

TRƯỜNG ĐẠI HỌC THỦY LỢI
KHOA CÔNG NGHỆ THÔNG TIN



GIÁO TRÌNH

THỰC HÀNH PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI ĐỘNG

Hà Nội, 2.2025

MỤC LỤC

CHƯƠNG 1.	LÀM QUEN	4
Bài 1)	Tạo ứng dụng đầu tiên	4
1.1)	Android Studio và Hello World	4
a.	Cài đặt Android Studio	6
b.	Tạo ứng dụng Hello World	7
c.	Sử dụng thiết bị vật lý	16
d.	Thay đổi cấu hình Gradle của ứng dụng	17
e.	Thêm câu lệnh nhật ký vào ứng dụng của bạn	18
1.2)	Giao diện người dùng tương tác đầu tiên	20
a.	Tạo và khám phá một dự án mới	23
b.	Thêm các phần tử View vào trình chỉnh sửa bố cục	24
c.	Thay đổi thuộc tính của phần tử giao diện người dùng	28
d.	Thêm một TextEdit và đặt thuộc tính cho nó	33
e.	Thiết kế bố cục trong XML	37
f.	Thêm bộ điều khiển onClick cho nút	39
1.3)	Trình chỉnh sửa bố cục	46
a.	Tạo biến thể bố cục	49
b.	Thay đổi bố cục thành LinearLayout	66
c.	Thay đổi bố cục thành RelativeLayout	74
1.4)	Văn bản và các chế độ cuộn	77
a.	Thêm và chỉnh sửa các thành phần TextView	80
b.	Thêm ScrollView và web link đang hoạt động	84
c.	Cuộn nhiều phần tử	86
1.5)	Tài nguyên có sẵn	88
a.	Thay đổi biểu tượng trình khởi chạy	88
b.	Sử dụng project templates	91

c.	Học hỏi từ code mẫu	92
d.	Nhiều tài nguyên khác	93
Bài 2)	Activities	93
2.1)	Activity và Intent	93
2.2)	Vòng đời của Activity và trạng thái	93
2.3)	Intent ngầm định	93
Bài 3)	Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ	93
3.1)	Trình gỡ lỗi	93
3.2)	Kiểm thử đơn vị	93
3.3)	Thư viện hỗ trợ	93
CHƯƠNG 2.	TRẢI NGHIỆM NGƯỜI DÙNG	94
Bài 1)	Tương tác người dùng	94
1.1)	Hình ảnh có thể chọn	94
1.2)	Các điều khiển nhập liệu	94
1.3)	Menu và bộ chọn	94
1.4)	Điều hướng người dùng	94
1.5)	RecyclerView	94
Bài 2)	Trải nghiệm người dùng thú vị	94
2.1)	Hình vẽ, định kiểu và chủ đề	94
2.2)	Thẻ và màu sắc	94
2.3)	Bố cục thích ứng	94
Bài 3)	Kiểm thử giao diện người dùng	94
3.1)	Espresso cho việc kiểm tra UI	94
CHƯƠNG 3.	LÀM VIỆC TRONG NỀN	94
Bài 1)	Các tác vụ nền	94
1.1)	AsyncTask	94
1.2)	AsyncTask và AsyncTaskLoader	94
1.3)	Broadcast receivers	94

Bài 2)	Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền	94
2.1)	Thông báo.....	94
2.2)	Trình quản lý cảnh báo	94
2.3)	JobScheduler	94
CHƯƠNG 4.	LƯU DỮ LIỆU NGƯỜI DÙNG	95
Bài 1)	Tùy chọn và cài đặt	95
1.1)	Shared preferences	95
1.2)	Cài đặt ứng dụng	95
Bài 2)	Lưu trữ dữ liệu với Room	95
2.1)	Room, LiveData và ViewModel	95
2.2)	Room, LiveData và ViewModel	95

CHƯƠNG 1. LÀM QUEN

Bài 1) Tạo ứng dụng đầu tiên

1.1) Android Studio và Hello World

Giới thiệu

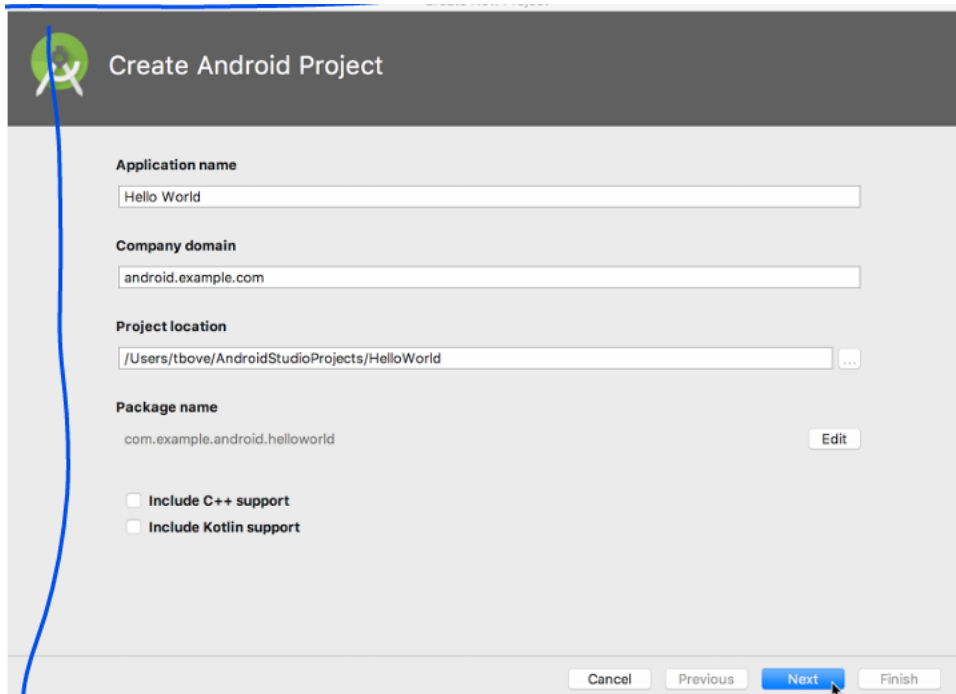
Trong bài thực hành này, bạn sẽ tìm hiểu cách cài đặt Android Studio, môi trường phát triển Android. Bạn cũng sẽ tạo và chạy ứng dụng Android đầu tiên của mình, Hello World, trên một trình giả lập và trên một thiết bị vật lý.

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 ứng dụng

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

a. Cài đặt Android Studio

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

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

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

1. Truy cập trang web của [nhà phát triển Android](#) và làm theo hướng dẫn để tải về và cài đặt Android Studio.

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

3. Sau khi cài đặt xong, Trình hướng dẫn Cài đặt sẽ tải xuống và cài đặt một số thành phần bổ sung bao gồm Android SDK. Hãy kiên nhẫn, điều này có thể mất một thời gian tùy thuộc vào tốc độ Internet của bạn, và một số bước có thể dường như lặp lại.

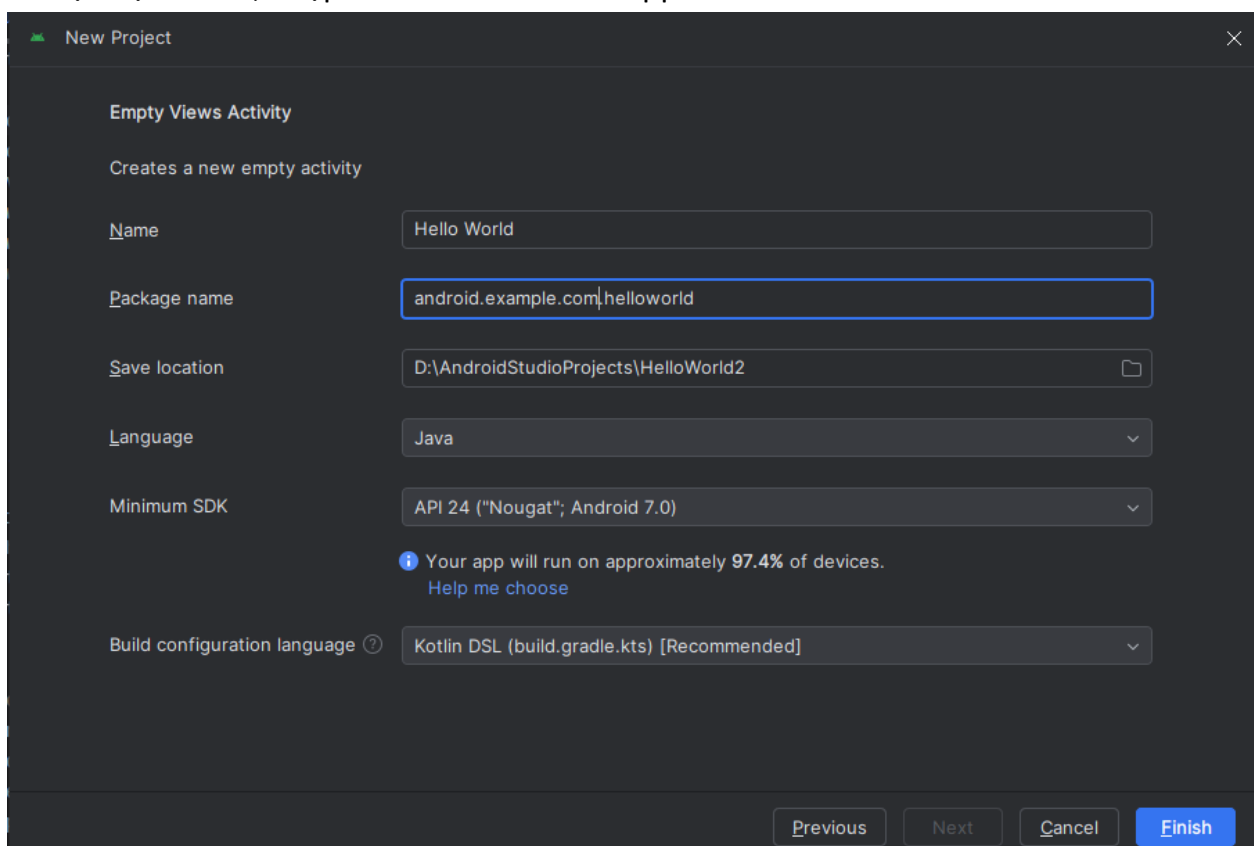
4. Khi quá trình tải xuống hoàn tất, Android Studio sẽ khởi động, và bạn đã sẵn sàng để tạo dự án đầu tiên của mình.

b. Tạo ứng dụng Hello World

Trong nhiệm vụ này, bạn sẽ tạo một ứng dụng hiển thị Hello World để xác minh rằng Android Studio đã được cài đặt đúng cách và để học cơ bản về phát triển với Android Studio.

i. Tạo dự án ứng dụng

1. Mở Android Studio nếu nó chưa được mở.
2. Trong cửa sổ **New Project**, chọn **Phone and Tablet**, trong Phone and Tablet chọn **Empty Views Activity** và nhấn **Next**.
3. Tại mục Name, nhập Hello World cho Application name.



4. Xác minh rằng **Save location** của Dự án là nơi bạn muốn lưu trữ ứng dụng Hello World và các dự án khác của Android Studio, hoặc thay đổi nó thành thư mục ưa thích của bạn.
5. Chấp nhận mặc định **android.example.com** cho **Company Domain** , hoặc tạo một tên miền công ty duy nhất.

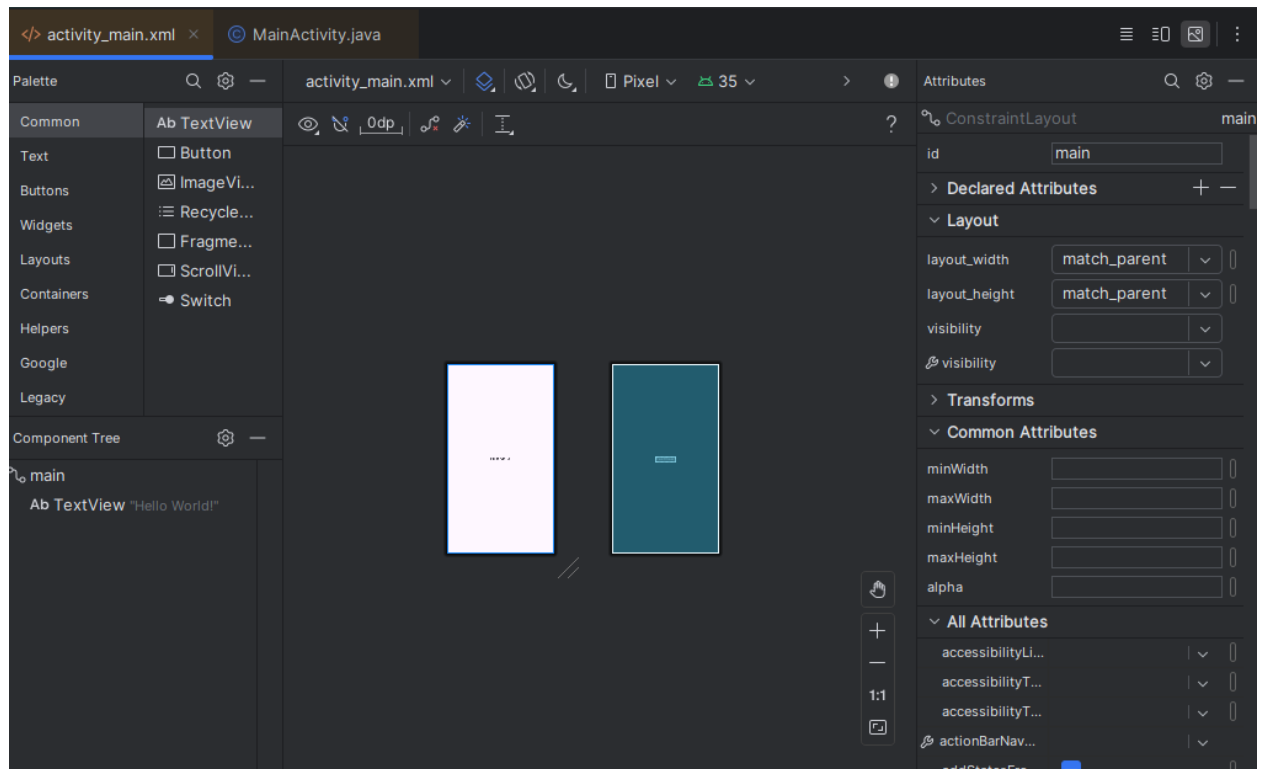
Nếu bạn không có kế hoạch công bố ứng dụng của mình, bạn có thể chấp nhận mặc định. Hãy nhớ rằng việc thay đổi tên gói của ứng dụng sau này là công việc phụ.

6. Chọn ngôn ngữ **Java** và nhấn **Finish**.

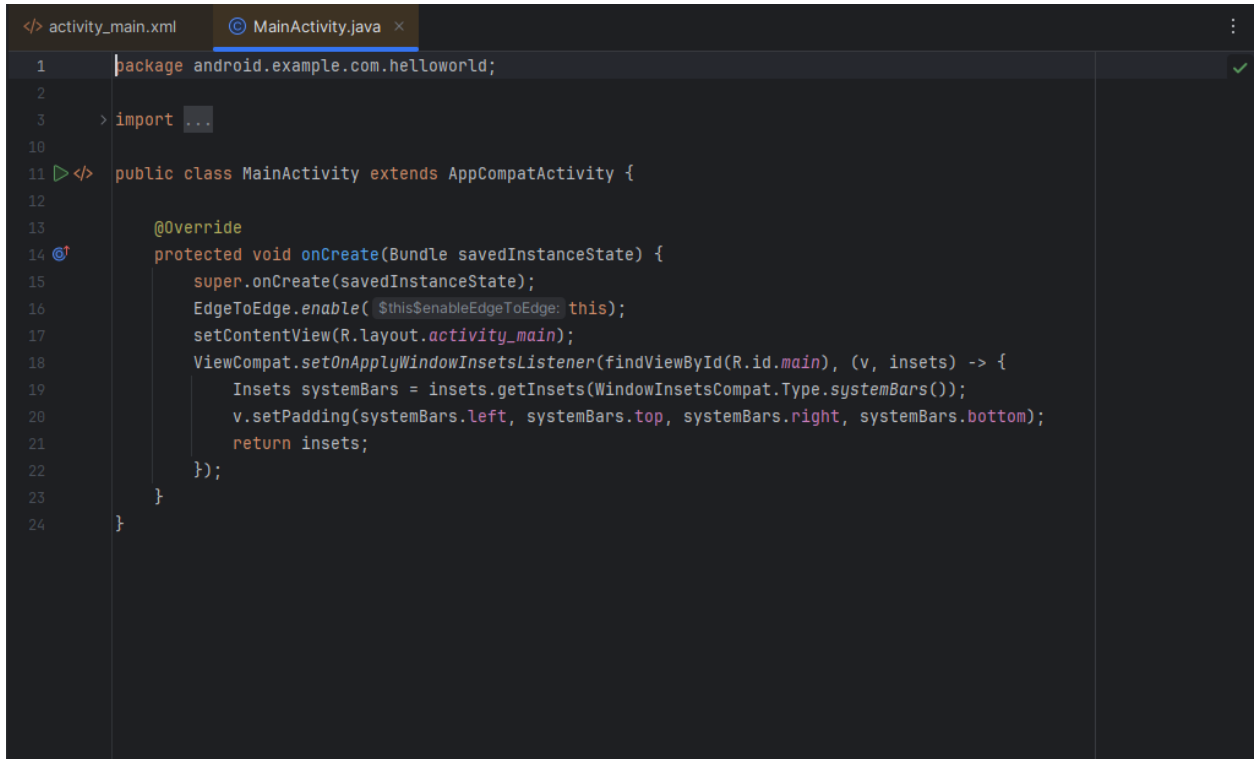
Android Studio tạo ra một thư mục cho các dự án của bạn và xây dựng dự án với Gradle (điều này có thể mất một chút thời gian).

Trình thiết kế Android Studio xuất hiện. Thực hiện theo các bước sau:

1. Nhấn the **activity_main.xml** tab để xem layout editor.
2. Nhấp vào tab Layout Editor Design, nếu chưa được chọn, để hiển thị biểu diễn đồ họa của bố cục như hình bên dưới.



3. Nhấp vào tab **MainActivity.java** để xem trình chỉnh sửa code như hiển thị bên dưới.



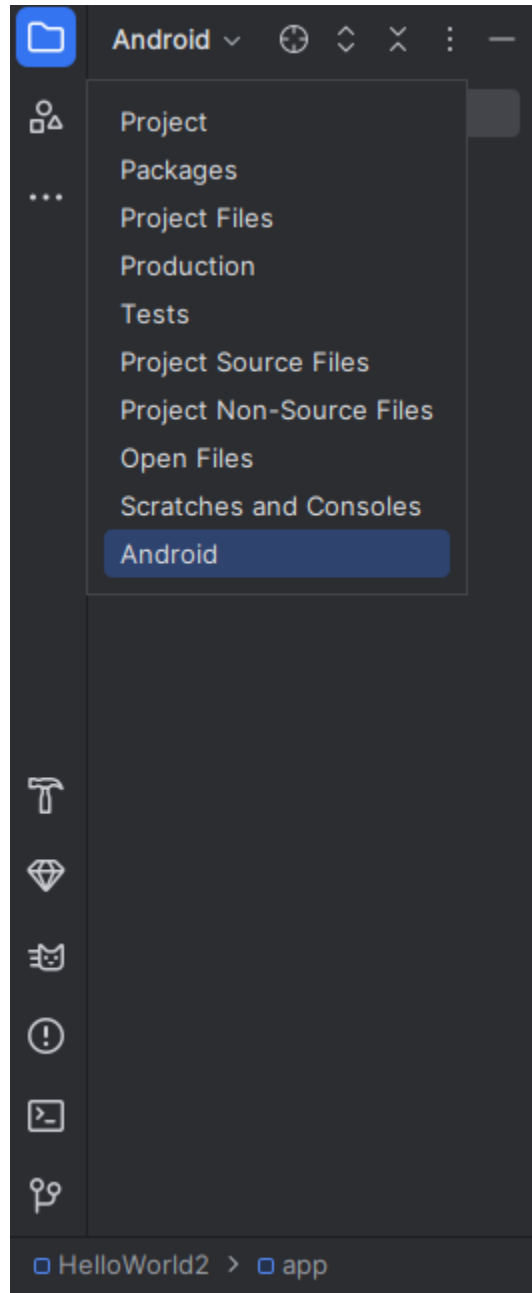
The screenshot shows the Android Studio interface with the **MainActivity.java** file open. The code is as follows:

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

ii. Khám phá dự án > Android pane

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

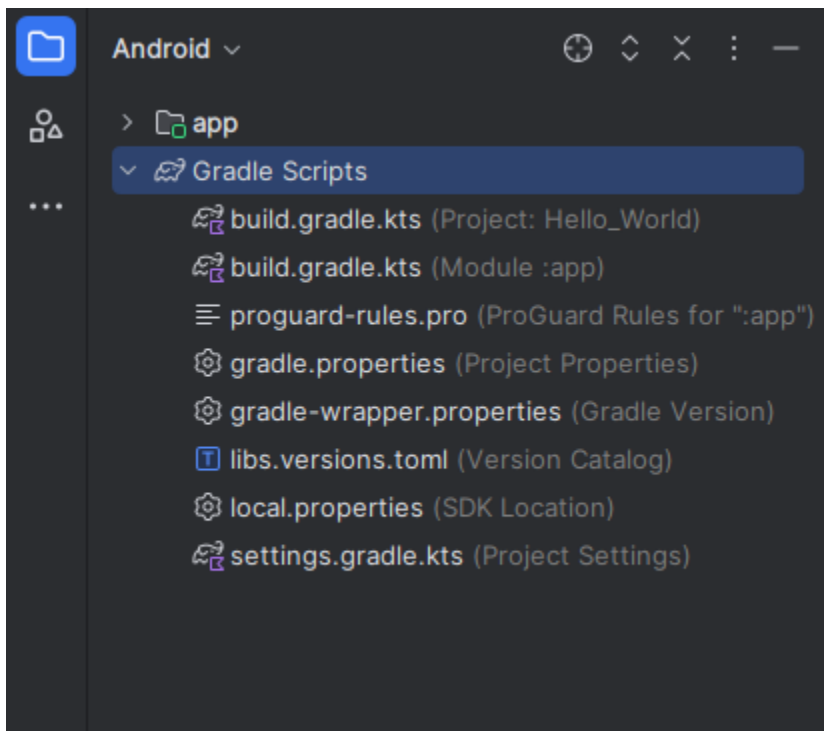
1. Nếu chưa chọn, hãy nhấp vào tab Project trong cột tab dọc ở bên trái cửa sổ Android Studio. Ngăn Dự án xuất hiện.
2. Để xem dự án trong hệ thống phân cấp dự án Android tiêu chuẩn, hãy chọn Android từ menu bật lên ở đầu ngăn Dự án, như được hiển thị bên dưới.



iii. Khám phá thư mục Gradle Scripts

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

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



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

1. Nếu thư mục Tập lệnh Gradle không được mở rộng, hãy nhấp vào hình dấu mũ để mở rộng thư mục đó. Thư mục này chứa tất cả các tệp cần thiết cho hệ thống xây dựng.
2. Tìm tệp **settings.gradle**(Project Settings).

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

Theo mặc định, tệp bản dựng cấp cao nhất sử dụng khối buildscript để xác định kho lưu trữ Gradle và các phần phụ thuộc chung cho tất cả các mô-đun trong dự án. Khi phần phụ thuộc của bạn không phải là thư viện cục bộ hoặc cây tệp, Gradle sẽ tìm kiếm các tệp trong bất kỳ kho lưu trữ trực tuyến nào được chỉ định trong khối kho lưu trữ của

tệp này. Theo mặc định, các dự án Android Studio mới khai báo JCenter và Google (bao gồm kho lưu trữ Google Maven) là vị trí kho lưu trữ:

```
repositories {  
    google {  
        content {  
            includeGroupByRegex( groupRegex: "com\\.android.*")  
            includeGroupByRegex( groupRegex: "com\\.google.*")  
            includeGroupByRegex( groupRegex: "androidx.*")  
        }  
    }  
    mavenCentral()  
    gradlePluginPortal()  
}
```

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

Ngoài tệp build.gradle cấp dự án, mỗi mô-đun có một tệp build.gradle riêng cho phép bạn định cấu hình cài đặt bản dựng cho từng mô-đun cụ thể (ứng dụng HelloWorld chỉ có một mô-đun). Việc định cấu hình các tùy chọn cài đặt bản dựng này cho phép bạn cung cấp các tùy chọn đóng gói tùy chỉnh, chẳng hạn như các loại bản dựng bổ sung và hương vị sản phẩm. Bạn cũng có thể ghi đè các tùy chọn cài đặt trong tệp AndroidManifest.xml hoặc tệp build.gradle cấp cao nhất.

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

```

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

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

    > defaultConfig {...}

    > buildTypes {...}
    compileOptions {
        sourceCompatibility = JavaVersion.VERSION_11
        targetCompatibility = JavaVersion.VERSION_11
    }
}

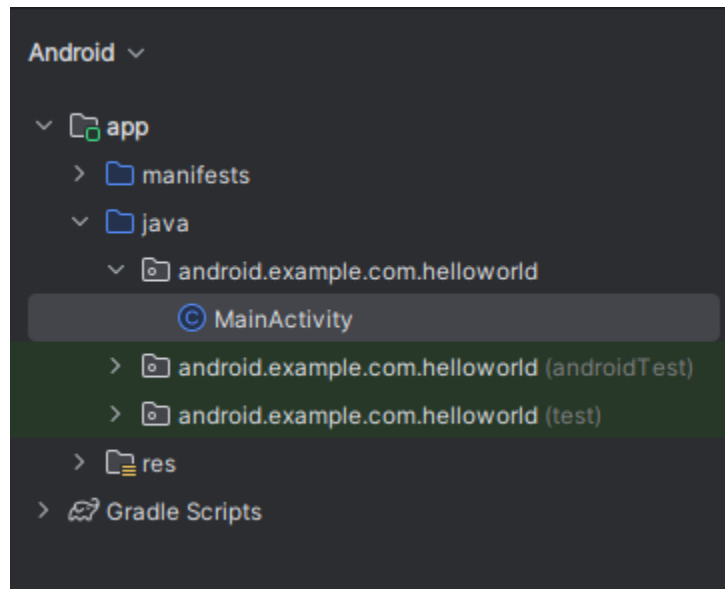
dependencies {
    implementation(libs.appcompat)
    implementation(libs.material)
    implementation(libs.activity)
    implementation(libs.constraintlayout)
    testImplementation(libs.junit)
    androidTestImplementation(libs.ext.junit)
    androidTestImplementation(libs.espresso.core)

```

iv. Khám phá ứng dụng và thư mục res

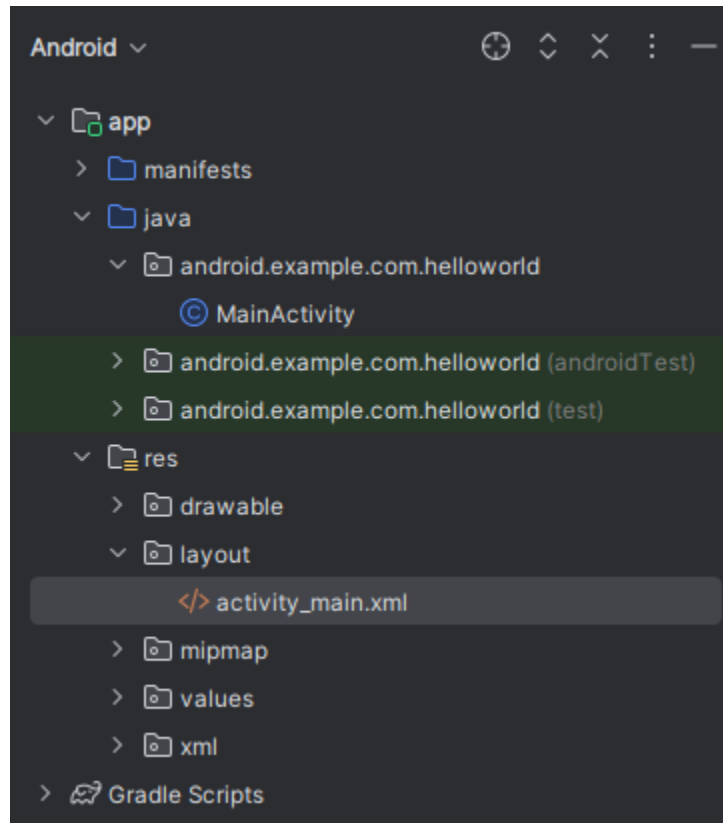
Tất cả mã và tài nguyên cho ứng dụng nằm trong thư mục ứng dụng và res.

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



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

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



Thư mục res chứa các tài nguyên, chẳng hạn như bố cục, chuỗi và hình ảnh. Hoạt động thường được liên kết với bố cục của các chế độ xem giao diện người dùng được xác định dưới dạng tệp XML. Tệp này thường được đặt tên theo hoạt động của nó

v. Khám phá thư mục manifests

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

1. Mở rộng thư mục **manifests**.

2. Mở tệp **AndroidManifest.xml**.

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

c. Sử dụng thiết bị vật lý

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

Những gì bạn cần:

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

i. Bật gỡ lỗi USB

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

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

1. Trên thiết bị của bạn, mở **Cài đặt**, tìm kiếm **Giới thiệu về điện thoại**, nhấp vào Giới thiệu về điện thoại và nhấn vào **Số bản dựng** (building number) bảy lần.
2. Quay lại màn hình trước đó (Cài đặt / Hệ thống). Tùy chọn nhà phát triển xuất hiện trong danh sách. Nhấn vào Tùy chọn nhà phát triển.
3. Chọn **USB Debugging**.

ii. Chạy ứng dụng trên thiết bị

Giờ đây, bạn có thể kết nối thiết bị của mình và chạy ứng dụng từ Android Studio.

1. Connect your device to your development machine with a USB cable.
2. Bấm vào nút Run trên thanh công cụ. Cửa sổ Select Deployment Target sẽ mở ra với danh sách các trình giả lập có sẵn và thiết bị được kết nối.
3. Chọn thiết bị của bạn và nhấp vào OK

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

Troubleshooting

Nếu Android Studio không nhận dạng thiết bị của bạn, hãy thử các 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 tuyên bố thiết bị là "không được phép", hãy làm theo các bước sau:

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

Bạn có thể cần cài đặt trình điều khiển USB thích hợp cho thiết bị của mình. Xem **Using Hardware Devices documentation**.

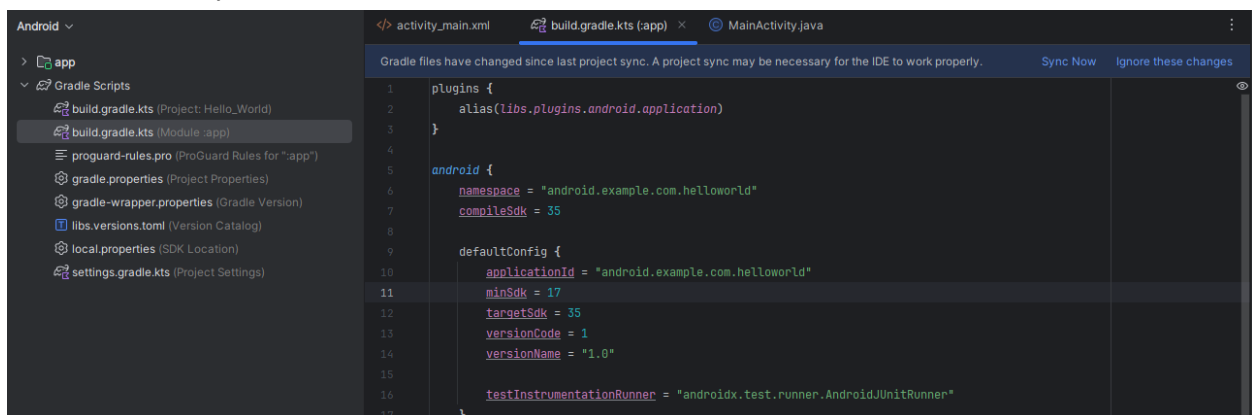
d. Thay đổi cấu hình Gradle của ứng dụng

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

i. Thay đổi phiên bản SDK tối thiểu cho ứng dụng

Các bước:

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 xuất hiện trong trình soạn thảo mã.
2. Trong khối `defaultConfig`, hãy thay đổi giá trị của `minSdkVersion` thành 17 như hình dưới đây (ban đầu nó được đặt thành 15).



Trình chỉnh sửa mã hiển thị thanh thông báo ở trên cùng với liên kết **Sync now**

ii. Đồng bộ hóa cấu hình Gradle mới

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

Để đồng bộ hóa các tệp dự án, hãy bấm vào **Sync now** trong thanh thông báo xuất hiện khi thực hiện thay đổi

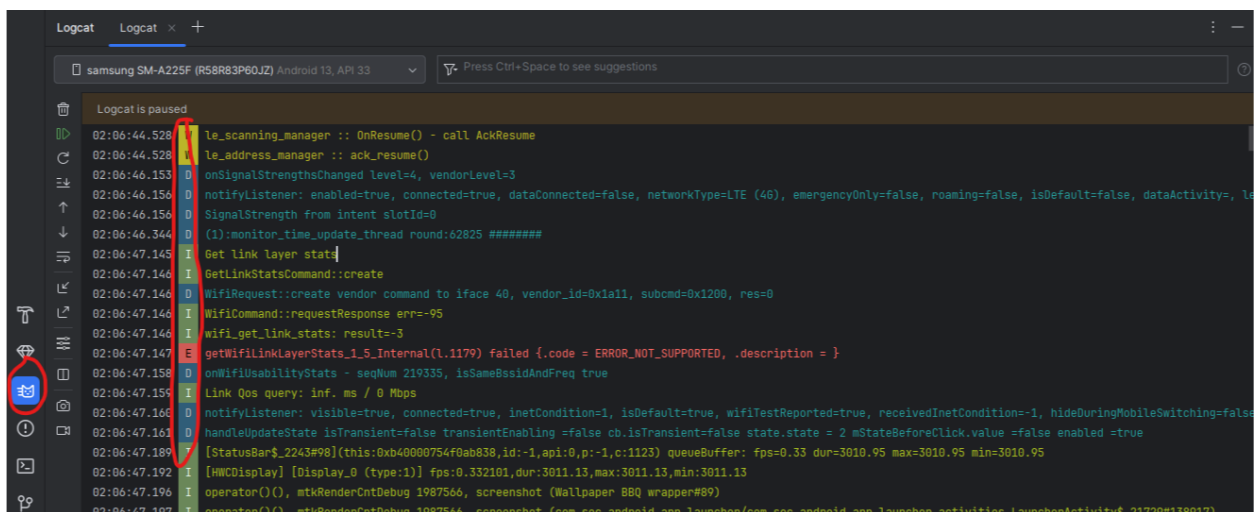
Khi quá trình đồng bộ hóa Gradle kết thúc, thông báo Bản dựng Gradle đã hoàn tất sẽ xuất hiện ở góc dưới cùng bên trái của cửa sổ Android Studio.

e. Thêm câu lệnh nhật ký vào ứng dụng của bạn

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

i. Xem ngăn Logcat

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



Trong hình trên:

1. Tab Logcat để mở và đóng ngăn Logcat, hiển thị thông tin về ứng dụng của bạn khi ứng dụng đang chạy. Nếu bạn thêm câu lệnh Nhật ký vào ứng dụng của mình, thông báo Nhật ký sẽ xuất hiện ở đây.
2. Menu Mức nhật ký được đặt thành Verbose (mặc định), hiển thị tất cả các thông báo Nhật ký. Các cài đặt khác bao gồm **Debug** , **Error** , **Info** , and **Warn** .

ii. Thêm câu lệnh nhật ký vào ứng dụng của bạn

Câu lệnh nhật ký trong mã ứng dụng của bạn sẽ hiển thị thông báo trong ngăn Logcat.

Ví dụ:

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

Các phần của thông điệp là:

- **Log** : Lớp Log để gửi thông báo nhật ký đến ngăn Logcat.
- **d** : Cài đặt mức nhật ký Debug để lọc thông báo nhật ký hiển thị trong ngăn Logcat. Các cấp độ nhật ký khác là **E** cho Error , **W** cho Warn và **I** cho Info .
- **"MainActivity"** : Đối số đầu tiên là một thẻ có thể được sử dụng để lọc tin nhắn trong ngăn Logcat. Đây thường là tên của Hoạt động mà thông điệp bắt đầu. Tuy nhiên, bạn có thể làm cho điều này bất cứ thứ gì hữu ích cho bạn để gỡ lỗi.

Theo quy ước, thẻ nhật ký được định nghĩa là hằng số cho Hoạt động:

```
private static final String LOG_TAG = MainActivity.class.getSimpleName();
```

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

Làm theo các bước sau:

1. Mở ứng dụng Hello World của bạn trong Android studio và mở MainActivity .
2. Để tự động thêm các mục nhập 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 Tùy chọn **Android Studio > Preferences** trong macOS.
3. Chọn **Editor > General > Auto Import**. Chọn tất cả các hộp kiểm và **Insert imports on paste to All**.
4. Nhấp vào **Apply** và sau đó nhấp vào **OK**
5. Trong phương thức onCreate() của MainActivity , thêm câu lệnh sau

```
Log.v("main", "Hello World");
```

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

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable(this);
    setContentView(R.layout.activity_main);
    Log.v( tag: "main", msg: "Hello World");
    ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
        Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
        v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
        return insets;
    });
}
```

6. Nếu ngăn Logcat chưa mở, hãy nhấp vào tab Logcat ở cuối Android Studio để mở.
7. Kiểm tra xem tên của mục tiêu và tên gói của ứng dụng có chính xác không.
8. Thay đổi mức nhật ký trong ngăn Logcat thành Debug (hoặc để nguyên ở chế độ Verbose vì có rất ít thông báo nhật ký).
9. Chạy ứng dụng của bạn.

2025-02-28 07:53:27.297 26270-26270 main android.example.com.helloworld V Hello World

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 views — mọi phần tử của màn hình là một View. Lớp View đại diện cho khối xây dựng cơ bản cho tất cả các thành phần giao UI và lớp cơ sở cho các lớp cung cấp các thành phần UI 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.
- Nút và các yếu tố 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 phần tử View khác và định vị chúng

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

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

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

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

Những gì Bạn sẽ cần:

Bạn nên làm quen với:

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

Những gì bạn sẽ học

- Cách tạo 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 thuật ngữ từ vựng và khái niệm để biết các định nghĩa thân thiện.

Những gì bạn sẽ làm

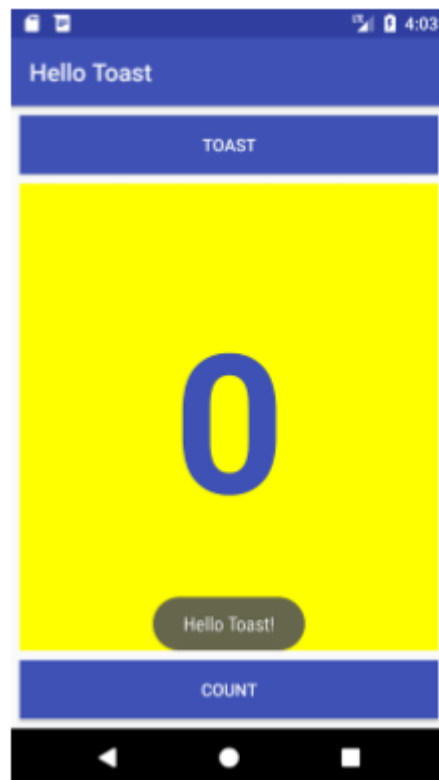
- Tạo một ứng dụng và thêm hai phần tử Button và một TextView vào bố cục.

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

Tổng quan ứng dụng

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

Đây là ứng dụng đã hoàn thành trông như thế nào




a. Tạo và khám phá một dự án mới

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

i. Tạo dự án Android Studio

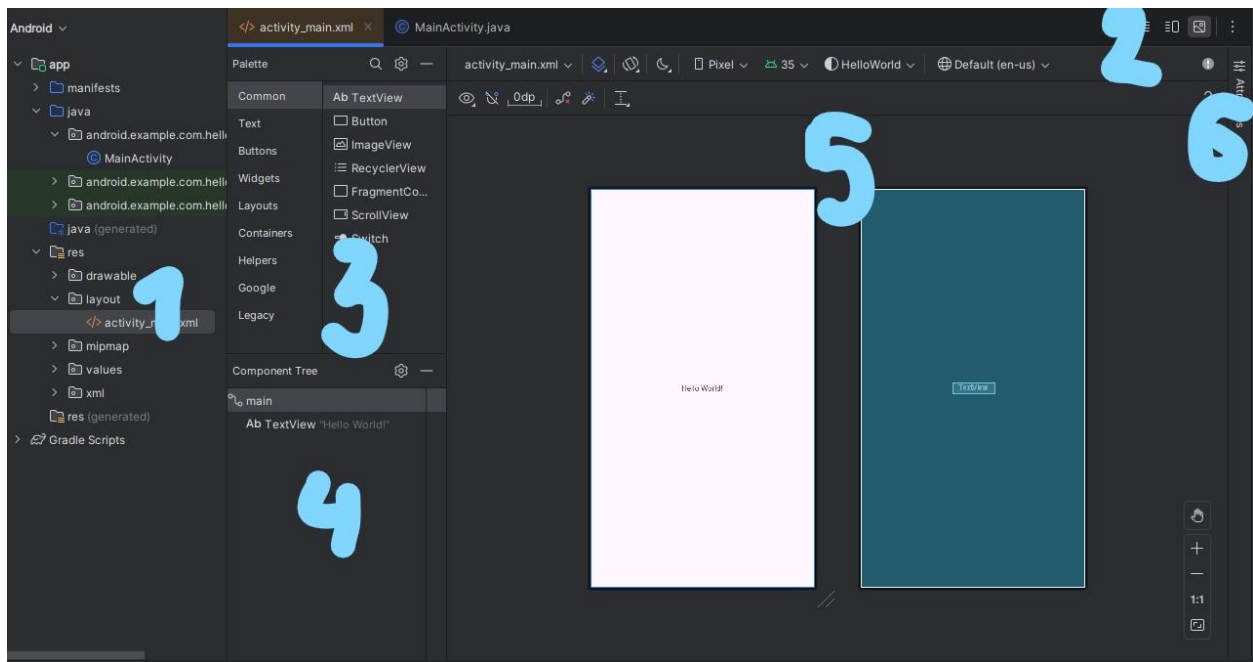
Khởi động Android Studio và tạo một dự án mới với các thông số sau:

Attribute	Value
Application Name	Hello Toast
Company Name	com.example.android(or your own domain)
Phone and Tablet Minimum SDK	API15: Android 4.0.3 IceCreamSandwich
Template	Empty Activity
Generate Layout file box	Selected
Backwards Compatibility box	Selected

Chọn Run > Run app hoặc nhấp vào biểu tượng Run  trên thanh công cụ để tạo và thực thi ứng dụng trên trình mô phỏng hoặc thiết bị của bạn.

ii. Khám phá trình chỉnh sửa bố cục

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



1. Trong app > res > layout trong ngăn Project > Android, nhấp đúp vào tệp activity_main.xml để mở nếu chưa 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. Ngăn Palettes hiển thị các thành phần giao diện người dùng mà bạn có thể sử dụng trong bố cục của ứng dụng.
4. Ngăn Component tree hiển thị hệ thống phân cấp chế độ xem của các phần tử giao diện người dùng. Các phần tử View được tổ chức thành một hệ thống phân cấp cây gồm cha mẹ và con, trong đó con kế thừa các thuộc tính của cha mẹ của nó. Trong hình trên, TextView là con của ConstraintLayout. Các em sẽ học về các yếu tố này ở phần sau của bài học này.
5. Các ngăn thiết kế và bản thiết kế của trình chỉnh sửa bố cục hiển thị các phần tử giao diện người dùng trong bố cục. Trong hình trên, bố cục chỉ hiển thị một phần tử: TextView hiển thị "Hello World".
6. Tab Attributes hiển thị Attributes để đặt thuộc tính cho phần tử giao diện người dùng





b. Thêm các phần tử View vào trình chỉnh sửa bố cục

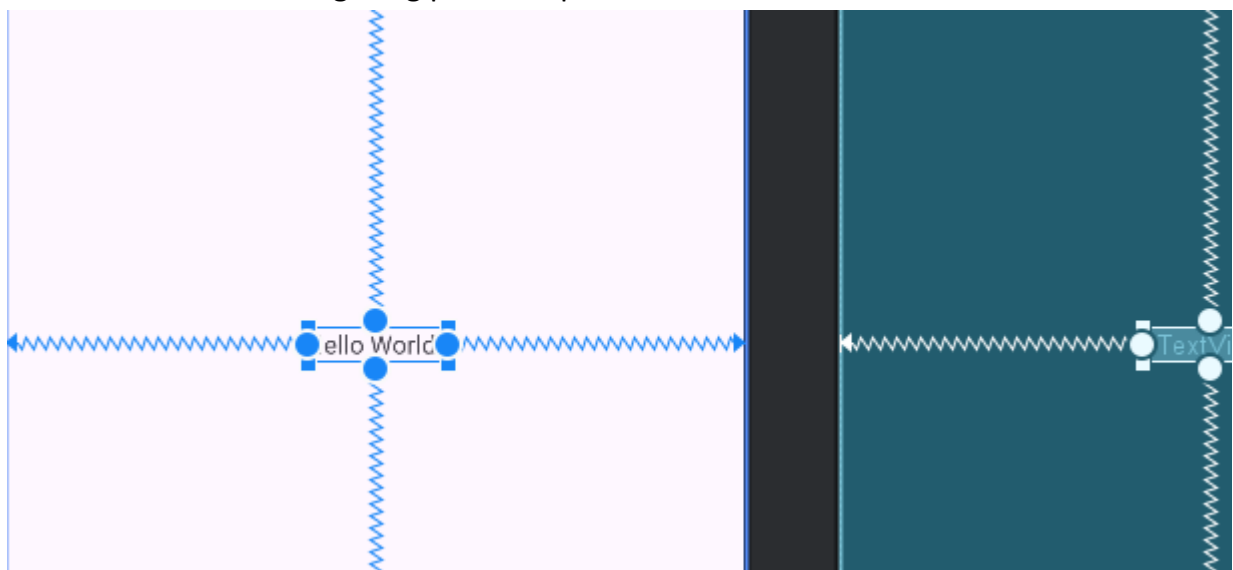
Trong tác vụ này, bạn tạo bố cục giao diện người dùng cho ứng dụng HelloToast trong trình chỉnh sửa bố cục bằng cách sử dụng các tính năng ConstraintLayout. Bạn có thể tạo các

ràng buộc theo cách thủ công, như được hiển thị sau hoặc tự động bằng cách sử dụng công cụ Tự động kết nối

i. Kiểm tra ràng buộc phần tử

Làm theo các bước sau:

1. Mở activity_main.xml từ ngăn Project > Android nếu chưa mở. Nếu tab Thiết kế chưa được chọn, hãy nhấp vào tab đó. Nếu không có bản thiết kế, hãy nhấp vào nút Select Design Surface  trên thanh công cụ và chọn Design + Blueprint .
2. Công cụ Tự động kết nối  cũng nằm trong thanh công cụ. Nó được bật theo mặc định. Đối với bước này, hãy đảm bảo rằng công cụ không bị tắt.
3. Nhấp vào nút phóng to  90%  để phóng to các ngăn thiết kế và bản thiết kế để có cái nhìn cận cảnh.
4. Chọn TextView trong ngăn Component Tree. TextView "Hello World" được đánh dấu trong các ngăn thiết kế và bản thiết kế và các ràng buộc cho phần tử được hiển thị.
5. Tham khảo hình động bên dưới cho bước này. Nhấp vào bộ điều khiển hình tròn ở phía bên phải của TextView để xóa ràng buộc ngang liên kết chế độ xem với phía bên phải của bố cục. TextView nhảy sang bên trái vì nó không còn bị ràng buộc ở phía bên phải nữa. Để thêm lại ràng buộc ngang, hãy nhấp vào cùng một tay cầm và kéo một đường sang phía bên phải của bố cục.



Trong ngăn blueprint hoặc thiết kế, các bộ điều khiển sau xuất hiện trên phần tử TextView:

● **Constraint handle** : Để tạo một ràng buộc như trong hình động ở trên, hãy nhấp vào một điều khiển ràng buộc, được hiển thị dưới dạng một vòng tròn ở bên cạnh của một phần tử. Sau đó, kéo tay cầm đến một tay cầm ràng buộc khác hoặc đến ranh giới mẹ.



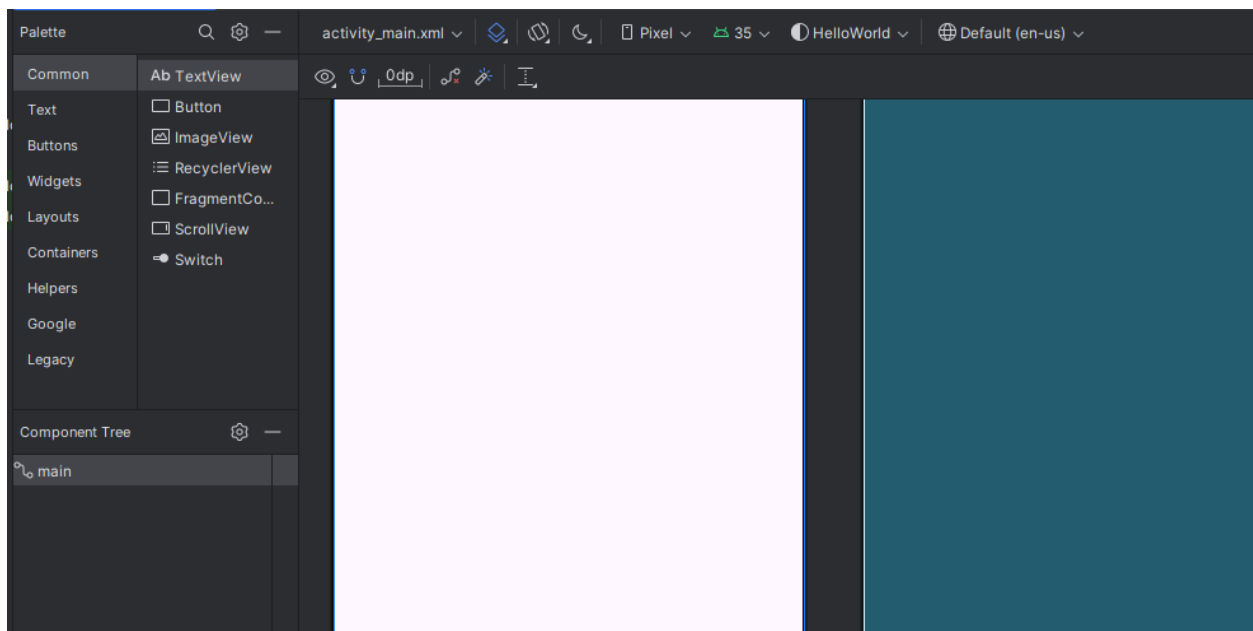
Một đường ngoằn ngoèo đại diện cho ràng buộc.

● **Resizing handle** : Để thay đổi kích thước phần tử, hãy kéo các bộ điều khiển thay đổi kích thước hình vuông. Tay cầm thay đổi thành một góc cạnh trong khi bạn đang kéo nó.



ii. Thêm một nút vào bố cục

Bố cục trống



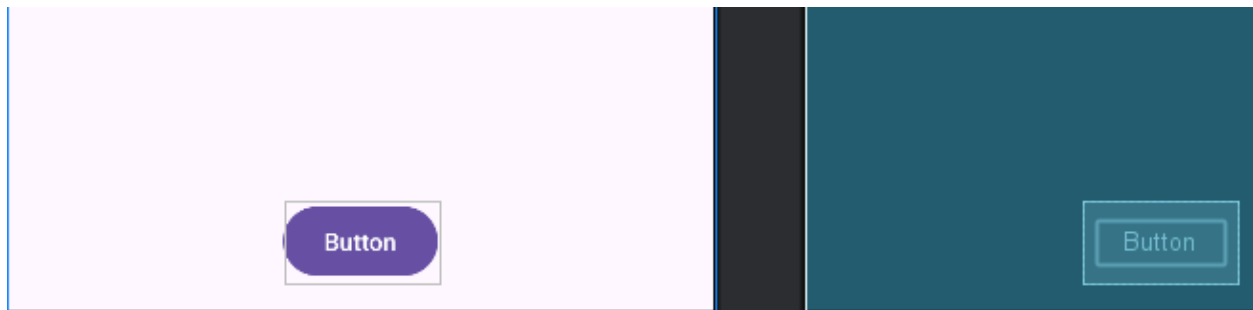
Khi được bật, công cụ Tự động kết nối sẽ tự động tạo hai hoặc nhiều ràng buộc cho một phần tử giao diện người dùng đối với bố cục mẹ. Sau khi bạn kéo phần tử vào bố cục, nó sẽ tạo ra các ràng buộc dựa trên vị trí của phần tử.

Làm theo các bước sau để thêm Nút:

1. Bắt đầu với một bảng sạch. Phần tử TextView là không cần thiết, vì vậy trong khi nó vẫn được chọn, hãy nhấn phím Delete hoặc chọn **Chỉnh sửa > Xóa** . Bây giờ bạn có một bố cục hoàn toàn trống.

2. Kéo một nút từ ngăn Bảng màu đến bất kỳ vị trí nào trong bố cục. Nếu bạn thả Nút ở khu vực giữa trên cùng của bố cục, các ràng buộc có thể tự động xuất hiện. Nếu không, bạn có thể kéo các ràng buộc vào phía trên, bên trái và bên phải của bố cục như trong hình động bên dưới.

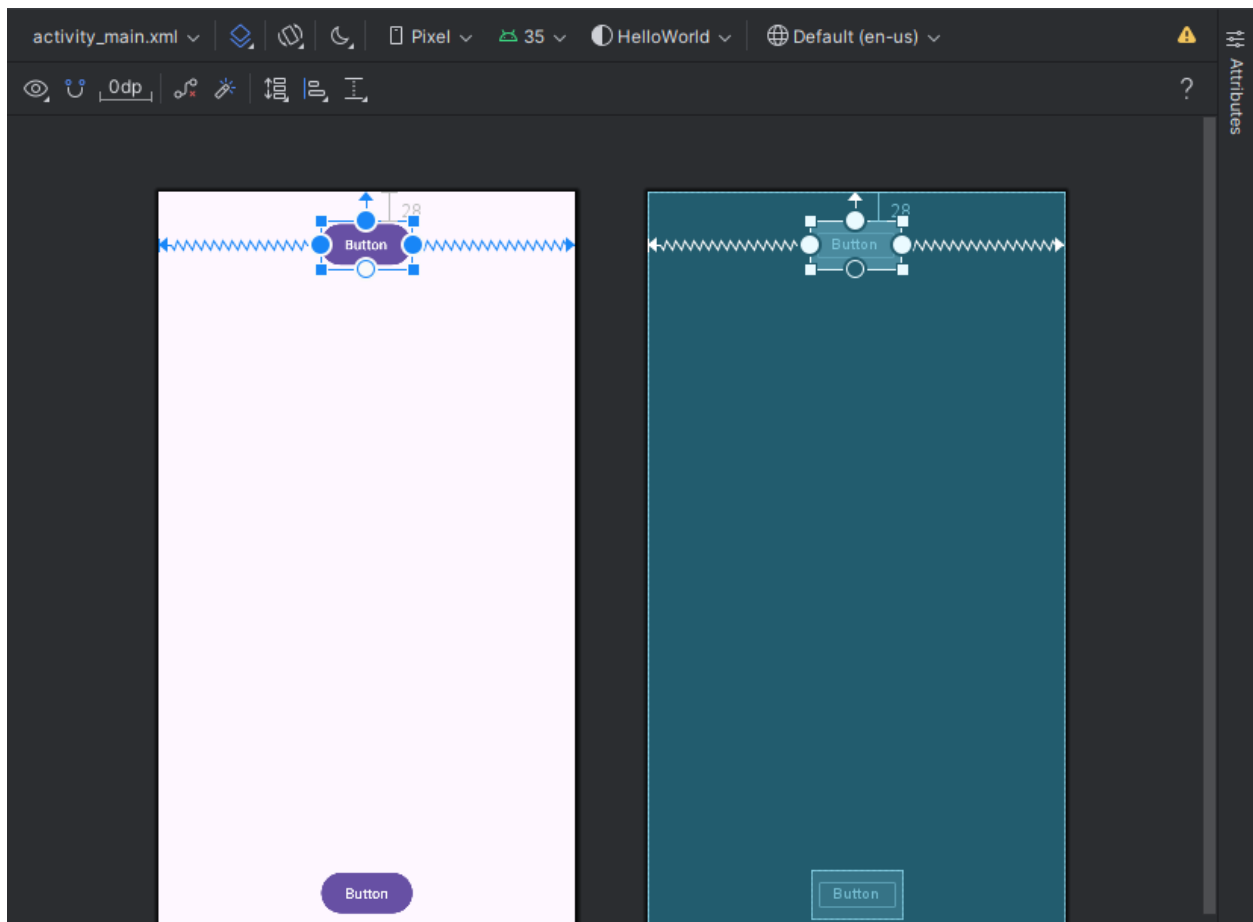
Thêm nút bấm




iii. Thêm nút thứ 2 vào layout

1. Kéo một nút khác từ ngăn Palette vào giữa bố cục như trong hình động bên dưới. Tự động kết nối có thể cung cấp các ràng buộc ngang cho bạn (nếu không, bạn có thể tự kéo chúng).

2. Kéo một ràng buộc dọc xuống cuối bố cục (tham khảo hình bên dưới).



Bạn có thể loại bỏ các ràng buộc khỏi một phần tử bằng cách chọn phần tử và di chuột con trỏ của bạn

trên đó để hiển thị nút Clear Constraints . Nhấp vào nút này để loại bỏ 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. Để xóa tất cả các ràng buộc trong toàn bộ bố cục, hãy nhấp vào công cụ Xóa tất cả các ràng buộc trên thanh công cụ. Công cụ này rất hữu ích nếu bạn muốn làm lại tất cả các ràng buộc trong bố cục của mình.

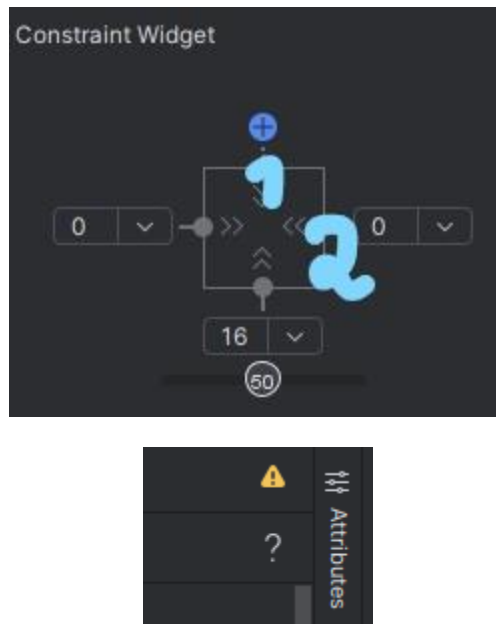
c. Thay đổi thuộc tính của phần tử giao diện người dùng

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

i. Thay đổi kích cỡ nút

Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước ở cả bốn góc của Chế độ xem để bạn có thể thay đổi kích thước Chế độ xem một cách nhanh chóng. Bạn có thể kéo các tay cầm trên mỗi góc của Chế độ xem để thay đổi kích thước, nhưng làm như vậy sẽ mã hóa cứng kích thước chiều rộng và chiều cao. Tránh mã hóa cứng kích thước cho hầu hết các thành phần Chế độ xem vì kích thước được mã hóa cứng không thể thích ứng với các nội dung và kích thước màn hình khác nhau.

Thay vào đó, hãy sử dụng ngăn Thuộc tính ở phía bên phải của trình soạn thảo bố cục để chọn chế độ định cỡ không sử dụng kích thước được mã hóa cứng. Ngăn Thuộc tính bao gồm một bảng điều chỉnh kích thước hình vuông được gọi là trình kiểm tra chế độ xem ở trên cùng. Các ký hiệu bên trong hình vuông đại diện cho cài đặt chiều cao và chiều rộng như sau:



Trong hình trên:

1. Kiểm soát chiều cao. Điều khiển này chỉ định thuộc tính `layout_height` và xuất hiện trong hai phân đoạn ở cạnh trên và dưới của hình vuông. Các góc chỉ ra rằng điều khiển này được đặt thành `wrap_content`, có nghĩa là Chế độ xem sẽ mở rộng theo chiều dọc khi cần thiết để phù hợp với nội dung của nó. Dấu "8" cho biết ký quỹ tiêu chuẩn được đặt thành 8dp.
2. Kiểm soát chiều rộng. Điều khiển này chỉ định `layout_width` và xuất hiện trong hai phân đoạn ở bên trái và bên phải của hình vuông. Các góc cho biết rằng điều khiển này

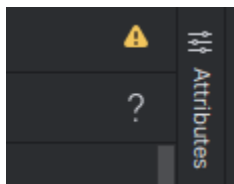
được đặt thành `wrap_content` có nghĩa là Chế độ xem sẽ mở rộng theo chiều ngang khi cần thiết để phù hợp với nội dung của nó, lên đến biên độ 8dp.

3. Nút đóng ngăn Thuộc tính. Bấm để đóng ngăn.

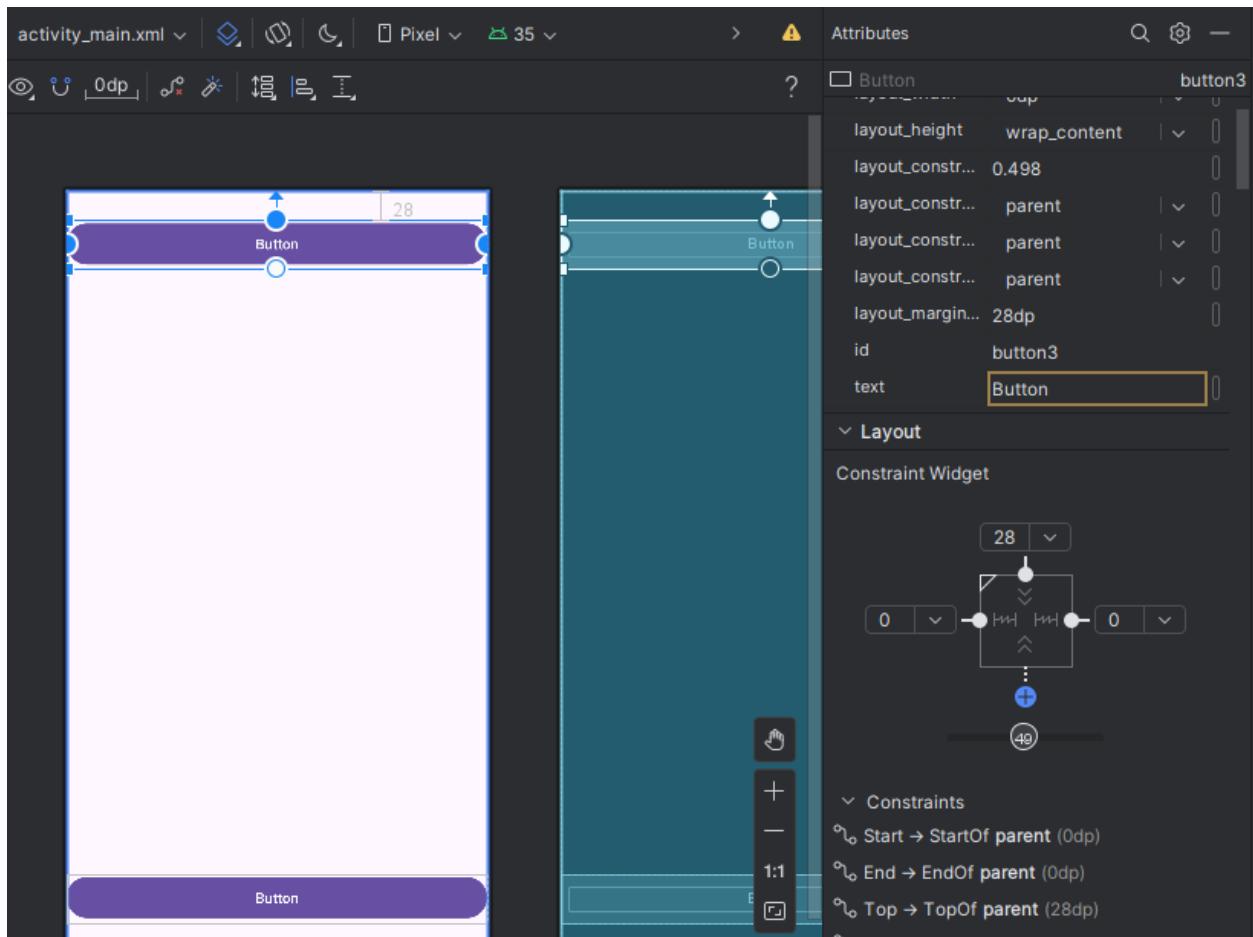
Thay đổi kích cỡ nút

Làm theo các bước sau:

1. Chọn nút trên cùng trong ngăn Cây thành phần.
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 thay đổi nó thành Cố định với các đường thẳng và lần nhấp thứ hai thay đổi nó thành Match Constraints với những đường uốn lượn như được hiển thị trong hình động bên dưới.



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

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

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

- Cài đặt `match_constraint` mở rộng phần tử View để lấp đầy phần tử cha của nó theo chiều rộng hoặc chiều cao — lên đến lề, nếu được đặt. Cha trong trường hợp này là `ConstraintLayout`. Bạn tìm hiểu thêm về `ConstraintLayout` trong nhiệm vụ tiếp theo.
- Cài đặt `wrap_content` thu nhỏ kích thước của phần tử View để nó vừa đủ lớn để bao bọc nội dung của nó. Nếu không có nội dung, phần tử View sẽ trở nên vô hình.
- Để chỉ định kích thước cố định điều chỉnh cho kích thước màn hình của thiết bị, hãy sử dụng một số pixel cố định không phụ thuộc vào mật độ (đơn vị dp). Ví dụ: 16dp có nghĩa là 16 pixel không phụ thuộc vào mật độ.

ii. Thay đổi thuộc tính Nút

Để xác định mỗi Chế độ xem duy nhất trong bố cục Hoạt động, mỗi lớp con Chế độ xem hoặc Chế độ xem (chẳng hạn như Nút) cần có một ID duy nhất. Và để có bất kỳ công dụng nào, các phần tử Button cần văn bản. Các thành phần chế độ xem cũng có thể có hình nền có thể là màu sắc hoặc hình ảnh.

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

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

1. Sau khi chọn Nút đầu tiên, hãy chỉnh sửa trường ID ở đầu ngăn Thuộc tính để `button_toast` thuộc tính `android:id`, thuộc tính này được sử dụng để xác định phần tử trong bố cục.
2. Đặt thuộc tính `background` thành `@color/colorPrimary`. (Khi bạn nhập `@c`, các lựa chọn sẽ xuất hiện để dễ dàng lựa chọn.)
3. Đặt thuộc tính `textColor` thành `@android:color/white`.
4. Chỉnh sửa thuộc tính `text` thành `Toast`.
5. Thực hiện các thay đổi thuộc tính tương tự cho Nút thứ hai, sử dụng `button_count` làm ID, Đếm cho thuộc tính văn bản và cùng màu cho nền và văn bản như các bước trước.

`colorPrimary` là màu chính của chủ đề, một trong những màu cơ sở chủ đề được xác định trước được xác định trong tệp tài nguyên `colors.xml`. Nó được sử dụng cho thanh ứng dụng. Sử dụng màu cơ bản cho các yếu tố giao diện người dùng khác sẽ tạo ra một giao

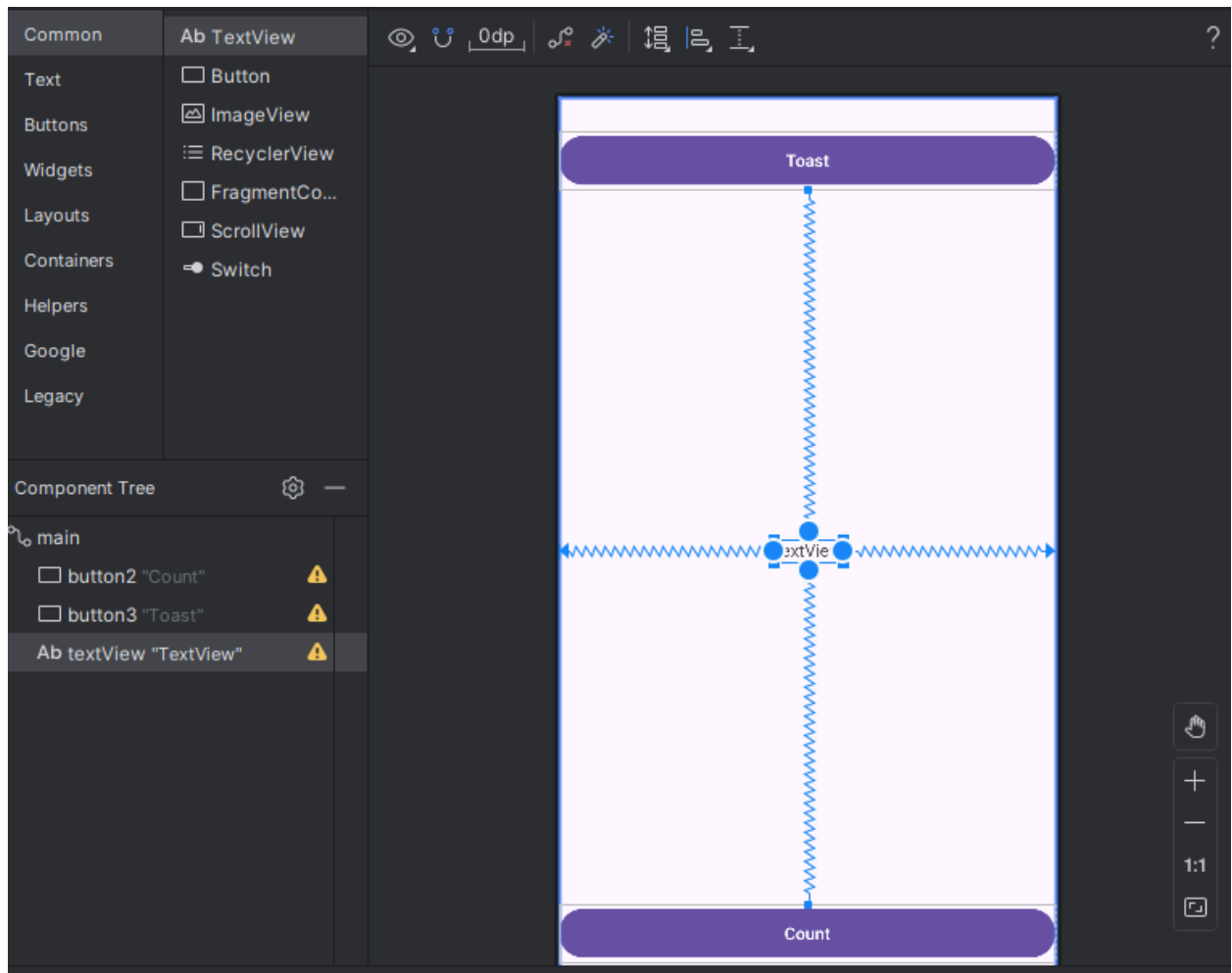
diện người dùng thống nhất. Bạn sẽ tìm hiểu thêm về chủ đề ứng dụng và Material Design trong một bài học khác.

d. Thêm một `TextEdit` và đặt thuộc tính cho nó

Một trong những lợi ích của `ConstraintLayout` là khả năng căn chỉnh hoặc hạn chế các phần tử liên quan đến các phần tử khác. Trong nhiệm vụ này, bạn sẽ thêm một `TextView` ở giữa bố cục và hạn chế nó theo chiều ngang vào lề và theo chiều dọc vào hai phần tử `Button`. Sau đó, bạn sẽ thay đổi các thuộc tính cho `TextView` trong ngăn `Attributes` (Thuộc tính).

i. Thêm TextView và các ràng buộc

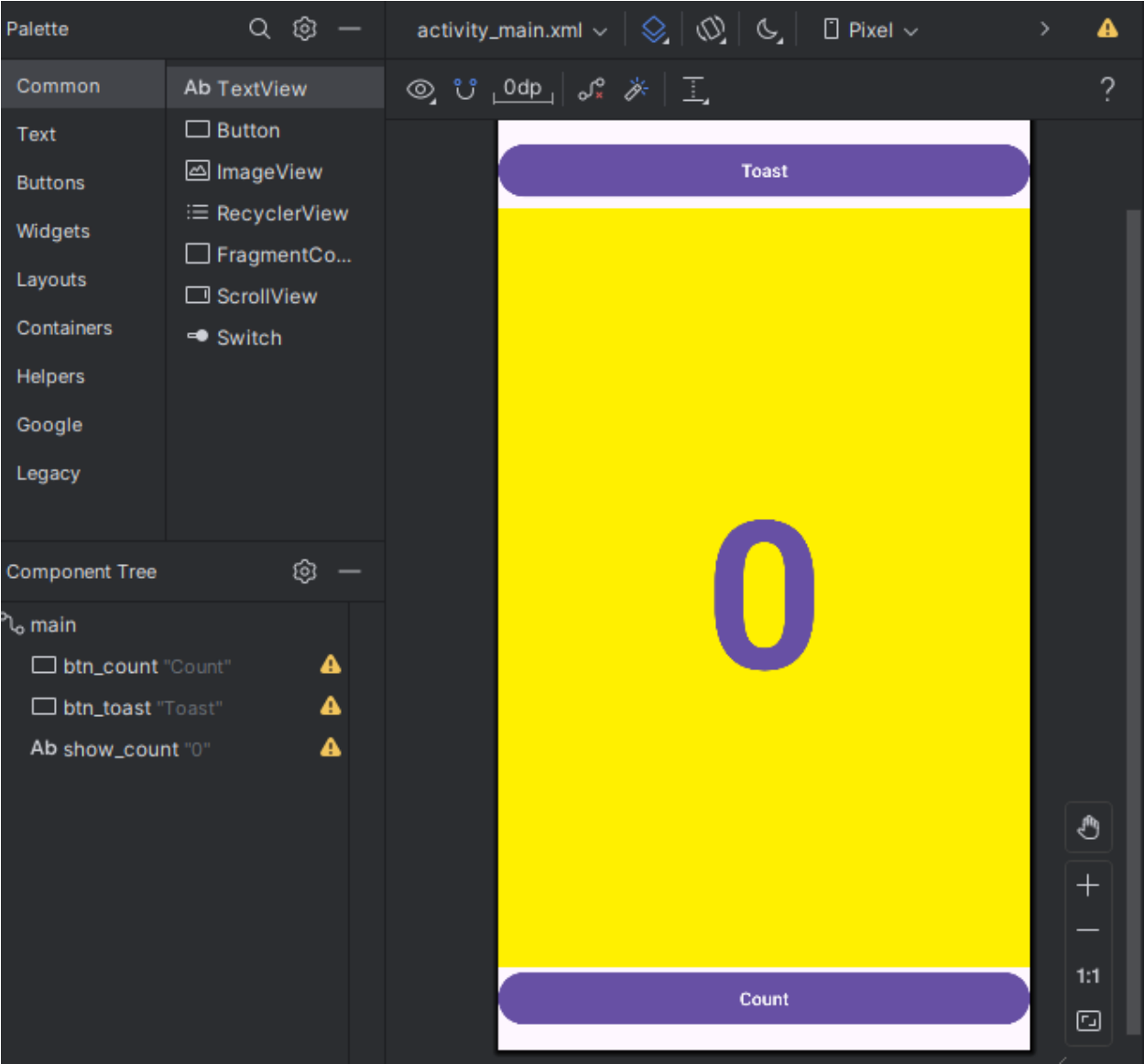
1. Như được hiển thị trong hình động bên dưới, kéo TextView từ ngăn Palette đến phần trên của bố cục và kéo một ràng buộc từ trên cùng của TextView đến tay cầm ở dưới cùng của Nút Toast . Điều này hạn chế TextView nằm bên dưới Button .
2. Như trong hình động bên dưới, hãy kéo một ràng buộc từ cuối TextView đến tay cầm ở trên cùng của Nút đếm và từ hai bên của TextView đến hai bên của bố cục. Điều này hạn chế TextView ở giữa bố cục giữa hai phần tử Button.




ii. Thay đổi thuộc tính TextView

With the TextView selected, open the Attributes pane, if it is not already open. Set attributes for the TextView as shown in the animated figure below. The attributes you haven't encountered yet are explained after the figure:

1. Đặt ID thành `show_count` .
2. Đặt văn bản thành `0` .
3. Đặt `textSize` thành `160sp` .
4. Đặt `textStyle` thành `B` (in đậm) và `textAlignment` thành `ALIGNCENTER` (căn giữa đoạn văn).
5. Thay đổi các điều khiển kích thước chế độ xem ngang và dọc (`layout_width` và `layout_height`) thành `match_constraint` .
6. Đặt `textColor` thành `@color/colorPrimary` .
7. Cuộn xuống ngăn và nhấp vào **View all attributes**, cuộn xuống trang thứ hai của các thuộc tính sang nền , sau đó nhập **#FFF00** cho màu vàng.
8. Cuộn xuống trọng lực , mở rộng trọng lực và chọn **center_ver** (đối với trung tâm-dọc).



- `textSize` : Kích thước văn bản của `TextView` . Đối với bài học này, kích thước được đặt thành 160sp . sp là viết tắt của pixel-independent scale, và giống như dp, là một đơn vị chia tỷ lệ 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 cho cả mật độ màn hình và sở thích 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).
- trọng lực : Thuộc tính trọng lực chỉ định cách một Chế độ xem được căn chỉnh trong Chế độ xem hoặc Nhóm chế độ xem cha của nó. Trong bước này, bạn căn giữa `TextView` theo chiều dọc trong `ConstraintLayout`

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

e. Thiết kế bố cục trong XML

Bố cục 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 Cây thành phần. Di con trỏ của bạn qua các dấu chấm than này để xem thông báo cảnh báo, như được hiển thị bên dưới. Cảnh báo tương tự xuất hiện cho cả ba phần tử: chuỗi được mã hóa cứng nên sử dụng tài nguyên.

Cách dễ nhất để khắc phục sự cố bố cục là chỉnh sửa bố cục trong XML. Mặc dù trình soạn thảo bố cục là một công cụ mạnh mẽ, nhưng một số thay đổi dễ dàng thực hiện trực tiếp trong mã nguồn XML.

i. Mở mã XML cho bố cục

Đối với tác vụ này, hãy mở tệp `activity_main.xml` nếu nó chưa được mở và nhấp vào tab Văn bản ở góc trên bên phải trình chỉnh sửa bố cục.

Trình soạn thảo XML xuất hiện, thay thế các ngăn thiết kế và bản thiết kế. Như bạn có thể thấy trong hình bên dưới, cho thấy một phần của mã XML cho bố cục, các cảnh báo được đánh dấu — các chuỗi được mã hóa cứng "Toast" và "Count" . (Mã hóa cứng "0" cũng

được đánh dấu nhưng không được hiển thị trong hình.) Di con trỏ của bạn lên chuỗi được mã hóa cứng "Toast" để xem thông báo cảnh báo.

```
2      <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
20      <Button
22          android:layout_width="0dp"
23          android:layout_height="wrap_content"
24          android:layout_marginTop="28dp"
25          android:text="Toast"
26          app:layout_constraintEnd_toEndOf="parent"
27          app:layout_constraintHorizontal_bias="0.498"
28          app:layout_constraintStart_toStartOf="parent"
29          app:layout_constraintTop_toTopOf="parent" />
30
31      <TextView
32          android:id="@+id/show_count"
33          android:layout_width="0dp"
34          android:layout_height="0dp"
35          android:layout_marginTop="5dp"
36          android:background="#FFF000"
37          android:gravity="center_horizontal"
38          android:text="0"
39          android:textAlignment="center"
40          android:textColor="?attr/colorPrimary"
41          android:textSize="160sp"
42          android:textStyle="bold"
43          app:layout_constraintBottom_toTopOf="@id/btn_count"
44          app:layout_constraintEnd_toEndOf="parent"
45          app:layout_constraintStart_toStartOf="parent"
46          app:layout_constraintTop_toBottomOf="@id/btn_toast" />
47  </androidx.constraintlayout.widget.ConstraintLayout>
```

ii. Trích xuất tài nguyên chuỗi

Instead of hard-coding strings, it is a best practice to use string resources, which represent the strings. Having the strings in a separate file makes it easier to manage them, especially if you use these strings more than once. Also, string resources are mandatory for translating and localizing your app, because you need to create a string resource file for each language.

1. Nhấp một lần vào từ "Toast" (cảnh báo được đánh dấu đầu tiên).
2. Nhấn **Alt-Enter** trong Windows hoặc **Option-Enter** trong macOS và **Extract string resource** từ menu bật lên.
3. Nhập **button_label_toast** cho **Resource name**.
4. Nhấp vào **OK**. Tài nguyên chuỗi được tạo trong tệp values/res/string.xml và chuỗi trong mã của bạn 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 ngăn **Project > Android**, hãy mở rộng các **value** trong **res**, sau đó nhấp đúp vào **strings.xml** để xem tài nguyên chuỗi của bạn trong tệp strings.xml:

```
<resources>
  <string name="app_name">Hello World</string>
  <string name="button_label_toast">Toast</string>
  <string name="button_label_count">Count</string>
  <string name="count_initial_value">0</string>
</resources>
```

7. Bạn cần một chuỗi khác để sử dụng trong tác vụ tiếp theo hiển thị thông báo. Thêm vào tệp strings.xml một tài nguyên chuỗi khác có tên toast_message cho cụm từ "Hello Toast!":

```
<resources>
  <string name="app_name">Hello World</string>
  <string name="button_label_toast">Toast</string>
  <string name="button_label_count">Count</string>
  <string name="count_initial_value">0</string>
  <string name="toast_message">Hello Toast!</string>
</resources>
```

f. Thêm bộ điều khiển onClick cho nút

Trong tác vụ này, bạn thêm một phương thức Java cho mỗi Button trong MainActivity thực thi khi người dùng nhấn vào Button

i. Thêm thuộc tính onClick và bộ điều khiển cho mỗi nút

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

1. Với trình chỉnh sửa XML đang mở (tab Văn bản), hãy tìm Nút với 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 chỉ báo kết thúc />:
3. Nhấp vào biểu tượng bóng đèn màu đỏ xuất hiện bên cạnh thuộc tính. Chọn 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 Tạo 'showToast(view)' trong MainActivity và nhấp vào OK . Hành động này tạo sơ khai phương thức 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 nút button_count : Thêm thuộc tính android:onClick vào cuối và thêm trình xử lý nhấp chuột:

```
android:onClick="showToast" />
```

```
android:onClick="countUp"/>
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/main"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity" >

    <Button
        android:id="@+id/btn_count"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="16dp"
        android:text="@string/button_label_count"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        android:onClick="countUp"/>

    <Button
        android:id="@+id/btn_toast"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:text="@string/button_label_toast"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        android:onClick="showToast" />

    <TextView
```



```

        android:id="@+id/show_count"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginTop="5dp"
        android:background="#FFF000"
        android:gravity="center_horizontal"
        android:text="@string/count_initial_value"
        android:textAlignment="center"
        android:textColor="?attr/colorPrimary"
        android:textSize="160sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toTopOf="@id/btn_count"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/btn_toast" />
</androidx.constraintlayout.widget.ConstraintLayout>

```

Mở tệp MainActivity.java

5. Nếu MainActivity.java chưa mở, hãy mở rộng java trong chế độ xem Project > Android, mở rộng com.example.android.hellotoast , sau đó nhấp đúp vào MainActivity . Trình chỉnh sửa mã xuất hiện cùng với mã trong MainActivity :

```

package android.example.com.helloworld;

import android.os.Bundle;
import android.util.Log;
import android.view.View;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        Log.v("main", "Hello World");
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets)
-> {
            Insets systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom);
            return insets;
        });
    }

    public void showToast(View view) {
    }

    public void countUp(View view) {
    }
}

```

ii. Thiết kế bộ điều khiển cho Toast Button

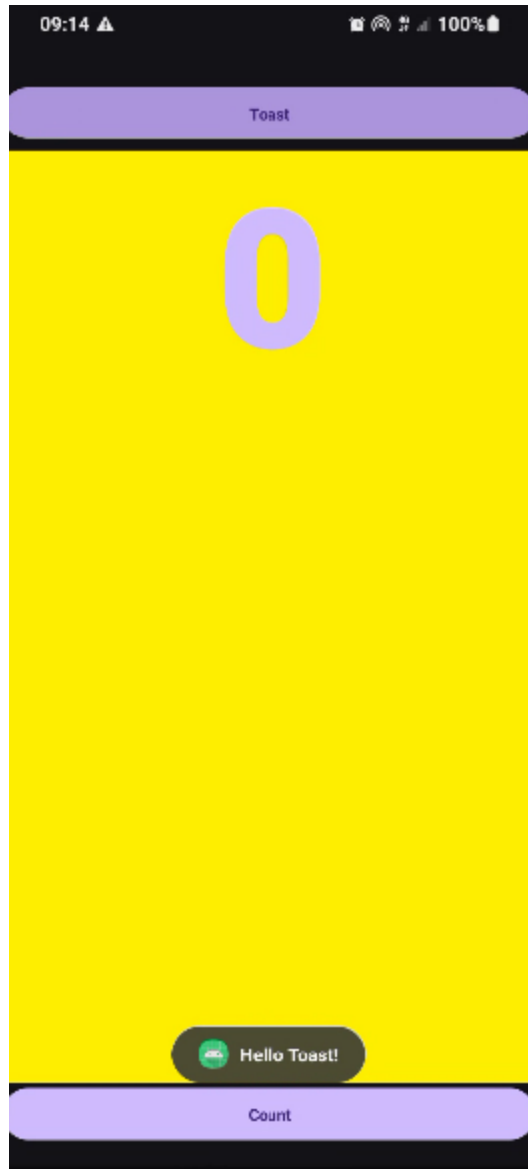
Bây giờ bạn sẽ chỉnh sửa phương thức `showToast()` — trình xử lý nhấp chuột Nút Toast trong `MainActivity` — để nó hiển thị một thông báo. Toast cung cấp một cách để hiển thị một thông báo đơn giản trong một cửa sổ bật lên nhỏ. Nó chỉ lấp đầy dung lượng cần thiết cho tin nhắn. Hoạt động hiện tại vẫn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

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

1. Xác định vị trí phương thức `showToast()` mới tạo.
2. Để tạo một thực thể của Toast, hãy gọi phương thức `factory makeText()` trên lớp Toast
3. Cung cấp ngữ cảnh của ứng dụng Hoạt động . Vì Toast hiển thị trên đầu giao diện người dùng Hoạt động, hệ thống cần thông tin về Hoạt động hiện tại. Khi bạn đã ở trong ngữ cảnh của Hoạt động mà bạn cần, hãy sử dụng nó như một phím tắt.
4. Cung cấp thông báo để hiển thị, chẳng hạn như tài nguyên chuỗi (`toast_message` bạn đã tạo ở bước trước). Tài nguyên chuỗi `toast_message` được xác định bởi `R.string` .
5. Cung cấp thời lượng cho màn hình. Ví dụ: `Toast.LENGTH_SHORT` hiển thị bánh mì nướng trong một thời gian tương đối ngắn
Thời lượng của màn hình Toast có thể là `Toast.LENGTH_LONG` hoặc `Toast.LENGTH_SHORT` . Độ dài thực tế là khoảng 3.5 giây đối với Bánh mì nướng dài và 2 giây đối với Bánh mì nướng ngắn.
6. Hiển thị Toast bằng cách gọi `show()` . Sau đây là toàn bộ phương thức `showToast()`:

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

Chạy chương trình



iii. Thiết kế bộ điều khiển cho Count Button

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

Mã cho trình xử lý phải:

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

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

1. Xác định vị trí phương thức `countUp()` mới tạo.
2. Để theo dõi số lượng, bạn cần một biến thành viên riêng. Mỗi lần nhấn vào nút Đếm sẽ tăng giá trị của biến này. Nhập thông tin sau, sẽ được đánh dấu bằng màu đỏ và hiển thị biểu tượng bóng đèn màu đỏ:

Nếu biểu tượng bóng đèn màu đỏ không xuất hiện, hãy chọn biểu thức `mCount++`. Bóng đèn màu đỏ cuối cùng cũng xuất hiện.

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

3. Nhấp vào biểu tượng bóng đèn màu đỏ và chọn **Create field 'mCount'** từ menu bật lên. Thao tác này sẽ tạo ra một biến thành viên riêng từ ở đầu MainActivity và Android Studio giả định rằng bạn muốn biến đó là một số nguyên (`int`):
4. Thay đổi câu lệnh biến thành viên riêng để khởi tạo biến về không:
5. Cùng với biến trên, bạn cũng cần một biến thành viên riêng từ để tham chiếu `show_count TextView` , bạn sẽ thêm vào trình xử lý nhấp chuột. Gọi biến này là `mShowCount`
6. Bây giờ bạn đã có `mShowCount` , bạn có thể lấy tham chiếu đến `TextView` bằng cách sử dụng ID bạn đã đặt trong tệp bố cục. Để có được tham chiếu này chỉ một lần, hãy chỉ định nó trong phương thức `onCreate()`. Như bạn đã học trong một bài học khác, phương thức `onCreate()` được sử dụng để thổi phồng 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 thành phần giao diện người dùng khác trong bố cục, chẳng hạn như `TextView` . Xác định vị trí phương thức `onCreate()` trong MainActivity:

```
public class MainActivity extends AppCompatActivity {  
  
    private int mCount = 0;  
    private TextView mShowCount;  

```

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

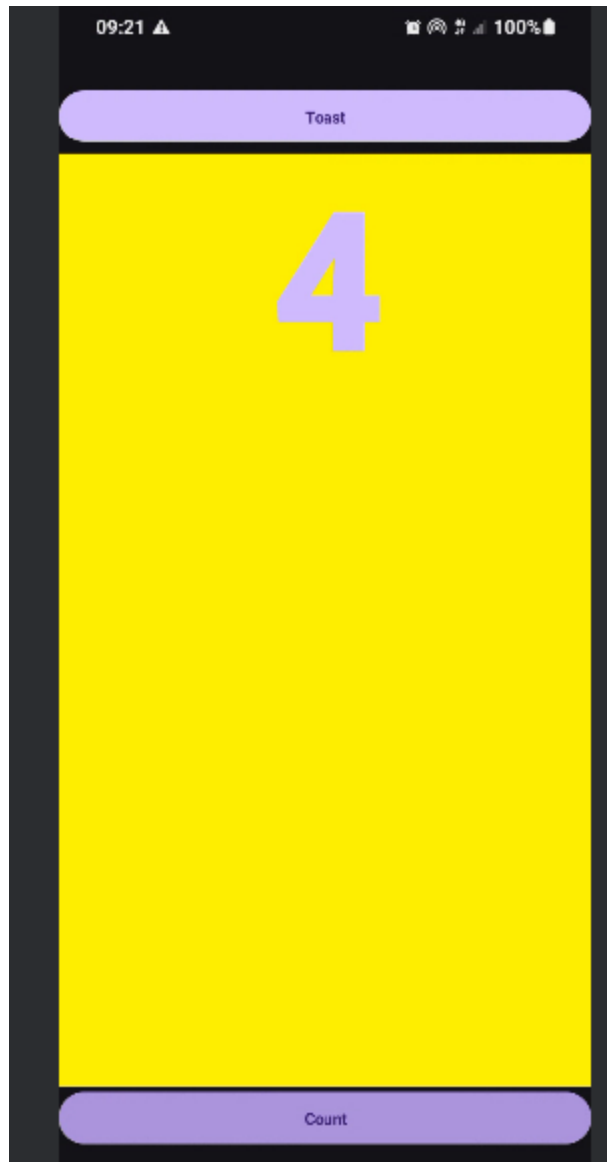
```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    EdgeToEdge.enable(this);  
    setContentView(R.layout.activity_main);  
    mShowCount = (TextView) findViewById(R.id.show_count);  
}
```

View , giống như một chuỗi, là một tài nguyên có thể có id. Lệnh gọi findViewById lấy ID của một chế độ xem làm tham số của nó và trả về View . Bởi vì phương thức trả về một View, bạn phải truyền kết quả sang loại view mà bạn mong đợi, trong trường hợp này là (TextView).

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

```
public void countUp(View view) {  
    mCount++;  
    if (mShowCount != null)  
        mShowCount.setText(Integer.toString(mCount));  
}
```

9. Chạy ứng dụng để xác minh rằng số lượng tăng lên khi bạn chạm vào nút **Count**.



1.3) Trình chỉnh sửa bố cục

Giới thiệu

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

ConstraintLayout là một ViewGroup , là một View đặc biệt có thể chứa các đối tượng View khác (được gọi là dạng xem con hoặc con). Thực tế này cho thấy nhiều tính năng hơn của ConstraintLayout và trình chỉnh sửa bố cục.

Bài 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 so với phần tử View khác trong ViewGroup. Vị trí của các phần tử View con được mô tả trong mối quan hệ với nhau hoặc với ViewGroup cha

Những gì Bạn sẽ cần:

Bạn sẽ có thể:

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

Những gì bạn sẽ học

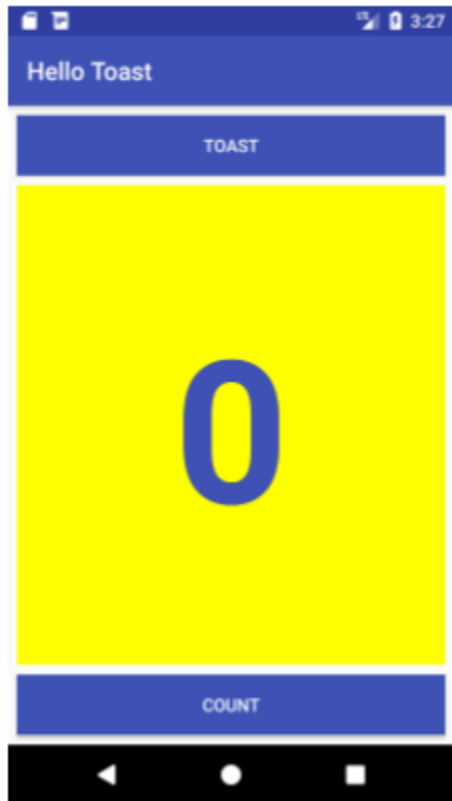
- Cách tạo một biến thể bố cục cho hướng ngang (ngang).
- Cách tạo biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Cách sử dụng ràng buộc đường cơ sở để căn chỉnh các thành phần giao diện người dùng với văn bản.
- Cách sử dụng các nút gói và căn chỉnh để căn chỉnh các phần tử trong bố cục.
- Cách định vị chế độ xem trong LinearLayout .
- Cách định vị chế độ xem trong RelativeLayout .

Những gì bạn sẽ làm

- Tạo biến thể bố cục cho hướng hiển thị ngang.
- Tạo biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Sửa đổi bố cục để thêm ràng buộc vào các thành phần giao diện người dùng.
- Sử dụng các ràng buộc đường cơ sở ConstraintLayout để căn chỉnh các phần tử với văn bản.
- Sử dụng gói ConstraintLayout và căn chỉnh các nút để căn chỉnh các phần tử.
- Thay đổi bố cục để sử dụng LinearLayout .
- Định vị các phần tử trong LinearLayout .
- Thay đổi bố cục để sử dụng RelativeLayout .
- Sắp xếp lại các chế độ xem trong bố cục chính để tương đối với nhau

Tổng quan ứng dụng

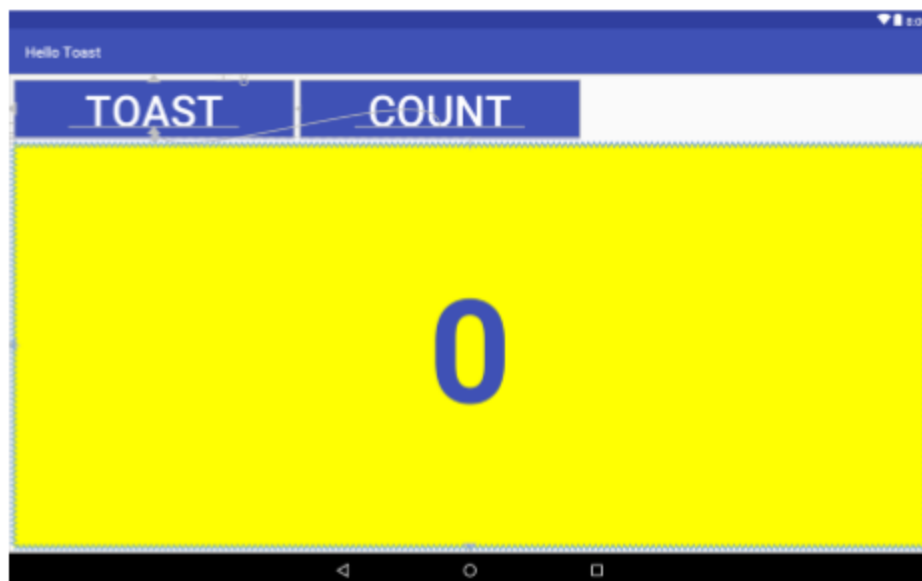
Ứng dụng Hello Toast trong bài học trước sử dụng ConstraintLayout để sắp xếp các thành phần giao diện người dùng trong bố cục Activity, như trong hình bên dưới.



Để thực hành thêm với `ConstraintLayout`, bạn sẽ tạo một biến thể của bố cục này cho hướng ngang như trong hình bên dưới.



Bạn cũng sẽ tìm hiểu cách sử dụng các ràng buộc đường cơ sở và một số tính năng căn chỉnh của `ConstraintLayout` bằng cách tạo một biến thể bố cục khác cho màn hình máy tính bảng.

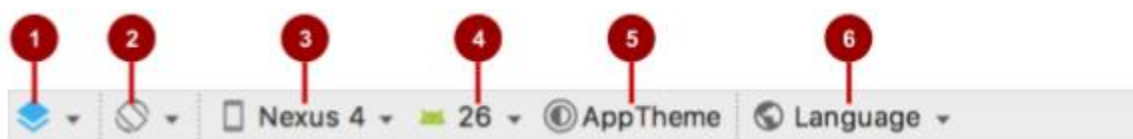


Bạn cũng tìm hiểu về các lớp con ViewGroup khác như LinearLayout và RelativeLayout , đồng thời thay đổi bố cục ứng dụng Hello Toast để sử dụng chúng

a. Tạo biến thể bố cục

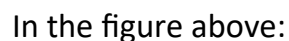
Trong bài học trước, coding challenge yêu cầu thay đổi bố cục của ứng dụng Hello Toast để nó vừa vặn theo hướng ngang hoặc dọc. Trong nhiệm vụ này, bạn sẽ học một cách dễ dàng hơn để tạo các biến thể bố cục của bạn cho horizontal (còn được gọi là landscape) và vertical (còn được gọi là portrait) cho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

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



In the figure above:

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

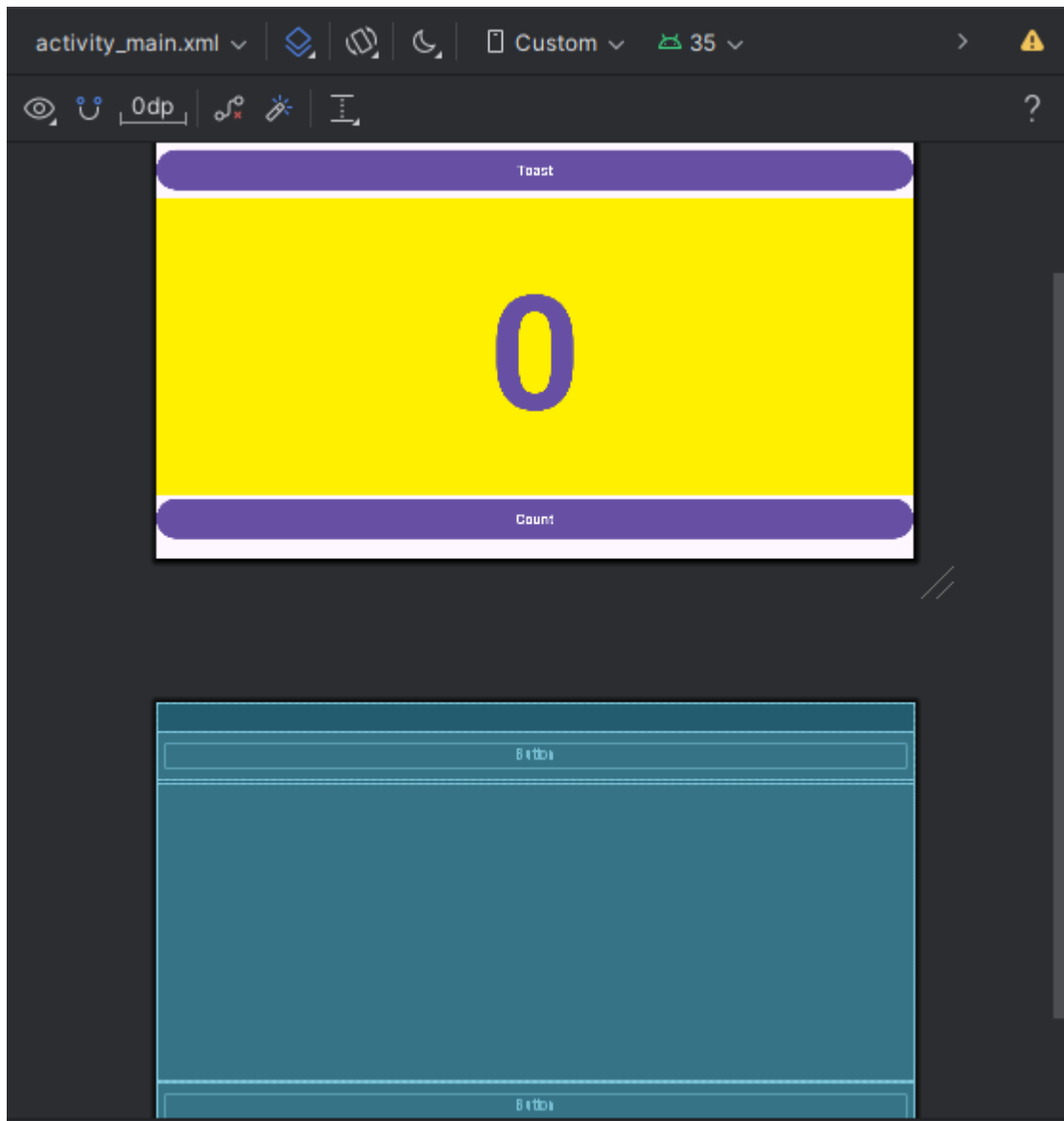


1. **Hiển thị :** Chọn **Hiển thị ràng buộc** và **Hiển thị lề** để hiển thị chúng trong bản xem trước hoặc để dừng hiển thị chúng.
2. **Tự động kết nối:** **Bật** hoặc **tắt** **Tự động kết nối**. Khi **bật** **Tự động kết nối**, bạn có thể kéo bất kỳ phần tử nào (chẳng hạn như **Nút**) vào bất kỳ phần nào của bố cục để tạo các ràng buộc đối với bố cục mẹ.
3. **Xóa tất cả các ràng buộc :** Xóa tất cả các ràng buộc trong toàn bộ bố cục.
4. **Infer Constraints:** Tạo ràng buộc bằng suy luận.
5. **Lề mặc định:** Đặt lề mặc định.
6. **Đóng gói:** Đóng gói hoặc mở rộng các yếu tố đã chọn.
7. **Căn chỉnh:** Căn chỉnh các phần tử đã chọn.
8. **Nguyên tắc :** Thêm hướng dẫn dọc hoặc ngang.
9. **Điều khiển thu phóng/xoay:** Phóng to hoặc thu nhỏ.

i. Xem trước bố cục theo hướng ngang

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

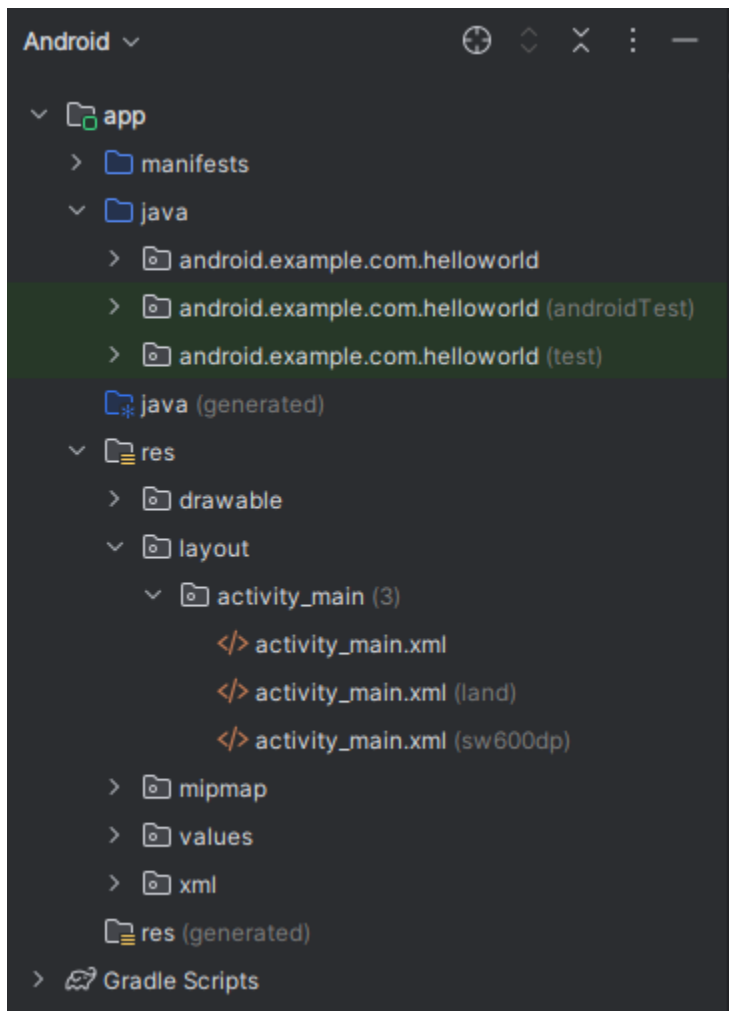
1. **Mở ứng dụng Hello Toast từ bài học trước.**
Lưu ý: Nếu bạn đã tải xuống mã giải pháp cho HelloToast, bạn cần xóa các bố cục ngang và cực lớn đã hoàn thành mà bạn sẽ tạo trong tác vụ này. Chuyển từ **Project > Android** sang **Project > Project Files** trong ngăn **Project**, mở rộng ứng dụng > ứng dụng > **src/main > res** , chọn cả thư mục **layout-land** và thư mục **layout-xlarge** và chọn **Edit > Delete** . Sau đó, chuyển ngăn **Dự án** trở lại **Dự án > Android** .
2. **Mở tệp bố cục activity_main.xml.** Nhấp vào tab **Thiết kế** nếu nó chưa được chọn.
3. Nhấp vào nút **Định hướng** trong **Trình chỉnh sửa** trên thanh công cụ trên cùng.
4. Chọn **Switch to Landscape** trong menu thả xuống. Bố cục xuất hiện theo hướng ngang như hình dưới đây. Để quay lại hướng dọc, hãy chọn **Switch to Portrait**

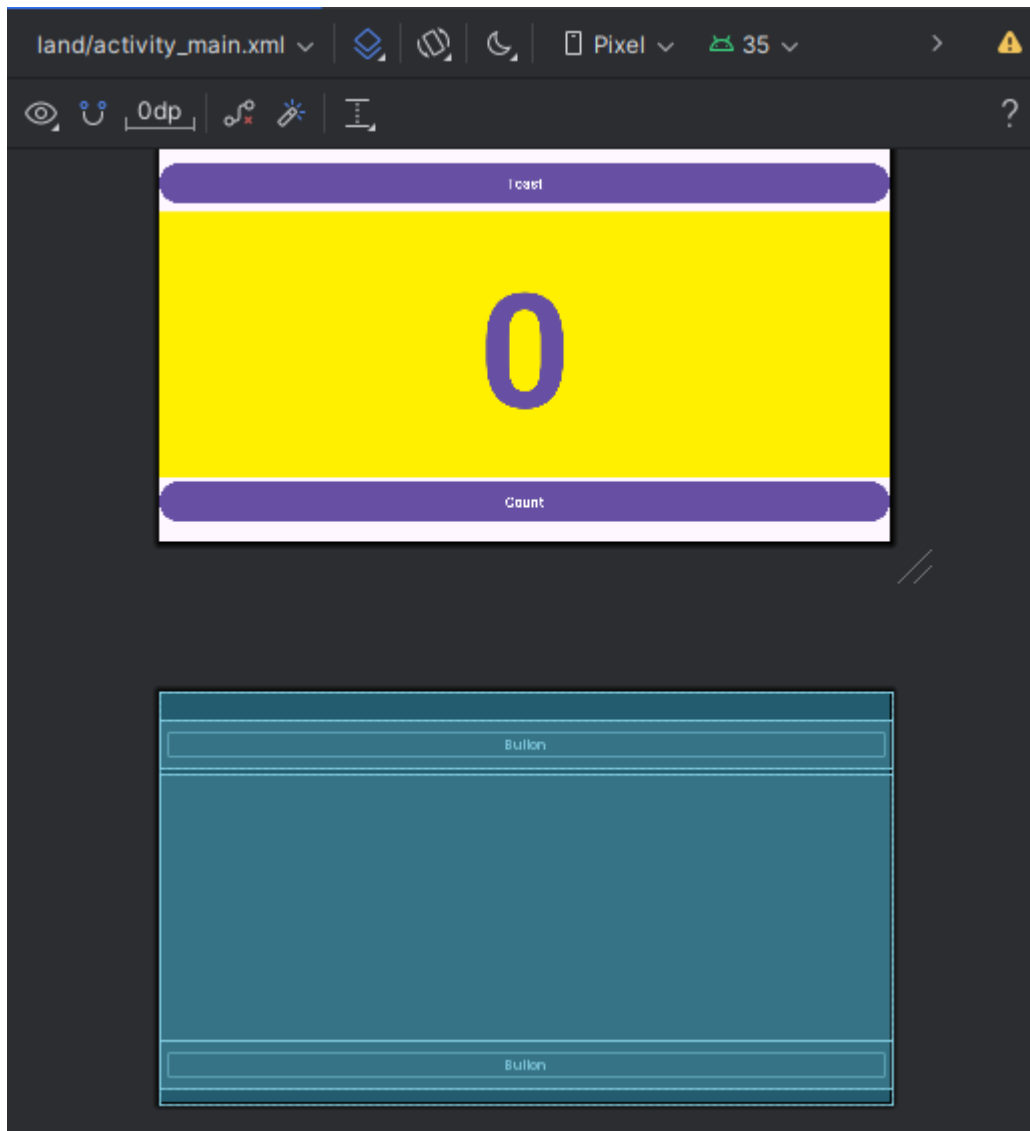


ii. Tạo biến thể bố cục cho hướng ngang

Sự khác biệt trực quan giữa hướng dọc và chiều ngang cho bố cục này là chữ số (0) trong phần tử `TextView show_count` quá thấp so với hướng ngang — quá gần với nút `Đếm`. Tùy thuộc vào thiết bị hoặc trình mô phỏng bạn sử dụng, phần tử `TextView` có thể xuất hiện quá lớn hoặc không ở giữa vì kích thước văn bản được cố định ở 160sp. Để khắc phục điều này đối với các hướng ngang trong khi vẫn giữ nguyên hướng dọc, bạn có thể tạo biến thể của bố cục ứng dụng `Hello Toast` khác với hướng ngang. Làm theo các bước sau:

1. Nhấp vào nút Định hướng trong Trình chỉnh sửa trên thanh công cụ trên cùng.
2. . Chọn Tạo biến thể phong cảnh .
3. Một cửa sổ trình chỉnh sửa mới mở ra với tab đất liền / activity_main.xml hiển thị bố cục cho hướng ngang (ngang). Bạn có thể thay đổi bố cục này, dành riêng cho hướng ngang, mà không cần thay đổi hướng dọc (dọc) ban đầu. . Trong ngăn Project > Android, hãy nhìn vào bên trong thư mục res > layout và bạn sẽ thấy rằng Android Studio đã tự động tạo biến thể cho bạn, được gọi là activity_main.xml (land) .

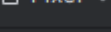




iii. Xem trước bố cục cho các thiết bị khác nhau

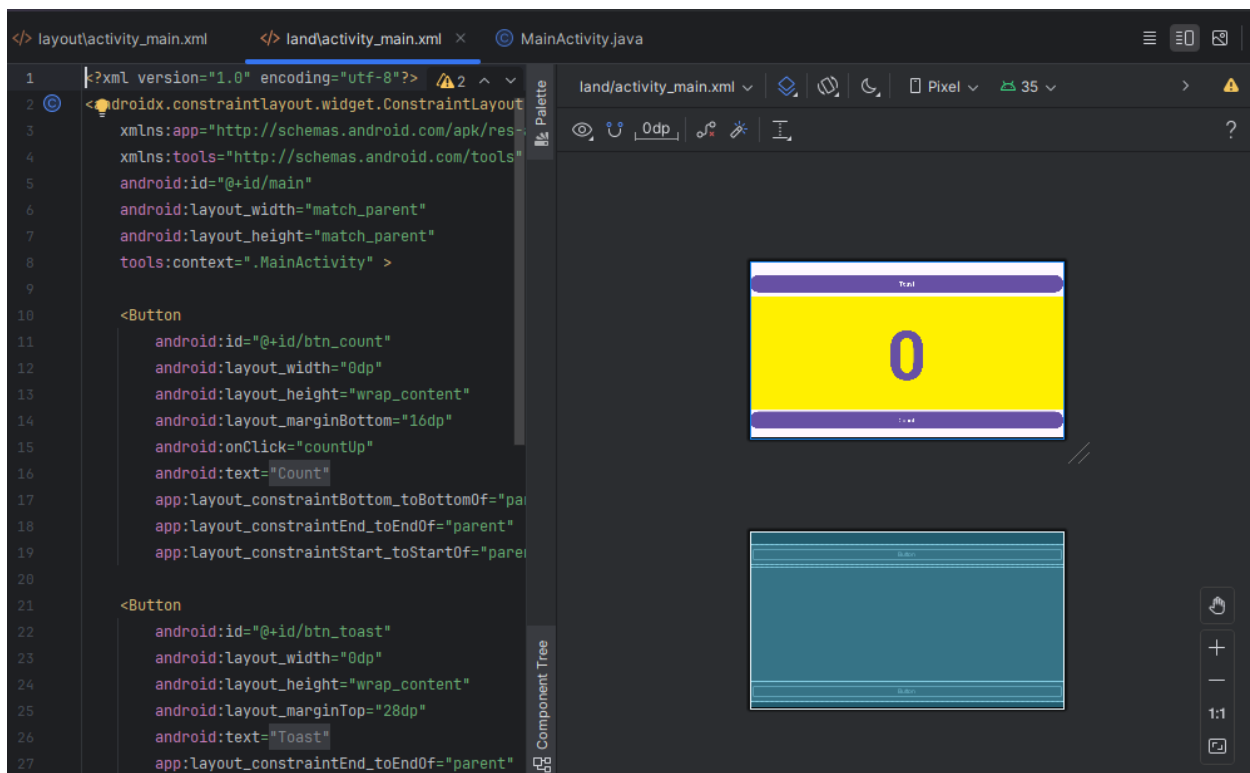
Bạn có thể xem trước bố cục cho các thiết bị khác nhau mà không cần phải chạy ứng dụng trên thiết bị hoặc trình mô phỏng. Làm theo các bước sau:

1. Tab `land/activity_main.xml` vẫn sẽ được mở trong trình chỉnh sửa bố cục; Nếu không, hãy nhấp đúp vào tệp `activity_main.xml (land)` trong thư mục bố cục.

2. Nhấp vào nút Device in Editor  trên thanh công cụ trên cùng.
3. Chọn một thiết bị khác trong menu thả xuống. Ví dụ: chọn Nexus 4 , Nexus 5 và sau đó chọn Pixel để xem sự khác biệt trong bản xem trước. Những khác biệt này là do kích thước văn bản cố định cho TextView .

iv. Thay đổi bố cục cho hướng ngang

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



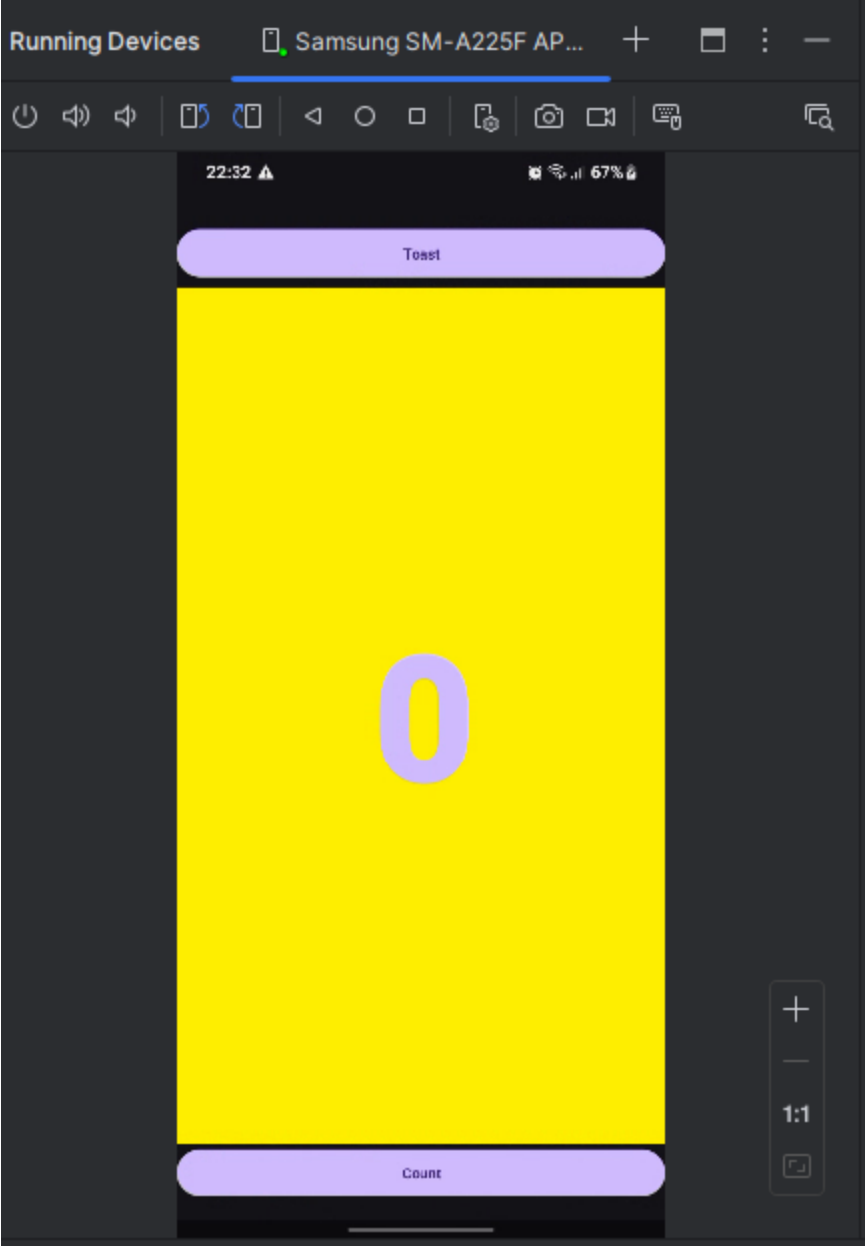
Hình trên cho thấy những điều sau:

1. Tab Xem trước, bạn sử dụng để hiển thị ngăn xem trước
2. Ngăn xem trước
3. Mã XML


Chạy ứng dụng trên trình mô phỏng hoặc thiết bị và chuyển hướng từ dọc sang ngang để xem cả bố cục

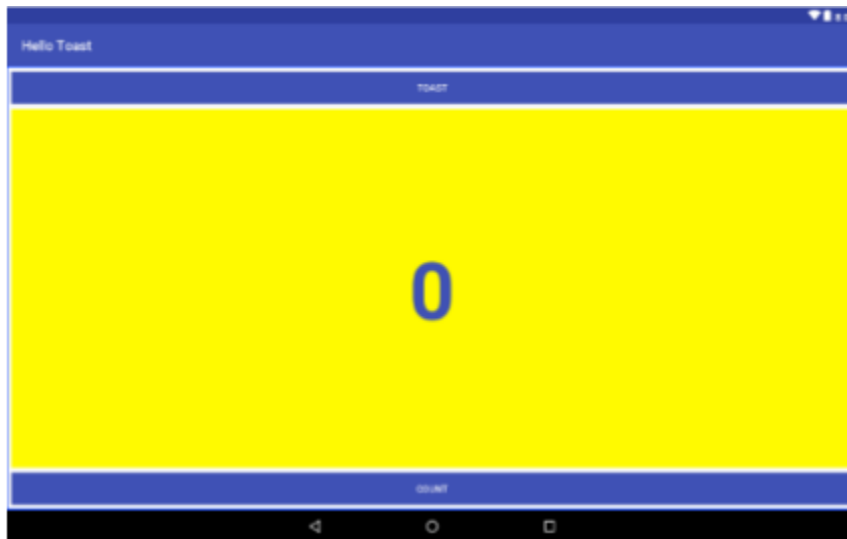
To change the layout, follow these steps:

1. Tab `and/activity_main.xml` vẫn sẽ được mở trong trình chỉnh sửa bố cục; Nếu không, hãy nhấp đúp vào tệp `activity_main.xml` (land) trong thư mục bố cục.
2. Nhấp vào tab Văn bản và tab Xem trước (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 Thiết bị trong Trình chỉnh sửa để xem bố cục trông như thế nào trên các thiết bị khác nhau theo hướng ngang. Trong ngăn trình chỉnh sửa, tab `and/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.
6. Chạy ứng dụng trên trình mô phỏng hoặc thiết bị và chuyển hướng từ dọc sang ngang để xem cả hai bố cục.



v. Tạo biến thể bố cục cho máy tính bảng

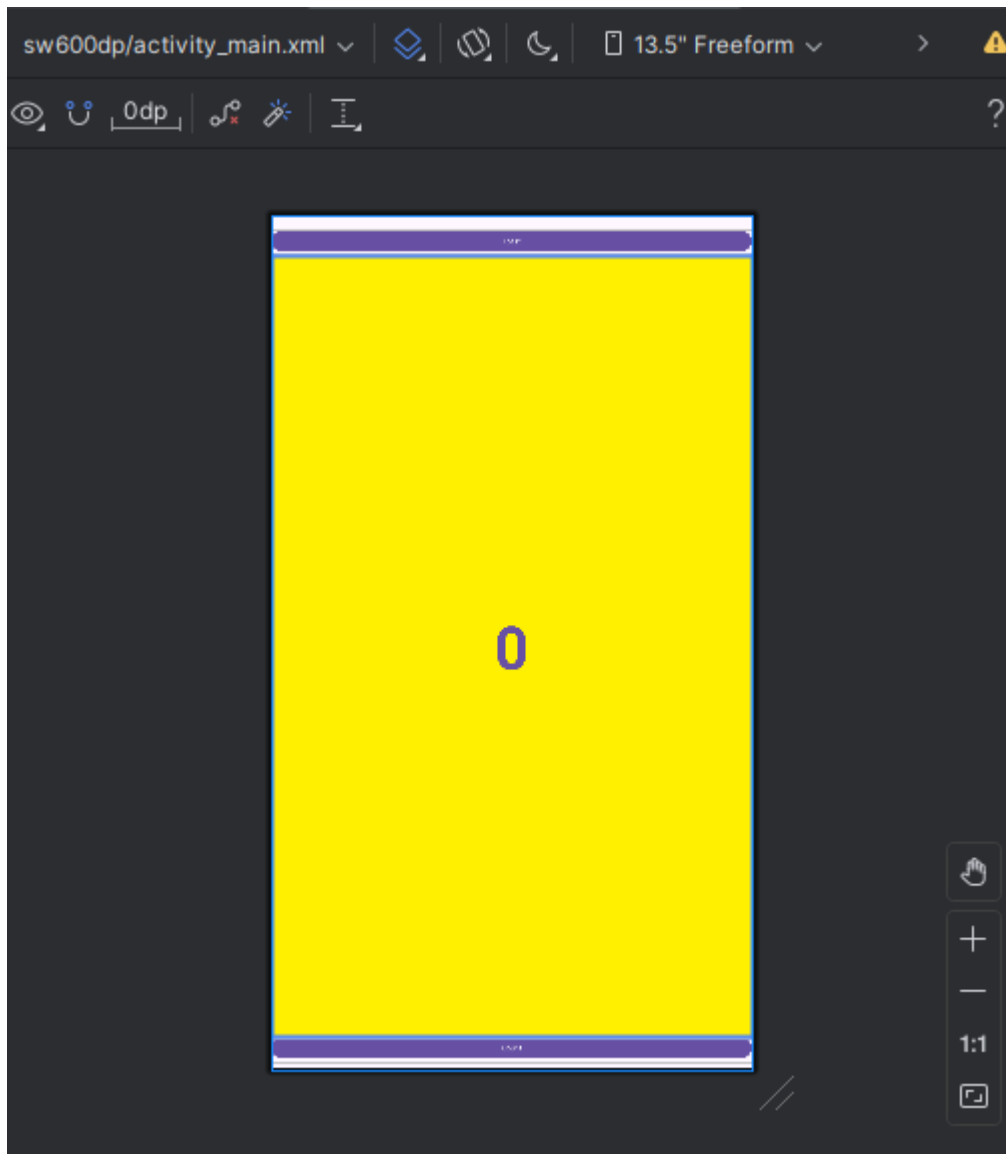
Như bạn đã học trước đây, bạn có thể xem trước bố cục cho các thiết bị khác nhau bằng cách nhấp vào nút Thiết bị  Nexus 5 trong Trình chỉnh sửa trên thanh công cụ trên cùng. Nếu bạn chọn một thiết bị như Nexus 10 (máy tính bảng) từ menu, bạn có thể thấy rằng bố cục không lý tưởng cho màn hình máy tính bảng — văn bản của mỗi Button quá nhỏ và cách sắp xếp các phần tử Button ở trên cùng và dưới cùng không lý tưởng cho máy tính bảng màn hình lớn.



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



1. Nhấp vào tab Thiết kế (nếu chưa được chọn) để hiển thị các ngăn thiết kế và bản thiết kế.
2. Nhấp vào nút Định hướng trong Trình chỉnh sửa trên thanh công cụ trên cùng.
3. Chọn Tạo bố cục x-large Variation .

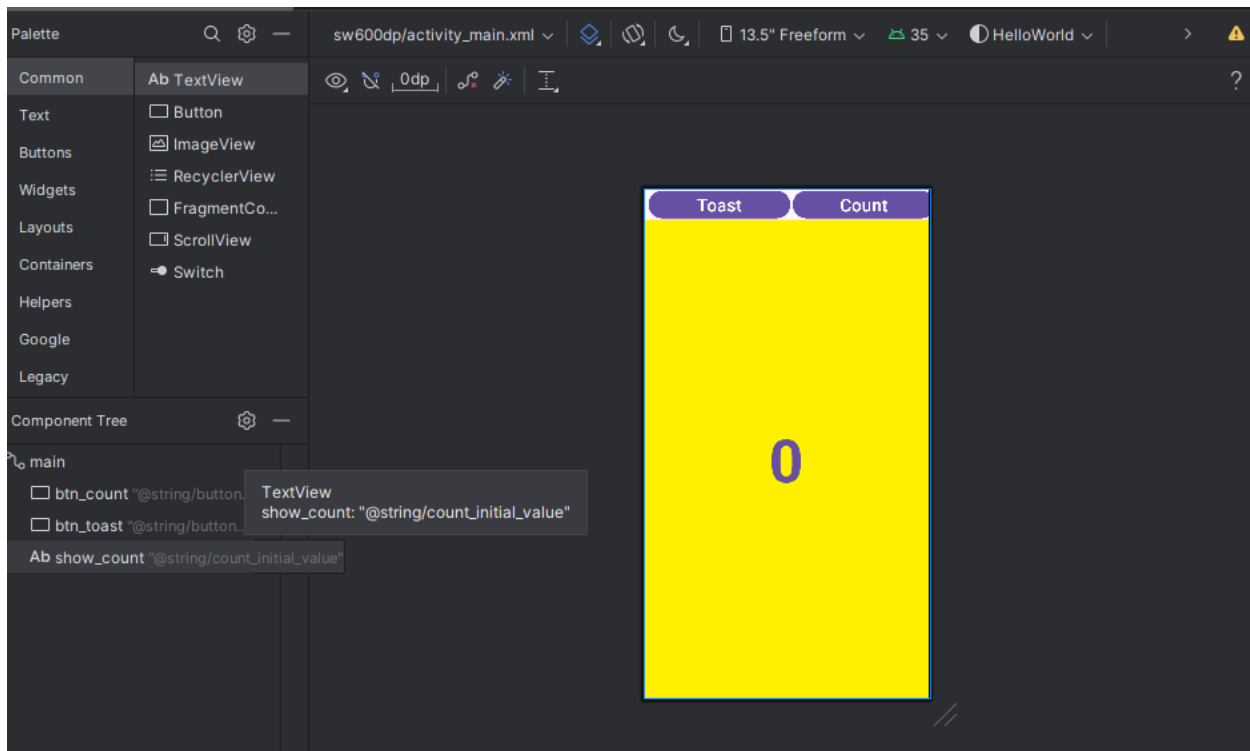
Một cửa sổ trình chỉnh sửa mới mở ra với tab xlarge / activity_main.xml hiển thị bố cục cho thiết bị có kích thước máy tính bảng. Biên tập viên cũng chọn một thiết bị máy tính bảng, chẳng hạn như Nexus 9 hoặc Nexus 10, để xem trước. Bạn có thể thay đổi bố cục này, dành riêng cho máy tính bảng, mà không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



vi. Thay đổi biến thể bố cục cho máy tính bảng

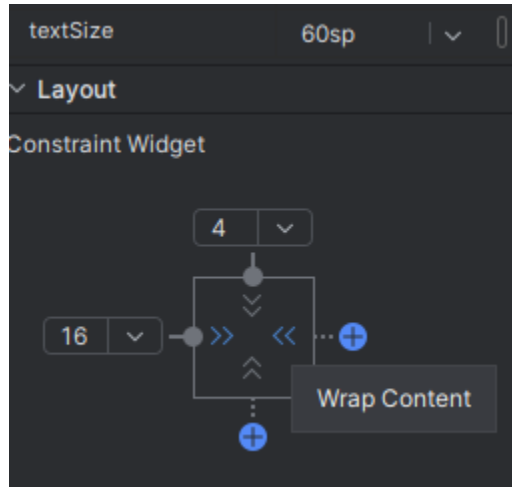
Bạn có thể sử dụng ngăn Thuộc tính trong tab Thiết kế để thay đổi thuộc tính cho bố cục này.

1. Tắt công cụ Tự động kết nối trên thanh công cụ. Đối với bước này, hãy đảm bảo rằng công cụ đã bị tắt: 
2. Xóa tất cả các ràng buộc trong bố cục bằng cách nhấp vào nút Xóa tất cả các ràng buộc  trên thanh công cụ. Với các ràng buộc được loại bỏ, bạn có thể di chuyển và thay đổi kích thước các phần tử trên bố cục một cách tự do.
3. Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước trên cả bốn góc của một phần tử để thay đổi kích thước của nó. Trong Component Tree , chọn TextView được gọi là show_count . Để loại bỏ TextView để bạn có thể tự do kéo các phần tử Button, hãy kéo một góc của nó để thay đổi kích thước, như trong hình động bên dưới.



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

4. Chọn nút `button_toast` trong Component Tree , nhấp vào tab Attributes để mở ngăn Attributes và thay đổi `textSize` thành 60sp (# 1 trong hình bên dưới) và `layout_width` thành `wrap_content` (# 2 trong hình bên dưới).



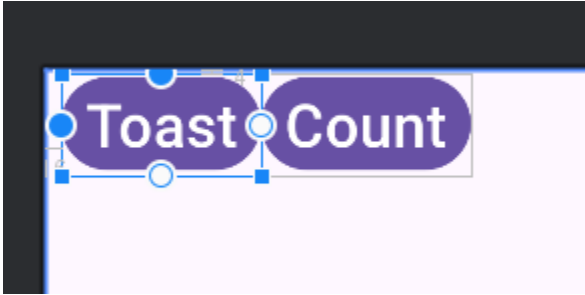
Như hiển thị ở phía bên phải của hình trên (2), bạn có thể nhấp vào điều khiển chiều rộng của trình kiểm tra chế độ xem, xuất hiện trong hai phân đoạn ở bên trái và bên phải của hình vuông, cho đến khi nó hiển thị `Wrap Content` . Thay vào đó, bạn có thể chọn `wrap_content` từ menu `layout_width`. Bạn 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 mỏng hơn để phù hợp với từ trong ngôn ngữ khác.

5. Chọn Nút `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.

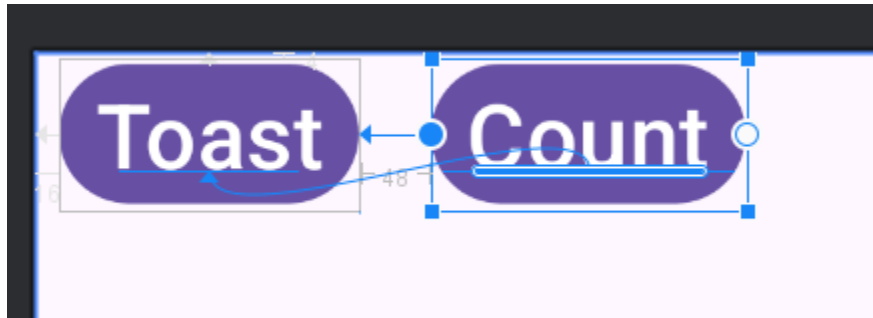
vii. **Sử dụng ràng buộc đường cơ sở**

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


1. Hạn chế Nút `button_toast` ở phía trên cùng và bên trái của bố cục, kéo Nút `button_count` vào một khoảng trống gần Nút `button_toast` và hạn chế Nút `button_count` ở phía bên trái của Nút `button_toast` , như trong hình động:




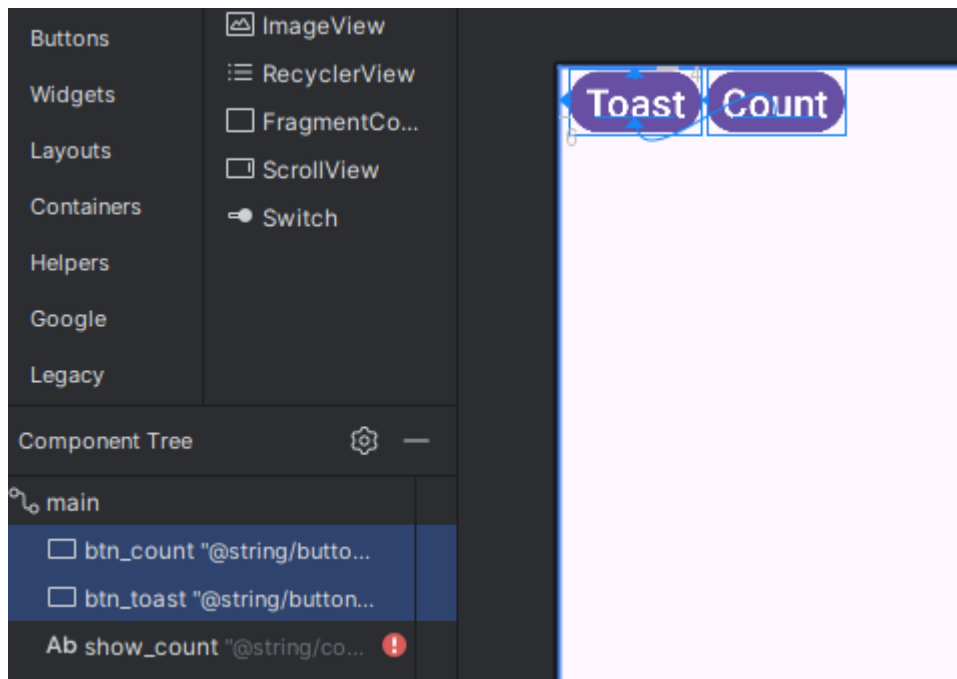
2. Sử dụng ràng buộc đường cơ sở , bạn có thể hạn chế Nút `button_count` để đường cơ sở văn bản của nó khớp với đường cơ sở văn bản của Nút `button_toast` . Chọn phần tử `button_count`, sau đó di con trỏ của bạn qua phần tử cho đến khi nút ràng buộc đường cơ sở xuất hiện bên dưới phần tử.
3. Nhấp vào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



viii. Mở rộng các nút theo chiều ngang

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

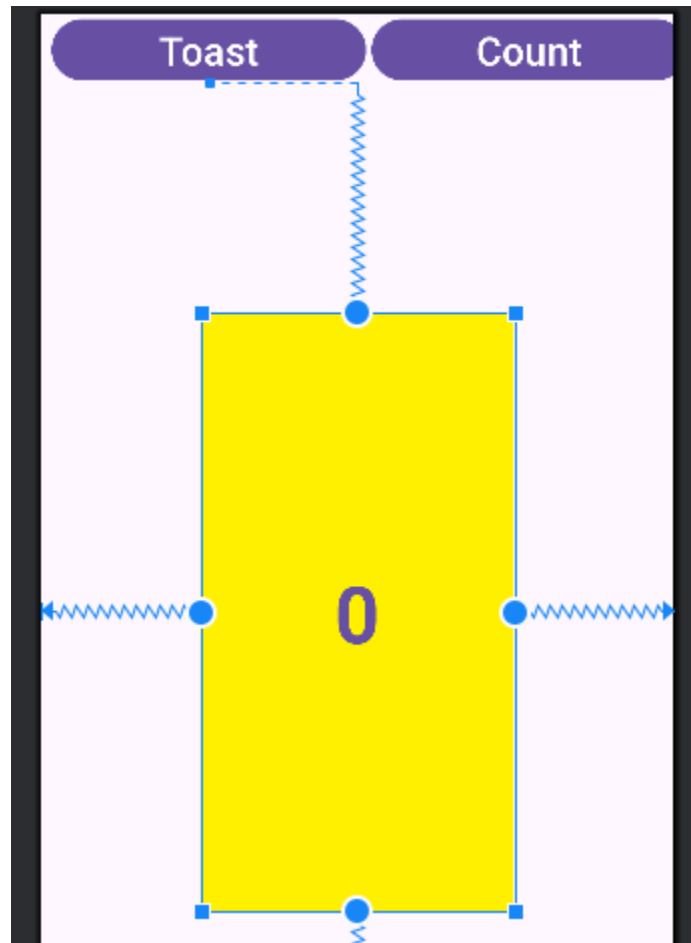
1. Chọn Nút `button_count` trong Cây thành phần và Shift-chọn Nút `button_toast` để cả hai đều được chọn.
2. Nhấp vào nút gói trên  thanh công cụ và chọn Mở rộng theo chiều ngang như trong hình bên dưới.



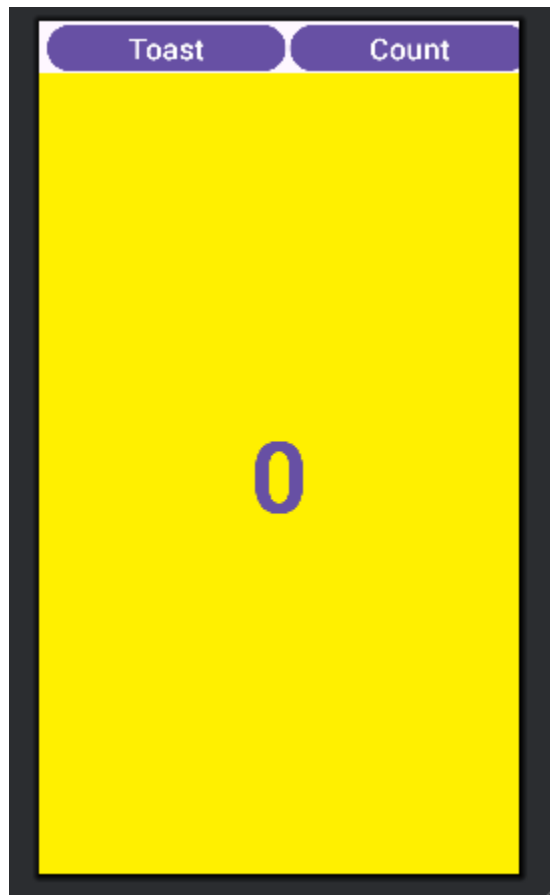
Các phần tử Button mở rộng theo chiều ngang để lấp đầy bố cục như hình dưới đây.



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



4. Các bước cuối cùng là thay đổi show_count TextView layout_width và layout_height thành Match Constraints và textSize thành 200sp . Bố cục cuối cùng trông giống như hình bên dưới.



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



6. Chạy ứng dụng trên các trình giả lập khác nhau và thay đổi hướng sau khi chạy ứng dụng để xem ứng dụng 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 thích hợp trên điện thoại và máy tính bảng có kích thước và mật độ màn hình khác nhau

b. Thay đổi bố cục thành LinearLayout

LinearLayout is a ViewGroup that arranges its collection of views in a horizontal or vertical row. A LinearLayout is one of the most common layouts because it is simple and fast. It is often used within another view group to arrange UI elements horizontally or vertically.

A LinearLayout is required to have these attributes:

- layout_width
- layout_height
- orientation

The layout_width and layout_height can take one of these values:

- `match_parent` : Expands the view to fill its parent by width or height. When the `LinearLayout` is the root view, it expands to the size of the screen (the parent view).
- `wrap_content` : Shrinks the view dimensions so the view is just big enough to enclose its content. If there is no content, the view becomes invisible.
- Fixed number of dp (density-independent pixels): Specify a fixed size, adjusted for the screen density of the device. For example, `16dp` means 16 density-independent pixels.

The orientation can be:

- `horizontal` : Views are arranged from left to right.
- `vertical` : Views are arranged from top to bottom.

In this task you will change the `ConstraintLayout` root view group for the Hello Toast app to `LinearLayout` so that you can gain practice in using `LinearLayout` .

i. **Thay đổi view group gốc thành `LinearLayout`**

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở tệp bố cục `activity_main.xml` (nếu chưa mở) và nhấp vào tab Văn bản ở cuối 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 thế này:

```
<LinearLayout xmlns:android="http:...
```

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

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

```
android: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` .

```
<?xml version="1.0" encoding="utf-8"?>
<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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity" >
```

ii. 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 được mở) và nhấp vào tab Văn bản.
3. Tìm phần tử button_toast Button và thay đổi thuộc tính sau:

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

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

app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toTopOf="parent"

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

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

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

app:layout_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_height="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"

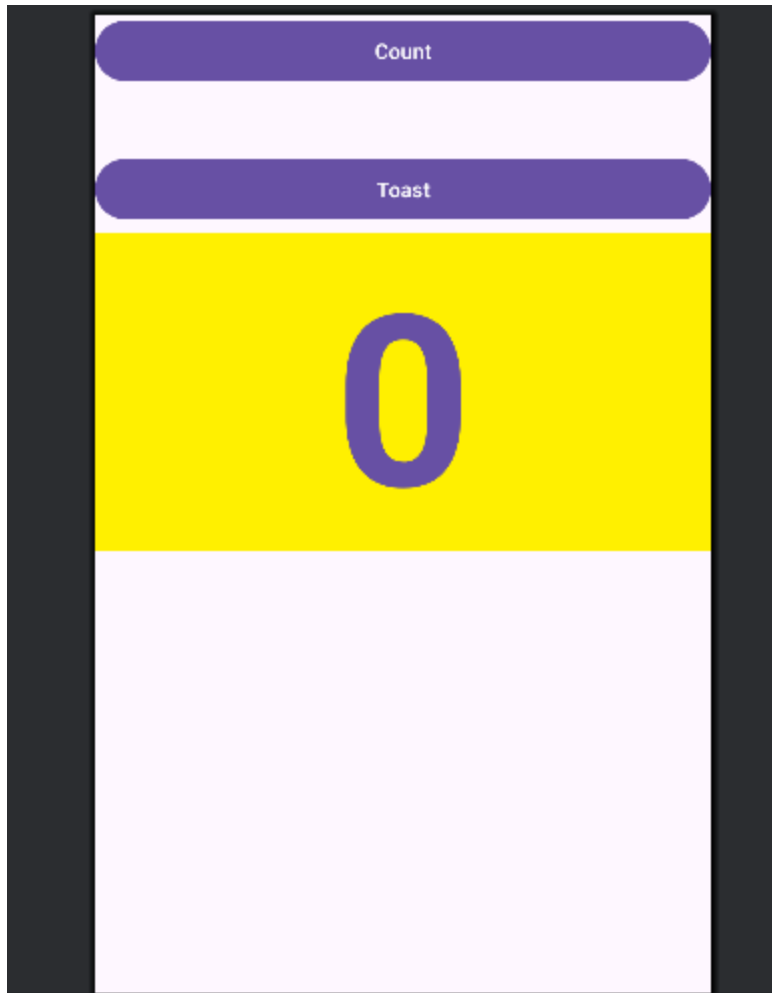
9. Nhấp vào tab Preview ở phía bên phải của cửa sổ Android Studio (nếu chưa được chọn) để xem trước bố cục cho đến nay:

```
<?xml version="1.0" encoding="utf-8"?>
<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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity" >

    <Button
        android:id="@+id/btn_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="16dp"
        android:onClick="countUp"
        android:text="@string/button_label_count" />

    <Button
        android:id="@+id/btn_toast"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:text="@string/button_label_toast"
        app:layout_constraintHorizontal_bias="0.498"
        android:onClick="showToast" />

    <TextView
        android:id="@+id/show_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dp"
        android:background="#FFF000"
        android:gravity="center_horizontal|center_vertical"
        android:text="@string/count_initial_value"
        android:textAlignment="center"
        android:textColor="?attr/colorPrimary"
        android:textSize="160sp"
        android:textStyle="bold" />
</LinearLayout>
```



iii. Thay đổi vị trí của các phần tử trong LinearLayout

LinearLayout arranges its elements in a horizontal or vertical row. You have already added the `android:orientation="vertical"` attribute for the LinearLayout, so the elements are stacked on top of each other vertically as shown in the previous figure.

To change their positions so that the Count button is on the bottom, follow these steps:

1. Open the Hello Toast app from the previous task.
2. Open the `activity_main.xml` layout file (if it is not already open), and click the Text tab.
3. Select the `button_count` Button and all of its attributes, from the `<Button` tag up to and including the closing `>` tag, and choose Edit > Cut .
4. Click after the closing `>` tag of the `TextView` element but before the closing `</LinearLayout>` tag, and choose Edit > Paste .
5. (Optional) To fix any indents or spacing issues for cosmetic purposes, choose Code > Reformat Code to reformat the XML code with proper spacing and indents.

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

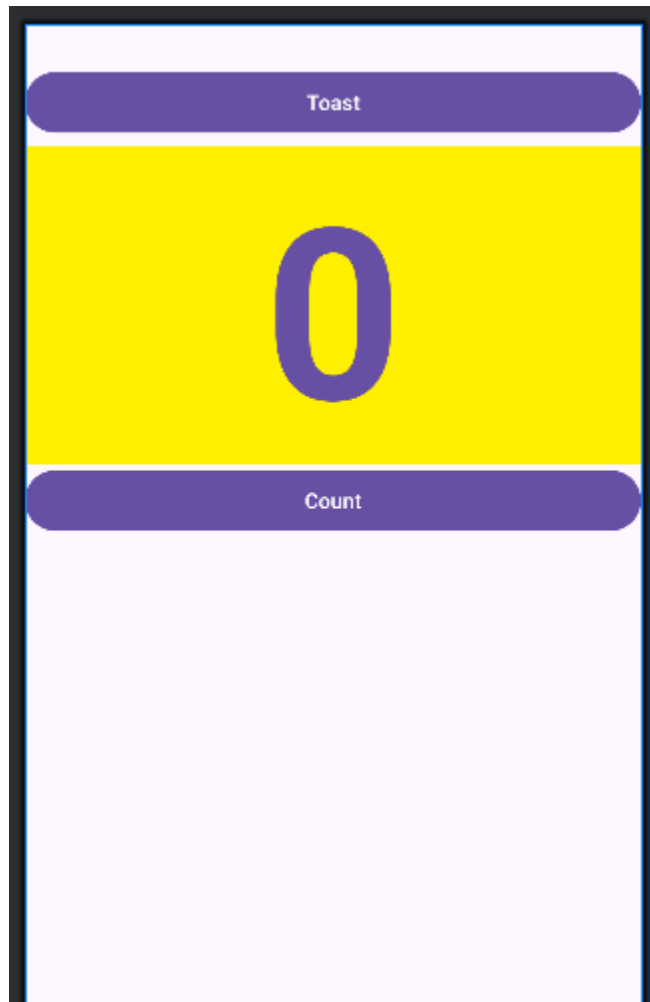
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"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

    <Button
        android:id="@+id/btn_toast"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:onClick="showToast"
        android:text="@string/button_label_toast"
        app:layout_constraintHorizontal_bias="0.498" />

    <TextView
        android:id="@+id/show_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dp"
        android:background="#FFF000"
        android:gravity="center_horizontal|center_vertical"
        android:text="@string/count_initial_value"
        android:textAlignment="center"
        android:textColor="?attr/colorPrimary"
        android:textSize="160sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/btn_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginBottom="16dp"
        android:onClick="countUp"
        android:text="@string/button_label_count" />
</LinearLayout>
```

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



iv. Thêm **weight** cho phần tử **TextView**

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

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

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

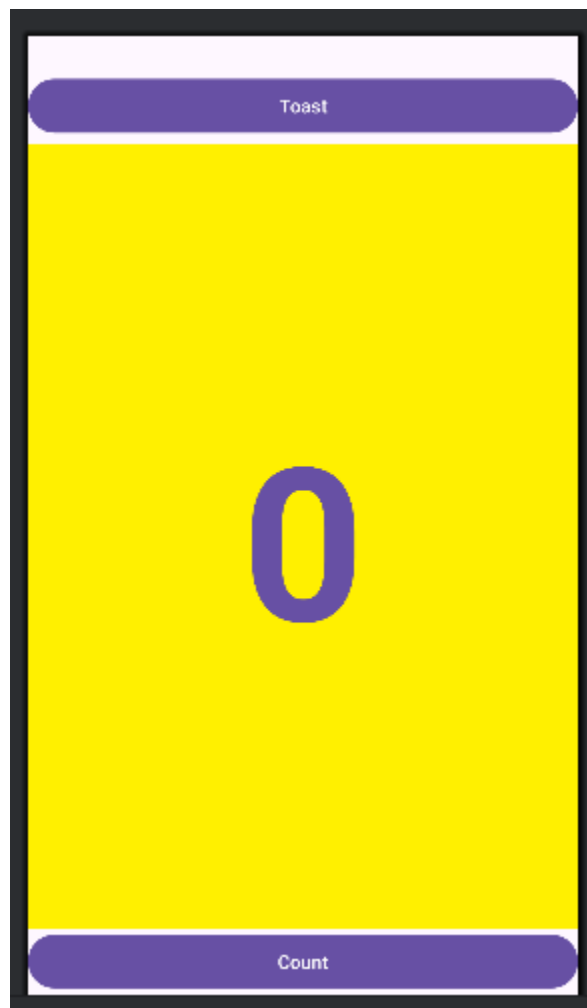
`android:layout_weight` tôi chỉ ra bao nhiêu không gian bổ sung trong `LinearLayout` sẽ được phân bổ cho View. Nếu chỉ có một Chế độ xem có thuộc tính này, thì nó sẽ nhận

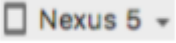
được tất cả không gian màn hình bổ sung. Đối với nhiều phần tử View, không gian được chia theo tỷ lệ. Ví dụ: nếu mỗi phần tử Button có trọng lượng là 1 và TextView là 2, tổng cộng là 4, thì các phần tử Button nhận được 1/4 khoảng trống mỗi phần tử và một nửa TextView.

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

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở tệp bố cục activity_main.xml (nếu nó chưa được mở) và nhấp vào tab Văn bản.
3. Tìm phần tử TextView show_count và thêm thuộc tính sau:

`android:layout_weight="1"`



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 bấm vào nút Thiết bị trong Trình chỉnh sửa  trên thanh công cụ trên cùng của ngăn xem trước và chọn một thiết bị khác. Bất kể bạn chọn thiết bị nào để xem trước, phần tử TextView show_count sẽ chiếm tất cả không gian giữa các nút.

c. Thay đổi bố cục thành RelativeLayout

i. Thay đổi LinearLayout thành RelativeLayout

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

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

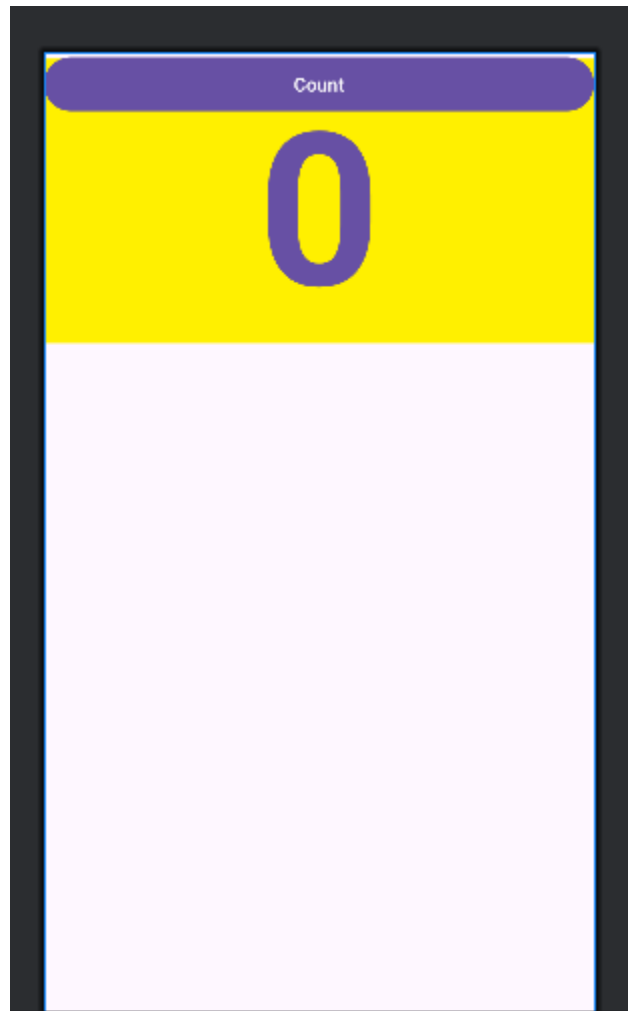
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

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

ii. Sắp xếp lại các chế độ xem trong RelativeLayout

An easy way to rearrange and position views in a RelativeLayout is to add XML attributes in the Text tab.

1. Click the `Preview` tab at the side of the editor (if it is not already selected) to see the layout preview, which now looks like the figure below.



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

- Nút đếm (button_count) phủ lên Nút Toast , đó là lý do tại sao bạn không thể nhìn thấy Nút Toast (button_toast).
 - Phần trên cùng của TextView (show_count) phủ các phần tử Button.
2. Thêm thuộc tính android:layout_below vào button_count Button để định vị Button ngay bên dưới show_count TextView . Thuộc tính này là một trong một số thuộc tính để định vị các dạng xem trong RelativeLayout —bạn đặt các dạng xem liên quan đến các dạng xem khác.
 3. Thêm thuộc tính android:layout_centerHorizontal vào cùng một Button để căn giữa chế độ xem theo chiều ngang trong phần tử của nó, trong trường hợp này là RelativeLayout

4. Thêm các thuộc tính sau vào show_count TextView

android:layout_below="@+id/button_toast"

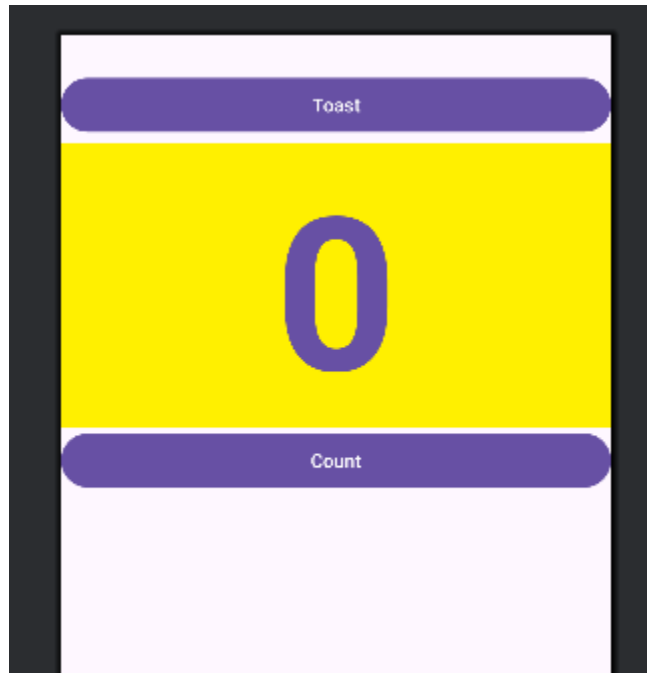
android:layout_alignParentLeft="true"

android:layout_alignParentStart="true"

```
<TextView
    android:id="@+id/show_count"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="5dp"
    android:background="#FFF000"
    android:gravity="center_horizontal|center_vertical"
    android:text="@string/count_initial_value"
    android:textAlignment="center"
    android:textColor="?attr/colorPrimary"
    android:textSize="160sp"
    android:textStyle="bold"
    android:layout_below="@+id/btn_toast"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"/>

<Button
    android:id="@+id/btn_count"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:onClick="countUp"
    android:text="@string/button_label_count"
    android:layout_below="@+id/show_count"
    android:layout_centerHorizontal="true" />
```

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



1.4) Văn bản và các chế độ cuộn

Giới thiệu

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

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

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

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

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

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

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

Những gì Bạn sẽ cần:

Bạn sẽ có thể:

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

Những gì bạn sẽ học

Bạn sẽ có thể:

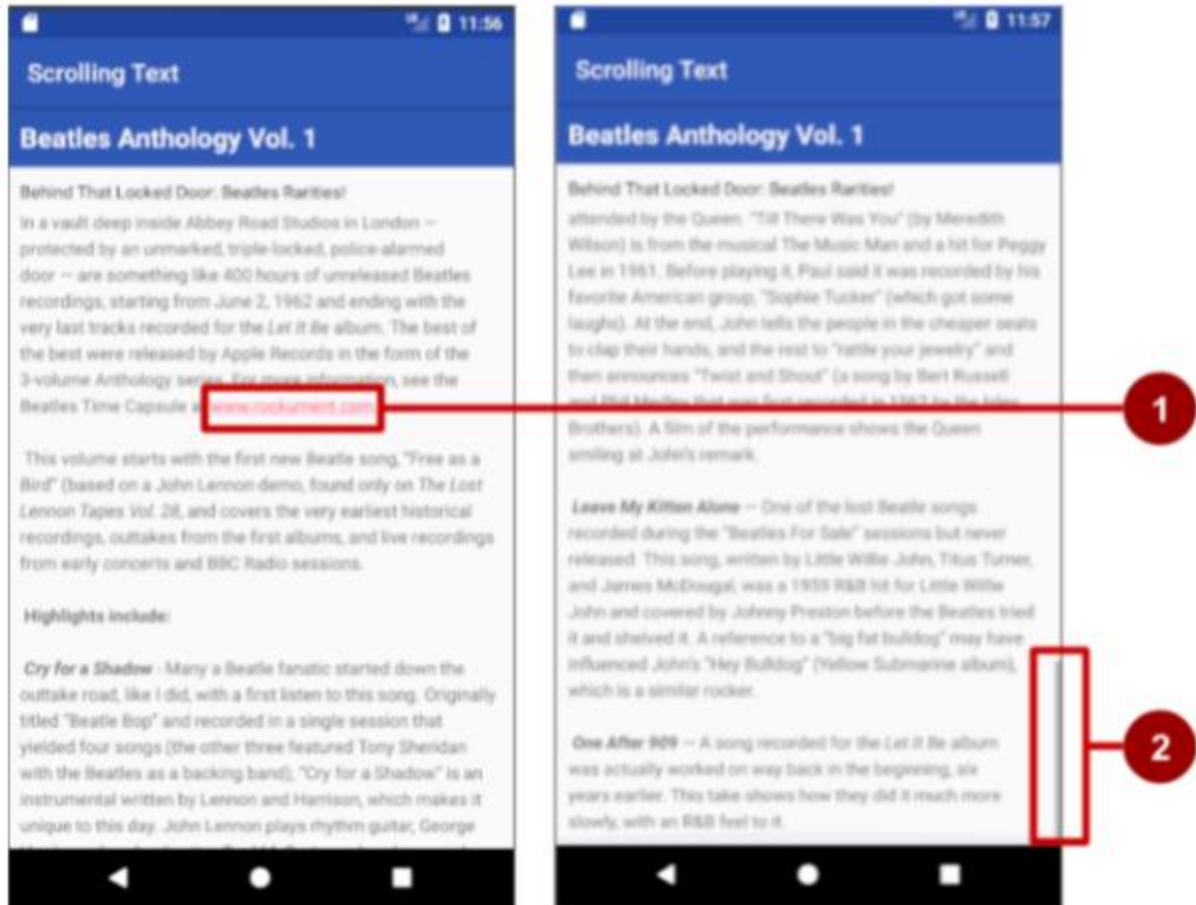
- Tạo ứng dụng Hello World bằng Android Studio.
- Chạy ứng dụng trên trình mô phỏng hoặc thiết bị.
- Triển khai TextView trong bố cục cho ứng dụng.
- Tạo và sử dụng tài nguyên chuỗi.

Những gì bạn sẽ làm

- Tạo ứng dụng ScrollingText.
- Thay đổi ConstraintLayout ViewGroup thành RelativeLayout .
- Thêm hai phần tử TextView cho tiêu đề và tiêu đề phụ của bài viết.
- Sử dụng kiểu và màu TextAppearance cho tiêu đề và tiêu đề phụ của bài viết.
- Sử dụng thẻ HTML trong chuỗi văn bản để kiểm soát định dạng.
- Sử dụng thuộc tính lineSpacingExtra để thêm khoảng cách dòng để dễ đọc.
- Thêm ScrollView vào bố cục để cho phép cuộn phần tử TextView.
- Thêm thuộc tính autoLink để cho phép URL trong văn bản hoạt động và có thể nhấp vào.

Tổng quan ứng dụng

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

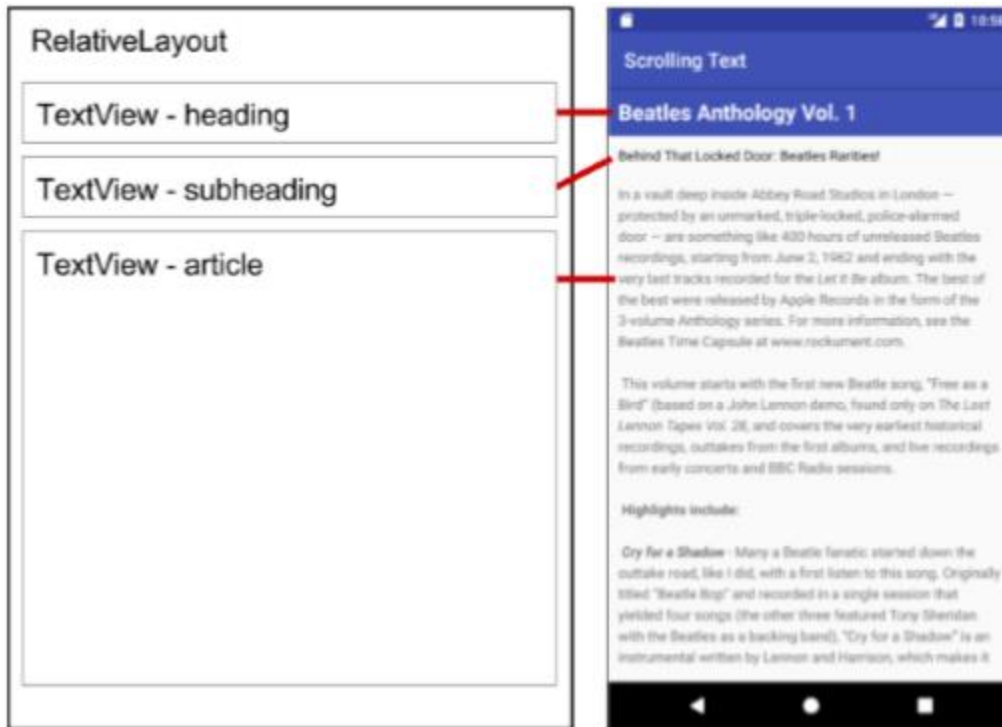


1. An active web link embedded in free-form text

2. The scroll bar that appears when scrolling the text

a. Thêm và chỉnh sửa các thành phần TextView

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



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

i. Tạo dự án và các thành phần TextView

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

1. Trong Android Studio, hãy tạo một dự án mới với các thông số sau:

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

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

Ở trên cùng hoặc gốc của hệ thống phân cấp View là ConstraintLayout ViewGroup

```
<androidx.constraintlayout.widget.ConstraintLayout
```

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

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

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

    <TextView
        android:id="@+id/article_heading"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/design_default_color_primary"
        android:padding="@dimen/padding_regular"
        android:text="@string/article_title"
        android:textAppearance="@android:style/TextAppearance.DeviceDefault.Large"
        android:textColor="@android:color/white"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/article_subheading"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/article_heading"
        android:padding="@dimen/padding_regular"
        android:text="@string/article_subtitle"
        android:textAppearance="@android:style/TextAppearance.DeviceDefault" />

    <TextView
        android:id="@+id/article"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/article_subheading"
        android:lineSpacingExtra="@dimen/line_spacing"
        android:padding="@dimen/padding_regular"
        android:text="@string/article_text" />

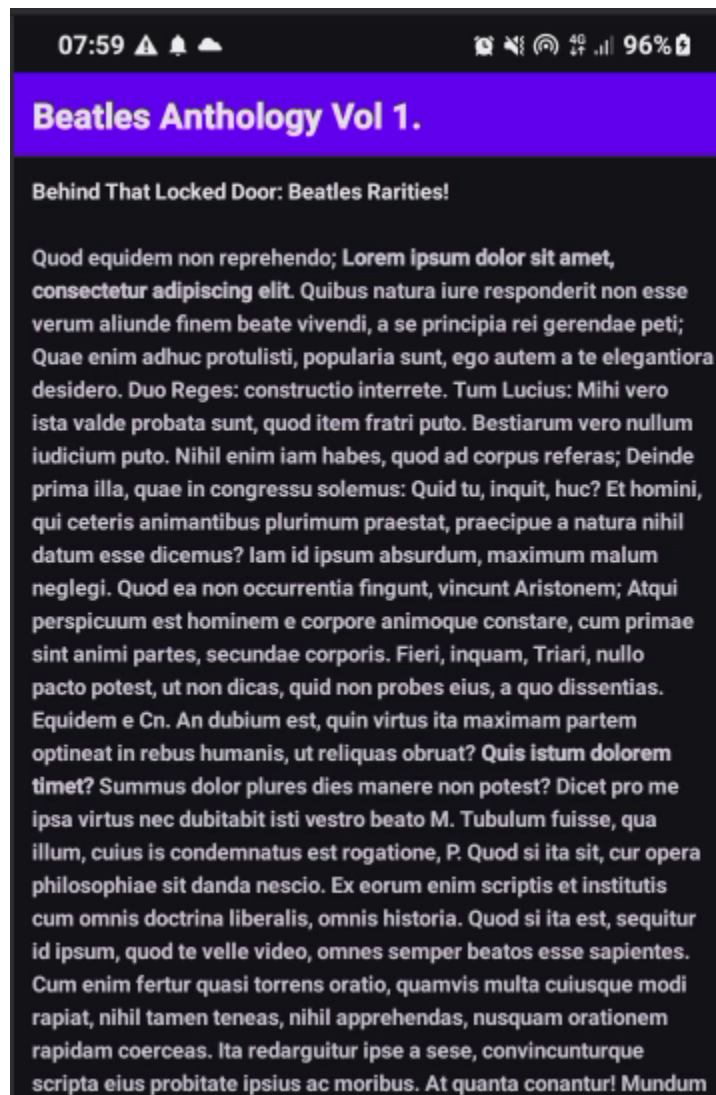
</RelativeLayout>
```

ii. Thêm văn bản của bài viết

```
</> activity_main.xml  </> strings.xml  MainActivity.java
1  <resources>
2      <string name="app_name">Scrolling Text</string>
3      <string name="article_title">Beatles Anthology Vol 1.</string>
4      <string name="article_subtitle">Behind That Locked Door: Beatles Rarities!</string>
5      <string name="article_text">Quod eguidem non reprehendo;
6      Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quibus natura iure responderit non esse verum aliunde finem beate vivendi,
7
8      Iam id ipsum absurdum, maximum malum negligi. Quod ea non occurrentia fingunt, vincunt Aristonem; Atqui perspicuum est hominem e co
9
10     Quis istum dolorem timet?
11     Summus dolor plures dies manere non potest? Dicet pro me ipsa virtus nec dubitabit isti vestro beato M. Tubulum fuisse, qua illum,
12
13     Ex eorum enim scriptis et institutis cum omnis doctrina liberalis, omnis historia.
14     Quod si ita est, sequitur id ipsum, quod te velle video, omnes semper beatos esse sapientes. Cum enim fertur quasi torrens oratio,
15
16     Quem si teneris, non modo meum Ciceronem, sed etiam me ipsum abducas licebit.
17     Stulti autem malorum memoria torquentur, sapientes bona praeterita grata recordatione renovata delectant.
18     Esse enim quam vellet iniquus iustus poterat inpune.
19     Quae autem natura suae primae institutionis oblita est?
20     Verum tamen cum de rebus grandioribus dicas, ipsae res verba rapiunt;
21     Hoc est non modo cor non habere, sed ne palatum quidem.
22     Voluptatem cum summum bonum diceret, primum in eo ipso parum vidit, deinde hoc quoque alienum; Sed tu istuc dixti bene Latine, parv
23
24     Idemne, quod iucunde?
25     Haec mihi videtur delicatior, ut ita dicam, molliorque ratio, quam virtutis vis gravitasque postulat. Sed quoniam et advesperascit
26     www.rockument.com</string>
27 </resources>
```

> values > </> strings.xml 25:354 LF UTF-8 4 spaces

iii. Chạy ứng dụng



b. Thêm ScrollView và web link đang hoạt động

i. Thêm thuộc tính autoLink cho các liên kết web đang hoạt động

```
<TextView
    android:id="@+id/article"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:autoLink="web"
    android:layout_below="@id/article_subheading"
    android:lineSpacingExtra="@dimen/line_spacing"
    android:padding="@dimen/padding_regular"
    android:text="@string/article_text" />
```

ii. Thêm ScrollView vào bố cục

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

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

        <TextView
            android:id="@+id/article_heading"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/design_default_color_primary"
            android:padding="@dimen/padding_regular"
            android:text="@string/article_title"
            android:textAppearance="@android:style/TextAppearance.DeviceDefault.Large"
            android:textStyle="bold" />

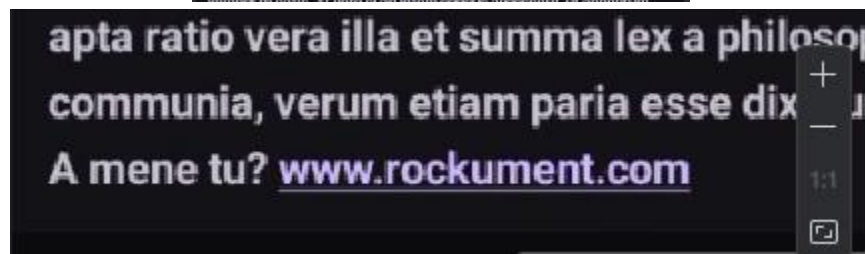
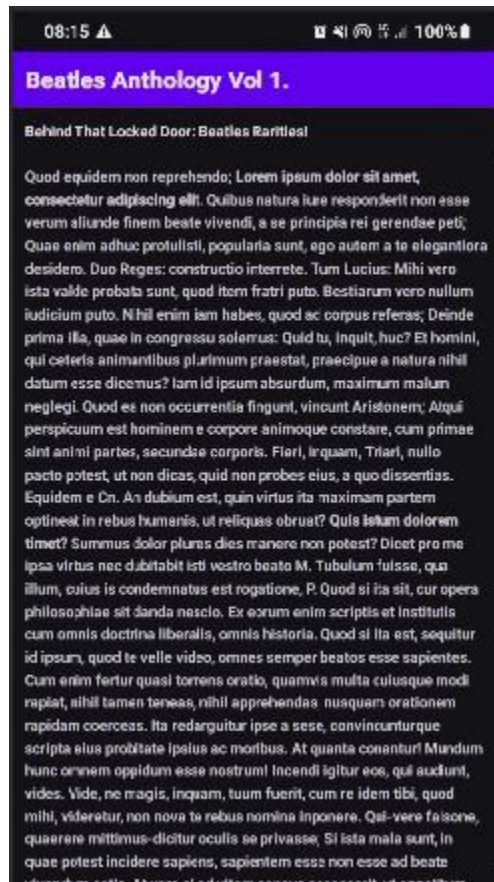
        <TextView
            android:id="@+id/article_subheading"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@id/article_heading"
            android:padding="@dimen/padding_regular"
            android:text="@string/article_subtitle"
            android:textAppearance="@android:style/TextAppearance.DeviceDefault" />

        <ScrollView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_below="@id/article_subheading">

            <TextView
                android:id="@+id/article"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:autoLink="web"
                android:lineSpacingExtra="@dimen/line_spacing"
                android:padding="@dimen/padding_regular"
                android:text="@string/article_text" />
        </ScrollView>

    </RelativeLayout>
```

iii. Chạy ứng dụng



c. Cuộn nhiều phần tử

i. Thêm LinearLayout vào ScrollView

```
<ScrollView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/article_subheading">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"></LinearLayout>
```

```
<ScrollView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/article_subheading">
```

```

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <TextView
        android:id="@+id/article"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:autoLink="web"
        android:lineSpacingExtra="@dimen/line_spacing"
        android:padding="@dimen/padding_regular"
        android:text="@string/article_text" />

</LinearLayout>
</ScrollView>

```

ii. Di chuyển các thành UI dùng trong LinearLayout

```

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

    <TextView
        android:id="@+id/article_heading"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/design_default_color_primary"
        android:padding="@dimen/padding_regular"
        android:text="@string/article_title"
        android:textAppearance="@android:style/TextAppearance.DeviceDefault.Large"
        android:textStyle="bold" />

    <ScrollView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/article_heading">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">
            <TextView
                android:id="@+id/article_subheading"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:padding="@dimen/padding_regular"
                android:text="@string/article_subtitle"
                android:textAppearance="@android:style/TextAppearance.DeviceDefault"
            />

            <TextView
                android:id="@+id/article"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:autoLink="web"
                android:lineSpacingExtra="@dimen/line_spacing"
                android:padding="@dimen/padding_regular"

```

```
        android:text="@string/article_text" />
    </LinearLayout>
</ScrollView>

</RelativeLayout>
```

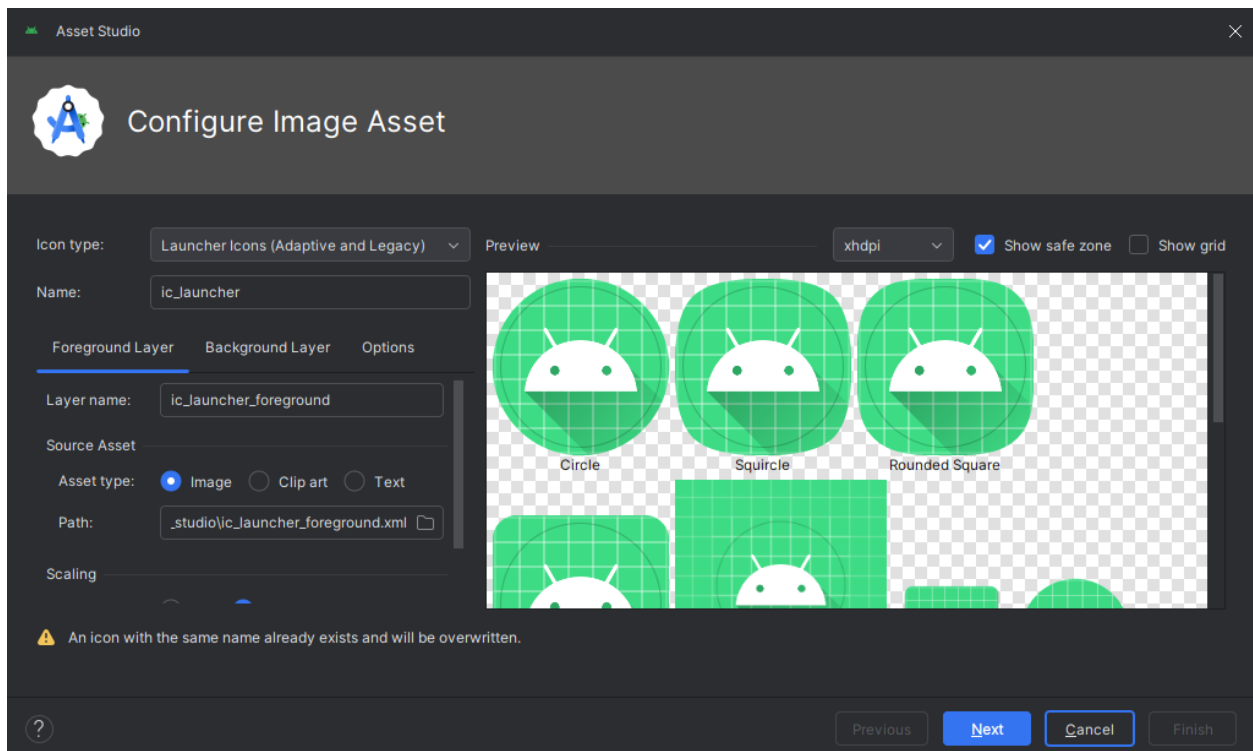
1.5) Tài nguyên có sẵn

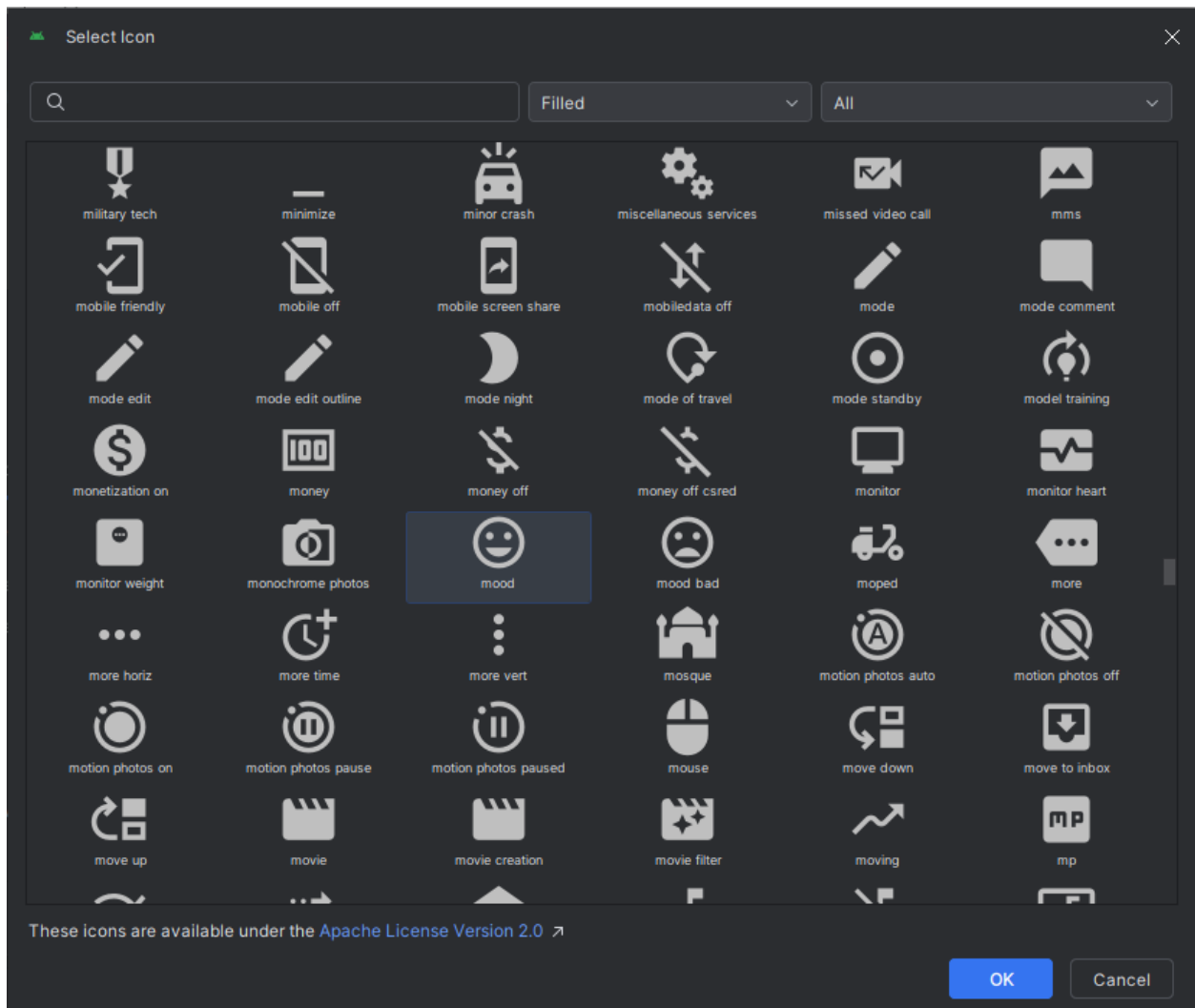
Giới thiệu

a. Thay đổi biểu tượng trình khởi chạy



- i. Khám phá tài liệu chính thức của Android
- ii. Thêm nội dung hình ảnh cho biểu tượng trình chạy





Chạy ứng dụng



b. Sử dụng project templates

i. Khám phá cấu trúc Basic Activity



ii. Tùy chỉnh ứng dụng do mẫu tạo

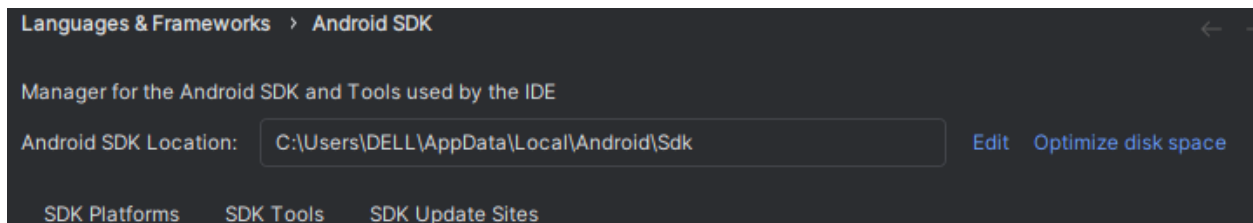


iii. Khám phá cách thêm activity bằng template

c. Học hỏi từ code mẫu

i. Android code samples

ii. Sử dụng Trình quản lý SDK để cài đặt tài liệu ngoại tuyến



d. Nhiều tài nguyên khác

i. Tìm kiếm trên Stack Overflow bằng thẻ

Bài 2) Activities

2.1) Activity và Intent

2.2) Vòng đời của Activity và trạng thái

2.3) Intent ngầm định

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

3.1) Trình gỡ lỗi

3.2) Kiểm thử đơn vị

3.3) Thư viện hỗ trợ

CHƯƠNG 2. TRẢI NGHIỆM NGƯỜI DÙNG

Bài 1) Tương tác người dùng

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

Bài 2) Trải nghiệm người dùng thú vị

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

Bài 3) Kiểm thử giao diện người dùng

- 3.1) Espresso cho việc kiểm tra UI**

CHƯƠNG 3. LÀM VIỆC TRONG NỀN

Bài 1) Các tác vụ nền

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

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

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

CHƯƠNG 4. LƯU DỮ LIỆU NGƯỜI DÙNG

Bài 1) Tùy chọn và cài đặt

1.1) Shared preferences

1.2) Cài đặt ứng dụng

Bài 2) Lưu trữ dữ liệu với Room

2.1) Room, LiveData và ViewModel

2.2) Room, LiveData và ViewModel