổi thẻ mở đầu). Nếu chưa thay đổi tự động, thay đổi thủ công.

#### 5. Trong dòng thẻ <LinearLayout>, thêm thuộc tính sau sau

android:layout\_height thuộc tính:

```
    android:orientation="vertical"
```

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 Hello Toast từ tác vụ trước đó.

2. Mở tệp bố cục activity\_main.xml (nếu nó chưa mở) và nhấp vào Văn bản chuyển hướng.

3. Tìm phần tử button\_toast Button và thay đổi thuộc tính sau:

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

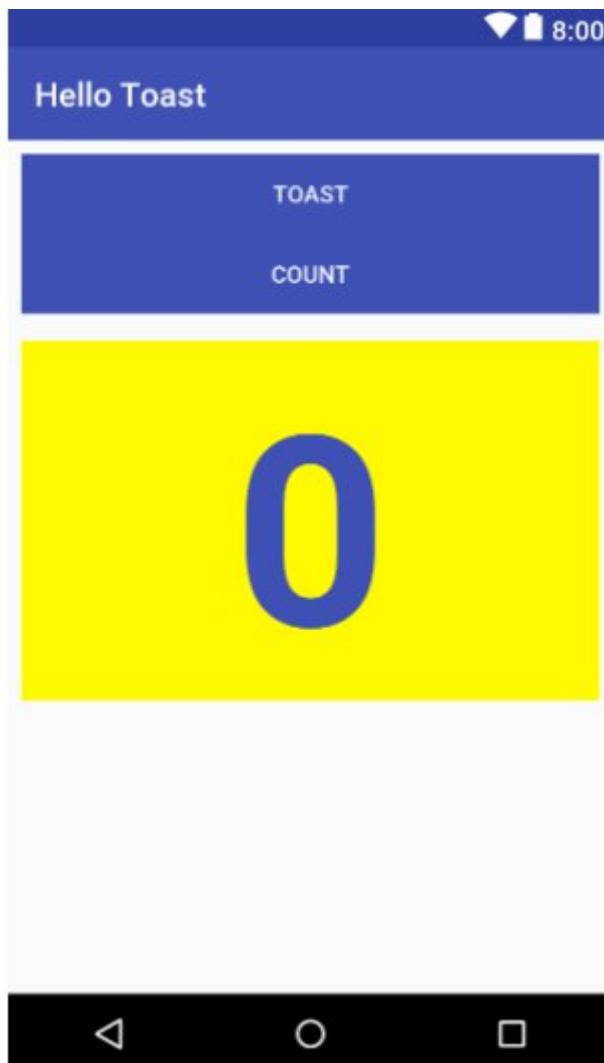
**7. Tìm phần tử TextView show\_count và thay đổi các thuộc tính sau:**

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 phía bên phải của cửa sổ Android Studio (nếu chưa được chọn) để xem trước bố cục cho đến nay:**



### 2.3 Thay đổi vị trí của các phần tử trong LinearLayout

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

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

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở tệp bố cục activity\_main.xml (nếu nó chưa được mở) và nhập vào Văn bản tab
3. Chọn Nút button\_count và tắt cả các thuộc tính của nó, từ thẻ và chọn Chính sửa > Cắt. 4. Nhấp vào sau thẻ /> đóng cửa phần tử TextView nhưng trước thẻ đóng và chọn Chính sửa > Dán.

5. (Tùy chọn) Để khắc phục bất kỳ vấn đề th襆 lè hoặc khoảng cách nào cho mục đích thẩm mĩ, hãy chọn Mã > Định dạng lại Mã để định dạng lại mã XML với khoảng cách và th襆 lè thích hợp. Mã XML cho các phần tử giao diện người dùng bây giờ trông như sau:

```
<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:text="@string/button_label_toast"
    android:textColor="@android:color/white" />

<TextView
    android:id="@+id/show_count"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:background="#FFFF00"
    android:gravity="center_vertical"
    android:text="@string/count_initial_value"
    android:textAlignment="center"
    android:textColor="@color/colorPrimary"
    android:textSize="160sp"
    android:textStyle="bold" />

<Button
    android:id="@+id/button_count"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:background="@color/colorPrimary"

    android:onClick="countUp"
    android:text="@string/button_label_count"
    android:textColor="@android:color/white" />
```

Bằng cách di chuyển Nút button\_count bên dưới TextView, bô cục bây giờ gần với những gì bạn có

trước đó, với nút Count ở phía dưới. Bạn xem trước của bô 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 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 Chế độ xem Bản thân. Trong bài học trước, bạn đặt thuộc tính này cho TextView show\_count để căn giữa nội dung (chữ số 0) ở giữa TextView:

```
android:gravity="center_vertical"
```

Thuộc tính android:layout\_weight cho biết có bao nhiêu không gian bổ sung trong LinearLayout

sẽ được phân bổ cho Chế độ xem. Nếu chỉ có một Chế độ xem có thuộc tính này, thì nó sẽ nhận được tất cả không gian màn hình bổ sung. Cho

nhiều phần tử View, không gian được chia theo tỷ lệ. Ví dụ: nếu mỗi phần tử Button có một trọng lượng của 1 và TextView 2, tổng cộng là 4, các phần tử Button nhận được 1/4 khoảng trống mỗi phần tử và

TextView một nửa.

Trên các thiết bị khác nhau, bố cục có thể hiển thị phần tử TextView show\_count là một phần hoặc hầu hết

của khoảng trống giữa các nút Toast và Count. Để mở rộng TextView để điền vào

Dung lượng khả dụng Bất kể thiết bị nào được sử dụng, hãy chỉ định thuộc tính android:gravity cho thuộc tính

TextView (bằng tiếng Anh). Làm theo các bước sau:

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở tệp bố cục activity\_main.xml (nếu nó chưa mở) và nhập vào Văn bản chuyên hướng.
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.



Phần tử TextView show\_count chiếm tất cả không gian giữa các nút. Bạn có thể xem trước bố cục cho các thiết bị khác nhau, như bạn đã làm trong tác vụ trước bằng cách nhấp vào nút Thiết bị trong Trình chỉnh sửa trên thanh công cụ trên cùng của ngăn xem trước và chọn một thiết bị khác. Không vấn đề gì thiết bị nào bạn chọn cho bản xem trước, phần tử TextView show\_count sẽ chiếm tất cả khoảng cách giữa các nút.

Mã giải pháp nhiệm vụ 2 Mã XML trong activity\_main.xml:

```
<LinearLayout 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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context="com.example.android.hellotoast.MainActivity">

    <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:text="@string/button_label_toast"
        android:textColor="@android:color/white" />

    <TextView
        android:id="@+id/show_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center_vertical"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:background="#FFFF00"
        android:text="@string/count_initial_value"
        android:textAlignment="center"
        android:textColor="@color/colorPrimary"
        android:textSize="160sp"
        android:textStyle="bold"
        android:layout_weight="1"/>

    <Button
        android:id="@+id/button_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"

        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:background="@color/colorPrimary"
        android:onClick="countUp"
        android:text="@string/button_label_count"
        android:textColor="@android:color/white" />
</LinearLayout>
```

### Nhiệm 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 chế độ xem khác

chế độ xem 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 Text

Tab.

1. Mở tệp bố cục activity\_main.xml và nhấp vào Văn bản tab ở cuối chỉnh sửa

để xem mã XML.

2. Thay đổi <LinearLayout ở trên cùng thành <RelativeLayout để câu lệnh trông giống như này:

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

**2.1) Trình chỉnh sửa bố cục**

**2.2) Văn bản và các chế độ cuộn**

**2.3) Tài nguyên có sẵn**

**Bài 3) Hoạt động**

**3.1) Hoạt động và Ý định**

**3.2) Vòng đời của Activity và trạng thái**

**3.3) Ý định ngầm định**

**Bài 4) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ**

**4.1) Trình gỡ lỗi**

**4.2) Kiểm thử đơn vị**

**4.3) Thư viện hỗ trợ**

## **CHƯƠNG 2. TRẢI NGHIỆM NGƯỜI DÙNG**

Bài 4: Tương tác người dùng

4.1 : Hình ảnh có thể nhấp

**Giới thiệu**

Giao diện người dùng (UI) xuất hiện trên màn hình của thiết bị chạy Android bao gồm một hệ thống phân cấp các đối tượng được gọi là chế độ xem. Mỗi 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. View là lớp cơ sở cho các lớp cung cấp các thành phần giao diện người dùng tương tác, chẳng hạn như phần tử Button. Nút là một phần tử giao diện người dùng mà người dùng có thể nhấn hoặc nhấp để thực hiện một hành động.

Bạn có thể biến bất kỳ Chế độ xem nào, chẳng hạn như ImageView, thành phần tử giao diện người dùng có thể nhấn hoặc nhấp vào. Bạn phải lưu trữ hình ảnh cho ImageView trong thư mục drawables của dự án.

Trong thực tế này, bạn học cách sử dụng hình ảnh làm yếu tố mà người dùng có thể nhấn hoặc nhấp vào.

#### a) Những điều bạn nên biết

Bạn sẽ có thể:

- Tạo một dự án Android Studio từ một mẫu và tạo bố cục chính.
- Chạy ứng dụng trên trình mô phỏng hoặc thiết bị được kết nối.
- Tạo và chỉnh sửa các thành phần giao diện người dùng bằng trình chỉnh sửa bố cục và mã XML.
- Truy cập các thành phần giao diện người dùng từ mã của bạn bằng cách sử dụng [findViewById\(\)](#).
- Xử lý một nút [bấm](#).
- Hiển thị thông báo Toast.
- Thêm hình ảnh vào thư mục có thể vẽ của dự án.

#### b) Những gì bạn sẽ học

- Cách sử dụng hình ảnh làm yếu tố tương tác để thực hiện một hành động.
- Cách đặt thuộc tính cho các phần tử ImageView trong trình chỉnh sửa bố cục.
- Cách thêm phương thức onClick() để hiển thị thông báo Toast.

#### c) Bạn sẽ làm gì

- Tạo một dự án Android Studio mới cho một ứng dụng đặt món tráng miệng giả sử dụng hình ảnh làm yếu tố tương tác.
- Đặt trình xử lý onClick() cho hình ảnh để hiển thị các thông báo Toast khác nhau.
- Thay đổi nút hành động nổi do mẫu cung cấp để hiển thị một biểu tượng khác và khởi chạy một Hoạt động khác.

## 1.2) Tổng quan về ứng dụng

Trong thực tế này, bạn tạo và xây dựng một ứng dụng mới bắt đầu với mẫu Hoạt động cơ bản bắt chước ứng dụng đặt món tráng miệng. Người dùng có thể nhấn

vào một hình ảnh để thực hiện một hành động — trong trường hợp này là hiển thị thông báo Toast — như thể hiện trong hình bên dưới. Người dùng cũng có thể nhấn vào nút giở hàng để chuyển sang Hoạt động tiếp theo.



### 1.3) Nhiệm vụ 1: Thêm hình ảnh vào bố cục

Bạn có thể làm cho chế độ xem có thể nhấp được, dưới dạng nút, bằng cách thêm thuộc tính android:onClick vào bố cục XML. Ví dụ: bạn có thể làm cho hình ảnh hoạt động giống như một nút bằng cách thêm android:onClick vào [ImageView](#).

Trong nhiệm vụ này, bạn tạo một nguyên mẫu của một ứng dụng để đặt món tráng miệng từ một quán cà phê. Sau khi bắt đầu một dự án mới dựa trên mẫu Hoạt động cơ bản, bạn sửa đổi TextView "Hello World" bằng văn bản thích hợp và thêm hình ảnh mà người dùng có thể chạm.

#### 1.1 Bắt đầu dự án mới

1. Bắt đầu một dự án Android Studio mới với tên ứng dụng **Droid Cafe**.

2. Chọn mẫu **Hoạt động cơ bản** và chấp nhận tên Hoạt động mặc định (MainActivity). Đảm bảo các tùy chọn Tạo tệp bố cục và **Tương thích ngược** (AppCompat) được chọn.
3. Nhấp vào Hoàn tất.

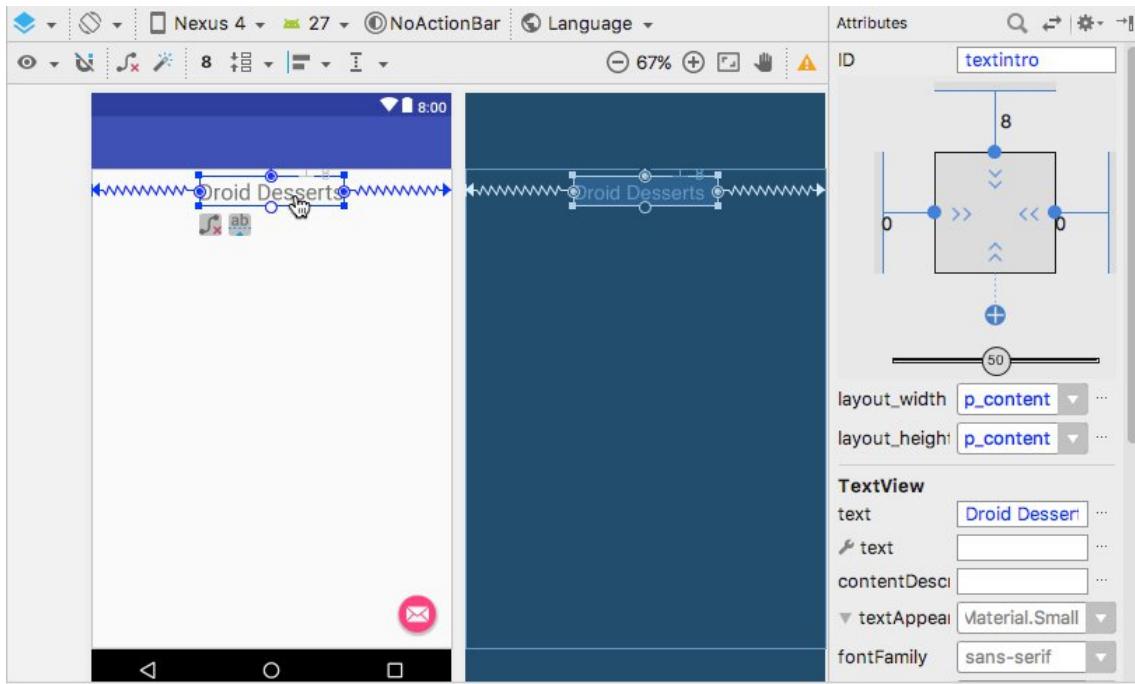
Dự án mở ra với hai bố cục trong thư mục bố cục res >: activity\_main.xml cho thanh ứng dụng và nút hành động nổi (bạn không thay đổi trong tác vụ này) và content\_main.xml cho mọi thứ khác trong bố cục.

4. Mở content\_main.xml và nhấp vào tab **Thiết kế** (nếu chưa được chọn) để hiển thị trình chỉnh sửa bố cục.
5. Chọn TextView "Hello World" trong bố cục và mở ngăn **Attributes**.
6. Thay đổi các thuộc tính textintro như sau:

Trường thuộc tính	Nhập như sau:
ID	văn bản giới thiệu
Nhắn tin	Thay đổi Hello World thành Droid Desserts
textStyle	B (tính bằng đậm)
Kích thước văn bản	24 điểm

Thao tác này sẽ thêm thuộc tính android:id vào TextView với id được đặt thành textintro, thay đổi văn bản, in đậm văn bản và đặt kích thước văn bản lớn hơn là 24sp.

7. Xóa ràng buộc kéo dài từ cuối TextView textintro đến cuối bố cục để TextView gắn vào đầu bố cục và chọn **8** (8dp) cho lề trên cùng như hình dưới đây.



Trong bài học trước, bạn đã học cách trích xuất tài nguyên chuỗi từ chuỗi văn bản theo nghĩa đen. Nhấp vào tab **Văn bản** để chuyển sang mã XML và trích xuất chuỗi "Droid Desserts" trong tệp TextView và nhập `intro_text` làm **tên** tài nguyên chuỗi.

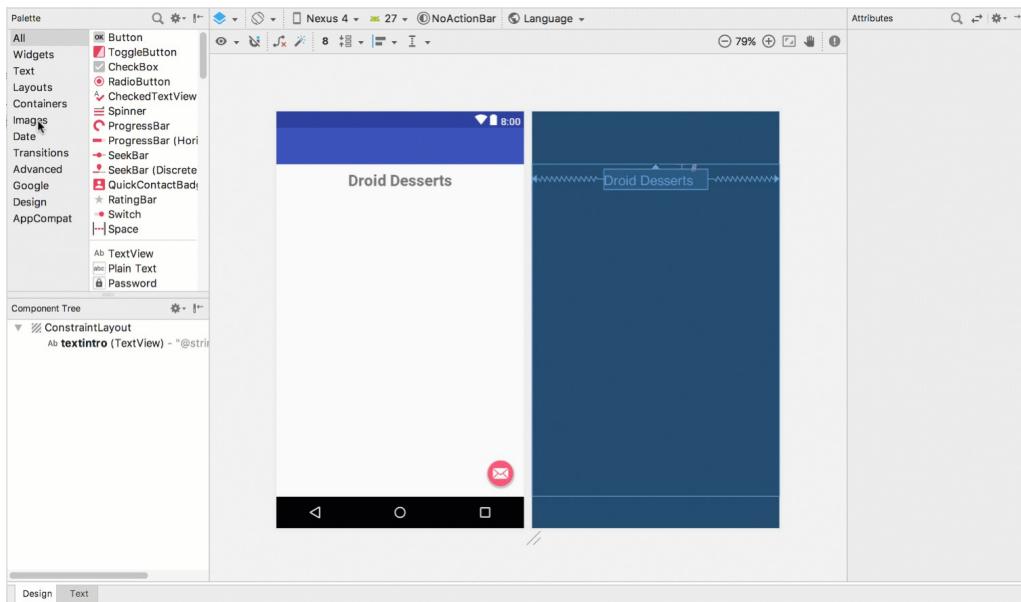
## 1.2 Thêm hình ảnh

Ba hình ảnh (`donut_circle.png`, `froyo_circle.png` và `icecream_circle.png`) được cung cấp cho ví dụ này mà bạn có thể tải xuống. Thay vào đó, bạn có thể thay thế hình ảnh của riêng mình dưới dạng tệp PNG, nhưng chúng phải có kích thước khoảng 113 x 113 pixel để sử dụng trong ví dụ này.

Bước này cũng giới thiệu một kỹ thuật mới trong trình chỉnh sửa bố cục: sử dụng **nút Fix** trong thông báo cảnh báo để trích xuất tài nguyên chuỗi.

- Để sao chép hình ảnh vào dự án của bạn, trước tiên hãy đóng dự án.
- Sao chép các tệp hình ảnh vào thư mục **có thể vẽ** của dự án của bạn.  
Tim thư mục **có thể vẽ** trong một dự án bằng cách sử dụng đường dẫn sau: `project_name ứng dụng > > src > độ phân giải > chính > có thể vẽ`.
- Mở lại dự án của bạn.

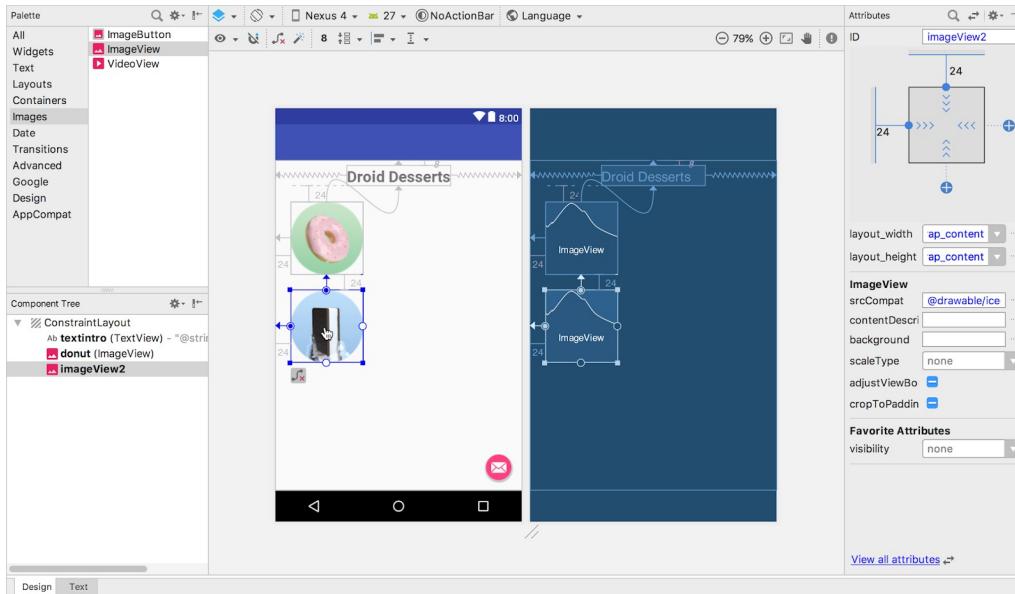
- Mở tệp **content\_main.xml** và nhấp vào tab **Thiết kế** (nếu chưa được chọn).
- Kéo ImageView vào bố cục, chọn hình ảnh **donut\_circle** cho bố cục đó và hạn chế hình ảnh đó ở TextView trên cùng và ở phía bên trái của bố cục với lề 24 (24dp) cho cả hai ràng buộc, như được hiển thị trong hình động bên dưới.



Trong ngăn **Thuộc tính**, nhập các giá trị sau cho các thuộc tính:

Trường thuộc tính	Nhập như sau:
ID	Donut
nội dungMô tả	Bánh rán được tráng men và rắc kẹo. (Bạn có thể sao chép/dán văn bản vào trường.)

- Kéo ImageView thứ hai vào bố cục, chọn hình ảnh **icecream\_circle** cho nó và hạn chế nó ở cuối ImageView đầu tiên và ở phía bên trái của bố cục với lề 24 (24dp) cho cả hai ràng buộc.



- Trong ngăn **Thuộc tính**, nhập các giá trị sau cho các thuộc tính:

Trường thuộc tính	Nhập như sau:
ID	ice_cream
nội dungMô tả	Bánh mì kẹp kem có bánh xốp sô cô la và nhân vani. (Bạn có thể sao chép/dán văn bản vào trường.)

- Kéo ImageView thứ ba vào bố cục, chọn hình ảnh froyo\_circle cho nó và hạn chế nó ở cuối ImageView thứ hai và sang phía bên trái của bố cục với lề **24** (24dp) cho cả hai ràng buộc.

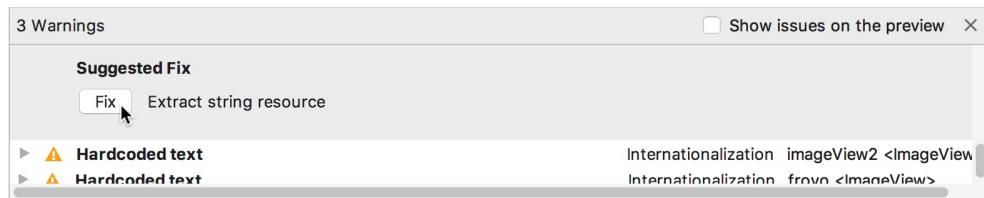
- Trong ngăn **Thuộc tính**, nhập các giá trị sau cho các thuộc tính:

Trường thuộc tính	Nhập như sau:
ID	Froyo

- Nhấp vào biểu tượng cảnh báo ở góc trên bên trái của trình chỉnh sửa bố cục để mở ngăn cảnh báo, ngăn này sẽ hiển thị cảnh báo về văn bản được mã hóa cứng:



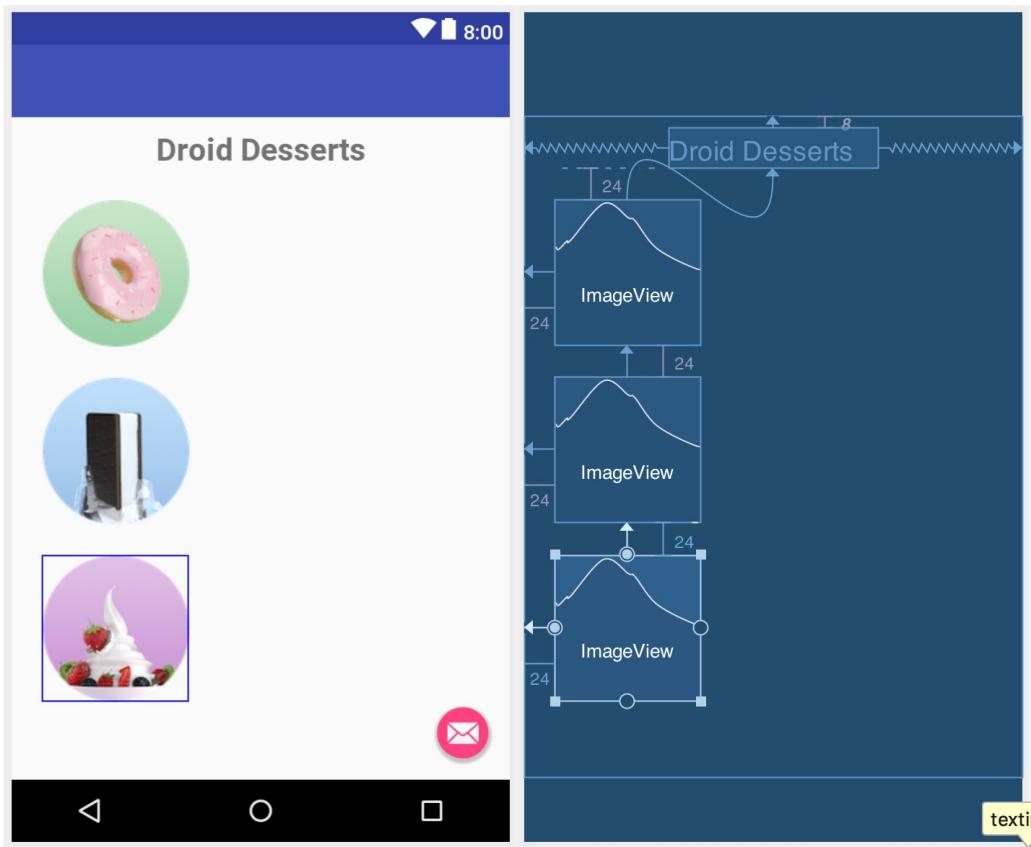
2. Mở rộng từng cảnh **báo văn bản được mã hóa cứng**, cuộn xuống cuối thông báo cảnh báo và nhấp vào nút **Khắc phục như hình dưới đây**:



Bản sửa lỗi cho mỗi cảnh báo văn bản được mã hóa cứng sẽ trích xuất tài nguyên chuỗi cho chuỗi. **Hộp thoại Trích xuất tài nguyên** xuất hiện và bạn có thể nhập tên cho tài nguyên chuỗi. Nhập các tên sau cho tài nguyên chuỗi:

Xâu	Nhập tên sau:
Bánh rán được tráng men và rắc kẹo.	Donuts
Bánh mì kẹp kem có bánh xốp sô cô la và nhân vani.	ice_cream_sandwiches

Bố cục bây giờ sẽ trông giống như hình bên dưới.



## 1.1 Thêm mô tả văn bản

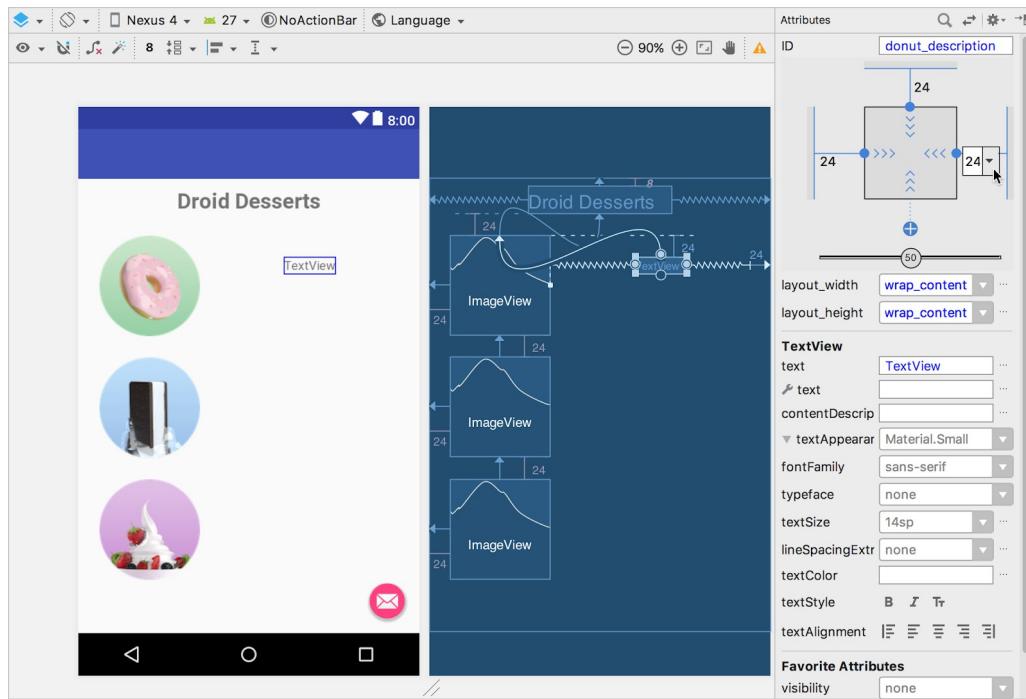
Trong bước này, bạn thêm mô tả văn bản (TextView) cho mỗi món tráng miệng. Vì bạn đã trích xuất tài nguyên chuỗi cho các trường contentDescription cho các phần tử ImageView, bạn có thể sử dụng cùng một tài nguyên chuỗi cho mỗi TextView mô tả.

1. Kéo phần tử TextView vào bố cục.

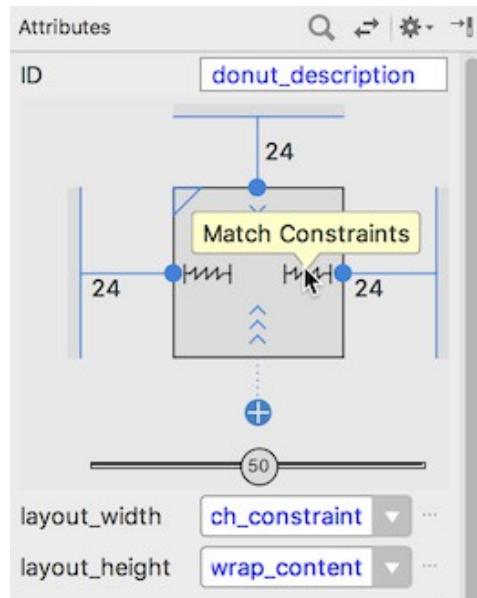
Hạn chế phía bên trái của phần tử ở phía bên phải của bánh rán ImageView và

1. trên cùng của phần tử ở trên cùng của bánh rán ImageView, cả hai đều có lề là 24 (24dp).
2. Hạn chế phía bên phải của phần tử ở phía bên phải của bố cục và sử dụng cùng một lề của 24 (24dp). Nhập **donut\_description** cho trường ID trong ngăn Thuộc tính. Kiểu dáng mới

TextView sẽ xuất hiện bên cạnh hình ảnh bánh rán như trong hình bên dưới.



- Trong ngăn **Thuộc tính**, thay đổi chiều rộng trong khung kiểm tra thành **Phù hợp với các ràng buộc**:



## **CHƯƠNG 3. LÀM VIỆC TRONG NỀN**

### **Bài 1) Các tác vụ nền**

- 1.1) Nhiệm vụ không đồng bộ**
- 1.2) AsyncTask và AsyncTaskLoader**
- 1.3) Máy thu phát sóng**

### **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) Tùy chọn được chia sẻ**
- 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**