

TRƯỜNG ĐẠI HỌC THỦY LỢI
KHOA CÔNG NGHỆ THÔNG TIN



GIÁO TRÌNH
**THỰC HÀNH PHÁT TRIỂN ỨNG DỤNG CHO THIẾT BỊ DI
ĐỘNG**

Hà Nội, 2.2025

MỤC LỤC

CHƯƠNG 1. LÀM QUEN

3

| | |
|--|-----|
| Bài 1) Tạo ứng dụng đầu tiên | 3 |
| 1.1) Android Studio và Hello World | 3 |
| 1.2) Giao diện người dùng tương tác đầu tiên..... | 33 |
| 1.3) Trình chỉnh sửa bố cục | 65 |
| 1.4) Văn bản và các chế độ cuộn..... | 101 |
| 1.5) Tài nguyên có sẵn..... | 123 |
| Bài 2) Activities | 131 |
| 2.1) Activity và Intent..... | 131 |
| 2.2) Vòng đời của Activity và trạng thái..... | 131 |
| 2.3) Intent ngầm định | 131 |
| Bài 3) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ | 131 |
| 3.1) Trình gỡ lỗi..... | 131 |
| 3.2) Kiểm thử đơn vị | 131 |
| 3.3) Thư viện hỗ trợ | 131 |

CHƯƠNG 2. TRẢI NGHIỆM NGƯỜI DÙNG

131

| | |
|--|-----|
| Bài 1) Tương tác người dùng | 131 |
| 1.1) Hình ảnh có thể chọn..... | 131 |
| 1.2) Các điều khiển nhập liệu..... | 131 |
| 1.3) Menu và bộ chọn | 131 |
| 1.4) Điều hướng người dùng | 131 |
| 1.5) RecyclerView..... | 131 |
| Bài 2) Trải nghiệm người dùng thú vị | 131 |
| 2.1) Hình vẽ, định kiểu và chủ đề..... | 131 |
| 2.2) Thẻ và màu sắc | 131 |

| | |
|---|------------|
| 2.3) Bố cục thích ứng | 131 |
| Bài 3) Kiểm thử giao diện người dùng..... | 131 |
| 3.1) Espresso cho việc kiểm tra UI | 131 |
| CHƯƠNG 3. LÀM VIỆC TRONG NỀN | 131 |
| Bài 1) Các tác vụ nền..... | 132 |
| 1.1) AsyncTask | 132 |
| 1.2) AsyncTask và AsyncTaskLoader | 132 |
| 1.3) Broadcast receivers | 132 |
| Bài 2) Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền | 132 |
| 2.1) Thông báo | 132 |
| 2.2) Trình quản lý cảnh báo | 132 |
| 2.3) JobScheduler | 132 |
| CHƯƠNG 4. LƯU DỮ LIỆU NGƯỜI DÙNG | 132 |
| Bài 1) Tùy chọn và cài đặt | 132 |
| 1.1) Shared preferences | 132 |
| 1.2) Cài đặt ứng dụng | 132 |
| Bài 2) Lưu trữ dữ liệu với Room | 132 |
| 2.1) Room, LiveData và ViewModel..... | 132 |
| 2.2) Room, LiveData và ViewModel | 132 |

CHƯƠNG 1. LÀM QUEN

Bài 1) Tạo ứng dụng đầu tiên

1.1) Android Studio và Hello World

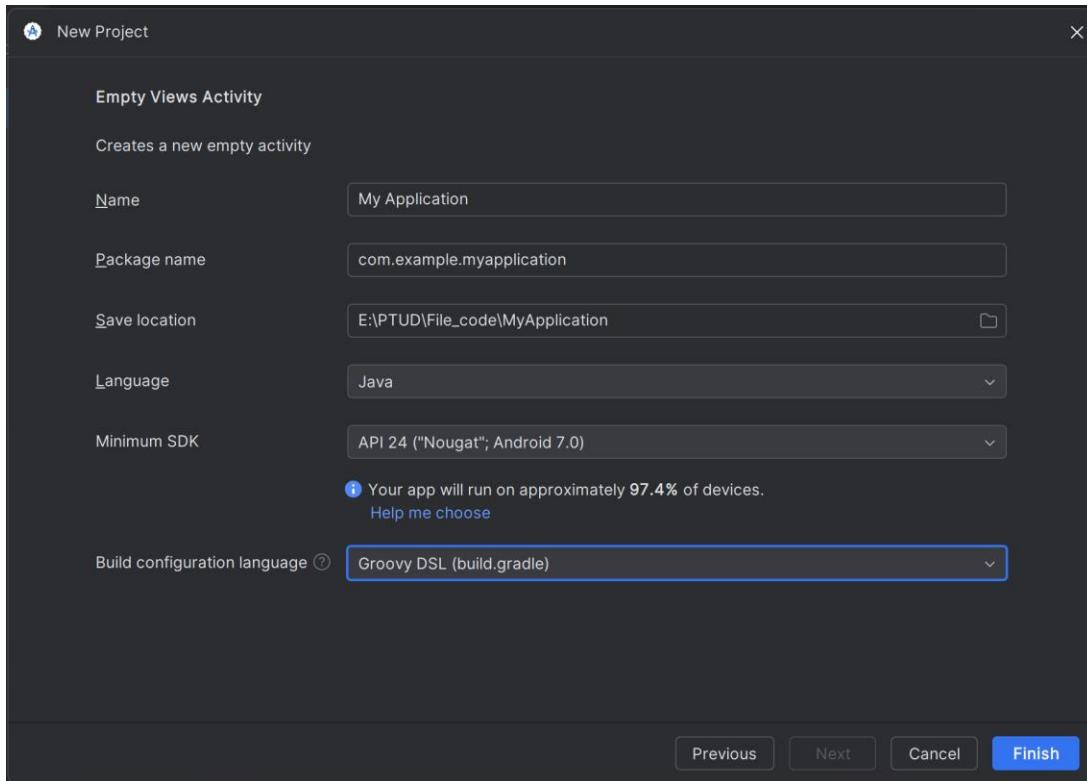
Giới thiệu

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 sẽ có thể:

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



Những gì Bạn sẽ cần:

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

Những gì bạn sẽ học

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

Những gì bạn sẽ làm

- Cài đặt môi trường phát triển **Android Studio**.
- Tạo một trình giả lập (thiết bị ảo) để chạy ứng dụng của bạn trên máy tính.
- Tạo và chạy ứng dụng **Hello World** trên các thiết bị ảo và vật lý.
- Khám phá cấu trúc dự án.
- Tạo và xem các thông điệp ghi lại từ ứng dụng của bạn.
- Khám phá tệp **AndroidManifest.xml**

Tổng quan về ứng dụng

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

Đây là ứng dụng đã hoàn thành sẽ như thế này:



Nhiệm vụ 1: Cài đặt Android Studio

Android Studio cung cấp một môi trường phát triển tích hợp hoàn chỉnh (IDE) bao gồm chỉnh sửa mã nâng cao và một bộ mẫu ứng dụng. Ngoài ra, nó còn chứa các công cụ để phát triển, gỡ lỗi, kiểm tra và hiệu suất giúp nó nhanh hơn và dễ dàng hơn để phát triển ứng dụng. Bạn có thể kiểm tra các ứng dụng của mình với một loạt các trình giả lập được cấu hình sẵn hoặc trên thiết bị di động của mình, xây dựng các ứng dụng sản xuất và xuất bản trên Google Play Store.

Android Studio có sẵn cho máy tính chạy Windows hoặc Linux, và cho máy Mac chạy macOS. Bộ phát triển phần mềm Java (JDK) mới nhất được tích hợp sẵn với Android Studio.

Để bắt đầu sử dụng Android Studio, trước tiên kiểm tra các yêu cầu hệ thống để đảm bảo hệ thống của bạn đáp ứng được chúng. Quá trình cài đặt tương tự nhau trên tất cả các nền tảng. Bất kỳ sự khác biệt nào sẽ được ghi chú bên dưới.

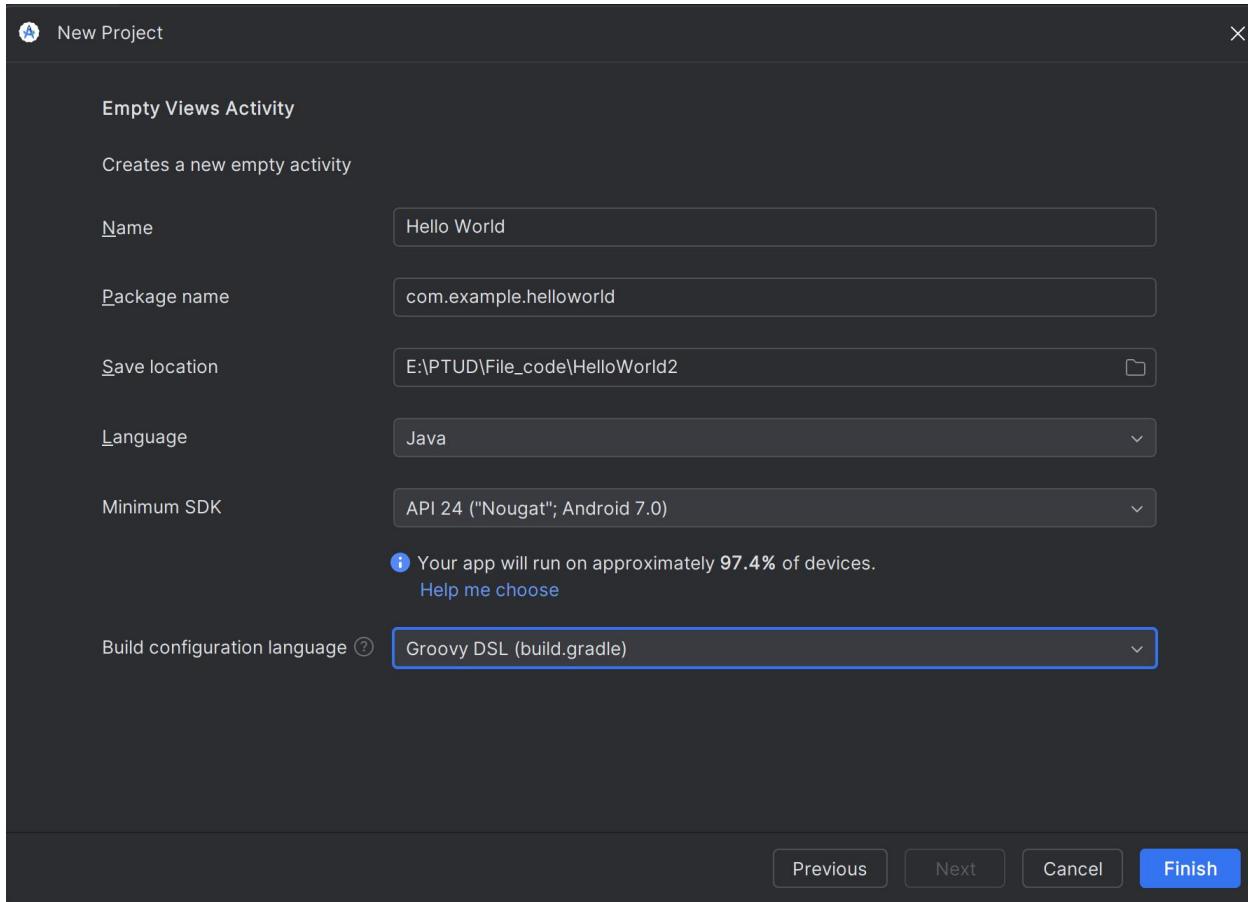
1. Điều hướng đến trang dành cho nhà phát triển Android và làm theo hướng dẫn để tải xuống và cài đặt Android Studio
2. Chấp nhận cấu hình mặc định cho tất cả các bước và đảm bảo rằng tất cả các thành phần đều được chọn để cài đặt
3. Sau khi hoàn tất cài đặt, Trình hướng dẫn thiết lập sẽ tải xuống và cài đặt một số thành phần bổ sung, bao gồm cả Android SDK. Hãy kiên nhẫn, điều này có thể mất một chút thời gian tùy thuộc vào tốc độ Internet của bạn và một số bước có thể thay đổi
4. Khi tải xuống hoàn tất, Android Studio sẽ khởi động và bạn đã sẵn sàng để tạo dự án đầu tiên của mình.

Nhiệm vụ 2: Tạo ứng dụng Hello World

Trong nhiệm vụ này, bạn sẽ tạo một ứng dụng hiển thị “Hello World” để xác nhận rằng Android Studio đã được cài đặt chính xác và để tìm hiểu những điều cơ bản của việc phát triển với Android Studio

2.1 Tạo dự án ứng dụng

1. Mở Android Studio nếu chưa mở
2. Trong cửa sổ chính chào mừng đến với Android Studio, nhập vào bắt đầu dự án Android Studio mới
3. Trong cửa sổ tạo dự án Android, nhập Hello World cho tên ứng dụng



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

5. Chấp nhận Android.example.com cho miền công ty hoặc tạo một miền công ty duy nhất.

Nếu bạn không có kế hoạch xuất ứng dụng của mình, bạn có thể chấp nhận mặc định. Xin lưu ý rằng việc thay đổi tên gói của ứng dụng của bạn sau này là công việc thêm.

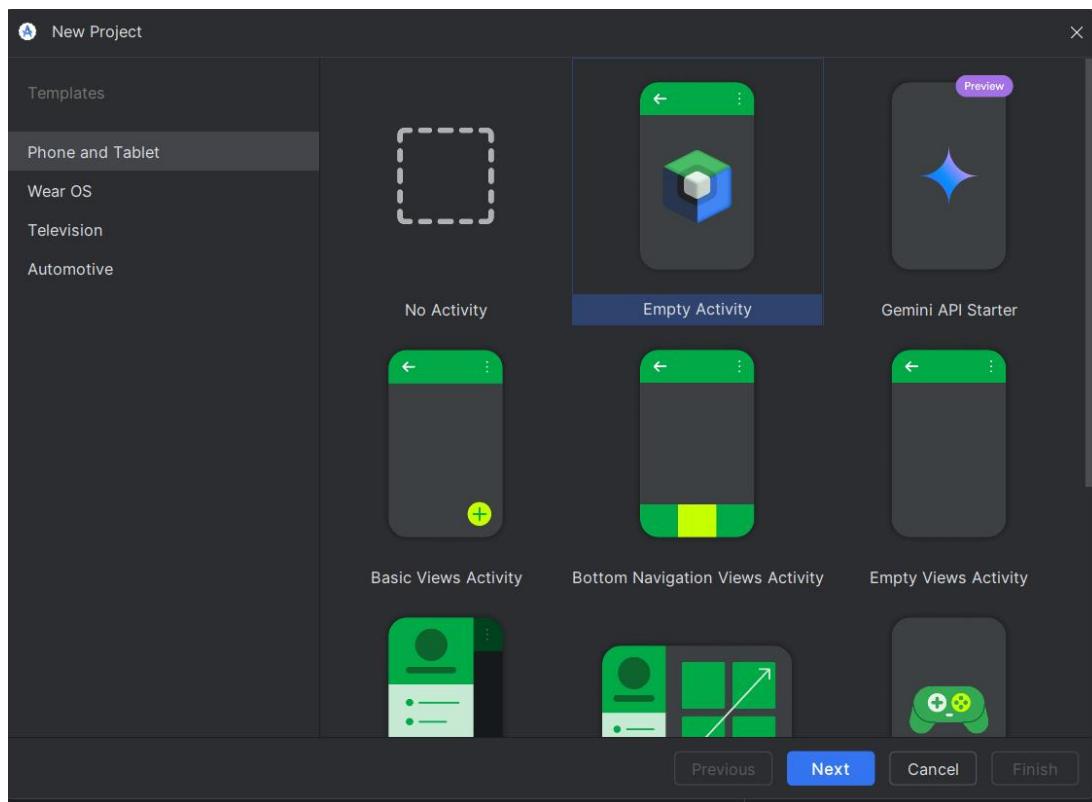
6. Bỏ chọn các tùy chọn bao gồm hỗ trợ C++ và bao gồm hỗ trợ Kotlin, và nhấp vào Tiếp theo

7. Trên màn hình Target Android thiết bị, điện thoại và máy tính bảng nên được chọn. Đảm bảo rằng API 15: Android 4.0.3 ICECreamSandwich được đặt làm SDK tối thiểu; Nếu không, hãy sử dụng menu bật lên để đặt nó

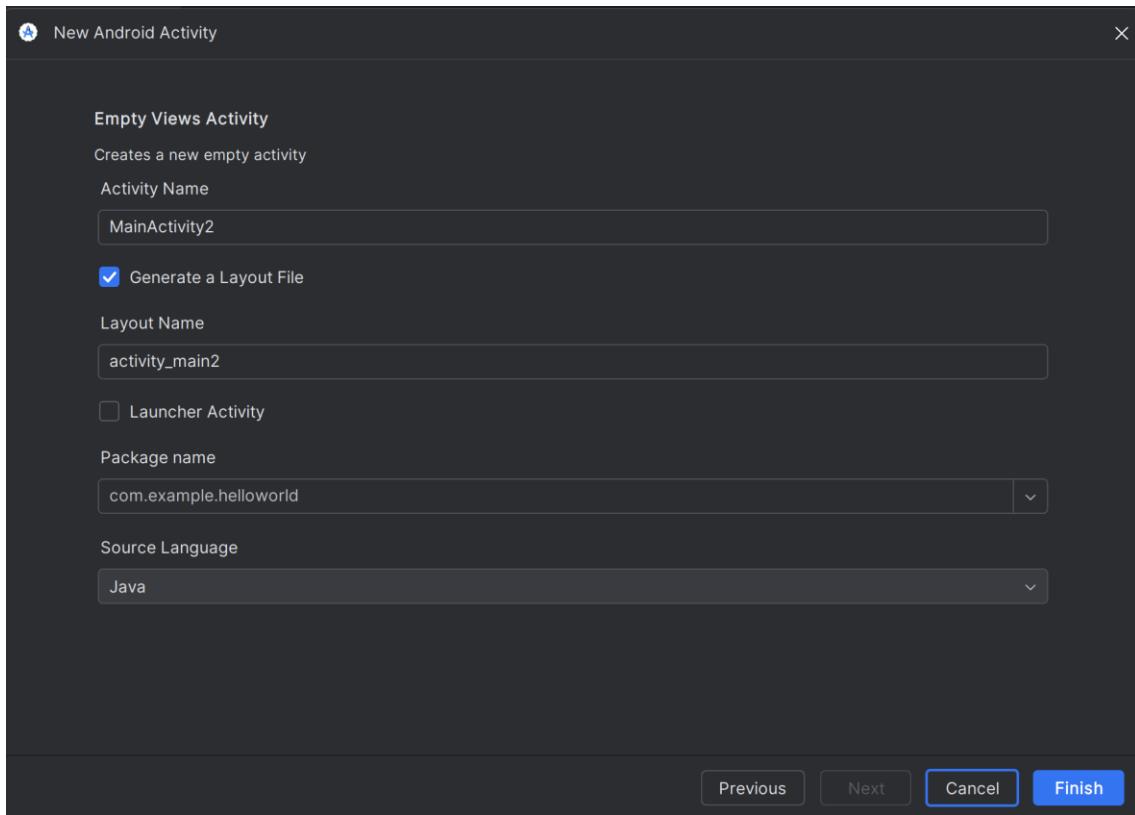
Đây là các cài đặt được sử dụng bởi các ví dụ trong các bài học cho khóa học này. Theo bài này, các cài đặt này làm cho ứng dụng Hello World của bạn tương thích với 97% thiết bị Android hoạt động trên Google Play Store.

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

9. Cửa sổ Thêm hoạt động xuất hiện. Hoạt động là một việc tập trung duy nhất mà người dùng có thể làm. Nó là một thành phần quan trọng của bất kỳ ứng dụng Android nào. Một hoạt động thường có bố cục được liên kết với nó xác định cách các thành phần giao diện người dùng xuất hiện trên màn hình. Android Studio cung cấp các mẫu Hoạt động để giúp bạn bắt đầu. Đối với dự án Hello World, chọn Empty Activity như hình dưới đây và nhấp vào Next



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



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

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

13. Nhấp vào Kết thúc.

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

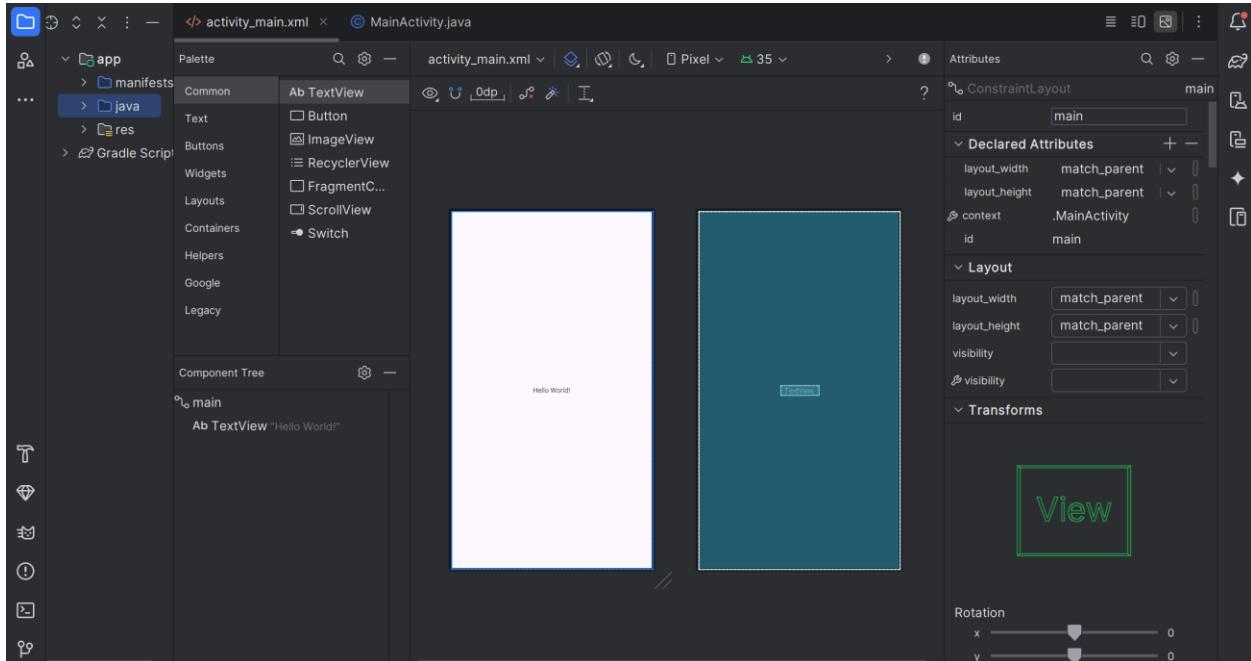
Mẹo: Xem trang Cấu hình nhà phát triển bản dựng của bạn để biết thông tin chi tiết.

Bạn cũng có thể thấy thông báo “Mẹo trong ngày” với các phím tắt và các mẹo hữu ích khác. Nhấp vào Đóng để đóng thông báo.

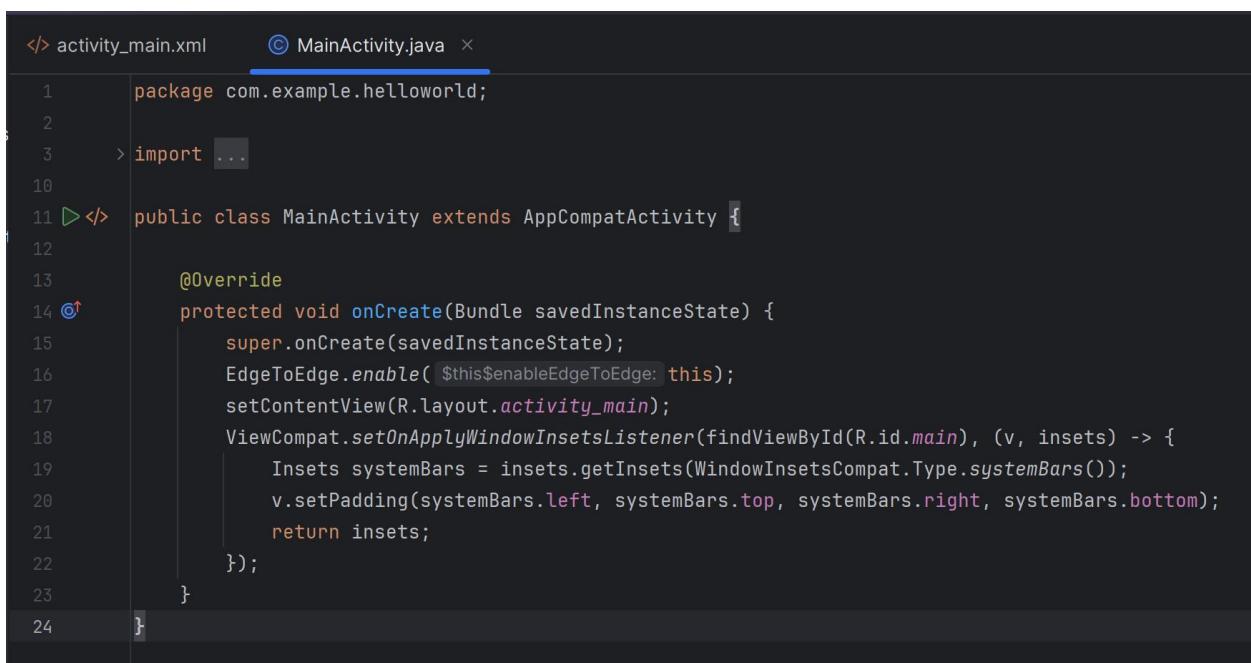
Trình chỉnh sửa Android Studio xuất hiện. Thực hiện theo các bước sau:

1. Nhấp vào tab **activity_main.xml** để xem trình chỉnh sửa bố cục

2. Nhập vào tab **Thiết kế** của trình chỉnh sửa bố cục , nếu chưa được chọn, để hiển thị bản đồ họa của bố cục như hình dưới đây



3. Nhập vào tab **MainActivity.java** để xem trình soạn thảo mã như hiển thị bên dưới

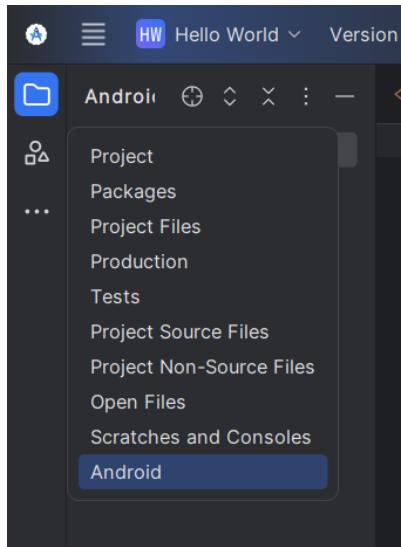


```
1 package com.example.helloworld;
2
3 > import ...
4
5 >>> public class MainActivity extends AppCompatActivity {
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
11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

2.2 Khám phá Project > Android

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

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

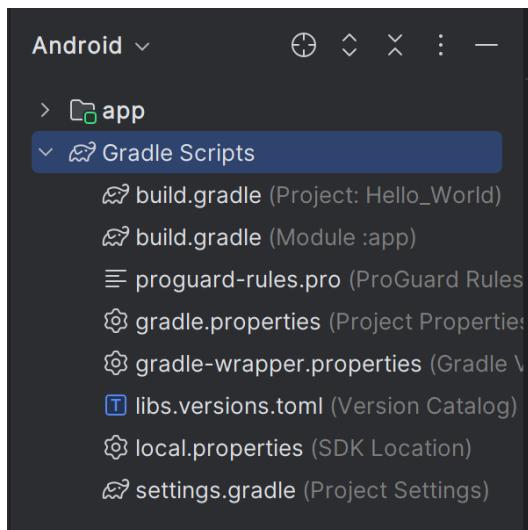


Lưu ý: Chương này và các chương khác đề cập đến ngăn Dự án khi được đặt thành Android là ngăn Dự án > Android

2.3 Khám phá thư mục Gradle Scripts

Hệ thống xây 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 phụ thuộc

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



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

1. Nếu thư mục **Gradle Scripts** không được mở rộng, hãy nhấp vào hình tam giác để mở rộng. Thư mục này chứa tất cả các tệp cần thiết cho hệ thống xây dựng.
2. Tìm tệp **build.gradle(Project: HelloWorld)**.

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

Theo mặc định, tệp dựng cấp cao nhất sử dụng khôi buildscript để xác định kho lưu trữ Gradle và các phụ thuộc chung cho tất cả các mô-đun trong dự án. Khi 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 khôi kho lưu trữ của tệp này. Theo mặc định, các dự án Android Studio mới khai báo JCenter và Google (bao gồm kho lưu trữ Google Maven) là các vị trí kho lưu trữ:

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

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

Ngoài tệp cấp dự án build.gradle, mỗi mô-đun đều có build.gradle tệp 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 cài đặt bản dựng này cho phép bạn cung cấp các tùy chọn đóng gói tùy chỉnh, chẳng hạn như các loại bản dựng bổ sung và hương vị sản phẩm. Bạn cũng có thể ghi đè cài đặt trong AndroidManifest.xml tệp hoặc cấp cao nhất build.gradle file.

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

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

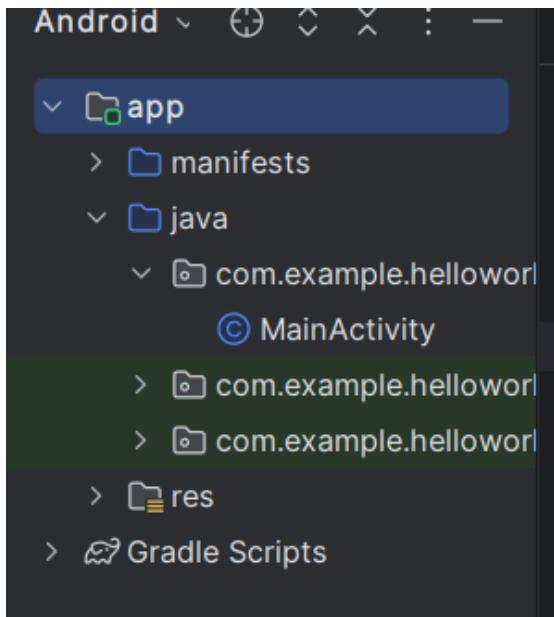
```
plugins {
    alias(libs.plugins.android.application)
}
android {
    namespace 'com.example.helloworld'
    compileSdk 35
    defaultConfig { DefaultConfig it ->
        applicationId "com.example.helloworld"
        minSdk 24
        targetSdk 35
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
    buildTypes { NamedDomainObjectContainer<BuildType> it ->
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions { CompileOptions it ->
        sourceCompatibility JavaVersion.VERSION_11
        targetCompatibility JavaVersion.VERSION_11
    }
}
dependencies {
    implementation libs.appcompat
    implementation libs.material
    implementation libs.activity
    implementation libs.constraintlayout
    testImplementation libs.junit
    androidTestImplementation libs.ext.junit
    androidTestImplementation libs.espresso.core
}
```

4. Nhập vào hình tam giác để đóng Gradle Scripts

2.4 Khám phá các thư mục ứng dụng và res

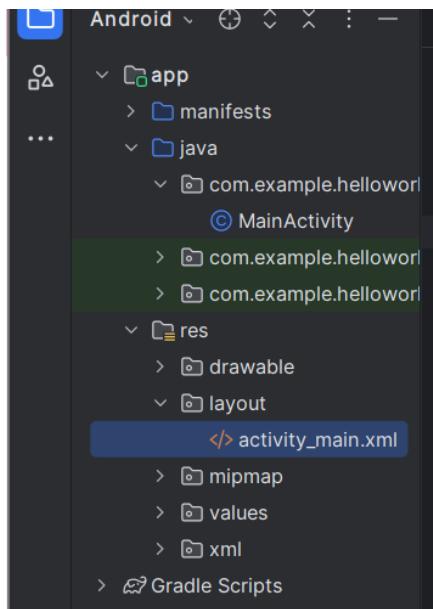
Tất cả mã và tài nguyên cho ứng dụng đều nằm trong các thư mục ứng dụng và res

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 để mở tệp 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 một gói ứng dụng. Hai thư mục khác được sử dụng để thử nghiệm và được mô tả trong 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 các 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`, sau đó nhấp đúp vào tệp `activity_main.xml` để mở tệp trong trình chỉnh sửa layout



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

2.5 Tìm hiểu 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 AndoirdManifests.xml

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

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

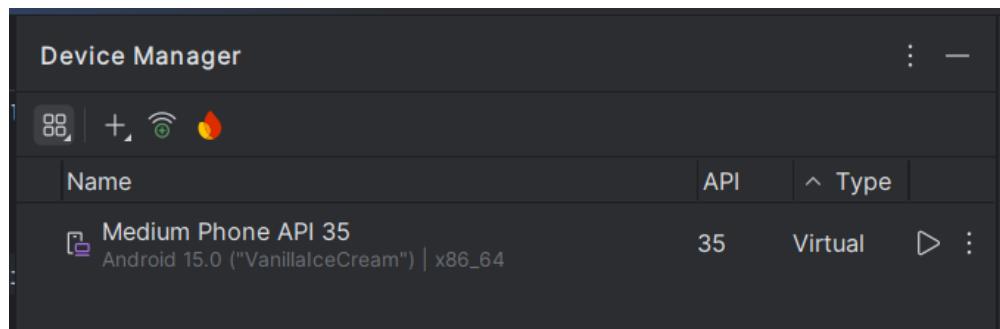
Trong tác vụ này, bạn sẽ sử dụng trình quản lý Thiết bị ảo Android (AVD) để tạo một thiết bị ảo (còn được gọi là trình giả lập) mô phỏng cấu hình cho một loại thiết bị Android cụ thể và sử dụng thiết bị ảo đó để chạy ứng dụng. Lưu ý rằng Trình giả lập Android có các yêu cầu bổ sung ngoài các yêu cầu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

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

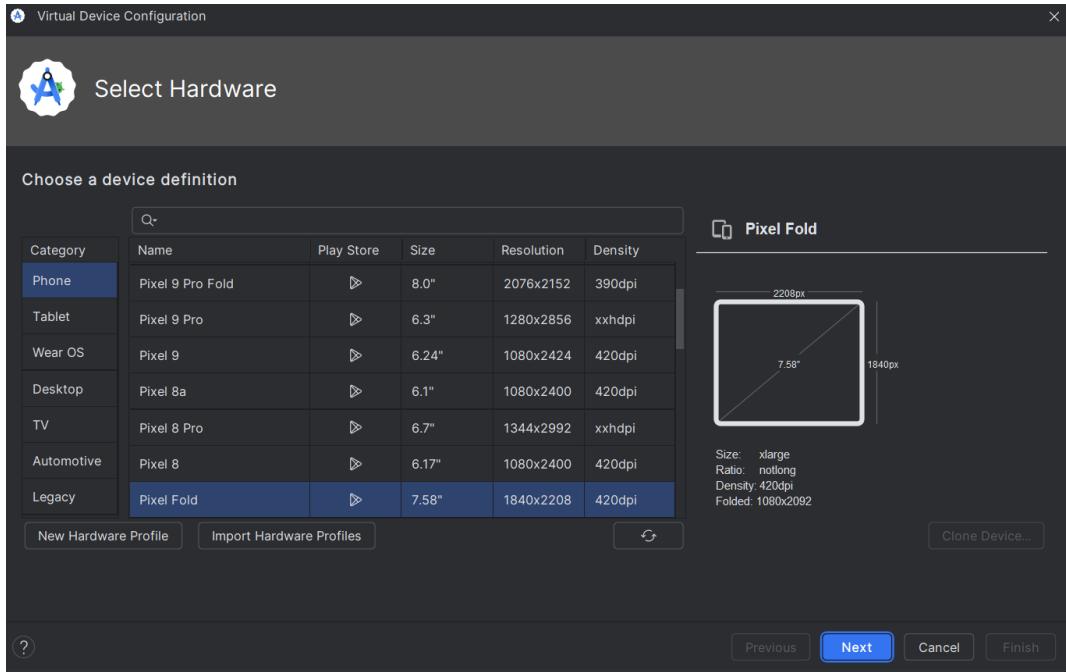
3.1 Tạo một thiết bị ảo Android (AVD)

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

1. Trong Android Studio, chọn Tools > Android > AVD Manager, hoặc nhấp vào biểu tượng AVD Manager trên thanh công cụ. Màn hình Your Virtual Devices xuất hiện. Nếu bạn đã tạo thiết bị ảo, màn hình sẽ hiển thị chúng; nếu không, bạn sẽ thấy màn hình sau:

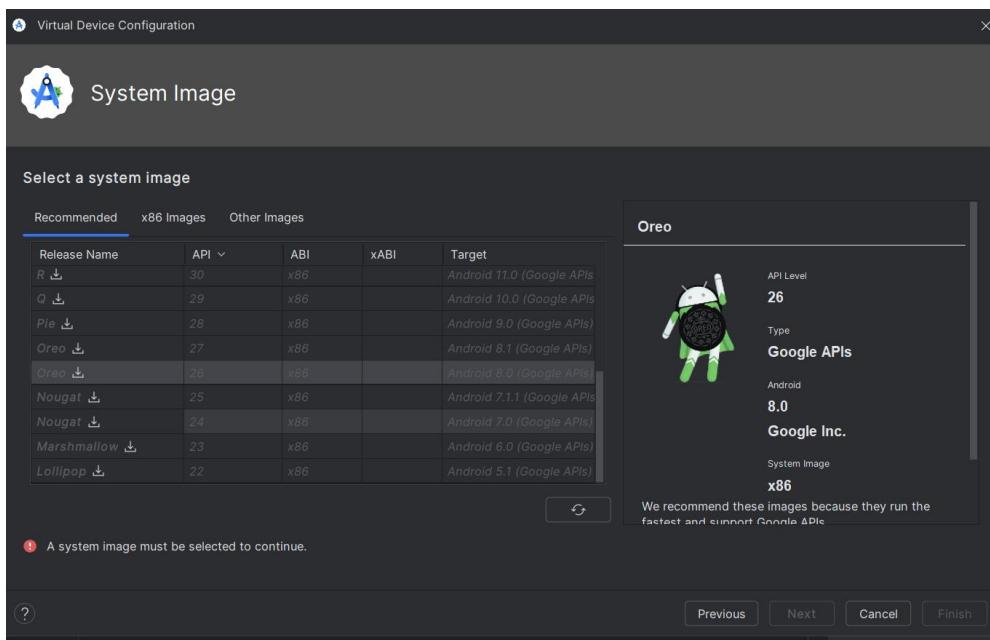


2. Nhấp vào +Create Virtual Device. Cửa sổ Select Hardware sẽ xuất hiện, hiển thị danh sách các thiết bị phần cứng được cấu hình trước. Đối với mỗi thiết bị, bảng cung cấp một cột cho kích thước hiển thị đường chéo (size), độ phân giải màn hình theo pixel (Resolution), và mật độ pixel (Density)



3. Chọn một thiết bị như Nexus 5X hoặc Pixel XL, và nhấp vào Tiếp theo. Màn hình ảnh hệ thống sẽ xuất hiện

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



Có nhiều phiên bản khả dụng hơn những phiên bản được hiển thị trong tab Recommended. Nhìn vào hình ảnh x86 và các hình ảnh khác để xem chúng

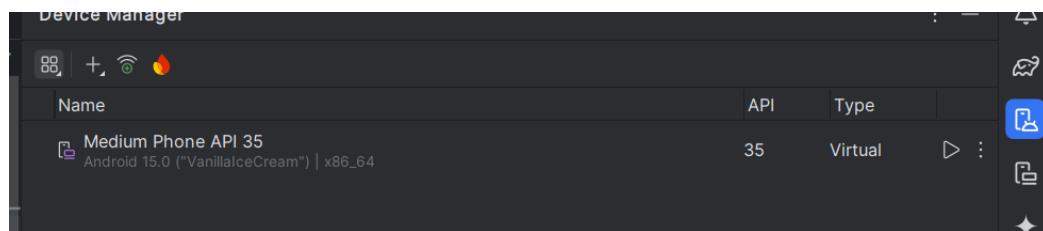
Nếu liên kết tải xuống hiển thị bên cạnh ảnh hệ thống bạn muốn sử dụng, thì ảnh đó vẫn chưa được cài đặt. Nhấp vào liên kết để bắt đầu tải xuống và nhấp vào kết thúc khi hoàn tất

5. Sau khi chọn một hình ảnh hệ thống, hãy nhấp vào Tiếp theo . Cửa sổ Thiết bị ảo Android (AVD) xuất hiện. Bạn cũng có thể thay đổi tên của AVD. Kiểm tra cấu hình của bạn và nhấp vào Kết thúc

3.2 Chạy ứng dụng trên thiết bị ảo

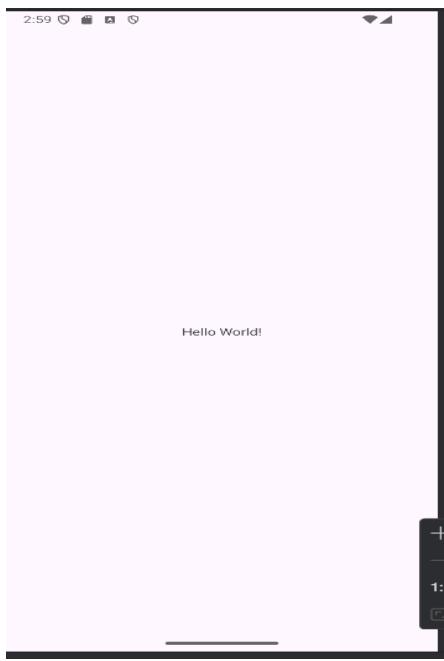
Trong nhiệm vụ này, cuối cùng bạn sẽ chạy ứng dụng Hello World của mình.

1. Trong Android Studio, chọn Run > Run app hoặc nhấp vào biểu tượng chạy trên thanh công cụ
2. Cửa sổ Select Deployment Target, trong Available Virtual Devices, chọn thiết bị ảo mà bạn vừa tạo và nhấp vào OK



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

Bạn sẽ thấy ứng dụng Hello World như hình dưới đây



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

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

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

Những gì bạn cần:

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

4.1 Bật gỡ lỗi USB

Để Android Studio giao tiếp với thiết bị của bạn, bạn phải bật USB Debugging trên thiết bị Android của mình. Tính năng này được bật trong cài đặt **tùy chọn nhà phát triển** của thiết bị

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

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

4.2 Chạy ứng dụng trên thiết bị của bạn

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

1. Kết nối thiết bị của bạn với máy phát triển bằng cáp USB.
2. Bấm vào nút Run trên thanh công cụ. Cửa sổ Select Deployment Target 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 OK

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

Xử lý sự cố

Nếu Android Studio không nhận ra thiết bị của bạn, thử cách sau:

1. Rút dây cáp và cắm lại thiết bị
2. Khởi động lại Android Studio

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

1. Rút dây cáp của thiết bị
2. Trên thiết bị, mở **Tùy chọn nhà phát triển** trong **ứng dụng cài đặt**

3. Nhấn vào thu hồi quyền gỡ lỗi USB
4. Kết nối lại thiết bị với máy tính
5. Khi được nhắc, hãy cấp quyền

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

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

Trong nhiệm vụ này, bạn sẽ thay đổi một số thông tin về cấu hình ứng dụng trong tệp build.gradle(Module:app) để tìm hiểu cách thực hiện thay đổi và đồng bộ hóa chúng với dự án Android Studio của bạn.

5.1 Thay đổi phiên bản SDK tối thiểu cho ứng dụng

Thực hiện theo các bước sau:

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

Nội dung của tệp sẽ xuất hiện trong trình soạn thảo mã

2. Trong khối defaultConfig, thay đổi giá trị của minSdkVersion thành 17 như hình dưới đây (ban đầu giá trị này được đặt thành 15)

```

gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.

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

android {
    namespace 'com.example.helloworld'
    compileSdk 35
    defaultConfig {
        applicationId "com.example.helloworld"
        minSdk 17
        targetSdk 35
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }
}

```

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

5.2 Đồng bộ hóa cấu hình Gradle mới

Khi bạn thực hiện thay đổi đối với tệp cấu hình bản dựng trong một dự án, Android Studio yêu cầu bạn đồng bộ hóa các tệp dự án để có thể nhập các

thay đổi cấu hình bản dựng và chạy một số kiểm tra để đảm bảo cấu hình sẽ không tạo ra lỗi bản dựng.

Để đồng bộ các tệp dự án, hãy nhấp vào **Đồng bộ ngay** trên thanh thông báo xuất hiện khi thực hiện thay đổi (như hiển thị trong hình trước) hoặc nhấp vào biểu tượng **Đồng bộ dự án với tệp Gradle**  trên thanh công cụ

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

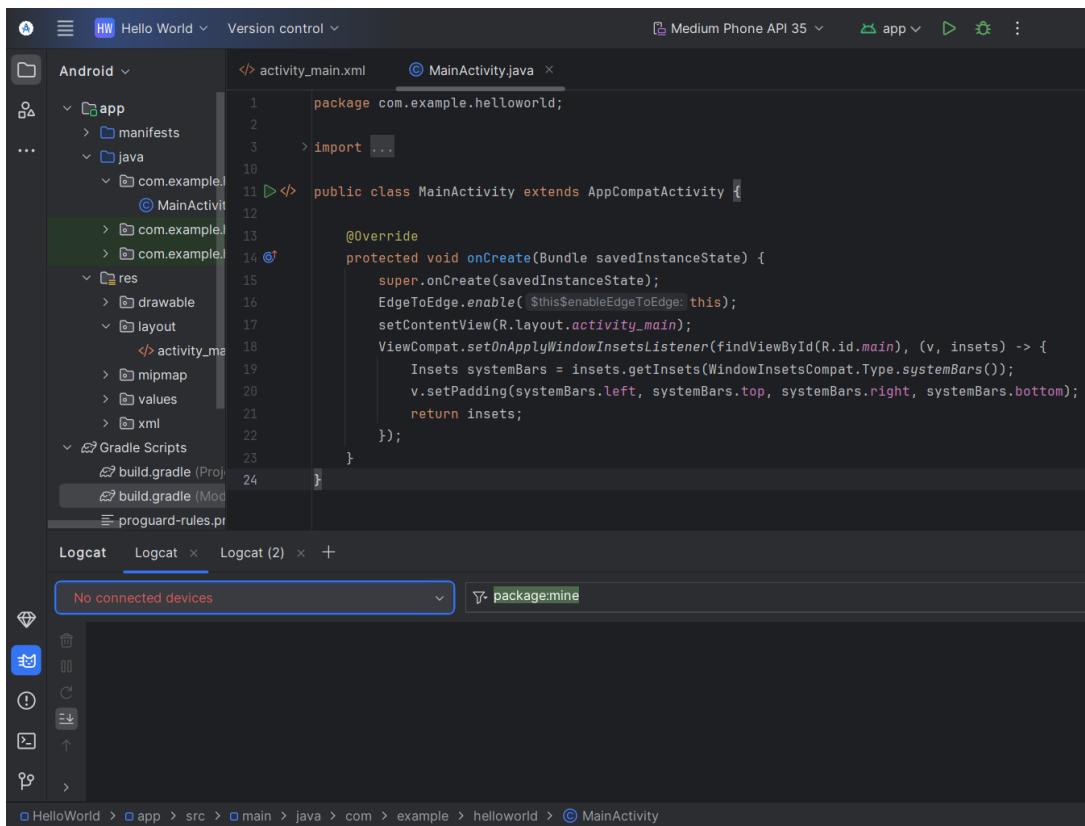
Để hiểu sâu hơn về Gradle, hãy tham khảo tài liệu Tổng quan về hệ thống xây dựng và Cấu hình bản dựng Gradle

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

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

6.1 Xem ngăn Logcat

Để xem ngăn **Logcat**, hãy nhấp vào tab **Logcat** ở cuối cửa sổ Android Studio như minh họa 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 Log vào ứng dụng, thông báo Log sẽ xuất hiện ở đây.
2. Menu cấp độ Log được đặt thành Verbose (mặc định), hiển thị tất cả các thông báo Log. Các thiết lập khác bao gồm Debug , Error , Info và Warn

6.2 Thêm câu lệnh log vào ứng dụng của bạn

Các câu lệnh Log trong mã ứng dụng của bạn hiển thị thông báo trong ngăn Logcat. Ví dụ:

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

Các phần của tin nhắn bao gồm:

- Log: Lớp Log để gửi tin nhắn nhật ký đến ngăn Logcat

- d: Cài đặt mức gỡ lỗi (Debug) Log để lọc thông báo Log hiển thị trong ngăn Logcat. Các cấp độ Log khác là e cho lỗi (Error), w cho cảnh báo (Warn) và i cho thông tin (Info).
- "MainActivity": Đối số đầu tiên là một thẻ có thể được sử dụng để lọc tin nhắn trong ngăn Logcat. Đây thường là tên của Activity 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 log được định nghĩa là hằng số cho Activity:

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

- Hello world":Đối số thứ hai là thông điệp thực tế được hiển thị. Thực hiện 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 mục android.util.Log nhập bắt buộc khi sử dụng Log), hãy chọn **File > Settings** trong Windows hoặc **Android Studio > Preferences** trong macOS

3. Chọn **Editor > General >Auto Import**. Chọn tất cả các hộp kiểm và đặt **Insert imports on paste** thành **All**.

4. Nhấp vào Apply và nhấp vào OK

5. Trong phương thức onCreate() của MainActivity, thêm câu lệnh sau:

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

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

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    Log.d("MainActivity", "Hello World");  
}
```

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 Log trong ngăn Logcat thành Debug (hoặc giữ nguyên mức Verbose vì có rất ít thông báo Log)

9. Chạy ứng dụng của bạn

Thông báo sẽ xuất hiện trong ngăn Logcat:

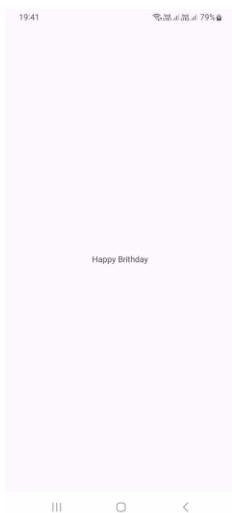
```
2025-03-12 11:08:03.907 10044-10044 MainActivity com.example.helloworld D [Hello World]
```

Coding challenge

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

Thách thức: Bây giờ bạn đã thiết lập xong và quen thuộc với quy trình phát triển cơ bản, hãy thực hiện như sau:

1. Tạo một dự án mới trong Android Studio
2. Thay đổi “Hello World” thành “Happy Birthday” và tên của người có sinh nhật gần đây.
3. (Tùy chọn) Chụp ảnh màn hình ứng dụng đã hoàn thành của bạn và gửi email cho người mà bạn quên ngày sinh.



4. Một cách sử dụng phổ biến của lớp Log là ghi nhật ký các ngoại lệ Java khi chúng xảy ra trong chương trình của bạn. Có một số phương thức hữu ích, chẳng hạn như Log.e(), mà bạn có thể sử dụng cho mục đích này. Khám phá các phương thức bạn có thể sử dụng để bao gồm ngoại lệ với thông báo Nhật ký. Sau đó, viết mã trong ứng dụng của bạn để kích hoạt và ghi nhật ký ngoại lệ

Tổng kết

- Để cài đặt Android Studio, hãy vào Android Studio và làm theo hướng dẫn để tải xuống và cài đặt.
- Khi tạo ứng dụng mới, hãy đảm bảo rằng API 15: Android 4.0.3 IceCreamSandwich được đặt làm SDK tối thiểu.
- Để xem hệ thống phân cấp Android của ứng dụng trong ngăn Dự án, hãy nhấp vào tab Dự án trong cột tab dọc, sau đó chọn Android trong menu bật lên ở trên cùng.
- Chỉnh sửa build.gradle(Module:app)tệp khi bạn cần thêm thư viện mới vào dự án hoặc thay đổi phiên bản thư viện.
- Tất cả mã và tài nguyên cho ứng dụng đều nằm trong các thư mục appvà res. Thư mục bao gồm các hoạt động, thử nghiệm và các thành phần khác trong mã nguồn Java. resThư mục chứa các tài nguyên, chẳng hạn như bộ cục, chuỗi và hình ảnh.
- Chỉnh sửa AndroidManifest.xmltệp để thêm các thành phần tính năng và quyền vào ứng dụng Android của bạn. Tất cả các thành phần cho một ứng dụng, chẳng hạn như nhiều hoạt động, phải được khai báo trong tệp XML này.
- Sử dụng trình quản lý Thiết bị ảo Android (AVD) để tạo thiết bị ảo (còn gọi là trình giả lập) để chạy ứng dụng của bạn.
- Thêm Logcác câu lệnh vào ứng dụng của bạn để hiển thị thông báo trong ngăn Logcat như một công cụ cơ bản để gỡ lỗi.
- Để chạy ứng dụng của bạn trên thiết bị Android vật lý bằng Android Studio, hãy bật Gỡ lỗi USB trên thiết bị. Mở Cài đặt > Giới thiệu về điện thoại và chạm vào Số bản dựng bảy lần. Quay lại màn hình trước đó (Cài đặt) và chạm vào Tùy chọn nhà phát triển . Chọn Gỡ lỗi USB

Các khái niệm liên quan

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

Tìm hiểu thêm

Tài liệu Android Studio

- Trang tải xuống Android Studio
- Ghi chú phát hành Android Studio

- Làm quen Android Studio
- Công cụ dòng lệnh Logcat
- Thiết bị ảo (AVD)
- Tổng quan về ứng dụng
- Cấu hình bản dựng của bạn
- Lớp Log
- Tạo và quản lý các thiết bị ảo

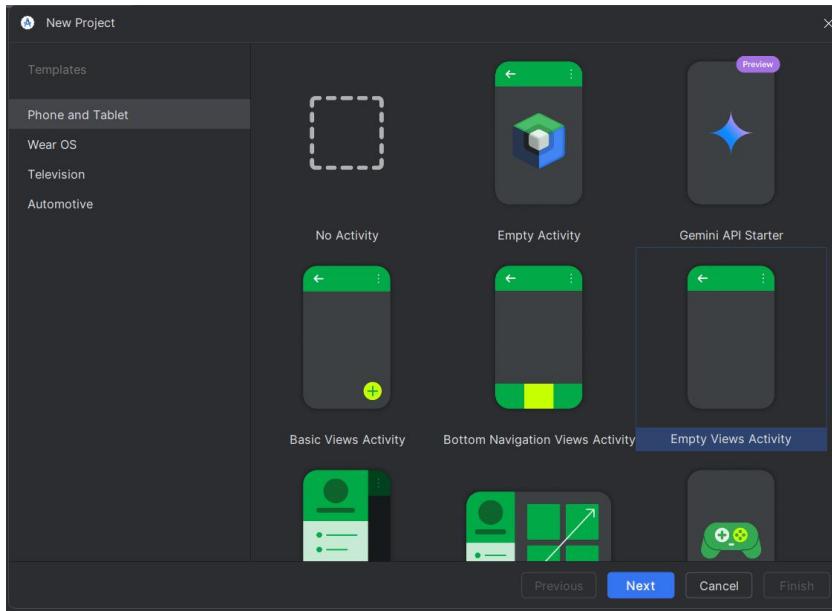
Khác:

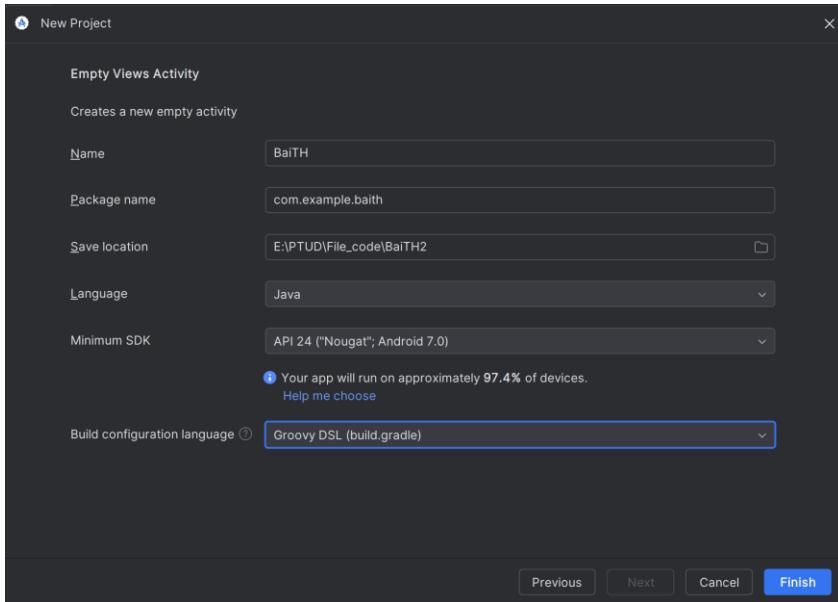
- Làm cách nào cài đặt Java?
- Cài đặt phần mềm JDK và cài đặt JAVA_HOME
- Trang web Gradle
- Cú pháp Apache Groovy
- Trang Wikipedia Gradle

Bài tập

Tạo và chạy một ứng dụng

- Tạo một dự án Android mới từ mẫu





- Thêm câu lệnh Log cho các mức Log khác nhau trong onCreate() trong hoạt động chính

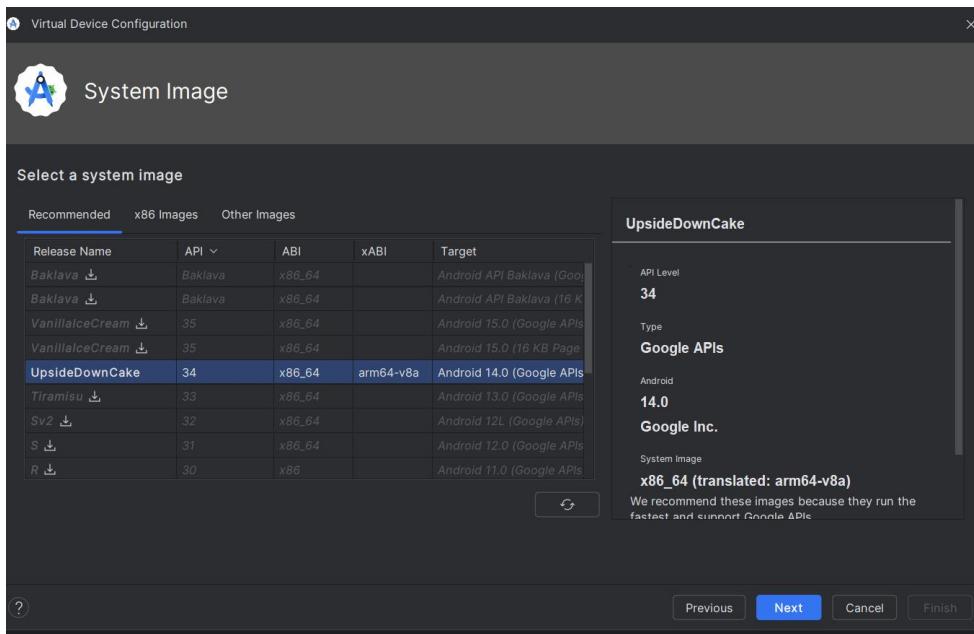
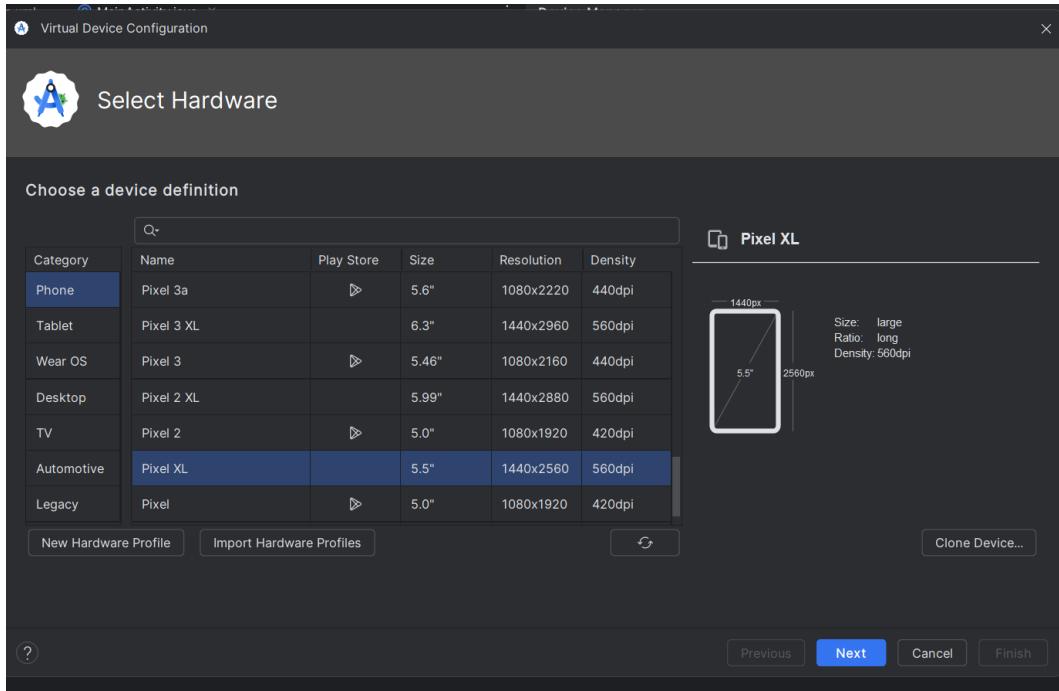
```

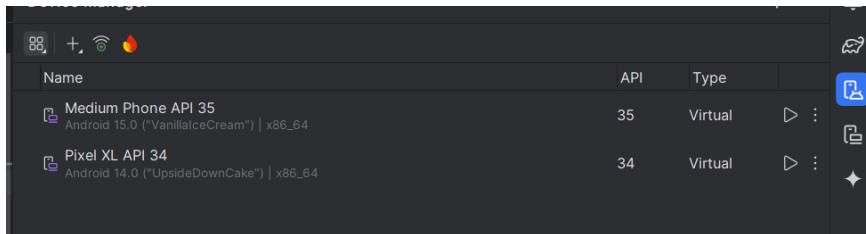
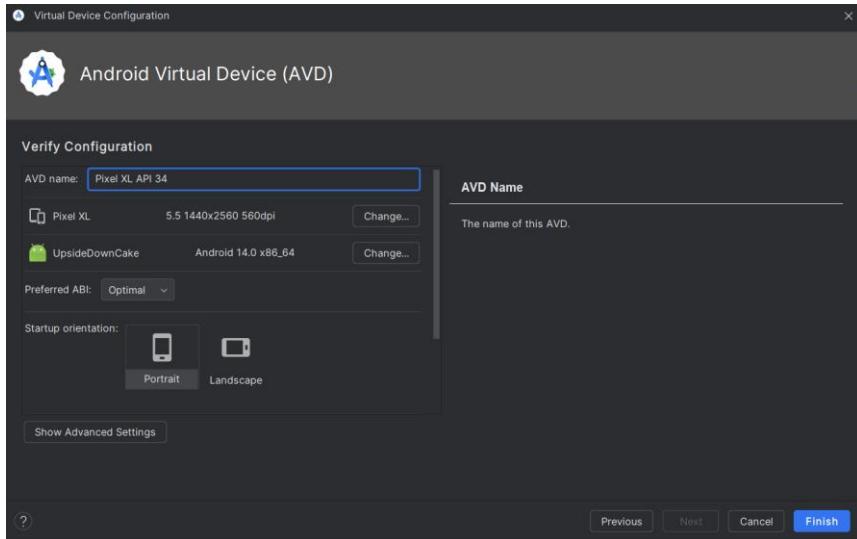
</> activity_main.xml  MainActivity.java <-
1 package com.example.baith;
2
3 > import ...
11
12 <></> public class MainActivity extends AppCompatActivity {
13
14     1 usage
15     private Log log;
16
17     @Override
18     protected void onCreate(Bundle savedInstanceState) {
19         super.onCreate(savedInstanceState);
20         EdgeToEdge.enable( $this$enableEdgeToEdge: this );
21         setContentView(R.layout.activity_main);
22         log.d( tag: "MainActivity", msg: "BaiTH" );
}

```

2025-03-12 11:14:43.951 12196-12196 MainActivity com.example.baith D | BaiTH

- Tạo trình giả lập cho thiết bị, chọn bất kỳ phiên bản Android nào bạn thích và chạy ứng dụng





- Sử dụng bộ lọc trong Logcat để tìm các câu lệnh Log của bạn và điều chỉnh các cấp độ để chỉ hiển thị các câu lệnh gỡ lỗi hoặc ghi lỗi

Trả lời các câu hỏi sau

Câu hỏi 1

Tên của tệp layout cho main activity là gì?

- MainActivity.java
- AndroidManifest.xml
- **Activity_main.xml**
- Build.gradle

Câu hỏi 2

Tên của resource string xác định tên ứng dụng là gì?

- app_name
- xmlns:app
- android:name

- applicationId

Câu hỏi 3

Công cụ nào để tạo trình giả lập mới?

- Android Device Monitor
- **AVD Manager**
- SDK Manager
- Theme Editor

Câu hỏi 4

Giả sử ứng dụng của bạn có chứa câu lệnh log sau đây:

Log.i (“MainActivity”, “MainActivity layout is complete”);

Bạn sẽ thấy dòng thông báo “MainActivity layout is complete” trong ngăn Logcat nếu menu Log được đặt ở mức nào trong các lựa chọn sau? (Gợi ý: nhiều câu trả lời đúng)

- **Verbose**
- Debug
- **Info**
- Warn
- Error
- Assert

Gửi ứng dụng của bạn để chấm điểm

Đảm bảo kiểm tra ứng dụng có những điều sau:

- Hoạt động hiển thị “Hello World” trên màn hình
- Các câu lệnh Log trong onCreate() trong hoạt động chính
- Mức Log trong ngăn Logcat hiển thị các câu lệnh gỡ lỗi hoặc ghi lỗi

1.2) Giao diện người dùng tương tác đầu tiên

Giới thiệu

Giao diện người dùng (UI) xuất hiện trên màn hình của thiết bị Android bao gồm một hệ thống phân cấp các đối tượng được gọi là view – mỗi phần tử trên màn hình là một view. Lớp view biểu thị khối xây dựng cơ bản cho tất

cả các thành phần UI, và là 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 để trình bày trong nhiều bài học:

- TextView : Hiển thị văn bản.
- EditText : cho phép người dùng nhập và chỉnh sửa văn bản.
- Button và các thành phần có thể nhấp khác (như RadioButton, Checkbox và Spinner) : để cung cấp hành vi tương tác.
- ScrollView và RecyclerView : để hiển thị các mục có thể cuộn được.
- ImageView : để hiện thị hình ảnh.
- ConstraintLayout và LinearLayout : để chứa các thành phần View khác và định vị của chúng.

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

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

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

Trong bài thực hành này, bạn sẽ học cách tạo ứng dụng tương tác đầu tiên của mình - một ứng dụng cho phép người dùng tương tác. Bạn sẽ tạo một ứng dụng sử dụng mẫu Empty Activity. Bạn cũng sẽ học cách sử dụng trình chỉnh sửa bố cục để thiết kế một 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 nên biết:

Bạn nên có quen thuộc 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 với hành vi tương tác.
- Cách sử dụng trình chỉnh sửa bố cục để thiết kế một bố cục
- Cách chỉnh sửa bố cục trong XML.

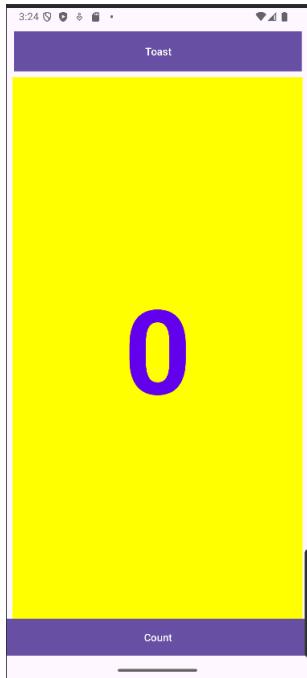
Những gì Bạn sẽ làm:

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

Tổng quan về ứng dụng

Ứng dụng HelloToast bao gồm hai thành phần Button và TextView. Khi người dùng nhấn vào Button đầu tiên, ứng dụng sẽ hiển thị một thông điệp ngắn(một Toast) trên màn hình. Việc nhấn vào button thứ hai sẽ tăng một bộ đếm “nhấp chuột” được hiển thị trong TextView, bắt đầu từ số 0.

Dưới đây là mô tả về giao diện của ứng dụng hoàn chỉnh:

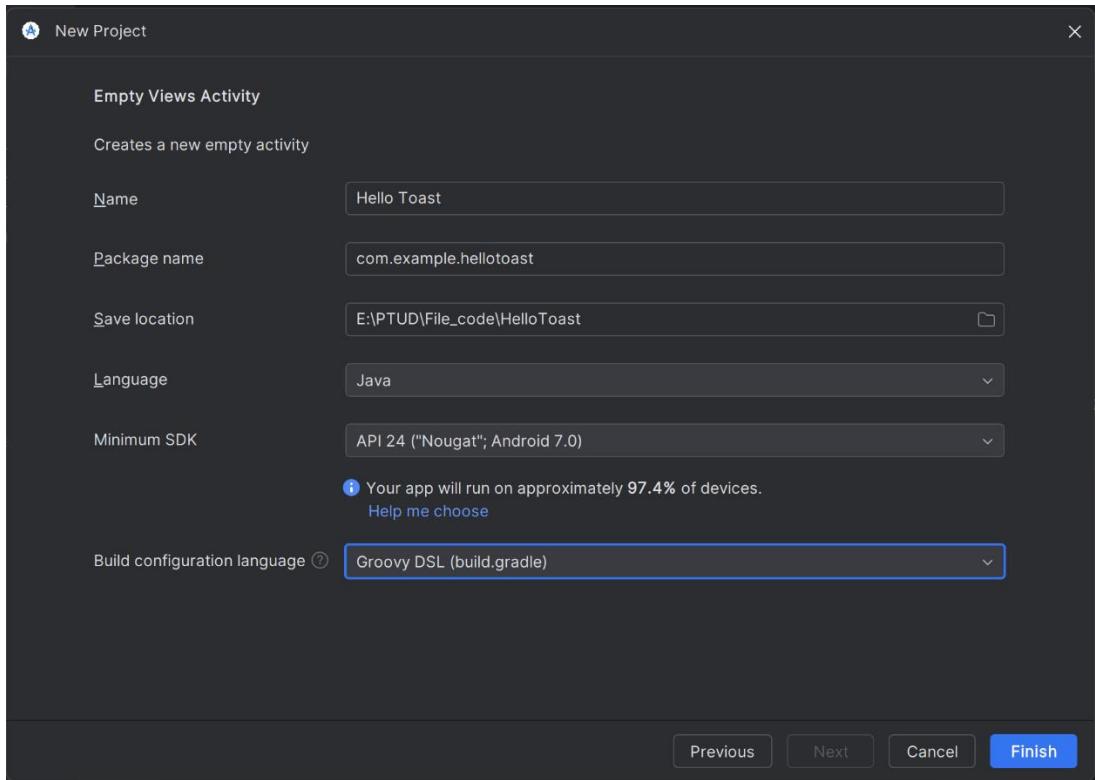


Nhiệm vụ 1: Tạo và khám phá một dự án mới

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

1.1 Tạo dự án Android Studio

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

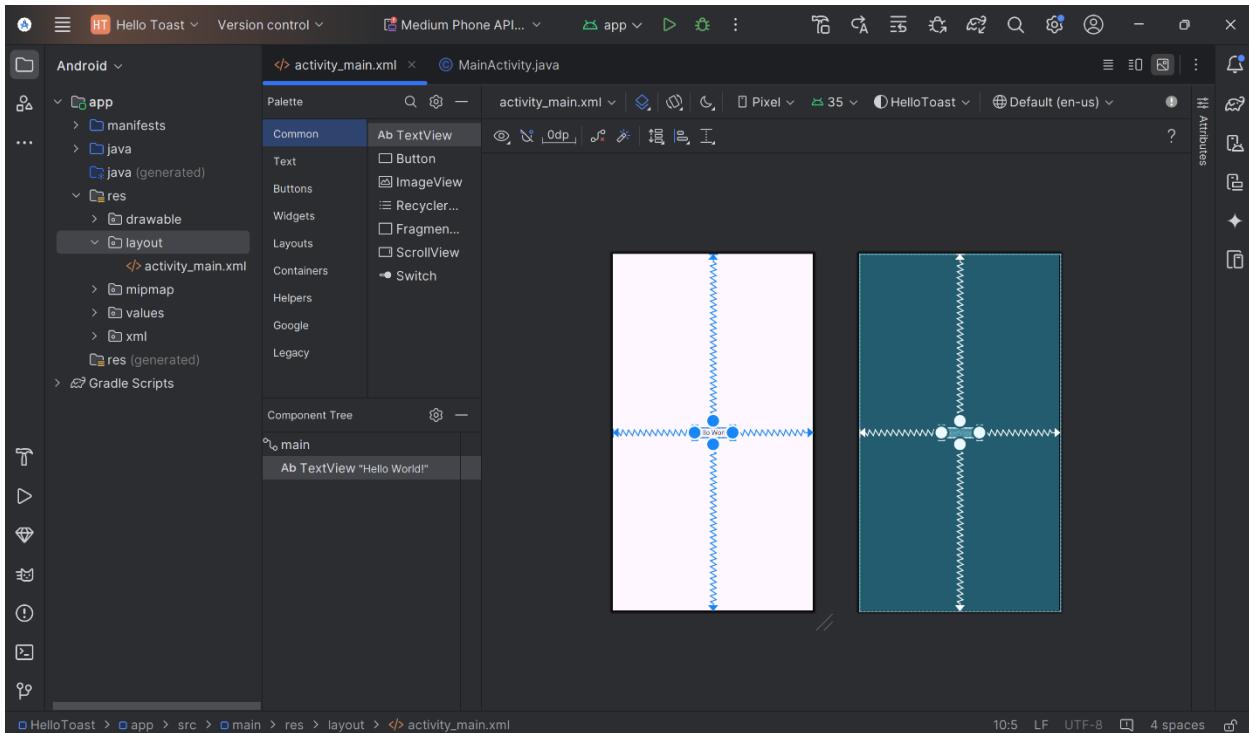


Chọn Run > Run app hoặc nhấp chuột vào biểu tượng Run  trên thanh công cụ để xây dựng và thực thi ứng dụng trên trình giả lập hoặc thiết bị của bạn.



1.2 Khám phá trình chỉnh sửa bố cục

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



- Trong thư mục app > res > layout trong bảng Project > Android, nhấp đúp vào tệp activity_main.xml để mở nó nếu nó chưa được mở.
- Nhấp vào tab Design nếu nó chưa được chọn. Bạn sử dụng tab Design để thao tác với các thành phần và bố cục, và tab Text để chỉnh sửa mã XML cho bố cục.
- Bảng Palettes hiển thị các thành phần UI mà bạn có thể sử dụng trong bố cục của ứng dụng.
- Bảng Componet hiển thị cấu trúc phân cấp các thành phần UI. Các thành phần View được tổ chức thành một cấu trúc cây với các thành phần cha và con, trong đó một phần tử con được kế thừa các thuộc tính với thành phần cha. Trong hình trên TextView là một phần tử con của ConstraintLayout. Bạn sẽ tìm hiểu các thành phần này sau trong bài học này.
- Các bảng thiết kế và bản vẽ của trình chỉnh sửa bố cục hiển thị các thành phần UI trong bố cục. Trong hình trên, bố cục chier hiển thị một thành phần: TextView hiển thị "Hello World".
- Tab Attributes hiển thị bảng Attributes để thiết lập các thuộc tính cho một thành phần UI.

Mẹo: Xem [Building a UI with Layout Editor](#) để biết chi tiết về việc sử dụng trình chỉnh sửa bố cục và [Meet Android Studio](#) để có tài liệu đầy đủ về Android Studio.

Nhiệm vụ 2: Thêm các thành phần View trong trình chỉnh sửa bố cục

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

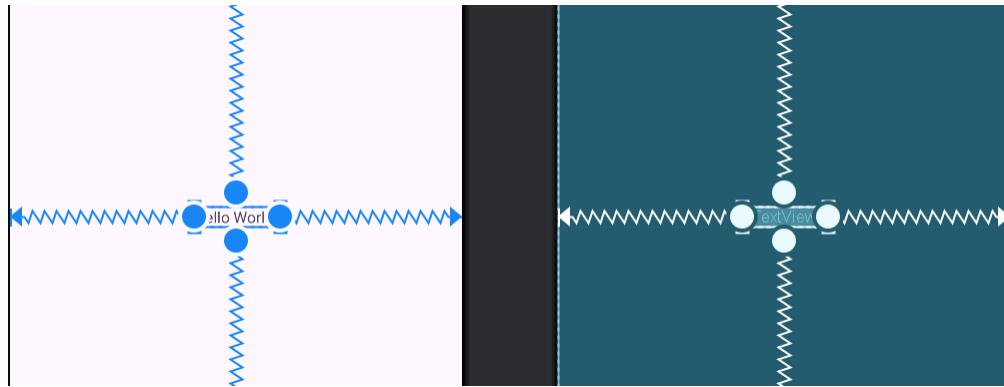
2.1 Kiểm tra ràng buộc của các thành phần

1. Mở tệp activity_main.xml:

- Trong bảng Project > Android, mở tệp activity_main.xml nếu nó chưa được mở. Nếu tab Design chưa được chọn, hãy nhấp vào nó.
- Nếu không có bản vẽ, hãy nhấp vào nút Select Design Surface  trong thanh công cụ và chọn Design and Blueprint.

2. Công cụ Autoconnect cũng nằm trong thanh công cụ. Nó được bật theo mặc định. Để thực hiện bước này, hãy đảm bảo rằng công cụ này không bị vô hiệu hóa.

3. Nhấn vào nút Zoom in  để phóng to khu vực thiết kế và bản vẽ nhằm có cái nhìn rõ hơn về các thành phần giao diện người dùng trong bố cục.
4. Chọn TextView trong bảng Component Tree. TextView có văn bản “Hello World” sẽ được làm nổi bật trong các khu vực thiết kế và bản vẽ và các ràng buộc cho thành phần này sẽ hiển thị rõ ràng.
5. Tham khảo hình ảnh động bên dưới cho các bước trên. Nhấp vào nút hình tròn ở phía bên phải của TextView để xóa ràng buộc ngang liên kết thành phần với bên phải bố cục. TextView sẽ nhảy sang bên trái vì nó không còn bị ràng buộc với bên phải. Để thêm lại ràng buộc ngang, nhấp vào cùng một nút và kéo một đường đến bên phải bố cục.



Trong các khung bản đồ hoặc thiết kế, các nút sau đây xuất hiện trên thành phần TextView:

- Xử lý ràng buộc: Để tạo một ràng buộc như trong hình trên, hãy nhấp vào một ràng buộc, được biểu thị bằng một hình tròn ở bên cạnh thành phần. Sau đó, kéo ràng buộc đó đến một ràng buộc khác, hoặc đến một ranh giới bố cục cha. Một đường zigzag sẽ đại diện cho ràng buộc.



- Thay đổi kích thước nút: Để thay đổi kích thước thành phần, hãy kéo các nút thay đổi kích thước hình vuông. Nút sẽ chuyển thành góc cạnh trong khi bạn đang kéo nó.

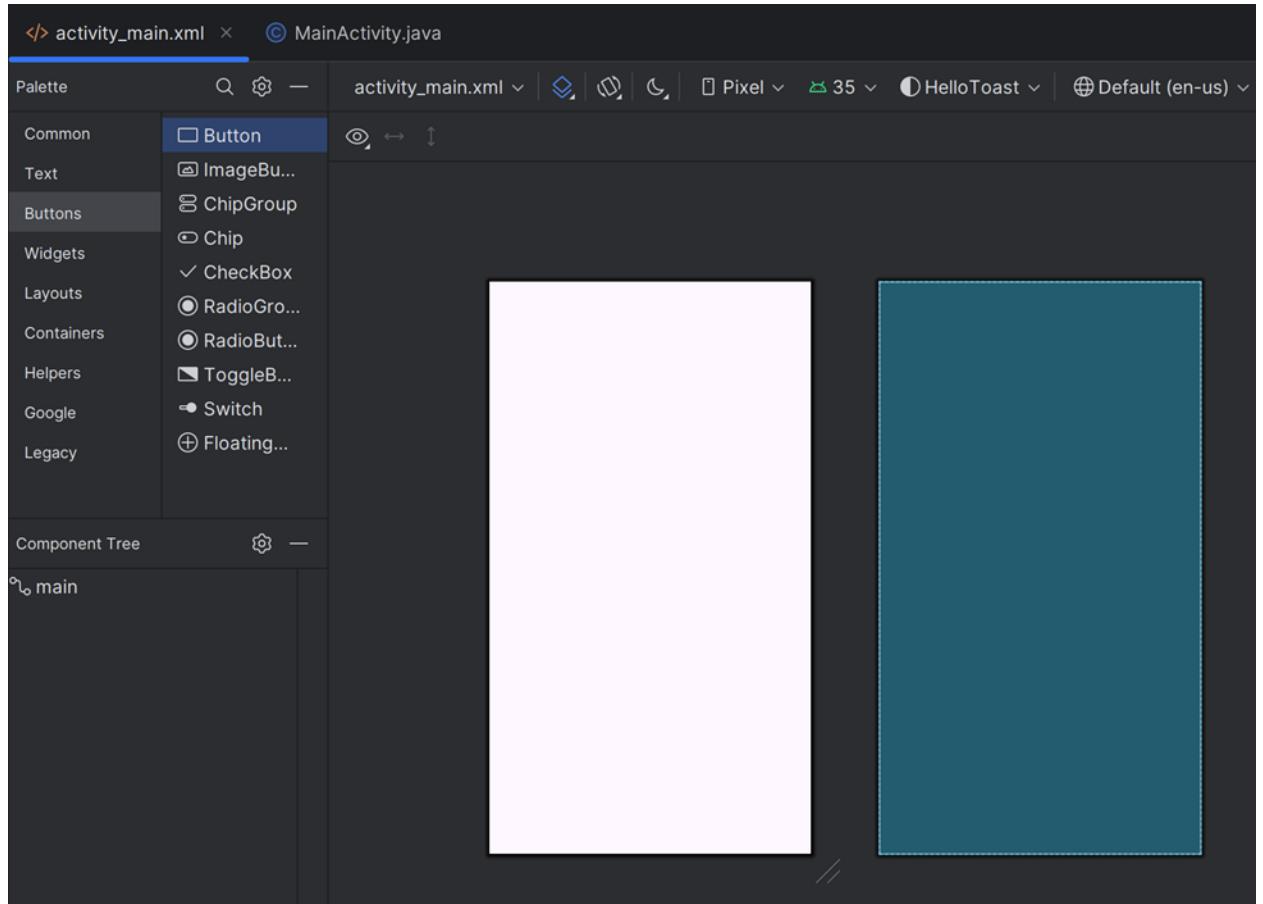


2.2 Thêm Button vào bố cục

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

Làm theo các bước sau để thêm Button:

1. Bắt đầu bằng một trang trắng. Thành phần TextView là không cần thiết, vì vậy trong khi nó vẫn được chọn, hãy nhấn phím Delete hoặc chọn Edit > Delete. Jetzt Sie haben eine leere Layoutschablone.

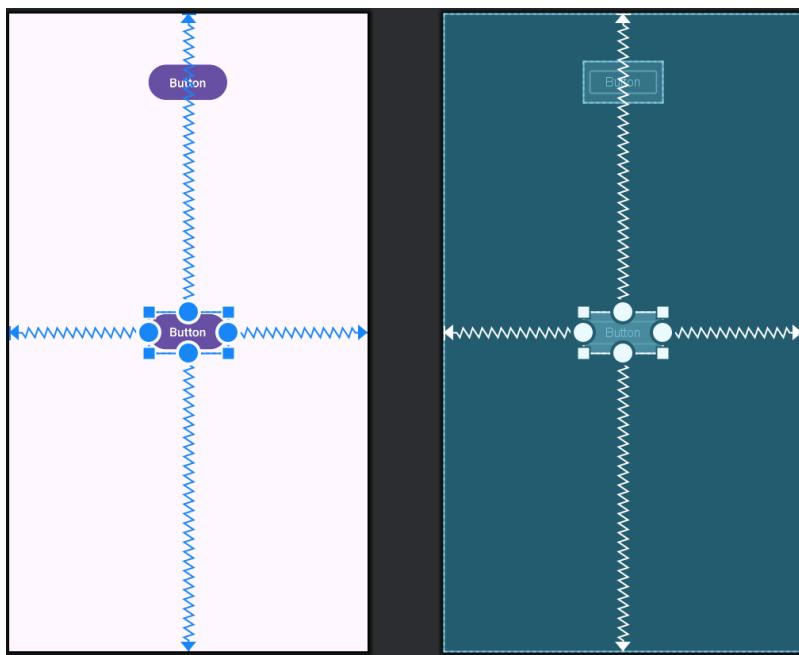


2. Kéo một Button từ bảng Palette vào bất kỳ vị trí nào trong bố cục. Nếu bạn thả Button ở khu vực giữa trên cùng của bố cục, các ràng buộc có thể tự động xuất hiện. Nếu không, bạn có thể kéo thả các ràng buộc đến phía trên, bên trái và bên phải của bố cục như được hiển thị bên dưới.

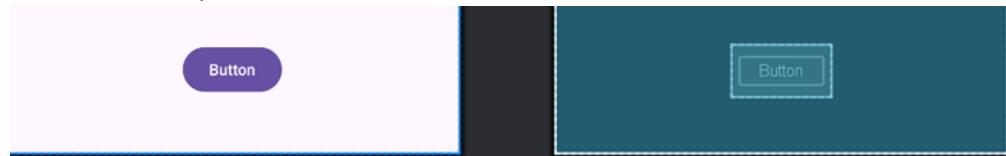


2.3 Thêm Button thứ hai vào bố cục

1. Kéo một Button khác từ bảng Palette vào giữa bố cục như được hiển thị trong hình bên dưới. Autoconnect có thể cung cấp ràng buộc ngang cho bạn (nếu không, bạn có thể tự kéo chúng).



2. Kéo một ràng buộc dọc đến đáy của bố cục (tham khảo hình bên dưới)



Bạn có thể xóa các ràng buộc khỏi một thành phần bằng cách chọn thành phần đó và di chuột qua nó để hiển thị nút Clear Constraints

Nhấp vào nút này để xóa tất cả các ràng buộc trên thành phần đã chọn. Để xóa một ràng buộc cụ thể, hãy nhấp vào tay cầm cụ thể thiết lập ràng buộc đó.

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

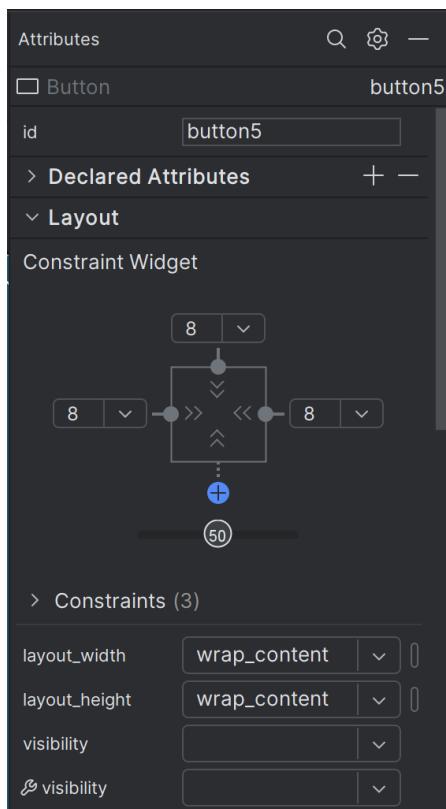
Nhiệm vụ 3: Thay đổi các thành phần giao diện người dùng(UI)

Bảng thuộc tính cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một thành phần UI. Bạn có thể tìm thấy các thuộc tính (được gọi là thuộc tính) chung cho tất cả các view trong tài liệu lớp view. Trong nhiệm vụ lần này, bạn sẽ nhập các giá trị mới và thay đổi các giá trị cho các thuộc tính quan trọng của Button, áp dụng cho hầu hết các loại view.

3.1 Thay đổi kích thước Button

Trình chỉnh sửa bô cục cung cấp các nút điều chỉnh kích thước ở cả bốn góc của View để bạn có thể thay đổi kích thước View một cách nhanh chóng. Bạn có thể kéo các nút điều chỉnh ở mỗi góc của view để thay đổi kích thước, nhưng làm như vậy sẽ cố định các kích thước chiều rộng và chiều cao. Tránh việc cố định kích thước cho hầu hết các thành phần của View, vì các kích thước cố định không thích ứng với nội dung và kích thước màn hình khác nhau.

Thay vào đó, hãy sử dụng bảng thuộc tính ở bên phải của trình chỉnh sửa bô cục để chọn chế độ kích thước không sử dụng kích thước cố định. Bảng thuộc tính bao gồm một bảng kích thước hình vuông gọi là trình kiểm tra chế độ xem ở phía trên. Các ký hiệu bên trong hình vuông đại diện cho các cài đặt chiều cao và chiều rộng như sau:



Trong hình trên có:

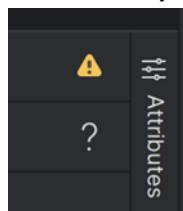
1. Điều chỉnh chiều cao: Điều chỉnh này được xác định bởi thuộc tính “layout_height” và xuất hiện ở hai phần trên và dưới của hình vuông. Các góc cho thấy điều chỉnh này được thiết lập để “wrap_content”, có nghĩa là “View” sẽ mở rộng theo chiều dọc khi cần thiết để phù hợp với nội dung của nó.

Số “8” chỉ ra rằng có một khoảng cách tiêu chuẩn được thiết lập là 8dp.

2. Điều chỉnh chiều rộng: Điều chỉnh này được xác định bởi thuộc tính “layout_width” và xuất hiện ở hai phần trái và phải của hình vuông. Các góc cho thấy điều chỉnh này được thiết lập để “wrap_content”, có nghĩa là “View” sẽ mở rộng theo chiều ngang khi cần thiết để phù hợp với nội dung của nó, tối đa đến một khoảng cách là 8dp.
3. Nút đóng bảng thuộc tính. Nhấn để đóng bảng.

Thực hiện các bước sau:

1. Chọn nút ở trên cùng trong bảng Component Tree.
2. Nhấp vào tab Attributes ở bên phải của cửa sổ trình chỉnh sửa bố cục.

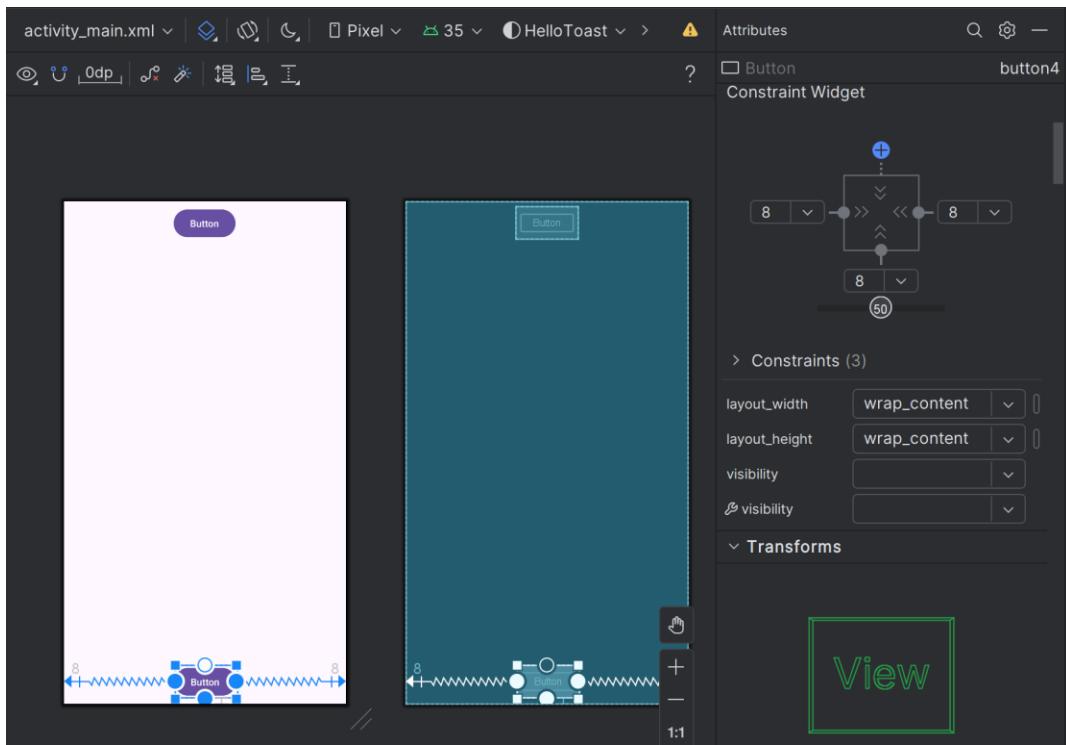


3. Nhấp vào điều chỉnh chiều rộng hai lần – lần nhấp đầu tiên sẽ thay đổi nó thành cố định với các đường thẳng, và nhấp lần thứ hai sẽ thay đổi nó thành khớp ràng buộc với các cuộn lò xo như hình bên dưới:



Do thay đổi điều khiển chiều rộng, layout_width thuộc tính trong ngắn Thuộc tính sẽ hiển thị giá trị match_constraint và Button phần tử sẽ 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 layout_width như ở bước trước, thể hiện trong hình bên dưới:



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

- Cài đặt `Match_constraint` mở rộng phần tử để xem để lấp đầy phần tử cha theo chiều rộng hoặc chiều cao – lên đến một lè, nếu có. Phần tử 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.

Mẹo: Nếu bạn thay đổi thuộc tính `layout_width` bằng menu bật lên của nó, thuộc tính `layout_width` được đặt thành không vì không có thứ nguyên được đặt. Cài đặt này giống như `match_constraint` - chế độ xem

có thể mở rộng càng nhiều càng tốt để đáp ứng các ràng buộc và cài đặt lề.

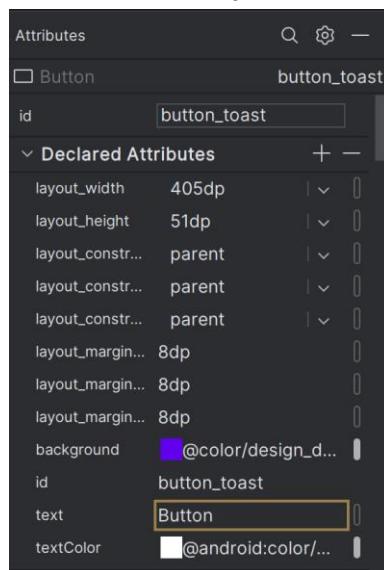
3.2 Thay đổi thuộc tính của Button

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

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

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

1. Sau khi chọn Button đầu tiên, hãy chỉnh sửa trường ID ở đầu ngăn Attributes thành button_toast cho thuộc tính android:id, được sử dụng để xác định phần tử trong bố cục.
2. Đặt thuộc tính background thành @color/colorPrimary . (Khi bạn nhập @c , các lựa chọn sẽ xuất hiện để bạn dễ dàng lựa chọn.)
3. Đặt thuộc tính textColor thành @android:color/white .
4. Chỉnh sửa thuộc tính text thành Toast



5. Thực hiện các thay đổi thuộc tính tương tự cho Button thứ hai, sử dụng button_count làm ID, Count cho thuộc tính text và cùng màu cho 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 trong tệp tài nguyên colors.xml. Nó được

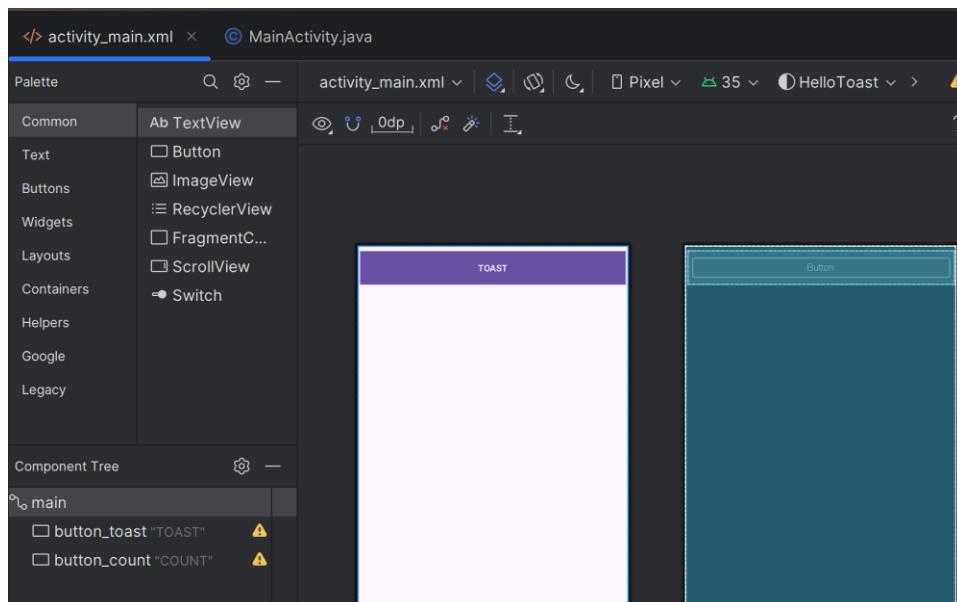
sử dụng cho thanh ứng dụng. Sử dụng màu cơ sở cho các thành phần UI khác sẽ tạo ra một UI thống nhất. Bạn sẽ tìm hiểu thêm về chủ đề ứng dụng và Material Design trong bài học khác.

Nhiệm vụ 4: Thêm Textedit và đặt các thuộc tính của nó

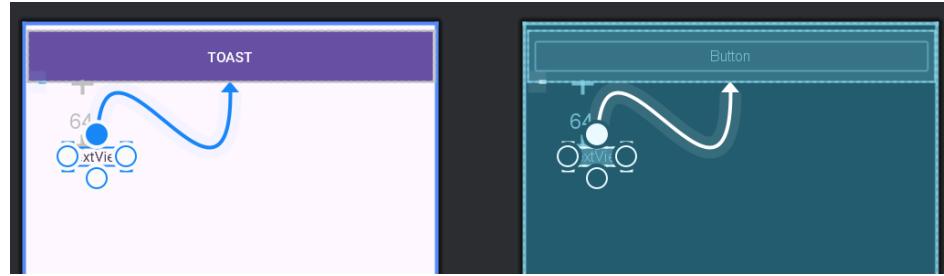
Một trong những lợi ích của ConstraintLayout là khả năng căn chỉnh hoặc hạn chế các phần tử liên quan đến các phần tử khác. Trong nhiệm vụ này, bạn sẽ thêm một TextView ở giữa bố cục và hạn chế nó theo chiều ngang vào lề và theo chiều dọc vào hai phần tử Button. Sau đó, bạn sẽ thay đổi các thuộc tính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

4.1 Thêm TextView và các ràng buộc

1. Như hình minh họa bên dưới, hãy kéo TextView từ ngăn Palette đến phần trên của bố cục và kéo một ràng buộc từ đầu TextView đến tay cầm ở cuối Nút Toast. Điều này hạn chế TextView nằm bên dưới Button .



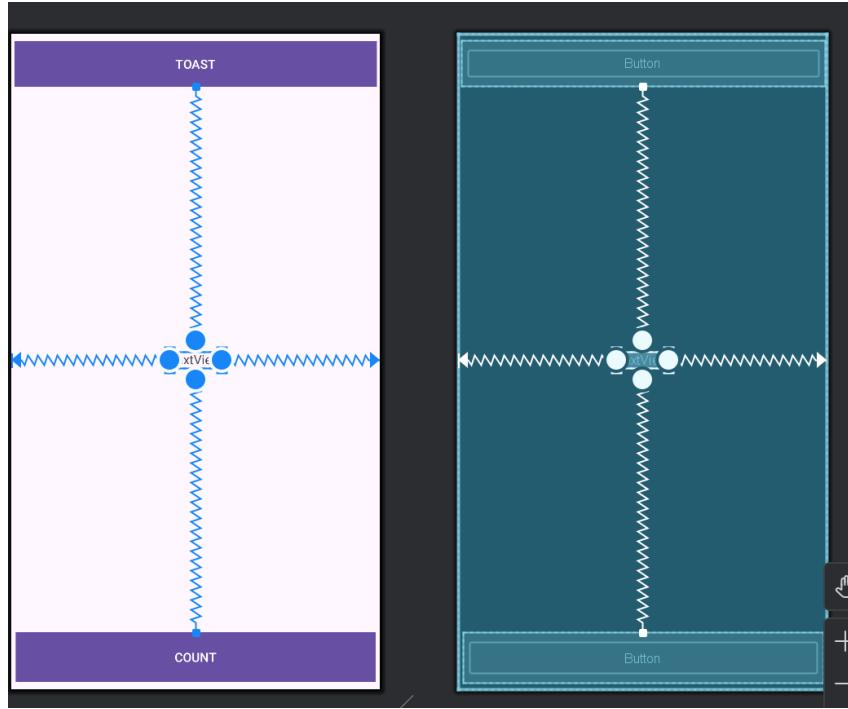
2. Như trong hình bên dưới, hãy kéo một ràng buộc từ cuối TextView đến tay cầm ở trên cùng của Nút count và từ hai bên của TextView đến hai bên của bố cục. Điều này hạn chế TextView ở giữa bố cục giữa hai phần tử Button.



4.2 Đặt các thuộc tính TextView

Với TextView được chọn, hãy mở ngăn Attributes, nếu nó chưa mở. Đặt thuộc tính cho TextView như trong hình động bên dưới. Các thuộc tính bạn chưa gặp được giải thích sau hình:

1. Đặt ID thành `show_count` .
2. Đặt `text` thành `0` .
3. Đặt `textSize` thành `160sp` .
4. Đặt `textStyle` thành `B` (in đậm) và `textAlignment` thành `ALIGNCENTER` (căn giữa đoạn văn).
5. Thay đổi các điều khiển kích thước chế độ xem theo chiều ngang và chiều dọc (`layout_width` và `layout_height`) thành `match_constraint` .
6. Đặt `textColor` thành `@color/colorPrimary` .
7. Cuộn xuống ngăn và nhấp để mở rộng xem tất cả thuộc tính , cuộn xuống trang thuộc tính thứ hai đến `background`, sau đó nhập `#FFFF00` để có màu vàng.
8. Cuộn xuống đến `gravity`, mở rộng `gravity` và chọn `center_vertical` (cho `center-vertical`).

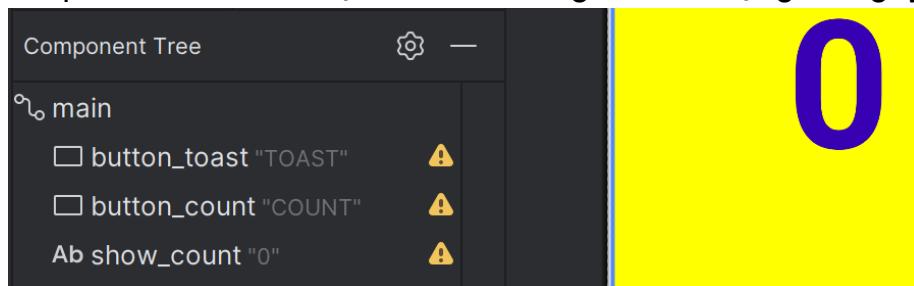


- textSize: Kích thước văn bản của TextView. Đối với bài học này, kích thước được đặt thành 160sp. Sp - chữ viết tắt của pixel không phụ thuộc tỷ lệ , và tương tự dp, là một đơn vị chia tỷ lệ theo mật độ màn hình và tùy chọn kích thước phông chữ của người dùng. Sử dụng đơn vị dp khi bạn chỉ định kích thước phông chữ để kích thước được điều chỉnh cho cả mật độ màn hình và sở thích của người dùng.
- textStyle và textAlign: Kiểu văn bản, được đặt thành B (in đậm) trong bài học này và căn chỉnh văn bản, được đặt thành ALIGNCENTER(căn giữa đoạn văn).
- gravity: Thuộc tính gravity này chỉ định cách View được căn chỉnh trong View hoặc ViewGroup. Trong bước này, bạn căn giữa phần tử TextView theo chiều dọc trong ConstraintLayout.

Bạn có thể nhận thấy rằng thuộc tính background nằm trên trang đầu tiên của ngăn Attributes cho Button, nhưng trên trang thứ hai của ngăn Attributes cho TextView. Ngăn Attributes thay đổi cho từng loại View. Các thuộc tính phổ biến nhất cho View loại này xuất hiện trên trang đầu tiên và các thuộc tính còn lại được liệt kê trên trang thứ hai. Để quay lại trang đầu tiên của ngăn Attributes , hãy bấm vào biểu tượng trên thanh công cụ ở đầu ngăn.

Nhiệm vụ 5: Chỉnh sửa bộ 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.

5.1 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 mã ở góc trên bên phải của 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, hiển thị một phần mã XML cho bố cục, các cảnh báo được tô sáng—các chuỗi được mã hóa cứng "Toast" và "Count". (Các chuỗi được mã hóa cứng "0" cũng được tô sáng nhưng không hiển thị trong hình.) Di con trỏ qua chuỗi được mã hóa cứng "Toast" để xem thông báo cảnh báo.

5.2 Trích xuất tài nguyên chuỗi

Thay vì mã hóa cứng các chuỗi, cách tốt nhất là sử dụng các tài nguyên chuỗi, đại diện cho các chuỗi. Việc có các chuỗi trong một tệp riêng biệt giúp quản lý chúng dễ dàng hơn, đặc biệt là nếu bạn sử dụng các chuỗi này nhiều lần. Ngoài ra, các tài nguyên chuỗi là bắt buộc để dịch và bản địa hóa ứng dụng của bạn, vì bạn cần tạo một tệp tài nguyên chuỗi cho mỗi ngôn ngữ.

1. Nhập một lần vào từ "Toast"(cảnh báo được tô sáng đầu tiên).
2. Nhấn **Alt-Enter** trong Windows/Linux hoặc **Option-Enter** trong macOS và chọn **Extract string resource** từ menu bật lên.
3. Nhập **button_label_toast** cho tên Tài nguyên .

4. Nhập vào **OK**. Một tài nguyên chuỗi được tạo trong 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_valuefor cho "0".
6. Trong ngăn **Project > Android**, hãy mở rộng **res** rồi mở rộng các giá trị trong **res** và nhấp đúp vào strings.xml để xem tài nguyên chuỗi của bạn trong tệp strings.xml:

```
<resources>
    <string name="app_name">Hello Toast</string>
    <string name="button_label_toast" translatable="false">Toast</string>
    <string name="button_label_count" translatable="false">Count</string>
    <string name="count_initial_value" translatable="false">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_messageecho cụm từ "Hello Toast!":

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

Mẹo : Tài nguyên chuỗi bao gồm tên ứng dụng, xuất hiện trên thanh ứng dụng ở đầu màn hình nếu bạn bắt đầu dự án ứng dụng của mình bằng mẫu trống. Bạn có thể thay đổi tên ứng dụng bằng cách chỉnh sửa tài nguyên app_name.

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

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

6.1 Thêm thuộc tính và trình xử dụng onClick vào 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 mở (tab văn bản), hãy tìm nút có Android: ID được đặt thành button_toast.

```
<Button  
    android:id="@+id/button_toast"  
    android:layout_width="405dp"  
    android:layout_height="51dp"  
    android:layout_marginStart="8dp"  
    android:layout_marginTop="8dp"  
    android:layout_marginEnd="8dp"  
    android:background="@color/design_default_color_primary"  
    android:text="TOAST"  
    android:textColor="@android:color/white"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent" />
```

2. Thêm thuộc tính android:onClick vào cuối phần tử button_toast sau thuộc tính cuối cùng và trước chỉ báo kết thúc </>.
 `android:onClick="showToast"/>`

3. Nhấp vào biểu tượng bóng đèn màu đỏ xuất hiện bên cạnh thuộc tính. Chọn **Create click handler**, chọn **MainActivity** và nhấp vào **OK**. Nếu biểu tượng bóng đèn màu đỏ không xuất hiện, hãy nhấp vào tên phương thức ('showToast'). Nhấn **Alt-Enter (Option-Enter** trên máy Mac), chọn **Create 'showToast(view)'** 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="countUp"/>
```

Mã XML cho các phần tử giao diện người dùng trong ConstraintLayout bây giờ trông như thế này

```

<Button
    android:id="@+id/button_toast"
    android:layout_width="405dp"
    android:layout_height="51dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:background="@color/design_default_color_primary"
    android:text="TOAST"
    android:textColor="@android:color/white"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    android:onClick="showToast"/>

<Button
    android:id="@+id/button_count"
    android:layout_width="407dp"
    android:layout_height="58dp"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="8dp"
    android:background="@color/design_default_color_primary"
    android:text="COUNT"
    android:textColor="@android:color/white"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    android:onClick="countUp"/>

<TextView
    android:id="@+id/show_count"
    android:layout_width="405dp"
    android:layout_height="606dp"
    android:background="#FFFF00"
    android:gravity="center_horizontal|center_vertical"
    android:text="0"
    android:textAlignment="center"
    android:textColor="@color/design_default_color_primary_dark"
    android:textSize="160sp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@+id/button_count"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/button_toast" />

```

- 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 com.example.hellotoast;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
        setContentView(R.layout.activity_main);
        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;
        });
    }

    1 usage
    public void showToast(View view) {
    }

    1 usage
    public void countUp(View view) {
    }
}

```

6.2 Chỉnh sửa trình xử lý nút Toast

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 nhấp 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

```

public void showToast(View view) {
}

```

2. Để tạo một thẻ hiện của a `Toast`, hãy gọi phương thức `makeText()` trên lớp `Toast`

```

public void showToast(View view) {
    Toast toast = Toast.makeText();
}

```

3. Cung cấp ngữ cảnh của app Activity . Vì Toast hiển thị trên đầu Activity UI, hệ thống cần thông tin về Activity hiện tại. Khi bạn đã ở trong ngữ cảnh của Activity mà bạn cần, hãy sử dụng nó như một phím tắt

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

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.

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

5. Cung cấp thời lượng hiển thị. Ví dụ: Toast.LENGTH_SHORT hiển thị toast trong thời gian tương đối ngắn

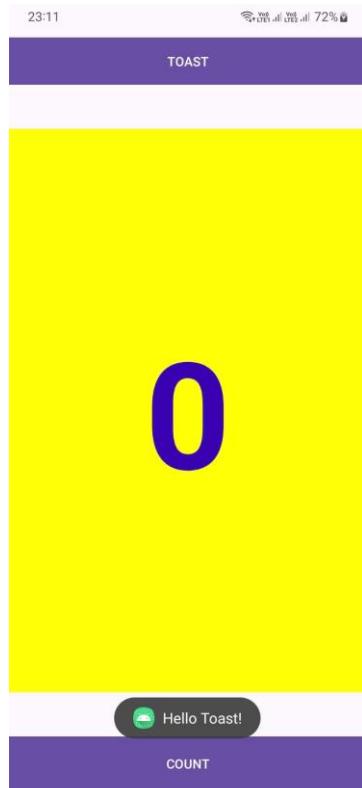
```
public void showToast(View view) {  
    Toast toast = Toast.makeText( context: this,R.string.toast_message, Toast.LENGTH_SHORT);  
}
```

Thời lượng Toast hiển thị có thể là Toast.LENGTH_LONG hoặc Toast.LENGTH_SHORT. Độ dài thực tế là khoảng 3,5 giây cho màn hình dài Toast và 2 giây cho màn hình ngắn Toast.

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( context: this,R.string.toast_message, Toast.LENGTH_SHORT);  
    toast.show();  
}
```

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



6.3 Chỉnh sửa trình xử lý nút Count

Bây giờ bạn sẽ chỉnh sửa phương thức `countUp()` — trình xử lý nhấp chuột Nút Count trong `MainActivity` — để nó hiển thị số lượng hiện tại sau khi nhấn vào Count. 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 cập nhật đến `TextView` để hiển thị.

Thực hiện theo các bước sau để chỉnh sửa trình xử lý nhấp chuột button Countt

1. Xác định vị trí phương thức `CountUp()` mới tạo.

```
public void countUp(View view) {  
}
```

2. Để theo dõi số lượng, bạn cần một biến thành viên riêng. Mỗi lần nhấn nút Count sẽ làm tăng giá trị của biến này. Nhập nội dung sau, nội dung này sẽ được tô sáng màu đỏ và hiển thị biểu tượng bóng đèn màu đỏ:

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

Nếu biểu tượng bóng đèn màu đỏ không xuất hiện, hãy nhấp vào biểu thức mCount++. Bóng đèn màu đỏ cuối cùng sẽ xuất hiện.

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

```
public class MainActivity extends AppCompatActivity {  
    1 usage  
    private int mCount;
```

4. Thay đổi câu lệnh biến thành viên riêng để khởi tạo biến về 0

```
public class MainActivity extends AppCompatActivity {  
    1 usage  
    private int mCount = 0;
```

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

```
public class MainActivity extends AppCompatActivity {  
    1 usage  
    private int mCount = 0;  
    no usages  
    private TextView mShowCount;
```

6. Nay bạn đã có mShowCount, bạn có thể lấy tham chiếu đến TextView bằng cách sử dụng ID mà bạn đã đặt trong tệp bố cục. Để lấy 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 "inflate" bố cục, có nghĩa là thiết lập nội dung của màn hình thành bố cục XML. Bạn có thể sử dụng nó để lấy tham chiếu đến các phần tử UI khác trong bố cục, chẳng hạn như "TextView". Tìm phương thức "onCreate()" trong "MainActivity"

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

7. Thêm câu lệnh findViewById vào cuối phương thức:

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    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`.

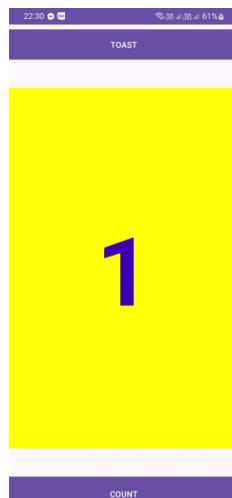
8. Bây giờ bạn đã gán “`mShowCount`” cho “`TextView`”, bạn có thể sử dụng biến này để thiết lập văn bản trong “`TextView`” thành giá trị của biến “`mCount`”. Thêm đoạn mã sau vào phương thức “`countUp()`”:

```
if(mShowCount != null)
    mShowCount.setText(Integer.toString(mCount));
```

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

```
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



Mẹo: Để biết hướng dẫn chuyên sâu về cách sử dụng `ConstraintLayout`, hãy xem [Using ConstraintLayout to design your views](#).

Giải pháp

Dự án HelloToast:

Thử thách mã hóa

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

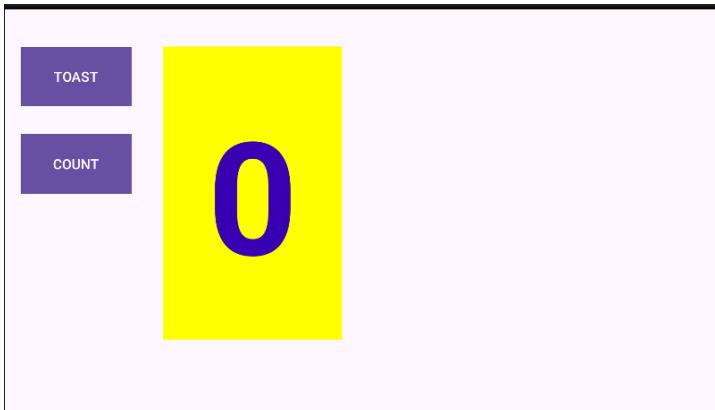
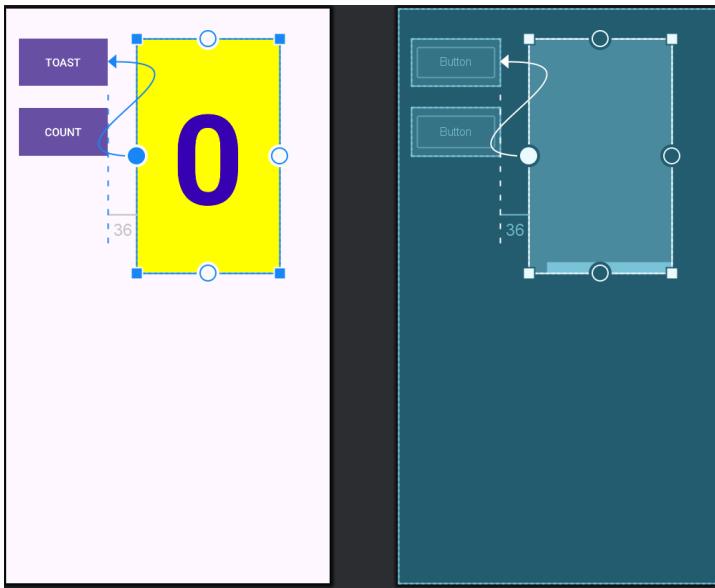
Ứng dụng HelloToast trông ổn khi thiết bị hoặc trình giả lập ở chế độ dọc.

Tuy nhiên, nếu bạn chuyển thiết bị hoặc trình giả lập sang chế độ ngang, nút Count có thể chồng chéo lên “TextView” ở phía dưới như được hiển thị trong hình dưới đây:



Thử thách: Thay đổi bối cảnh sao cho nó trông đẹp trong cả hai chế độ ngang và dọc.

1. Trên máy tính của bạn tạo một bản sao của thư mục dự án HelloToast và đổi tên nó thành HelloToastChallenge.
2. Mở HelloToastChallenge trong Android Studio và thực hiện tái cấu trúc nó.(Xem phụ lục: Tiện ích để biết hướng dẫn về việc sao chép và tái cấu trúc một dự án.)
3. Thay đổi bối cảnh sao cho nút Toastr và nút Count xuất hiện ở bên trái, như được hiển thị trong hình dưới đây. TextView xuất hiện bên cạnh chúng, nhưng chỉ đủ rộng để hiển thị nội dung của nó (Gợi ý: sử dụng “wrap_content”)
4. Chạy ứng dụng trong cả hai chế độ ngang dọc.



Mã giải pháp thử thách

Dự án Android Studio: [HelloToastChallenge](#)

View, ViewGroup và Layout:

- Tất cả các thành phần UI đều là các lớp con của lớp View , do đó kế thừa nhiều thuộc tính của lớp cha View.
- Các thành phần View có thể được nhóm lại bên trong một ViewGroup, hoạt động như một container. Mỗi quan hệ này là mối quan hệ cha-con, trong đó cha là một ViewGroup và con là một View hoặc một ViewGroup khác.
- Phương thức `onCreate()` được sử dụng để inflate bố cục, có nghĩa là thiết lập nội dung màn hình thành bố cục XML. Bạn có thể sử dụng nó để lấy tham chiếu đến các phần tử giao diện người dùng khác trong bố cục.

- Một View giống như một chuỗi, là một tài nguyên có thể có một ID. Lệnh gọi findViewById nhận ID của một View làm tham số và trả về View tương ứng.

Sử dụng trình chỉnh sửa bố cục:

- Nhập vào tab Design để thao tác với các phần tử và bố cục, và tab Text để chỉnh sửa mã XML cho bố cục.
- Trong tab Design, bảng Palettes hiển thị các thành phần UI mà bạn có thể sử dụng trong bố cục của ứng dụng, và bảng Component tree hiển thị cấu trúc phân cấp của các phần tử UI.
- Các bảng thiết kế và bản vẽ của trình chỉnh sửa bố cục hiển thị các thành phần UI trong bố cục.
- Tab Attributes hiển thị bảng Attributes để thiết lập các thuộc tính cho một thành phần UI.
- Xử lý ràng buộc: nhấp vào một ràng buộc, được hiển thị dưới dạng hình tròn ở mỗi bên của một thành phần, sau đó kéo đến một ràng buộc khác hoặc đến ranh giới của cha để tạo một ràng buộc. Ràng buộc đó được biểu diễn bằng đường zigzag.
- Tay cầm thay đổi kích thước: Bạn có thể kéo tay cầm thay đổi kích thước hình vuông để thay đổi kích thước phần tử. Trong khi kéo, tay cầm sẽ thay đổi thành góc nghiêng.
- Khi được bật, công cụ Autoconnect tự động tạo hai hoặc nhiều ràng buộc cho một phần tử UI đến bố cục cha. Sau khi bạn kéo thành phần vào bố cục, công cụ sẽ tạo ra các ràng buộc dựa trên vị trí của thành phần.
- Bạn có thể xóa ràng buộc khỏi một phần tử bằng cách chọn phần tử đó và di con trỏ qua để hiển thị nút Xóa ràng buộc  . Nhấp vào nút này để xóa tất cả các ràng buộc trên phần tử đã chọn. Để xóa một ràng buộc duy nhất, hãy nhấp vào tay cầm cụ thể đặt ràng buộc đó.
- Ngăn Attributes cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một phần tử UI. Ngăn này cũng bao gồm một bảng điều khiển kích thước hình vuông được gọi là trình kiểm tra chế độ xem ở trên cùng. Các ký hiệu bên trong hình vuông biểu thị các thiết lập chiều rộng và chiều cao.

Thiết lập chiều rộng và chiều cao của bố cục:

Các thuộc tính layout_width và layout_height thay đổi khi bạn thay đổi các điều khiển kích thước chiều cao và chiều rộng trong trình kiểm tra chế độ xem. Các thuộc tính này có thể lấy một trong ba giá trị cho ConstraintLayout:

- Thiết lập match_constraint mở rộng view để lấp đầy phần tử cha theo chiều rộng và chiều cao - tối đa đến một khoảng cách nhất định (nếu có).
- Thiết lập wrap_content thu nhỏ kích thước view sao cho view chỉ đủ lớn để bao gồm nội dung của nó. Nếu không có nội dung, view sẽ trở nên vô hình.
- Sử dụng số lượng cố định dp (density-independent pixels) để chỉ định Kích thước cố định, được điều chỉnh theo kích thước màn hình của thiết bị.

Trích xuất tài nguyên chuỗi:

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

- Nhấp một lần vào chuỗi được mã hóa cứng để trích xuất, nhấn Alt-Enter (Option – Enter trên máy Mac) và chọn Extract string resources từ menu pop-up.
- Đặt tên tài nguyên.
- Nhấp vào Ok. Thao tác này sẽ tạo một tài nguyên chuỗi trong values/res/string.xml và chuỗi trong mã hóa của bạn sẽ được thay thế bằng một tham chiếu đến tài nguyên: "@string/button_label_toast"

Xử lý nhấp chuột:

- Trình xử lý nhấp chuột là phương thức được gọi khi người dùng nhấp hoặc chạm vào phần tử UI.
- Chỉ định trình xử lý nhấp chuột cho một phần tử UI như Button bằng cách nhập tên của nó vào trường onClick trong ngăn Attributes của tab Design hoặc trong trình soạn thảo XML bằng cách thêm thuộc tính android:onClick vào một phần tử UI như button.
- Tạo trình xử lý sự kiện nhấp chuột trong Activity chính bằng cách sử dụng tham số View. Ví dụ: public void showToast(View view){...}.

- Bạn có thể tìm thấy thông tin về tất cả các thuộc tính của Button trong tài liệu lớp Button và tất cả các thuộc tính của TextView trong tài liệu lớp TextView.

Hiển thị tin nhắn Toast:

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

- Gọi phương thức factory makeText() trên lớp Toast.
- Cung cấp bối cảnh của ứng dụng Activity và thông báo cần hiển thị(chẳng hạn như tài nguyên chuỗi).
- Cung cấp thời lượng hiển thị, ví dụ Toast.LENGTH_SHORT trong một khoảng thời gian ngắn. Thời lượng có thể là Toast.LENGTH hoặc Toast.LENGTH_SHORT.
- Hiển thị Toast bằng cách gọi show().

Khái niệm liên quan

Tài liệu và khái niệm liên quan có trong 1.2: Bố cục và tài nguyên cho UI.

Tìm hiểu thêm

Tài liệu dành cho nhà phát triển Android:

- Android Studio
- Xây dựng giao diện người dùng và trình chỉnh sửa bố cục
- Xây dựng giao diện người dùng với ConstraintLayout
- Bố cục
- View
- Button
- TextView
- Tài nguyên Android
- Tài nguyên R.color tiêu chuẩn của Android
- Hỗ trợ các độ phân giải khác nhau
- Sự kiện đầu vào Android
- Context

Khác

- Codelabs: Sử dụng ConstraintLayout để thiết kế các view của bạn
- Từ vựng và bảng chú giải các khái niệm

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

1.3) Trình chỉnh sửa bố cục

Giới thiệu

Như bạn đã họ trong mục 1.2A: Giao diện người dùng tương tác đầu tiên, bạn có thể xây dựng giao diện người dùng (UI) bằng ConstraintLayout trong trình chỉnh sửa bố cục, nơi chứa các thành phần UI vào bố cục bằng các kết nối ràng buộc với các thành phần khác với các cạnh bố cục. ConstraintLayout được thiết kế để giúp bạn dễ dàng kéo thả các thành phần UI vào trình chỉnh sửa bố cục.

ConstraintLayout là một ViewGroup, là một đối tượng này có thể chứa các đối tượng khác (gọi là children hoặc child views). Bài thực hành này sẽ trình bày thêm nhiều tính năng của ConstraintLayout trình chỉnh sửa bố cục.

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

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

Những gì bạn nên biết

Bạn sẽ có thể:

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

Những gì bạn sẽ học được

- Cách tạo biến thể bố cục cho hướng ngang (phong cảnh).
- 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 UI với văn bản.

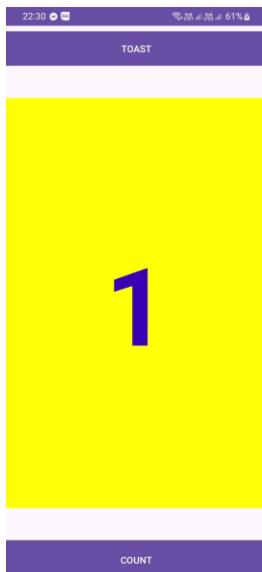
- Cách sử dụng các nút đóng gói và căn chỉnh để căn chỉnh các thành phần trong bố cục.
- Cách định vị các chế độ xem trong LinearLayout.
- Cách định vị các chế độ xem trong RelativeLayout.

Những gì bạn sẽ làm

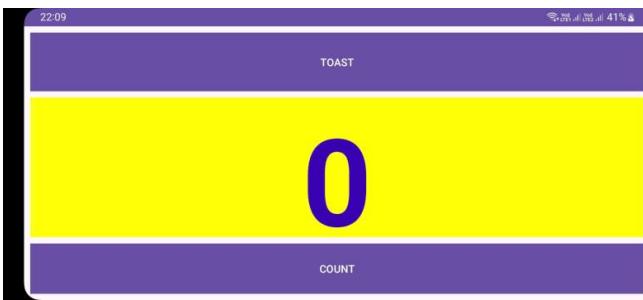
- Tạo một biến thể bố cục cho hướng hiển thị ngang.
- Tạo một biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Sửa đổi bố cục để thêm các ràng buộc vào các thành phần UI.
- Sử dụng ConstraintLayout ràng buộc đường cơ sở để căn chỉnh các thành phần với văn bản.
- Sử dụng ConstraintLayout các nút đóng gói và căn chỉnh để căn chỉnh các thành phần.
- Thay đổi bố cục để sử dụng LinearLayout.
- Vị trí các phần tử trong LinearLayout.
- Thay đổi bố cục để sử dụng RelativeLayout.
- Sắp xếp lại các chế độ xem trong bố cục chính sao cho phù hợp với nhau.

Tổng quan về ứng dụng

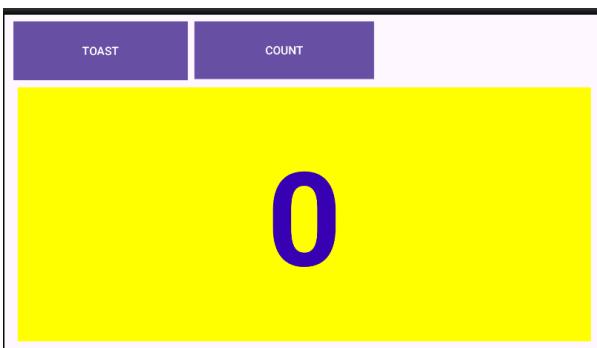
Ứng dụng Hello Toast trong bài học trước sử dụng ConstraintLayout để sắp xếp các thành phần UI trong bố cục, như trong hình bên dưới.



Để thực hành nhiều hơn ConstraintLayout, bạn sẽ tạo một biến thể của bố cục này theo hướng ngang như thể hiện trong hình bên dưới.



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



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

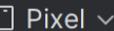
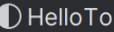
Nhiệm vụ 1: Tạo các biến thể bố cục

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

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



Trong hình trên:

- Chọn Design Surface  : Chọn Design để hiển thị bản xem trước màu sắc của bố cục, hoặc Blueprint để chỉ hiển thị phác thảo cho từng thành phần UI. Để xem cả hai ngăn cạnh nhau, hãy chọn Design + Blueprint.
- Định hướng trong trình chỉnh sửa  : Chọn Portrait hoặc Landscape để hiển thị bản xem trước ở hướng dọc hoặc ngang. Điều này hữu ích để xem trước bố cục mà không cần phải chạy ứng dụng trên trình giả lập hoặc thiết bị. Để tạo các bố cục thay thế, chọn Create Landscape Variation hoặc các biến thể khác.
- Thiết bị trong trình chỉnh sửa  Pixel : Chọn loại thiết bị(điện thoại/máy tính bảng, Android TV, hoặc Android Wear)
- Phiên bản API trong trình chỉnh sửa  35 : Chọn phiên bản Android sử dụng để hiển thị bản xem trước.
- Chủ đề trong trình chỉnh sửa  HelloToast : chọn một chủ đề (như Hello Toast) để áp dụng cho bản xem trước.
- Ngôn ngữ trong trình chỉnh sửa  Default (en-us) : chọn ngôn ngữ và khu vực cho bản xem trước. Danh mục này chỉ hiển thị các ngôn ngữ có sẵn trong tài nguyên chuỗi (xem bài học về địa phương hóa để biết chi tiết về cách thêm ngôn ngữ). Bạn có thể chọn Preview as Right to Left để xem bố cục như thể một ngôn ngữ RTL đã được chọn.

Thanh công cụ thứ hai cho phép bạn cấu hình giao diện của các thành phần UI trong ConstraintLayout, cũng như phóng to và thu nhỏ bản xem trước:



Trong hình trên có:

- Hiển thị: chọn Show Constraints và Show Margins để hiển thị chúng trong bản xem trước hoặc để ngừng hiển thị chúng.
- Tự động kết nối: bật hoặc tắt tự động kết nối. Khi tự động kết nối được bật, bạn có thể kéo bất kỳ phần tử nào (chẳng hạn như một button) đến bất kỳ phần tử nào của bố cục để tạo ra các ràng buộc với bố cục cha.
- Lề mặc định: đặt các lề mặc định
- Xóa tất cả các ràng buộc: xóa tất cả các ràng buộc trong toàn bộ bố cục

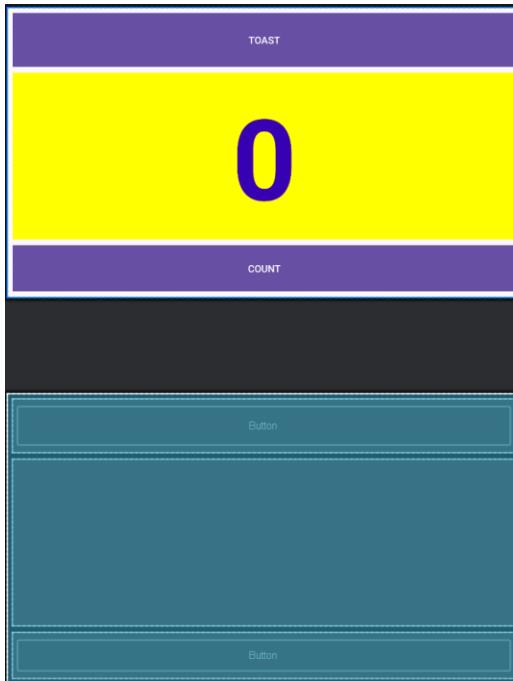
5. Suy diễn ràng buộc: tạo các ràng buộc bằng cách suy diễn
6. Gói: gói hoặc mở rộng các phần tử đã chọn
7. Căn chỉnh: căn chỉnh các phần tử đã chọn
8. Hướng dẫn: thêm các hướng dẫn đọc hoặc ngang
9. Điều khiển phóng to/thu nhỏ: phóng to hoặc thu nhỏ

Mẹo: Để tìm hiểu thêm về cách sử dụng trình chỉnh sửa bố cục, hãy xem Xây dựng giao diện người dùng bằng trình chỉnh sửa bố cục. Để tìm hiểu về cách xây dựng bố cục với ConstraintLayout, hãy xem Xây dựng giao diện người dùng phản hồi với ConstraintLayout.

1.1 Xem trước bố cục theo chiều ngang

Để xem trước bố cục của ứng dụng Hello Toast theo chiều 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 Hello Toast, bạn cần xóa các bố cục finished landscape và extra-large mà bạn sẽ tạo trong nhiệm vụ này. Chuyển từ Project > Android sang Project > Project Files trong bảng Dự án, mở rộng app > app > src/main > res, chọn cả thư mục layout-land và thư mục layout-xlarge, sau đó chọn Edit > Delete. Sau đó, chuyển bảng dự án trở lại Project > Android.
2. Mở tệp bố cục activity_main.xml. Nhấp vào tab Design nếu nó chưa được chọn.
3. Nhấp vào nút Orientation in Editor trong thanh công cụ trên cùng.
4. Chọn Switch to Landscape trong menu thả xuống. Bố cục sẽ xuất hiện ở chiều ngang như hình dưới đây. Để quay lại chiều dọc, chọn Switch to Portrait.



1.2 Tạo một biến thể bố cục cho chiều ngang

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

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

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

Một cửa sổ mới mở ra với tab land/activity_main.xml hiển thị bố cục cho chiều ngang. Bạn có thể thay đổi bố cục này, dành riêng cho chiều ngang, mà bạn không cần thay đổi chiều dọc ban đầu.

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

```
▼ activity_main (2)
  </> activity_main.xml
  </> activity_main.xml (land)
```

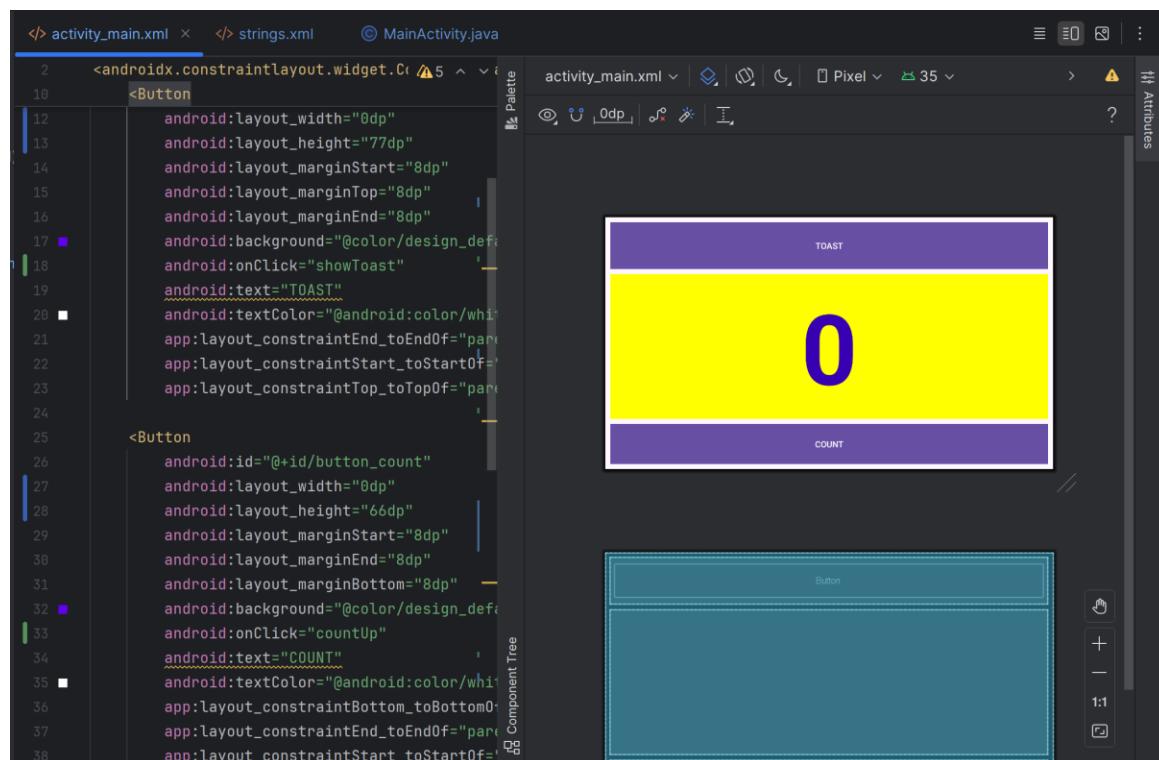
1.3 Xem trước bố cục cho các thiết bị khác nhau

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

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

1.4 Thay đổi bố cục theo chiều ngang

Bạn có thể sử dụng bảng Attributes trong tab Design để thiết lập hoặc thay đổi các thuộc tính, nhưng đôi khi việc sử dụng tab Text để chỉnh sửa mã XML trực tiếp sẽ nhanh hơn. Tab Text hiển thị mã XML và cung cấp một tab Preview ở bên phải cửa sổ để hiển thị bản xem trước bố cục, như được hiển thị trong hình dưới đây.



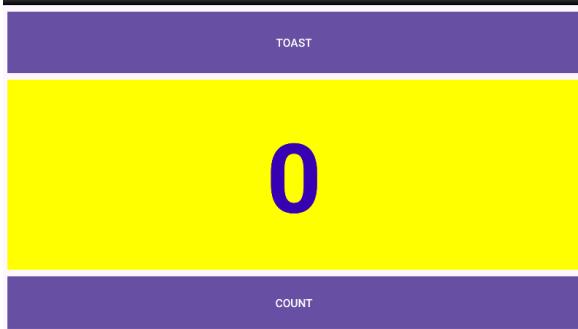
Hình trên cho thấy:

1. Tab Preview, bạn sử dụng để hiển thị ngăn xem trước.
2. Khung xem trước

3. Mã XML

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

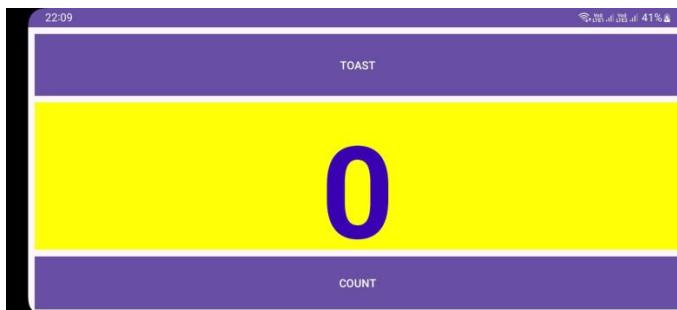
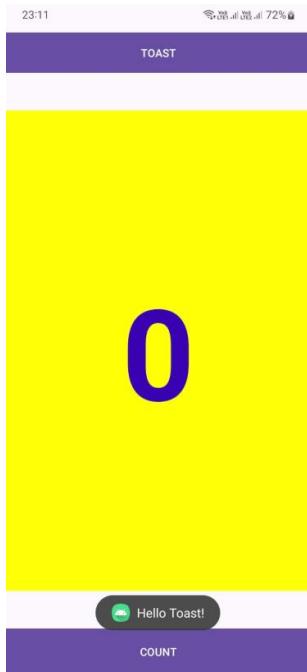
1. Tab activity_main.xml(Land) nên vẫn mở trong trình chỉnh sửa bố cục, nếu không hãy nhấp vào tệp activity_main.xml(land) trong thư mục layout.
2. Nhấp vào tab Text và tav Preview (nếu chưa được chọn).
3. Tìm phần tử TextView trong mã XML
4. Thay đổi thuộc tính android:textSize="160sp" thành android:textSize="120sp". Bản xem trước bố cục sẽ hiển thị kết quả:



5. Chọn các thiết bị khác trong menu thả xuống Device in Editor để xem cách bố cục hiển thị trên các thiết bị khác nhau ở chiều ngang.

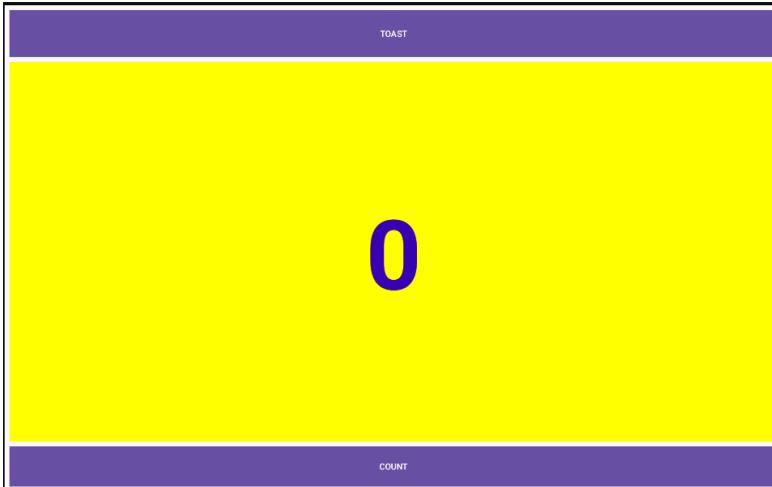
Trong bảng chỉnh sửa, tab activity_main.xml(land) hiển thị bố cục cho chiều ngang. Tab activity_main.xml hiển thị bố cục không thay đổi cho chiều dọc. Bạn có thể chuyển đổi qua lại bằng cách nhấp vào các tab.

6. Chạy ứng dụng trên trình giả lập hoặc thiết bị và chuyển hướng từ dọc sang ngang để xem cả hai bố cục.



1.5 Tạo biến thẻ bố cục cho máy tính bảng

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



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

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

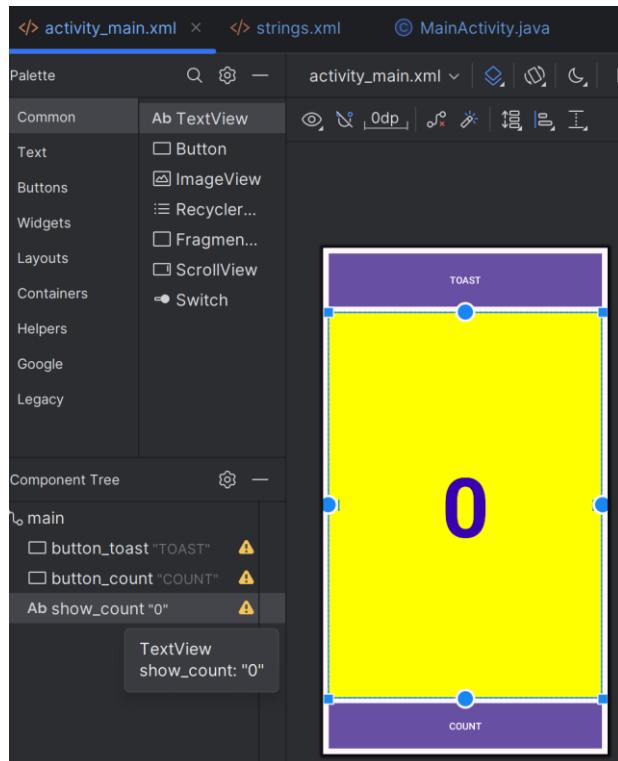
Một cửa sổ biên tập mới mở ra với tab `xlarge/activity_main.xml` hiển thị bố cục cho thiết bị có kích thước máy tính bảng. Trình biên tập cũng chọn một thiết bị máy tính bảng, chẳng hạn như Nexus 9 hoặc 10, để xem trước. Bạn có thể thay đổi bố cục này, dành riêng cho máy tính bảng, mà không cần thay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

1.6 Thay đổi biến thể bố cục cho máy tính bảng

Bạn có thể sử dụng bảng Attributes trong tab Design để thay đổi thuộc tính cho bố cục này:

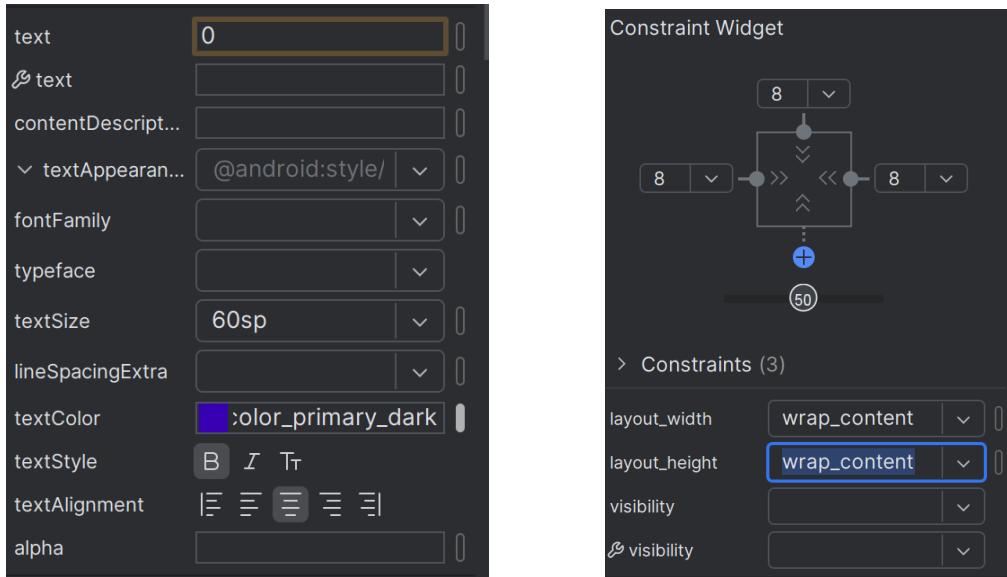
1. Tắt công cụ Autoconnect trên thanh công cụ. Để thực hiện bước này, hãy đảm bảo rằng công cụ đã bị vô hiệu hóa.
2. Xóa tất cả các ràng buộc trong bố cục bằng cách nhấp vào nút Clear All Constraint trên thanh công cụ.
Khi các ràng buộc đặc được loại bỏ, bạn có thể di chuyển và thay đổi kích thước các thành phần trên bố cục một cách tự do.
3. Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước ở bốn góc của một phần tử để thay đổi kích thước nó. Trong Componet Tree, chọn `TextView` có tên `show_count`. Để di chuyển `TextView` ra khỏi đường đi để

bạn có thể tự do kéo các phần tử Button, hãy kéo một góc của nó để thay đổi kích thước, như được hiển thị trong hình động bên dưới.



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

4. Chọn nút button_toast trong Component Tree, nhấp vào tab Attributes ở bảng Attributes và thay đổi thuộc tính textSize thành 60sp (#1 trong hình dưới đây) và width thành wrap_content(#2 trong hình dưới đây)



Như được hiển thị trên hình bên dưới, bạn có thể nhấp vào điều khiển chiều rộng của 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. Ngoài ra, bạn có thể chọn wrap_content từ menu layout_width.

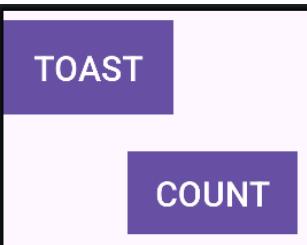
Bạn sử dụng wrap_content để nếu văn bản nút được định nghĩa thành một ngôn ngữ khác, Button sẽ xuất hiện rộng hơn hoặc mỏng hơn để phù hợp với tú trong ngôn ngữ khác.

- Chọn button_count trong Component Tree, thay đổi textSize thành 60SP và Layout_width thành Wrap_Content và kéo nút phía trên TextView vào một khoảng trống trong Layout.

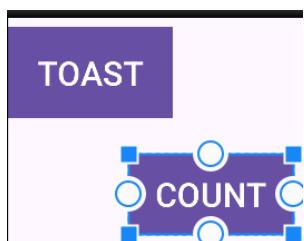
1.7 Sử dụng ràng buộc cơ sở

Bạn có thể căn chỉnh một phần tử UI có chứa text, chẳng hạn như TextView hoặc Button, với một phần tử UI khác có chứa text. Một ràng buộc cơ bản cho phép bạn ràng buộc các phần tử sao cho đường cơ sở văn bản khớp với nhau

- Giới hạn button_toast ở phía trên và phía bên trái của bố cục, kéo button_count đến một khoảng trống gần button_toast, và giới hạn button_count ở phía bên trái của button_toast, như thể hiện trong hình động:



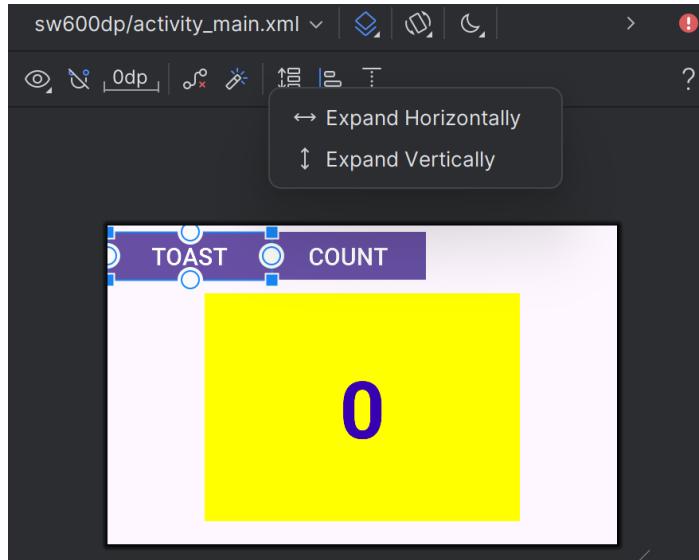
2. Sử dụng ràng buộc đường cơ sở, bạn có thể ràng buộc button_count sao cho đường cơ sở văn bản của nó khớp với đường cơ sở văn bản của button_toast. Nhấp chuột phải vào button_count và chọn Hiển thị đường cơ sở từ menu.
3. Nhấp vào nút ràng buộc đường cơ sở. Tay cầm cơ sở xuất hiện, nhấp nháy màu xanh lục như trong hình động. Nhấp và kéo một đường ràng buộc đường cơ sở vào đường cơ sở của phần tử button_toast



1.8 Mở rộng các nút theo chiều ngang

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

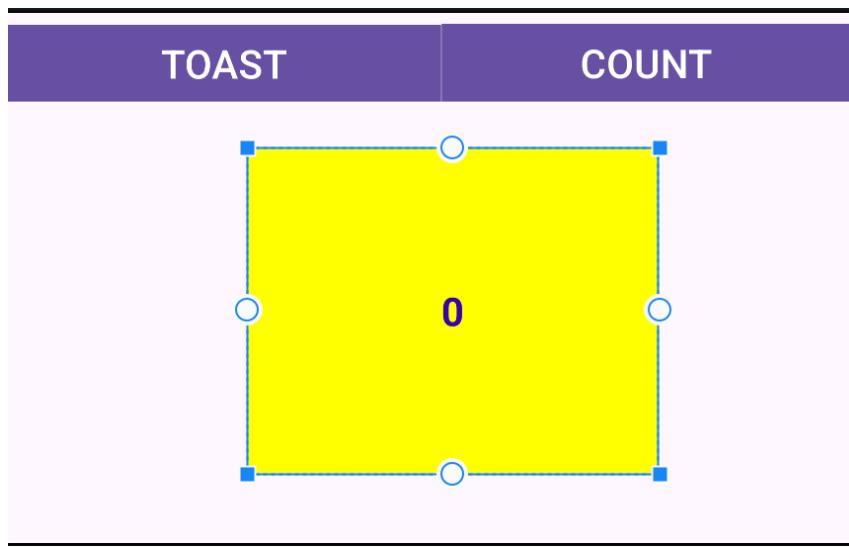
1. Chọn nút button_count trong Component Tree và chọn nút button_toast để cả hai được chọn.
2. Nhấp vào nút Pack trên thanh công cụ và chọn Expand Horizontally như trong hình bên dưới .



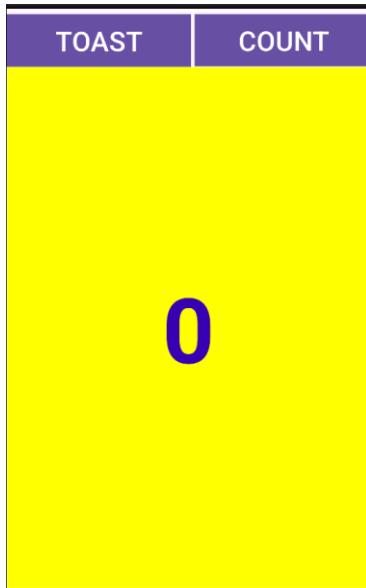
Các phần tử Button mở rộng theo chiều ngang để lấp đầy bối cảnh như hình dưới đây



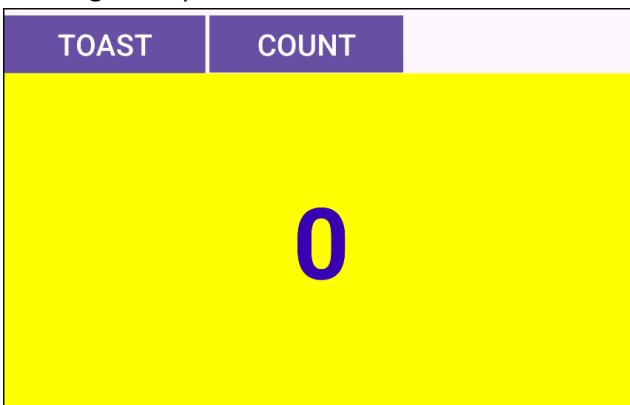
- Để hoàn tất bố cục, hãy hạn chế TextView show_count ở cuối Nút button_toast và ở hai bên và dưới cùng của bố cục, như trong hình động bên dưới.



- 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 phù hợp trên điện thoại và máy tính bảng có kích thước và mật độ màn hình khác nhau

Mẹo: Để biết hướng dẫn chuyên sâu về cách sử dụng ConstraintLayout, hãy xem Sử dụng ConstraintLayout để thiết kế chế độ xem của bạn.

Mã giải pháp nhiệm vụ 1

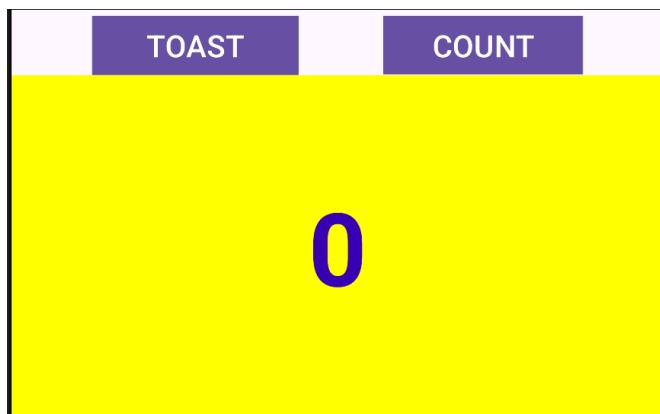
Dự án Android Studio: HelloToast

Thử thách mã hóa 1

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

Thử thách : Để phù hợp với hướng ngang (ngang) cho máy tính bảng, bạn có thể căn giữa các phần tử Button trong activity_main.xml (xlarge) để chúng xuất hiện như trong hình bên dưới

Gợi ý: Chọn các phần tử, nhấp vào nút căn chỉnh trên thanh công cụ và chọn Căn giữa theo chiều ngang



Mã giải pháp thử thách 1

Dự án HelloToastChallenge2

Nhiệm vụ 2: Thay đổi bố cục thành LinearLayout

LinearLayout là một ViewGroup sắp xếp tập hợp các view của nó theo dạng hàng ngang hoặc dọc. LinearLayout là một trong những bố cục phổ biến nhất vì nó đơn giản và nhanh. Nó thường được sử dụng trong một nhóm view khác để sắp xếp các phần tử UI theo chiều ngang hoặc dọc.

LinearLayout yêu cầu những thuộc tính sau:

- Layout_width
- Layout_height
- Orientation

Layout_width và layout_height có thể nhận một trong các giá trị sau:

- Match_parent: Mở rộng view để lấp đầy view thao chiều rộng hoặc chiều cao. Khi LinearLayout là view gốc, nó sẽ mở rộng để chiếm toàn bộ màn hình(view cha).

- Wrap_content: Thu nhỏ kích thước view để view chỉ vừa đủ để chứa nội dung. Nếu không có nội dung view sẽ trở nên vô hình.
- Số dp cố định(đơn vị pixel không phụ thuộc mật độ): Xác định kích thước cố định, được điều chỉnh theo mật độ màn hình của thiết bị. Ví dụ, 16dp có nghĩa là 16 pixel không phụ thuộc mật độ.

Orientation có thể là:

- Horizontal: Các view được sắp xếp từ trái sang phải
- Vertical: Các view được sắp xếp từ trên xuống dưới

Trong nhiệm vụ này, bạn sẽ thay đổi nhóm view gốc từ ConstraintLayout thành LinearLayout cho ứng dụng Hello Toast để thực hành sử dụng ứng dụng LinearLayout.

2.1 Thay đổi nhóm view gốc thành LinearLayou

1. Mở ứng dụng Hello Toast từ nhiệm vụ trước đó
2. Mở tệp activity_main.xml (nếu nó chưa được mở), và nhấp vào tab Text ở dưới cùng của cửa sổ chỉnh sửa để xem mã XML. Phía trên cùng của mã XML là dòng mã sau:

```
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/
```

3. Thay đổi thẻ <sandrodx.constraintLayout.widget.ConstraintLayout> thành <LinearLayout> sao cho mã có dạng sau:

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

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

5. Dưới dòng thẻ <LinearLayout>, thêm thuộc tính sau vào 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 cởi constraintLayout và không liên quan đến LinerLayout

2.2 Thay đổi thuộc tính phần tử cho LinearLayout

Thực hiện theo các bước sau để thay đổi các thuộc tính của phần tử UI sao cho chúng hoạt động với LinearLayout:

1. Mở ứng dụng Hello Toast từ nhiệm vụ trước đó

2. Mở file layout activity_main.xml (nếu chưa mở), và nhấp vào tab Text.

3. Tìm phần tử button_toast kiểu Button và thay đổi thuộc tính sau:

```
    android:layout_width="798dp"
```

Thành : `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 và thay đổi thuộc tính sau:

```
    android:layout_width="0dp"
```

Thành `android:layout_width="match_parent"`

6. Xóa các thuộc tính sau khỏi button_count:

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

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

```
    android:layout_width="0dp"
```

```
    android:layout_height="0dp"
```

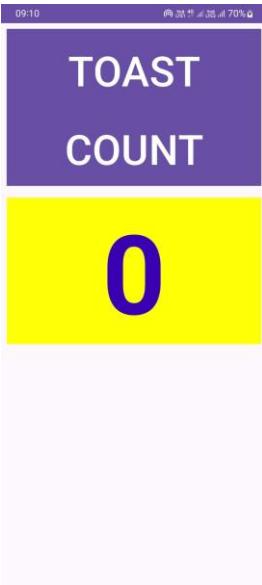
```
        android:layout_width="match_parent"
```

Thành: `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_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/button_toast"
    app:layout_constraintVertical_bias="1.0" />
```

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



2.3 Thay đổi vị trí các phần tử trong LinearLayout

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

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

1. Mở ứng dụng Hello Toast từ nhiệm vụ trước đó
2. Mở tệp layout `activity_main.xml`(nếu chưa được mở) và nhấp vào tab Text.
3. Chọn nút `button_count` và tất cả các thuộc tính của nó, từ thẻ `<Button>` cho đến và bao gồm thẻ đóng `'/>'`, sau đó chọn Edit > Cut
4. Nhấp vào vị trí sau thẻ đóng `'/>'` của phần tử `TextView` nhưng trước thẻ đóng `'</LinearLayout>'` và chọn Edit > Paste
5. (Tùy chọn) Để sửa bất kỳ vấn đề về thụt lề hoặc khoảng cách cho mục đích thẩm mỹ, chọn Code > Reformat Code để định dạng lại mã XML với khoảng cách và thụt lè đúng.

Mã XML cho các phần tử UI bây giờ trông như sau:

```
</> layout\activity_main.xml </> land\activity_main.xml </> sw600dp\activity_main.xml

1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:id="@+id/main"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      android:orientation="vertical"
9      tools:context=".MainActivity">
10
11      <Button
12          android:id="@+id/button_toast"
13          android:layout_width="390dp"
14          android:layout_height="wrap_content"
15          android:layout_marginStart="8dp"
16          android:layout_marginTop="8dp"
17          android:layout_marginEnd="8dp"
18          android:background="@color/design_default_color_primary"
19          android:onClick="showToast"
20          android:text="TOAST"
21          android:textColor="@android:color/white"
22          android:textSize="30sp" />
23
24      <TextView
25          android:id="@+id/show_count"
26          android:layout_width="match_parent"
27          android:layout_height="wrap_content"
28          android:layout_marginStart="8dp"
29          android:layout_marginTop="8dp"
30          android:layout_marginEnd="8dp"
31
32          android:layout_marginStart="8dp"
33          android:layout_marginTop="8dp"
34          android:layout_marginEnd="8dp"
35          android:layout_marginBottom="8dp"
36          android:background="#FFFF00"
37          android:gravity="center_horizontal|center_vertical"
38          android:text="0"
39          android:textAlignment="center"
40          android:textColor="@color/design_default_color_primary_dark"
41          android:textSize="160sp"
42          android:textStyle="bold" />
43
44      <Button
45          android:id="@+id/button_count"
46          android:layout_width="match_parent"
47          android:layout_height="108dp"
48          android:layout_marginStart="8dp"
49          android:layout_marginEnd="8dp"
50          android:layout_marginBottom="8dp"
51          android:background="@color/design_default_color_primary"
52          android:onClick="countUp"
53          android:text="COUNT"
54          android:textColor="@android:color/white"
55          android:textSize="60sp" />
56
57  </LinearLayout>
```

Bằng cách di chuyển nút button_count xuống dưới phần tử TextView, bố cục hiện tại đã gần giống như trước, với nút Count ở dưới cùng. Bản xem trước của bố cục bây giờ trong như sau:



2.4 Thêm trọng số cho phần tử TextView

Việc chỉ định các thuộc tính gravity và weight sẽ giúp bạn kiểm soát tốt hơn việc sắp xếp các view và nội dung trong một LinearLayout.

Thuộc tính “android:gravity” xác định cách căn chỉnh nội dung của một view bên trong chính view đó. Trong bài học trước, bạn đã thiết lập thuộc tính này cho phần tử “show_count”. TextView để căn giữa nội dung (số 0) ở giữa TextView:

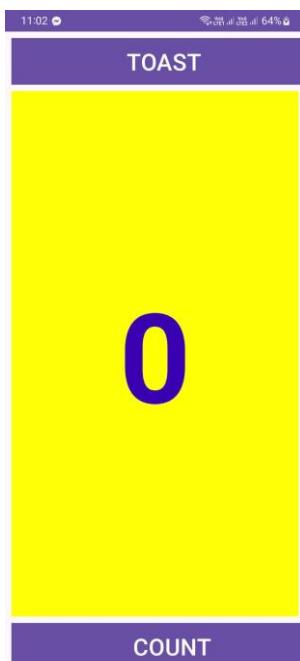
```
android:gravity="center_horizontal|center_vertical"
```

Thuộc tính “android:layout_weight” cho biết không gian dư thừa trong LinearLayout sẽ được phân bổ cho View. Nếu chỉ có một View có thuộc tính này, nó sẽ nhận toàn bộ không gian màn hình dư thừa. Đối với nhiều phần tử View, không gian sẽ được phân bổ theo tỷ lệ. Ví dụ, nếu các phần tử button mỗi cái có trọng số là 1 và TextView có trọng số là 2, tổng cộng là 4 thì mỗi phần tử button sẽ nhận được một không gian và TextView sẽ nhận được một nửa còn lại.

Trên các thiết bị khác nhau, bố cục có thể hiển thị phần tử “show_count” TextView chiếm một phần hoặc hầu hết không gian 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
Hãy làm theo các bước sau:

1. Mở ứng dụng Hello Toast từ nhiệm vụ trước
2. Mở tệp layout activity_main.xml (nếu chưa được mở) và nhấp vào tab Text
3. Tìm phần tử “show_count” TextView và thêm thuộc tính sau:
`android:layout_weight="1"`

Bản xem trước bây giờ trông như hình sau:



Phần tử “show_count” TextView chiếm toàn bộ không gian các nút. Bạn có thể xem trước bối cảnh cho các thiết bị khác nhau, như bạn đã làm trong nhiệm vụ trước, bằng cách nhấp vào nút Device in Editor

Pixel trên thanh công cụ ở phía trên của khung xem trước, và chọn một thiết bị khác. Dù bạn chọn thiết bị nào xem trước, phần tử “show_count” TextView sẽ luôn chiếm toàn bộ không gian giữa các nút.

Mã giải pháp nhiệm vụ 2

Mã XML trong activity_main.xml

```
</> layout\activity_main.xml </> land\activity_main.xml </> sw600dp\activity_main.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  @<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:id="@+id/main"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      android:orientation="vertical"
9      tools:context=".MainActivity">
10
11     <Button
12         android:id="@+id/button_toast"
13         android:layout_width="390dp"
14         android:layout_height="wrap_content"
15         android:layout_marginStart="8dp"
16         android:layout_marginTop="8dp"
17         android:layout_marginEnd="8dp"
18         android:background="@color/design_default_color_primary"
19         android:onClick="showToast"
20         android:text="TOAST"
21         android:textColor="@android:color/white"
22         android:textSize="30sp" />
23
24     <TextView
25         android:id="@+id/show_count"
26         android:layout_width="match_parent"
27         android:layout_height="wrap_content"
28         android:layout_weight="1"
29         android:layout_marginStart="8dp"
30         android:layout_marginTop="8dp"
```

```
        android:layout_marginBottom="8dp"
        android:background="#FFFF00"
        android:gravity="center_horizontal|center_vertical"
        android:text="0"
        android:textAlignment="center"
        android:textColor="@color/design_default_color_primary_dark"
        android:textSize="160sp"
        android:textStyle="bold" />

    <Button
        android:id="@+id/button_count"
        android:layout_width="411dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:background="@color/design_default_color_primary"
        android:onClick="countUp"
        android:text="COUNT"
        android:textColor="@android:color/white"
        android:textSize="30sp" />
</LinearLayout>
```

Nhiệm vụ 3: Thay đổi bố cục thành RelativeLayout

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

3.1 Thay đổi LinearLayout thành RelativeLayout

Một cách dễ dàng chuyển LinearLayout thành RelativeLayout là thêm các thuộc tính XML trong tab Text.

1. Mở tệp layout activity_main.xml và nhấp vào tab Text ở dưới cùng của cửa sổ chỉnh sửa để xem mã XML
2. Thay đổi <LinearLayout> ở trên cùng thành <RelativeLayout> sao cho dòng mã 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 được thay đổi thành </RelativeLayout>; nếu nó chưa thay đổi, hãy chỉnh sửa thủ công.

3.2 Sắp xếp lại các view trong RelativeLayout

Một cách dễ dàng chuyển LinearLayout thành RelativeLayout là thêm các thuộc tính XML trong tab Text.

1. Nhấp vào tab Preview ở bên cạnh trình soạn thảo(nếu chưa được chọn) để xem bản xem trước layout, hiện tại trông giống như hình dưới đây



Với việc chuyển sang RelativeLayout, trình chỉnh sửa bố cục cũng đã thay đổi một số thuộc tính của các view. Ví dụ:

- Nút Count(button_count) phủ lên nút Toast(button_toast), đó là lý do bạn không thấy nút Toast
 - Phần trên của TextView (show_count) phủ lên các nút Button
2. Thêm thuộc tính android:layout_below vào nút button_count để đặt nút Button này ngay phía dưới TextView show_count. Thuộc tính này là một trong nhiều thuộc tính để định vị các view trong RelativeLayout - bạn sẽ đặt các view theo mối quan hệ với các view khác.
 3. Thêm thuộc tính android:layout_centerHorizontal vào cùng một button để căn giữa view theo chiều ngang trong RelativeLayout, đây là nhóm view cha trong trường hợp này:

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

Mã XML đầy đủ cho nút button_count là như sau:

```
<Button  
    android:id="@+id/button_count"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_below="@+id/show_count"  
    android:layout_centerHorizontal="true"  
    android:layout_marginStart="8dp"  
    android:layout_marginEnd="8dp"  
    android:layout_marginBottom="8dp"  
    android:background="@color/design_default_color_primary"  
    android:onClick="countUp"  
    android:text="COUNT"  
    android:textColor="@android:color/white"  
    android:textSize="30sp" />
```

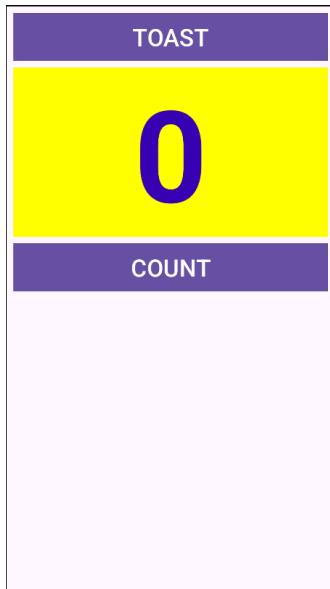
4. Thêm các thuộc tính sau vào TextView show_count:

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

Thuộc tính android:layout_alignParentLeft căn chỉnh view về phía bên trái của nhóm view cha RelativeLayout. Mặc dù thuộc tính này tự nó đã đủ để căn chỉnh view sang bên trái, bạn có thể muốn view căn chỉnh về phía bên phải nếu ứng dụng đang chạy trên một thiết bị sử dụng ngôn ngữ từ phải sang trái. Do đó, thuộc tính android:layout_alignParentStart sẽ giúp mặt bắt đầu của view này căn chỉnh với mặt bắt đầu của nhóm view cha. Mặt bắt đầu là cạnh trái của màn hình nếu ứng dụng sử dụng ngôn ngữ từ trái sang

phải hoặc là cạnh phải của màn hình nếu ứng dụng sử dụng ngôn ngữ từ phải sang trái.

5. Xóa thuộc tính android:layout_weight = “1” khỏi TextView show_count, vì thuộc tính này không liên quan đến RelativeLayout. Bạn xem trước layout bây giờ trông như hình dưới đây



Mẹo: RelativeLayout giúp bạn dễ dàng và nhanh chóng đặt các phần tử UI trong một bối cảnh. Để tìm hiểu thêm về cách định vị các view trong RelativeLayout, hãy xem phần “Positioning Views” trong chủ đề “RelativeLayout” của hướng dẫn API

Giải pháp cho bài tập 3

Mã XML trong tệp activity_main.xml

```

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

    <Button
        android:id="@+id/button_toast"
        android:layout_width="397dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:background="@color/design_default_color_primary"
        android:onClick="showToast"
        android:text="TOAST"
        android:textColor="@android:color/white"
        android:textSize="30sp" />
    <TextView
        android:id="@+id/show_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/button_toast"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentEnd="true"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:background="#FFFF00"
        android:gravity="center_horizontal|center_vertical"
        android:text="0"
        android:textAlignment="center"
        android:textColor="@color/design_default_color_primary_dark"
        android:textSize="160sp"
        android:textStyle="bold" />
    <Button
        android:id="@+id/button_count"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/show_count"
        android:layout_centerHorizontal="true"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="8dp"
        android:background="@color/design_default_color_primary"
        android:onClick="countUp"
        android:text="COUNT"
        android:textColor="@android:color/white"
        android:textSize="30sp" />
</RelativeLayout>

```

Tóm tắt

Sử dụng trình chỉnh sửa bộ cục để xem trước và tạo các biến thể:

- Để xem trước bố cục ứng dụng với định hướng ngang trong trình chỉnh sửa bố cục, nhấp vào nút Orientation in Editor  trên thanh công cụ và chọn Switch to Landscape. Chọn Switch to Portrait để quay lại định hướng dọc.
- Để tạo biến thể của bố cục khác biệt cho định hướng ngang, nhấp vào nút Orientation in Editor và chọn Create Landscape Variation. Một cửa sổ chỉnh sửa mới sẽ mở ra với tab land/activity_main.xml hiển thị bố cục cho định dạng ngang (landscape)
- Để xem trước bố cục cho các thiết bị khác nhau mà không cần chạy ứng dụng trên thiết bị giả lập, nhấp vào nút Device in Editor  Pixel trên thanh công cụ và chọn một thiết bị
- Để tạo biến thể của bố cục khác biệt cho máy tính bảng (màn hình lớn hơn) nhấp vào nút Orientation in Editor và chọn Create layout x-large Variation. Một cửa sổ chỉnh sửa mới sẽ mở ra với tab xlarge/activity_main.xml hiển thị bố cục cho thiết bị có màn hình lớn (máy tính bảng)

Sử dụng ConstraintLayout:

- Để xóa tất cả các ràng buộc trong một bố cục có ConstraintLayout làm gốc, nhấp vào nút Clear All Constraints  trên thanh công cụ.
- Bạn có thể căn chỉnh một phần tử giao diện người dùng chứa văn bản, chẳng hạn như TextView hoặc Button, với một phần tử giao diện người dùng khác chứa văn bản. Một baseline constraint giúp bạn ràng buộc các phần tử sao cho các đường cơ sở của văn bản khớp với nhau.
- Để tạo một baseline constraint, di chuột qua phần tử giao diện người dùng cho đến khi nút baseline constraint  xuất hiện dưới phần tử đó.
- Nút pack  trên thanh công cụ cung cấp các tùy chọn để gói gọn hoặc mở rộng các phần tử giao diện người dùng đã chọn. Bạn có thể sử dụng nó để sắp xếp đều các phần tử Button theo chiều ngang trên bố cục.

Sử dụng LinearLayout:

- LinearLayout là một nhóm View sắp xếp bộ sưu tập các view của nó theo hàng ngang hoặc dọc.
- LinearLayout yêu cầu có các thuộc tính layout_width, layout_height và orientation

- `match_parent` cho `layout_width` hoặc `layout_height`: Mở rộng View để lấp đầy phần tử cha theo chiều rộng hoặc chiều cao. Khi LinearLayout là View gốc, nó sẽ mở rộng đến kích thước của màn hình (View cha).
- `wrap_content` cho `layout_width` hoặc `layout_height`: Thu nhỏ kích thước của View sao cho chỉ đủ lớn để chứa nội dung của nó. Nếu không có nội dung, View sẽ trở nên vô hình.
- Số dp (density-independent pixels) cố định cho `layout_width` hoặc `layout_height`: Xác định kích thước cố định, được điều chỉnh cho mật độ màn hình của thiết bị. Ví dụ, 16dp có nghĩa là 16 điểm ảnh độc lập với mật độ.
- Hướng của LinearLayout có thể là horizontal để sắp xếp các phần tử từ trái sang phải, hoặc vertical để sắp xếp các phần tử từ trên xuống dưới.
- Việc chỉ định các thuộc tính `gravity` và `weight` cung cấp cho bạn sự kiểm soát thêm trong việc sắp xếp các view và nội dung trong một LinearLayout.
- Thuộc tính `android:gravity` xác định cách căn chỉnh nội dung của một View bên trong chính nó.
- Thuộc tính `android:layout_weight` chỉ ra mức độ không gian dư thừa trong LinearLayout sẽ được phân bổ cho View. Nếu chỉ có một View có thuộc tính này, nó sẽ chiếm toàn bộ không gian dư thừa trên màn hình. Đối với nhiều phần tử View, không gian sẽ được phân bổ theo tỷ lệ. Ví dụ, nếu hai phần tử Button mỗi cái có trọng số 1 và một TextView có trọng số 2, tổng cộng là 4, thì các phần tử Button sẽ chiếm $\frac{1}{4}$ không gian mỗi cái, và TextView sẽ chiếm một nửa.

Sử dụng RelativeLayout:

- RelativeLayout là một nhóm view trong đó mỗi view được định vị và căn chỉnh liên quan đến các view khác trong cùng nhóm.
- Sử dụng `android:layout_alignParentTop` để căn chỉnh View lên trên cùng của phần tử cha.
- Sử dụng `android:layout_alignParentLeft` để căn chỉnh View sang bên trái của phần tử cha.
- Sử dụng `android:layout_alignParentStart` để căn chỉnh cạnh bắt đầu của View khớp với cạnh bắt đầu của phần tử cha. Thuộc tính này rất hữu ích nếu bạn muốn ứng dụng của mình hoạt động trên các thiết bị sử dụng các ngôn ngữ hoặc vùng khác nhau. Cạnh bắt đầu là cạnh bên trái của màn hình nếu cài đặt là từ trái qua phải, hoặc là cạnh bên phải của màn hình nếu cài đặt là từ phải qua trái.

Các khái niệm liên quan

Tài liệu về các khái niệm liên quan có thể tìm thấy trong mục 1.2: Bố cục và tài nguyên cho UI

Tìm hiểu thêm

Tài liệu Android Studio:

- Làm quen với Android Studio
- Tạo biểu tượng ứng dụng với Image Asset Studio

Tài liệu dành cho nhà phát triển Android:

- Bố cục
- Xây dựng UI bằng trình chỉnh sửa bố cục
- Xây dựng giao diện người dùng phản hồi bằng ConstraintLayout
- Bố cục
- LinearLayout
- RelativeLayout
- View
- Button
- TextView
- Hỗ trợ mật độ các điểm ảnh khác nhau

Khác:

- CodeLabs: Sử dụng ConstraintLayout để thiết kế các view tổng Android
- Danh sách thuật ngữ và khái niệm

Bài tập về nhà:

Thay đổi ứng dụng

Mở ứng dụng HelloToast:

1. Thay đổi tên dự án thành HelloConstraint và tái cấu trúc dự án thành HelloConstraint (để biết hướng dẫn cách sao chép và tái cấu trúc dự án, tham khảo phần phụ lục: tiện ích)
2. Chỉnh sửa layout activity_main.xml để căn chỉnh các phần tử nút Toast và Count sao cho chúng nằm theo chiều dọc bên trái của show_count TextView (hiển thị "0"). Tham khảo các hình minh họa dưới đây để biết bố cục
3. Thêm một nút thứ ba gọi là Zozo xuất hiện giữa nút Toast và Count

4. Phân phôi các phần tử Button theo chiều dọc giữa phần trên và phần dưới của TextView show_count
5. Đặt nút Zezo có nền màu xám
6. Đảm bảo rằng bạn đã bao gồm nút Zero để định hướng theo chiều ngang trong activity_main.xml (land), và cũng chọn màn hình có kích thước máy tính bảng trong activity_main (xlarge)
7. Làm cho nút Zezo thay đổi giá trị TextView show_count thành 0
8. Cập nhật trình xử lý nhấp chuột cho nút Count để nó thay đổi màu nền của chính nó, tùy thuộc vào việc số đếm mới là lẻ hay chẵn. Gợi ý: Đừng sử dụng findViewById để tìm nút Count. Có điều gì khác mà bạn có thể sử dụng không? Cảm thấy tự do để sử dụng các hằng số trong lớp Color cho hai màu nền khác nhau.
9. Cập nhật trình xử lý nhấp chuột cho nút Count để đặt màu nền cho nút Zero thành một màu khác ngoài màu xám để cho thấy nó hiện đang hoạt động. Gợi ý: Bạn có thể sử dụng findViewById trong trường hợp này.
10. Cập nhật trình xử lý nhấp chuột cho nút Zero để đặt lại màu sắc thành màu xám, để nó có màu xám khi số đếm là 0.

Activity_main.xml

```
<?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"
    android:orientation="vertical"
    tools:context=".MainActivity">

    <Button
        android:id="@+id/button_toast"
        android:layout_width="122dp"
        android:layout_height="50dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="64dp"
        android:backgroundTint="#7241C8"
        android:onClick="showToast"
        android:text="Toast"
        android:textColor="@android:color/white"
        android:textSize="25sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button_zezo"
        android:layout_width="111dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
```

```
        android:layout_width="111dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="168dp"
        android:backgroundTint="#AEAB6"
        android:onClick="resetCount"
        android:text="Zero"
        android:textSize="25dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/show_count"
        android:layout_width="224dp"
        android:layout_height="310dp"
        android:layout_below="@+id/button_toast"
        android:layout_alignParentStart="true"
        android:layout_alignParentLeft="true"
        android:background="#F8FFEB3B"
        android:gravity="center_horizontal|center_vertical"
        android:text="0"
        android:textAlignment="center"
        android:textColor="@color/design_default_color_primary_dark"
        android:textSize="160sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
```

```
        android:gravity="center_horizontal|center_vertical"
        android:text="0"
        android:textAlignment="center"
        android:textColor="@color/design_default_color_primary_dark"
        android:textSize="160sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.914"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.078" />
    <Button
        android:id="@+id/button_count"
        android:layout_width="120dp"
        android:layout_height="51dp"
        android:layout_centerHorizontal="true"
        android:layout_marginStart="16dp"
        android:layout_marginTop="268dp"
        android:backgroundTint="#7241C8"
        android:onClick="countUp"
        android:text="Count"
        android:textColor="@android:color/white"
        android:textSize="25dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

Land/Activity_main.xml

```
<?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"
    android:orientation="vertical"
    tools:context=".MainActivity">
    <Button
        android:id="@+id/button_toast"
        android:layout_width="122dp"
        android:layout_height="50dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="64dp"
        android:backgroundTint="#7241C8"
        android:onClick="showToast"
        android:text="Toast"
        android:textColor="@android:color/white"
        android:textSize="25sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <Button
        android:id="@+id/button_zzero"
        android:layout_width="111dp"
        android:layout_height="60dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="168dp"
        android:backgroundTint="#AEAAE6"
        android:backgroundTint="#AEAAE6"
        android:onClick="resetCount"
        android:text="Zzero"
        android:textSize="25dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
        android:backgroundTint="#AEAAE6"
        android:onClick="resetCount"
        android:text="Zzero"
        android:textSize="25dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/show_count"
        android:layout_width="538dp"
        android:layout_height="337dp"
        android:layout_below="@+id/button_toast"
        android:layout_alignParentStart="true"
        android:layout_alignParentLeft="true"
        android:background="#F8FFEB3B"
        android:gravity="center_horizontal|center_vertical"
        android:text="0"
        android:textAlignment="center"
        android:textColor="@color/design_default_color_primary_dark"
        android:textSize="160sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.792"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.351" />
```

```
        android:gravity="center_horizontal|center_vertical"
        android:text="0"
        android:textAlignment="center"
        android:textColor="@color/design_default_color_primary_dark"
        android:textSize="160sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.792"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.351" />
    <Button
        android:id="@+id/button_count"
        android:layout_width="120dp"
        android:layout_height="51dp"
        android:layout_centerHorizontal="true"
        android:layout_marginStart="16dp"
        android:layout_marginTop="268dp"
        android:backgroundTint="#7241C8"
        android:onClick="countUp"
        android:text="Count"
        android:textColor="@android:color/white"
        android:textSize="25dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java

```
package com.example.helloconstraint;

import ...

public class MainActivity extends AppCompatActivity {
    5 usages
    private int mCount = 0;
    3 usages
    private TextView mShowCount;
    5 usages
    private Button Zezo;
    4 usages
    private Button count;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mShowCount = (TextView) findViewById(R.id.show_count);
        Zezo = (Button) findViewById(R.id.button_zozo);
        count = (Button) findViewById(R.id.button_count);
        Zezo.setBackgroundColor(Color.DKGRAY);
    }

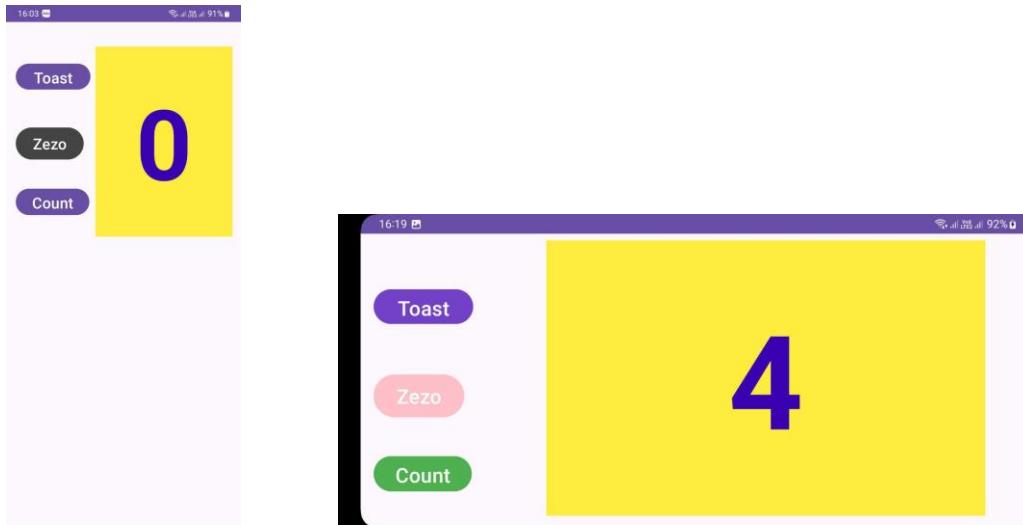
    3 usages
    public void showToast(View view) {
        Toast toast = Toast.makeText( context: this,"Hello Toast!", Toast.LENGTH_SHORT);
        toast.show();
    }
}
```

```

3 usages
public void countUp(View view) {
    mCount++;
    mShowCount.setText(String.valueOf(mCount));
    if(mCount % 2 == 0) {
        Zezo.setBackgroundColor(ContextCompat.getColor(context, R.color.pink));
        count.setBackgroundColor(ContextCompat.getColor(context, R.color.green));
    }else{
        Zezo.setBackgroundColor(ContextCompat.getColor(context, R.color.gray));
        count.setBackgroundColor(ContextCompat.getColor(context, R.color.green));
    }
}
1 usage
public void resetCount(View view){
    mCount = 0;
    mShowCount.setText(String.valueOf(mCount));
    Zezo.setBackgroundColor(Color.DKGRAY);
    count.setBackgroundColor(getResources().getColor(com.google.android.material.R.color.design_default_color_on_primary));
}

```

App



Trả lời những câu hỏi

Câu hỏi 1: Hai thuộc tính ràng buộc bô cục nào trên vị trí nút 0 khoảng cách bằng nhau giữa hai phần tử nút khác? (Chọn 2 câu trả lời.)

- app:layout_constraintBottom_toTopOf="@+id/button_count"
- android:layout_marginBottom="8dp"
- android:layout_marginStart="16dp"
- app:layout_constraintTop_toBottomOf="@+id/button_toast"
- android:layout_marginTop="8dp"

Câu hỏi 2: Thuộc tính ràng buộc bô cục nào trên nút Zero định vị theo chiều ngang với sự liên kết với hai phần tử nút khác?

- app:layout_constraintLeft_toLeftOf="parent"
- app:layout_constraintBottom_toTopOf="@+id/button_count"
- android:layout_marginBottom="8dp"
- app:layout_constraintTop_toBottomOf="@+id/button_toast"

Câu hỏi 3: Chữ ký chính xác cho một phương thức được sử dụng với thuộc tính android:onClick XML là gì?

- public void callMethod()
- **public void callMethod(View view)**
- private void callMethod(View view)
- public boolean callMethod(View view)

Câu hỏi 4: Trình xử lý click cho nút Count bắt đầu bằng chữ ký phương thức sau: *Public void countUp (View view)*

Kỹ thuật nào sau đây hiệu quả hơn để sử dụng trong trình xử lý này để thay đổi màu nền của phần tử Button? Chọn một:

- Sử dụng findViewById để tìm nút Count . Gán kết quả cho một biến View, sau đó sử dụng setBackgroundColor()
- **Sử dụng tham số view được chuyển đến trình xử lý nhấp chuột với setBackgroundColor() : view.setBackgroundColor()**

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Kiểm tra xem ứng dụng có các tính năng sau không

- Nó hiển thị nút Zero
- Nút Zero nằm giữa các nút Toast và Count

- Ứng dụng bao gồm việc triển khai activity_main.xml, activity_main.xml (land) và activity_main.xml (xlarge)
- Ứng dụng bao gồm một cách triển khai phuong thức xử lý nhấp chuột cho nút Zero để đặt lại số đếm về 0. Phương thức phải hiển thị số lượng không trong TextView show_count. Trình xử lý nhấp chuột cũng phải đặt lại màu nền của nút Zero thành màu xám
- Phương thức xử lý nhấp chuột cho nút Count đã được cập nhật để thay đổi màu nền của riêng nó tùy thuộc vào việc số đếm mới là lẻ hay chẵn. Phương thức này phải sử dụng tham số view để truy cập nút. Phương pháp này cũng phải thay đổi nền của nút Zero thành màu khác với màu xám.

1.4) Văn bản và các chế độ cuộn

Giới thiệu

Lớp TextView là một lớp con của View dùng để hiển thị văn bản trên màn hình. Bạn có thể điều khiển cách văn bản xuất hiện bằng các thuộc tính TextView trong tệp bố cục XML. Bài thực hành này hướng dẫn cách làm việc với nhiều phần tử TextView, bao gồm một phần tử cho phép người dùng cuộn nội dung theo chiều dọc.

Nếu bạn có nhiều thông tin hơn những gì có thể hiển thị trên màn hình của thiết bị, bạn có thể tạo một chế độ 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 sang trái. Bạn thường sử dụng chế độ cuộn cho các câu chuyện tin tức, bài viết, hoặc bất kì văn bản dài nào không hoàn toàn vừa với màn hình. Bạn cũng có thể sử dụng chế độ cuộn để cho phép người dùng nhập nhiều dòng văn bản hoặc để kết hợp với các phần tử UI (như trường văn bản và nút bấm) trong một chế độ cuộn.

Lớp ScrollView cung cấp bố cục cho chế độ cuộn. ScrollView là một lớp con của FrameLayout. Chỉ nên đặt một view duy nhất làm con bên trong nó - một view con chứa toàn bộ nội dung để cuộn. View con này có thể là một nhóm View (như LinearLayout) chứa các phần tử UI.

Các bố cục phức tạp có thể gấp vấn đề về hiệu suất với các view con như hình ảnh. Một lựa chọn tốt cho một view bên trong ScrollView là một LinearLayout được sắp xếp theo chiều dọc, trình bày các mục mà người dùng có thể cuộn qua (chẳng hạn như các phần tử TextView).

Với ScrollView át cả các phần tử UI đều nằm trong bộ nhớ và trong cấu trúc view ngay cả khi chúng không được hiển thị trên màn hình. Điều này làm cho ScrollView trở nên lý tưởng cho việc cuộn các trang văn bản tự do một cách mượt mà, vì văn bản đã có sẵn 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 của bạn. Để hiển thị danh sách dài các mục mà người dùng có thể thêm vào, xóa hoặc chỉnh sửa, hãy xem xét việc sử dụng RecyclerView được mô tả trong một bài học riêng.

Những điều bạn nên biết

Bạn có thể:

- Tạo ứng dụng Hello World với Android Studio
- Chạy một ứng dụng trên trình giả lập hoặc thiết bị
- Triển khai một TextView trong bố cục của ứng dụng
- Tạo và sử dụng tài nguyên chuỗi

Những gì bạn sẽ học:

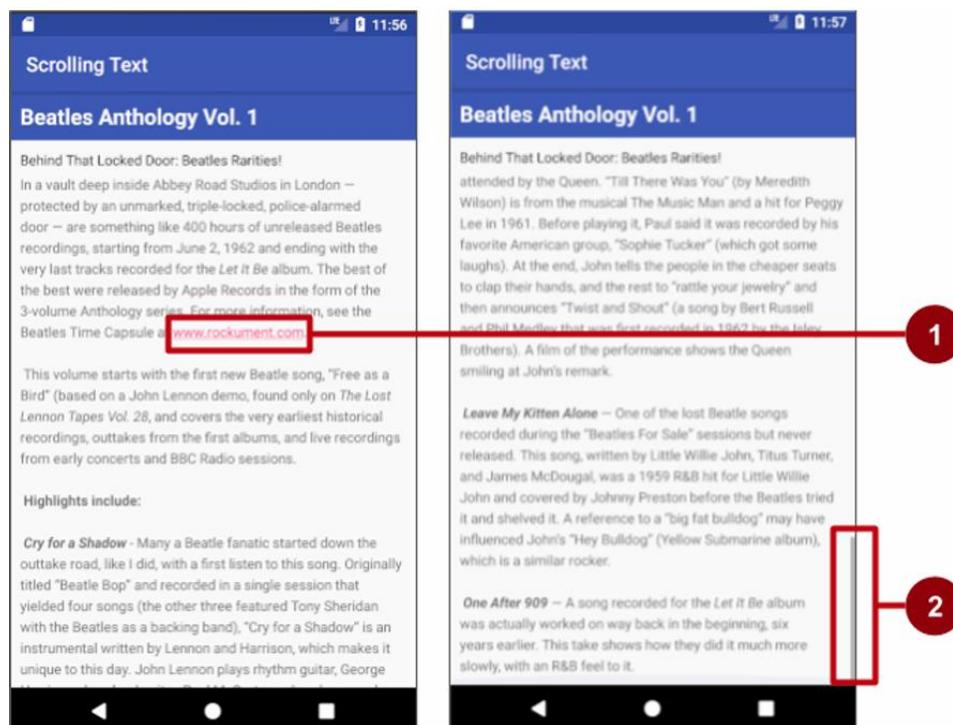
- Cách sử dụng mã XML để định nghĩa một View cuộn
- Cách hiển thị văn bản tự do với một thẻ định dạng HTML
- Cách định kiểu màu nền và màu văn bản cho TextView
- Cách bao gồm một liên kết web trong văn bản

Những gì bạn sẽ làm:

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

Tổng quan về ứng dụng

Ứng dụng Scrolling Text minh họa thành phần UI 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 một bài 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 chữ đậm hoặc chữ nghiêng, và với các ký tự xuống dòng để tách các đoạn văn. Bạn cũng có thể bao gồm các liên kết web hoạt động trong văn bản



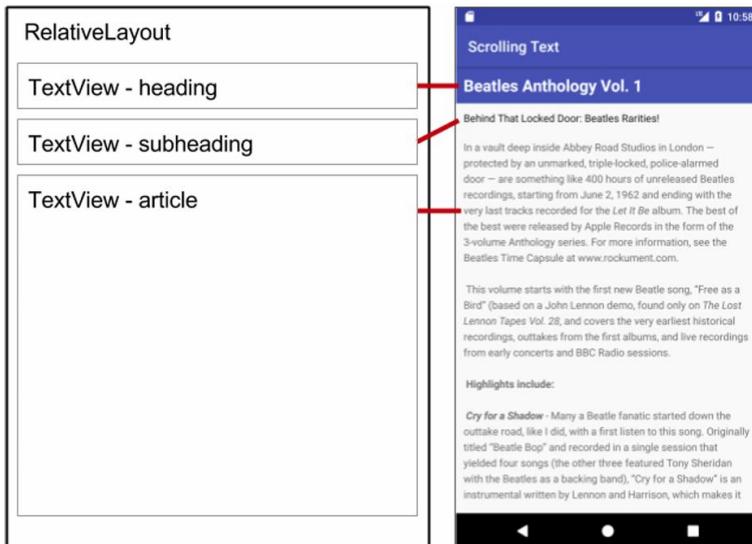
Trong hình trên, có các yếu tố sau xuất hiện:

1. Một liên kết web hoạt động được nhúng trong văn bản tự do.
2. Thanh cuộn xuất hiện khi cuộn văn bản.

Nhiệm vụ 1: Thêm và chỉnh sửa các phần tử của 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 tiêu đề phụ

của bài viết, và thay đổi phần tử TextView "Hello World" hiện có để hiển thị một bài viết dài. Hình dưới đây là sơ đồ của bố cục.



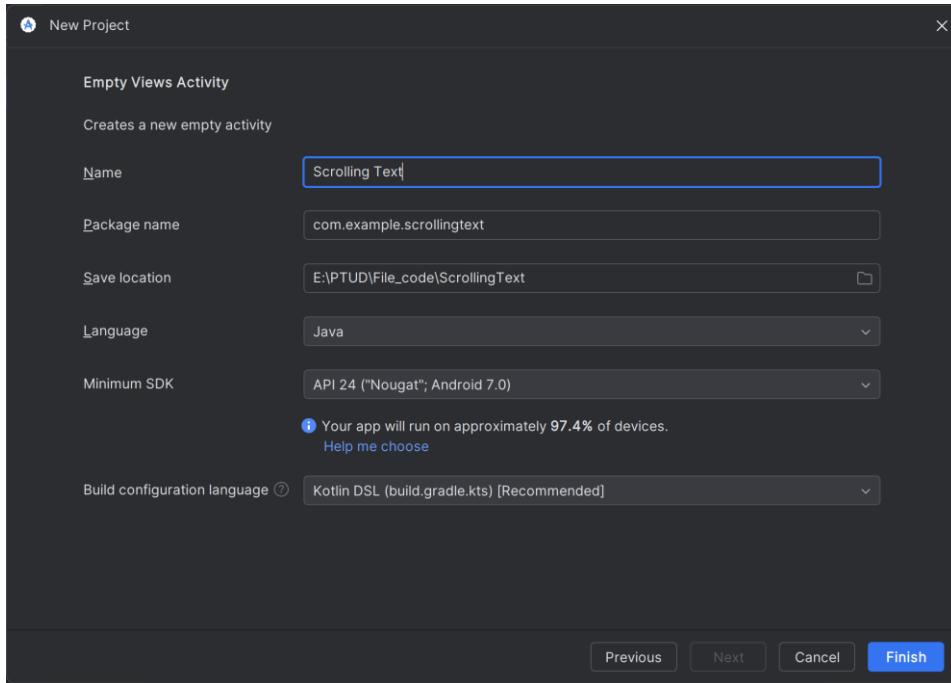
Bạn sẽ thực hiện tất cả những thay đổi này trong mã XML và trong tệp strings.xml. Bạn sẽ chỉnh sửa mã XML cho bố cục trong phần Text, mà bạn có thể hiển thị bằng cách nhấp vào tab Text, thay vì nhấp vào tab Design cho phần thiết kế. Một số thay đổi đối với các phần tử và thuộc tính giao diện người dùng dễ thực hiện hơn trực tiếp trong phần Text bằng cách sử dụng mã XML.

1.1 Tạo dự án và các phần tử TextView

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

Mẹo: Để tìm hiểu thêm về các thuộc tính này, hãy xem tài liệu tham khảo về TextView.

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



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

Ở đầu hoặc gốc của cấu trúc View là ViewGroup ConstraintLayout

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

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

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

RelativeLayout cho phép bạn đặt các phần tử giao diện người dùng tương đối với nhau, hoặc tương đối với chính RelativeLayout cha

Phần tử TextView mặc định Hello World được tạo bởi mẫu Layout trống vẫn có các thuộc tính ràng buộc (như app:layout_constraintBottom_toBottomOf="parent"). Để ng lo lắng, bạn sẽ xóa chúng trong bước tiếp theo

- Xóa dòng mã XML sau đây, liên quan đến ConstraintLayout

```
xmlns:app="http://schemas.android.com/apk/res-auto"
```

Khôi mã XML ở đầu bây giờ trông như sau:

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

5. Thêm một phần tử TextView ở trên phần tử TextView Hello World bằng cách nhập TextView. Một khối TextView sẽ xuất hiện và kết thúc bằng “/”, hiển thị các thuộc tính layout_width và layout_height là những thuộc tính bắt buộc cho TextView
6. Nhập các thuộc tính sau cho TextView. Khi bạn nhập mỗi thuộc tính và giá trị, các gợi ý sẽ xuất hiện để hoàn thành tên hoặc giá trị của thuộc tính.

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

7. Trích xuất tài nguyên chuỗi cho thuộc tính android:text có chuỗi cứng Article Title trong TextView để tạo một mục cho nó trong strings.xml. Đặt con trỏ vào chuỗi cứng, nhấn Alt-Enter (Option-Enter trên Mac), và chọn Extract string resource. Đảm bảo rằng tùy chọn Create the resource in directories được chọn, sau đó chỉnh sửa tên tài nguyên cho giá trị chuỗi thành article_title.
Tài nguyên chuỗi được mô tả chi tiết trong phần Tài nguyên Chuỗi.
8. Trích xuất tài nguyên kích thước cho thuộc tính android:padding có chuỗi cứng 10dp trong TextView để tạo dimens.xml và thêm một mục vào đó. Đặt con trỏ vào chuỗi cứng, nhấn Alt-Enter (Option-Enter trên Mac), và chọn Extract dimension resource. Đảm bảo rằng tùy chọn Create the resource in directories được chọn, sau đó chỉnh sửa tên tài nguyên thành padding_regular.
9. Thêm một phần tử TextView khác ở trên phần tử TextView Hello World và bên dưới TextView mà bạn đã tạo trong các bước trước. Thêm các thuộc tính sau cho TextView

```
<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:textAppearance="@android:style/TextAppearance.DeviceDefault"  
    android:text="@string/article_subtitle" />
```

Bởi vì bạn đã trích xuất tài nguyên kích thước cho chuỗi 10dp thành padding_regular trong TextView đã tạo trước đó, bạn có thể sử dụng @dimen/padding_regular cho thuộc tính android:padding trong TextView này

10. Trích xuất tài nguyên chuỗi cho thuộc tính android:text có chuỗi cứng Article Subtitle trong TextView thành article_subtitle

11. Trong phần tử TextView HelloWorld, xóa các thuộc tính layout_constraint

```
app:layout_constraintBottom_toBottomOf="parent"  
app:layout_constraintEnd_toEndOf="parent"  
app:layout_constraintStart_toStartOf="parent"  
app:layout_constraintTop_toTopOf="parent" />
```

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

```
<TextView  
    android:id="@+id/article"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_below="@+id/article_subheading"  
    android:lineSpacingExtra="5sp"  
    android:padding="@dimen/padding_regular"  
    android:text="@string/article_text" />
```

13. Trích xuất tài nguyên chuỗi cho Article text thành article_text và trích xuất tài nguyên kích thước cho 5sp thành line_spacing

14. Định dạng lại và căn chỉnh mã bằng cách chọn Code > Reformat Code. Đây là một thói quen tốt để định dạng và căn chỉnh mã của bạn, giúp cho bạn và những người khác dễ dàng hiểu hơn.

1.2 Thêm văn bản cho bài viết

Trong một ứng dụng thực tế, cập các bài viết từ tạp chí hoặc báo, các bài viết xuất hiện có thể đến từ một nguồn trực tuyến thông qua một nhà cung cấp nội dung, hoặc có thể được lưu trữ trước trong một cơ sở dữ liệu trên thiết bị.

Đối với bài thực hành này, bạn sẽ tạo bài viết dưới dạng một chuỗi dài trong tài nguyên strings.xml.

1. Trong thư mục app > res > values, mở file strings.xml
2. Mở bất kỳ tệp văn bản nào có một lượng lớn văn bản, hoặc mở file strings.xml của ứng dụng ScrollingText đã hoàn thành.
3. Nhập giá trị cho các chuỗi article_title và article_subtitle với một tiêu đề và phụ đề tự tạo, hoặc sử dụng các giá trị trong file strings.xml của ứng dụng ScrollingText đã hoàn thành. Đảm bảo rằng các giá trị chuỗi là văn bản một dòng mà không có thẻ HTML hoặc nhiều dòng.
4. Nhập hoặc sao chép và dán văn bản cho chuỗi article_text.

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

Hãy nhớ những điều sau (tham khảo hình bên dưới để có ví dụ):

- Khi bạn nhập hoặc dán văn bản vào file strings.xml, các dòng văn bản không tự động xuống dòng mà kéo dài ra ngoài lề phải. Đây là hành vi đúng - mỗi dòng văn bản mới bắt đầu từ lề trái đại diện cho một đoạn văn hoàn chỉnh. Nếu bạn muốn văn bản trong strings.xml được xuống dòng, bạn có thể nhấn Return để nhập các kết thúc dòng cứng, hoặc định dạng văn bản trước trong một trình soạn thảo văn bản với các kết thúc dòng cứng.
- Nhập \n để đại diện cho sự kết thúc của một dòng, và một \n khác để đại diện cho một dòng trống. Bạn cần thêm các ký tự kết thúc dòng để giữ cho các đoạn văn không bị chồng lên nhau.
- Nếu bạn có một dấu nháy đơn (') trong văn bản của mình, bạn phải thoát nó bằng cách đặt một dấu gạch chéo ngược (\) trước nó ('). Nếu bạn có một dấu nháy kép trong văn bản của mình, bạn cũng phải thoát nó ("). Bạn cũng phải thoát bất kỳ ký tự không phải ASCII nào khác. Xem phần Định dạng và kiểu dáng trong Tài nguyên Chuỗi để biết thêm chi tiết.
- Nhập các thẻ HTML và xung quanh các từ cần được in đậm.
- Nhập các thẻ HTML <i> và </i> xung quanh các từ cần được in nghiêng. Nếu bạn sử dụng dấu nháy đơn cong trong một cụm từ in nghiêng, hãy thay thế chúng bằng dấu nháy đơn thẳng.

- Bạn có thể kết hợp in đậm và in nghiêng bằng cách kết hợp các thẻ, như trong `<i>... từ... </i>`. Các thẻ HTML khác sẽ bị bỏ qua.
- Đặt toàn bộ văn bản bên trong `<string name="article_text"> </string>` trong file strings.xml.
- Bao gồm một liên kết web để kiểm tra, chẳng hạn như www.google.com. (Ví dụ dưới đây sử dụng www.rockument.com).
Đừng sử dụng thẻ HTML, vì bất kỳ thẻ HTML nào ngoại trừ thẻ in đậm và in nghiêng đều bị bỏ qua và được trình bày dưới dạng văn bản, điều này không phải là điều bạn muốn.

```

<resources>
    <string name="app_name">Scrolling Text</string>
    <string name="article_title">Beatles Anthology Vol. 1</string>
    <string name="article_subtitle">Behind That Locked Door: Beatles Rarities</string>
    <string name="article_text">In a vault deep inside Abbey Road Studios in London – protected by an unmarked, triple-locked, padlocked door – lies a time capsule containing a collection of rare Beatles recordings. This volume starts with the first new Beatle song, "Free as a Bird" (based on a John Lennon demo, found only on the bootleg <i>The Beatles Anthology Vol. 1</i>). Highlights include:<br><br><b><i>Cry for a Shadow</i></b> - Many a Beatle fanatic started down the bootleg road, like I did, with a first listen to this song. <b><i>My Bonnie</i></b> and <b><i>Ain't She Sweet</i></b> - At the same session, the Beatles played on "My Bonnie" (the first-ever recording of the song) and "Ain't She Sweet". <b><i>Searchin'</i></b> - A Jerry Leiber - Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular live favorite of the Beatles. <b><i>Love Me Do</i></b> - An early version of the song, played a bit slower and with more of a blues feeling, and a cool bossa-nova feel. <b><i>She Loves You - Till There Was You - Twist and Shout</i></b> - Live at the Princess Wales Theatre by Leicester Square in London, 1963. <b><i>Leave My Kitten Alone</i></b> - One of the lost Beatle songs recorded during the "Beatles For Sale" sessions but never released. <b><i>One After 999</i></b> - A song recorded for the <i>Let It Be</i> album was actually worked on way back in the beginning, so it's included here. </string>
</resources>

```

1.3 Chạy ứng dụng

Chạy ứng dụng. Bài viết xuất hiện, nhưng người dùng không thể cuộn bài viết vì bạn chưa bao gồm một ScrollView (điều này sẽ được thực hiện trong nhiệm vụ tiếp theo). Lưu ý rằng việc chạm vào một liên kết web hiện tại không có tác dụng gì. Bạn cũng sẽ sửa điều đó trong nhiệm vụ tiếp theo.



Giải pháp cho nhiệm vụ 1:

Tệp layout activity_main.xml trông như sau:

```
<?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=".com.example.scrollingtext.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:textColor="@color/white"
        android:padding="10dp"
        android:textAppearance="@android:style/TextAppearance.DeviceDefault.Large"
        android:textStyle="bold"
        android:text="Beatles Anthology Vol. 1" />
    <TextView
        android:id="@+id/article_subheading"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/article_heading"
        android:padding="0dp"
        android:textAppearance="@android:style/TextAppearance.DeviceDefault"
        android:text="Behind That Locked Door: Beatles Rarities" />
    <TextView
        android:id="@+id/article"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
<TextView  
    android:id="@+id/article"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_below="@+id/article_subheading"  
    android:lineSpacingExtra="5sp"  
    android:padding="0dp"  
    android:text="In a vault deep inside Abbey Road Studios in London – p..." />  
  
</RelativeLayout>
```

Nhiệm vụ 2: Thêm một ScrollView và một liên kết web hoạt động

Trong nhiệm vụ trước, bạn đã tạo ứng dụng ScrollingText với các phần tử TextView cho tiêu đề bài viết, phụ đề và văn bản bài viết dài. Bạn cũng đã bao gồm một liên kết web, nhưng liên kết đó vẫn chưa hoạt động. Bạn sẽ thêm mã để làm cho nó hoạt động.

Ngoài ra, TextView không thể cho phép người dùng cuộn văn bản bài viết để xem tất cả nội dung. Bạn sẽ thêm một ViewGroup mới gọi là ScrollView vào bộ cục XML để làm cho TextView có thể cuộn được.

2.1 Thêm thuộc tính autoLink cho các liên kết web hoạt động

Thêm thuộc tính android:autoLink= “web” vào TextView của bài viết. Mã XML cho TextView này bây giờ trông như sau:

```
<TextView  
    android:id="@+id/article"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_below="@+id/article_subheading"  
    android:lineSpacingExtra="5sp"  
    android:autoLink="web"  
    android:padding="@dimen/padding_regular"  
    android:text="In a vault deep inside Abbey Road Studios in London – p..." />
```

2.2 Thêm một ScrollView vào bộ cục

Để làm cho một View (chẳng hạn như một TextView) có thể cuộn được, hãy nhúng View đó bên trong một ScrollView.

1. Thêm một ScrollView giữa TextView của article_subheading và TextView của article.

Khi bạn nhập ScrollView Android Studio sẽ tự động thêm </ScrollView> ở cuối và hiển thị các thuộc tính android:layout_width và android:layout_height và các gợi ý

2. Chọn wrap_content từ các gợi ý cho cả hai thuộc tính.

Mã cho hai phần tử TextView và ScrollView bây giờ trông như sau:

```
<ScrollView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content">  
</ScrollView>
```

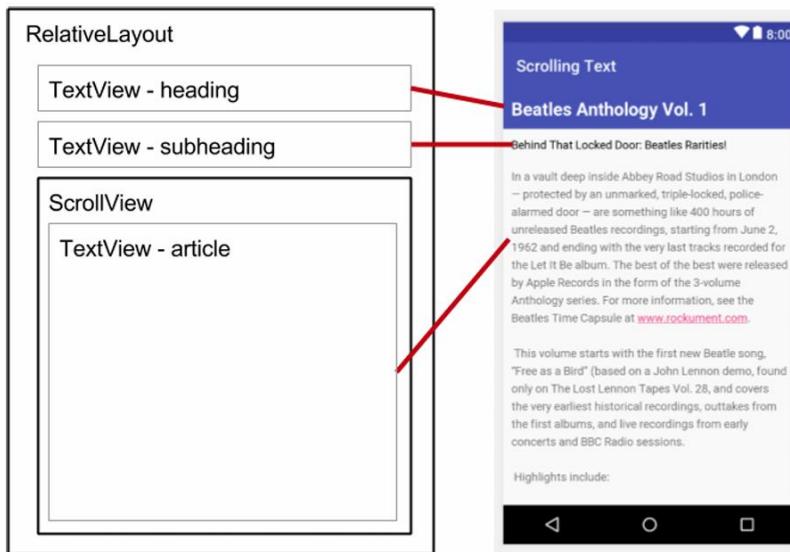
3. Di chuyển mã kết thúc</ScrollView> sau TextView của article để các thuộc tính của TextView bài viết hoàn toàn nằm bên trong ScrollView.
4. Di chuyển thuộc tính sau khỏi TextView của article và thêm nó vào ScrollView:

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

Với thuộc tính trên, phần tử ScrollView sẽ xuất hiện bên dưới tiêu đề phụ của bài viết. Bài viết nằm bên trong phần tử ScrollView.

5. Chọn Code > Reformat Code để định dạng lại mã XML, để TextView của bài viết bây giờ xuất hiện được thụt lề bên trong mã <ScrollView>
6. Nhấp vào tab Preview ở phía bên phải của trình chỉnh sửa bố cục để xem trước bố cục.

Bố cục bây giờ trông giống như bên phải của hình dưới đây:

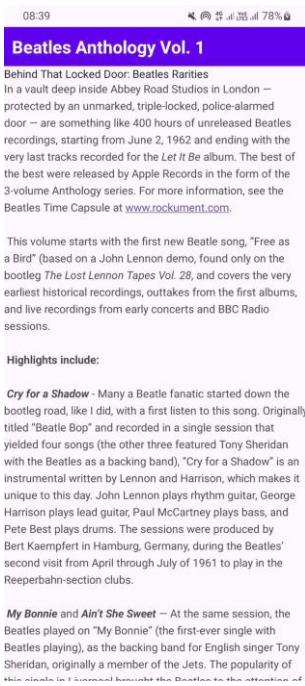


2.3 Chạy ứng dụng

Để kiểm tra cách văn bản cuộn:

1. Chạy ứng dụng trên một thiết bị hoặc trình giả lập.
Kéo lên và xuống để cuộn bài viết. Thanh cuộn sẽ xuất hiện ở lề phải khi bạn cuộn.
Nhấn vào liên kết web để truy cập trang web. Thuộc tính `android:autoLink` sẽ biến bất kỳ URL nào có thể nhận diện trong TextView (chẳng hạn như www.rockument.com) thành một liên kết web.

2. Xoay thiết bị hoặc trình giả lập trong khi chạy ứng dụng. Lưu ý cách mà vùng cuộn mở rộng để sử dụng toàn bộ màn hình và vẫn cuộn đúng cách.
3. Chạy ứng dụng trên một máy tính bảng hoặc trình giả lập máy tính bảng. Lưu ý cách mà vùng cuộn mở rộng để sử dụng toàn bộ màn hình và vẫn cuộn đúng cách.



Trong hình trên, các yếu tố sau đây xuất hiện:

1. Một liên kết web hoạt động được nhúng trong văn bản tự do.
2. Thanh cuộn xuất hiện khi cuộn văn bản.

Giải pháp nhiệm vụ 2

Mã XML cho bố cục với ScrollView như sau:

```

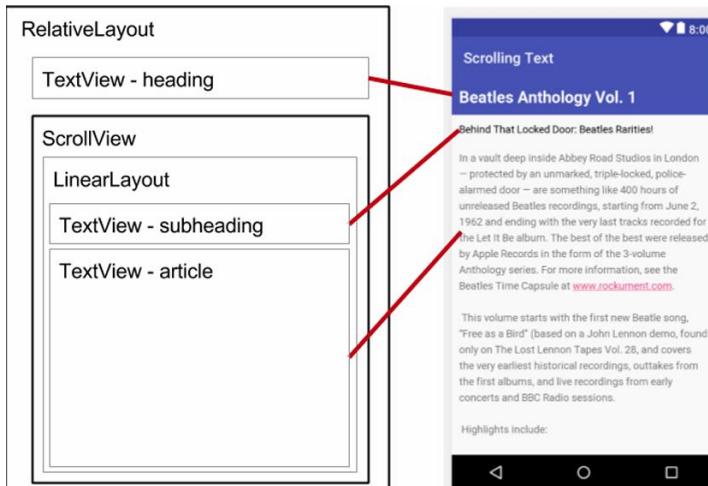
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.scrollingtext.MainActivity">
    <TextView
        android:id="@+id/article_heading"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/design_default_color_primary"
        android:textColor="@color/white"
        android:padding="10dp"
        android:textAppearance="@android:style/TextAppearance.DeviceDefault.Large"
        android:textStyle="bold"
        android:text="Beatles Anthology Vol. 1" />
    <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:textAppearance="@android:style/TextAppearance.DeviceDefault"
        android:text="Behind That Locked Door: Beatles Rarities" />
    <ScrollView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/article_subheading">
        <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="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginEnd="0dp"
                android:layout_marginBottom="0dp"
                android:autoLink="web"
                android:lineSpacingExtra="5sp"
                android:padding=""
                android:text="In a vault deep inside Abbey Road Studios in London – p..." />
        </ScrollView>
    </ScrollView>
</RelativeLayout>

```

Nhiệm vụ 3: Cuộn nhiều phần tử

Như đã đề cập trước đó, một ScrollView chỉ có thể chứa một View con duy nhất (chẳng hạn như TextView của bài viết mà bạn đã tạo). Tuy nhiên, View đó có thể là một ViewGroup khác chứa các phần tử View, chẳng hạn như LinearLayout. Bạn có thể nhúng một ViewGroup như LinearLayout bên trong ScrollView, từ đó cuộn tất cả các phần tử bên trong LinearLayout.

Ví dụ, nếu bạn muốn tiêu đề phụ của bài viết cuộn cùng với bài viết, hãy thêm một LinearLayout bên trong ScrollView và di chuyển tiêu đề phụ và bài viết vào trong LinearLayout. LinearLayout sẽ trở thành View con duy nhất trong ScrollView như được hiển thị trong hình dưới đây, và người dùng có thể cuộn toàn bộ LinearLayout: tiêu đề phụ và bài viết.



3.1 Thêm một LinearLayout vào ScrollView

1. Mở tệp activity_main.xml của dự án ScrollingText, và chọn tab Text để chỉnh sửa mã XML (nếu nó chưa được chọn).
2. Thêm một LinearLayout ở trên TextView của bài viết bên trong ScrollView. Khi bạn nhập <LinearLayout> Android Studio sẽ tự động thêm </LinearLayout> ở cuối và hiển thị các thuộc tính android:layout_width và android:layout_height với các gợi ý. Chọn match_parent cho chiều rộng và wrap_content cho chiều cao từ các gợi ý. Mã ở đầu ScrollView bây giờ trông như sau:

```
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="match_parent">
</LinearLayout>
```

Bạn sử dụng match_parent để khớp chiều rộng với ViewGroup cha. Bạn sử dụng wrap_content để thay đổi kích thước của LinearLayout sao cho nó chỉ đủ lớn để bao bọc nội dung của nó.

3. Di chuyển mã kết thúc </LinearLayout> sau TextView của bài viết nhưng trước thẻ đóng </ScrollView>. LinearLayout bây giờ bao gồm TextView của bài viết và hoàn toàn nằm bên trong ScrollView.

- Thêm thuộc tính `android:orientation=“vertical”` vào vào `LinearLayout` để đặt hướng của nó thành dọc.
- Chọn `Code > Reformat Code` để thuần le mã một cách chính xác.

Bố cục `LinearLayout` trông như sau:

```
<ScrollView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/article_subheading">
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <TextView
            android:id="@+id/article"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginEnd="0dp"
            android:layout_marginBottom="0dp"
            android:autoLink="web"
            android:lineSpacingExtra="5sp"
            android:padding=""
            android:text="In a vault deep inside Abbey Road Studios in London – p..." />
    </LinearLayout>
</ScrollView>
</RelativeLayout>
```

3.2 Di chuyển các phần tử UI bên trong `LinearLayout`

`LinearLayout` hiện chỉ có một phần tử UI—`TextView` của bài viết. Bạn muốn bao gồm `TextView` của tiêu đề phụ trong `LinearLayout` để cả hai đều có thể cuộn.

- Để di chuyển `TextView` của tiêu đề phụ, hãy chọn mã, chọn `Edit > Cut` nhấp vào vị trí phía trên `TextView` của bài viết bên trong `LinearLayout`, và chọn `Edit > Paste`
 - Xóa thuộc tính `android:layout_below = “@+id/article_heading”` khỏi `TextView` của tiêu đề phụ. Bởi vì `TextView` này hiện nằm trong `LinearLayout`, thuộc tính này sẽ gây xung đột với các thuộc tính của `LinearLayout`.
 - Thay đổi thuộc tính bố cục của `ScrollView` từ `android:layout_below="“@+id/article_subheading”"` Thành `android:layout_below="“@+id/article_heading”"`
- Bây giờ mà tiêu đề phụ là một phần của `LinearLayout`, `ScrollView` phải được đặt bên dưới tiêu đề, không phải bên dưới tiêu đề phụ.
- Mã XML cho `ScrollView` bây giờ trông như sau:

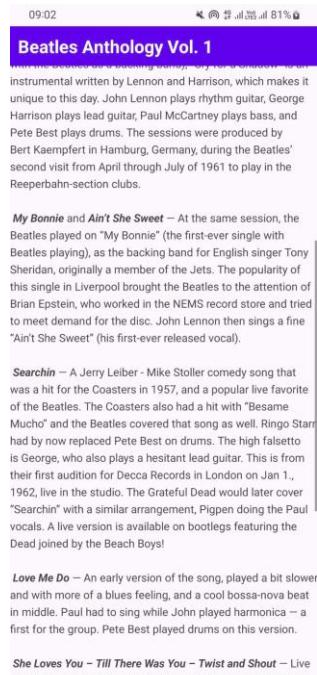
```

<ScrollView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/article_heading">
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <TextView
            android:id="@+id/article_subheading"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_below="@+id/article_heading"
            android:padding="16dp"
            android:textAppearance="@android:style/TextAppearance.DeviceDefault"
            android:text="Behind That Locked Door: Beatles Rarities" />
        <TextView
            android:id="@+id/article"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginEnd="0dp"
            android:layout_marginBottom="0dp"
            android:autoLink="web"
            android:lineSpacingExtra="5sp"
            android:padding="16dp"
            android:text="In a vault deep inside Abbey Road Studios in London - p..." />
    </LinearLayout>

```

4. Chạy ứng dụng

Kéo lên và xuống để cuộn bài viết, và lưu ý rằng tiêu đề phụ bây giờ cuộn cùng với bài viết trong khi tiêu đề chính vẫn giữ nguyên vị trí.



Giải pháp

Dự án Android Studio : Scrolling Text

Thử thách lập trình

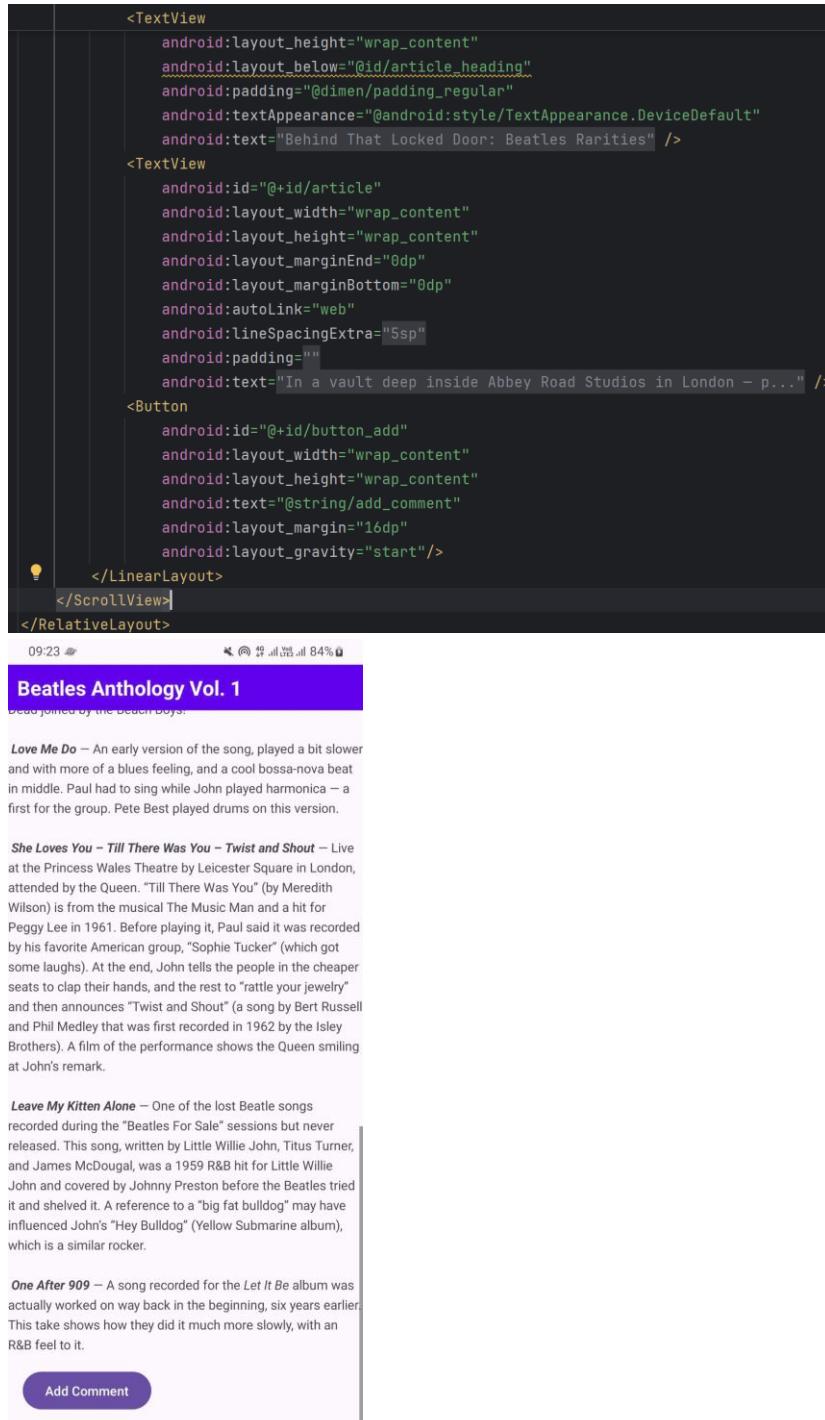
Lưu ý: Tất cả các thử thách lập trình là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách: Thêm một phần tử UI khác – một button – vào LinearLayout bên trong ScrollView để nó cuộn cùng với văn bản.

Hướng dẫn thực hiện:

1. Mở tệp activity_main.xml trong dự án Scrolling Text
2. Thêm một button vào bên trong LinearLayout, dưới TextView của bài viết.

```
<?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="com.example.scrollingtext.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:textColor="@color/white"
        android:padding="10dp"
        android:textAppearance="@android:style/TextAppearance.DeviceDefault.Large"
        android:textStyle="bold"
        android:text="Beatles Anthology Vol. 1" />
    <ScrollView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/article_heading">
        <LinearLayout
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:orientation="vertical">
            <TextView
                android:id="@+id/article_subheading"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
```



Mã giải pháp cho thử thách

Dự án Android Studio: ScrollingTextChallenge

Tóm tắt

- Sử dụng ScrollView để cuộn một View con duy nhất (chẳng hạn như TextView). Một ScrollView chỉ có thể chứa một View hoặc ViewGroup con.
- Sử dụng một ViewGroup như LinearLayout làm View con bên trong ScrollView để cuộn nhiều phần tử View. Bao bọc các phần tử bên trong LinearLayout
- Hiển thị văn bản tự do trong TextView với các thẻ định dạng HTML cho chữ đậm và chữ nghiêng.
- Sử dụng “\n” như một ký tự kết thúc dòng trong văn bản tự do để giữ cho một đoạn văn không bị dồn vào đoạn văn tiếp theo.
- Sử dụng thuộc tính android:autoLink="web" để làm cho các liên kết web trong văn bản có thể nhấp được.

Các khái niệm liên quan

Tài liệu về các khái niệm liên quan có tổng 1.3 Văn bản và các chế độ cuộn

Tìm hiểu thêm

Tài liệu Android Studio

- Trang tải xuống Android Studio
- Gặp gỡ Android Studio

Tài liệu dành cho nhà phát triển Android

- ScrollView
- LinearLayout
- RelativeLayout
- View
- Button
- TextView
- String resources
- Relative layout

Khác

- Blog dành cho nhà phát triển Android: Linkify your Text
- Đường dẫn mã: Working with a TextView

Bài tập về nhà

Thay đổi ứng dụng

Mở ứng dụng ScrollingText2 mà bạn đã tạo trong bài học Làm việc với các phần tử TextView.

1. Thay đổi tiêu đề phụ sao cho nó được bọc trong một cột bên trái rộng 100 dp, như hình dưới đây.
2. Đặt văn bản của bài viết ở bên phải tiêu đề phụ như hình dưới đây.

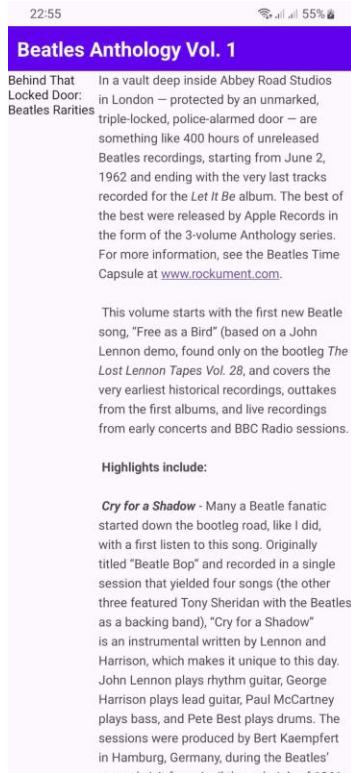
The screenshot shows an Android XML layout file in a code editor. The code defines a relative layout containing a scroll view and a linear layout. The scroll view contains a text view for the main heading and a linear layout for the subtitle. The linear layout has a horizontal orientation and contains another linear layout for the subheading and a final text view for the article content. A specific section of the code is highlighted with a yellow background, showing the modification of the inner linear layout's width to "100dp".

```
<?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="com.example.scrollingtext.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:textColor="@color/white"
        android:padding="10dp"
        android:textAppearance="@android:style/TextAppearance.DeviceDefault.Large"
        android:textStyle="bold"
        android:text="Beatles Anthology Vol. 1" />
    <ScrollView
        android:id="@+id/scrollView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/article_heading">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal">
            <LinearLayout
                android:layout_width="100dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:orientation="vertical">
                <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:textAppearance="@android:style/TextAppearance.DeviceDefault"
                    android:text="@string/article_subtitle" />
            </LinearLayout>
            <LinearLayout
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:orientation="vertical">
                <TextView
                    android:id="@+id/article"
                    android:layout_width="match_parent"
                    android:layout_height="wrap_content"
                    android:layout_marginEnd="0dp"
                    android:layout_marginBottom="0dp"
                    android:layout_weight="1"
                    android:autoLink="web"
                    android:lineSpacingExtra="5sp" />
            </LinearLayout>
        </LinearLayout>
    </ScrollView>
</RelativeLayout>
```

```

    android:orientation="vertical">
<TextView
    android:id="@+id/article"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginEnd="0dp"
    android:layout_marginBottom="0dp"
    android:layout_weight="1"
    android:autoLink="web"
    android:lineSpacingExtra="5sp"
    android:padding="16dp"
    android:text="In a vault deep inside Abbey Road Studios in London - p..." />

    <Button
        android:id="@+id/button_add"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="16dp"
        android:text="Add Comment" />
</LinearLayout>
</LinearLayout>
</ScrollView>
</RelativeLayout>
```



Trả lời câu hỏi

Câu 1: Bạn có thể sử dụng bao nhiêu view trong một ScrollView? Chọn một:

- A. Một view duy nhất
- B. Một view hoặc một nhóm view
- C. Bao nhiêu tùy thích

Câu 2: Thuộc tính XML nào bạn sử dụng trong LinearLayout để hiển thị các view bên cạnh nhau? Chọn một

- A. android:orientation="horizontal"
- B. android:orientation="vertical"
- C. android:layout_width="wrap_content"

Câu 3: Thuộc tính XML nào bạn sử dụng để định nghĩa chiều rộng của LinearLayout bên trong ScrollView? Chọn một:

- A. android:layout_width="wrap_content"
- B. android:layout_width="match_parent"
- C. android:layout_width="200dp"

Nộp ứng dụng để chấm điểm

Hướng dẫn cho người chấm điểm:

Kiểm tra rằng ứng dụng có các tính năng sau:

- Bố cục hiển thị tiêu đề phụ ở cột bên trái và văn bản bài viết ở cột bên phải, như được hiển thị trong hình trên.
- ScrollView bao gồm một LinearLayout với hai phần tử TextView
- Hướng của LinearLayout được đặt ngang

1.5) Tài nguyên có sẵn

Giới thiệu

Những gì bạn nên biết

Bạn nên có khả năng:

- Hiểu quy trình làm việc cơ bản của Android Studio.
- Tạo một ứng dụng từ đầu bằng cách sử dụng mẫu Empty Activity
- Sử dụng trình chỉnh sửa bố cục.

Những gì bạn sẽ học

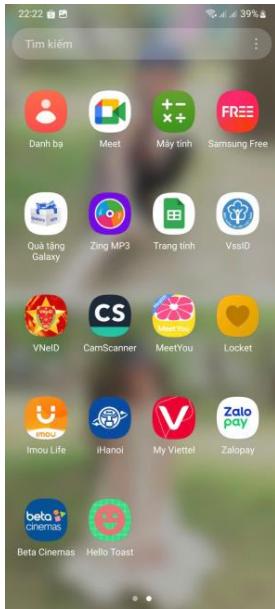
- Nơi tìm thông tin và tài nguyên dành cho nhà phát triển.
- Cách thêm biểu tượng khởi động cho ứng dụng của bạn.
- Cách tìm kiếm sự trợ giúp khi bạn đang phát triển ứng dụng Android.

Những gì bạn sẽ làm

- Khám phá một số tài nguyên có sẵn cho các nhà phát triển Android ở mọi cấp độ.
- Thêm biểu tượng khởi động cho ứng dụng của bạn.

Tổng quan về ứng dụng

Bạn sẽ thêm biểu tượng khởi động cho ứng dụng HelloToast mà bạn đã tạo trước đó hoặc cho một ứng dụng mới.



Nhiệm vụ 1: Thay đổi biểu tượng khởi động

Mỗi ứng dụng mới mà bạn tạo bằng Android Studio bắt đầu với một biểu tượng khởi động tiêu chuẩn đại diện cho ứng dụng. Biểu tượng khởi động xuất hiện trong danh sách trên Google Play. Khi người dùng tìm kiếm trong cửa hàng Google Play, biểu tượng của ứng dụng bạn sẽ xuất hiện trong kết quả tìm kiếm.

Khi người dùng đã cài đặt ứng dụng, biểu tượng khởi động sẽ xuất hiện trên thiết bị ở nhiều vị trí khác nhau, bao gồm màn hình chính và màn hình Tìm kiếm Ứng dụng. Ví dụ, ứng dụng HelloToast xuất hiện trong màn hình Tìm kiếm Ứng dụng của trình giả lập với biểu tượng tiêu chuẩn cho các dự án ứng dụng mới, như được hiển thị bên dưới.



Việc thay đổi biểu tượng khởi động là một quy trình đơn giản theo từng bước, giúp bạn làm quen với các tính năng hình ảnh của Android Studio. Trong nhiệm vụ này, bạn cũng sẽ tìm hiểu thêm về cách truy cập tài liệu chính thức của Android.

1.1 khám phá tài liệu chính thức của Android

Bạn có thể tìm thấy tài liệu dành cho nhà phát triển Android chính thức tại developer.android.com.

Tài liệu này chứa một lượng lớn thông tin và được Google cập nhật liên tục.

1. Truy cập vào developer.android.com/design/.

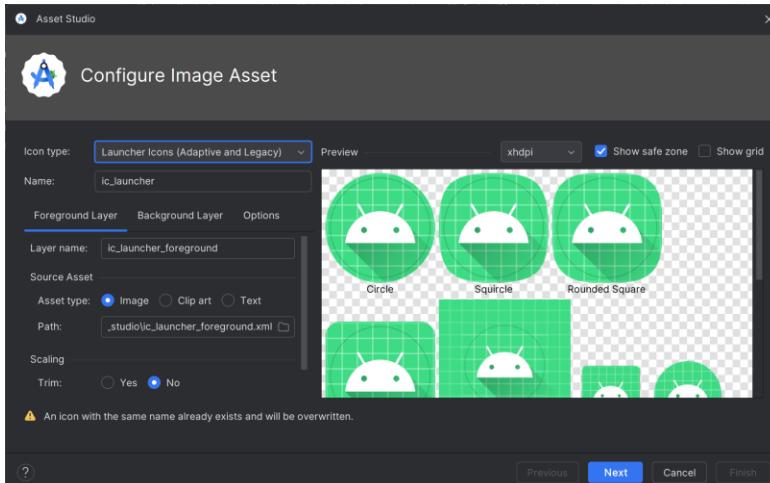
Mục này nói về Material Design, một triết lý thiết kế khái niệm định nghĩa cách các ứng dụng nên trông như thế nào và hoạt động ra sao trên các thiết bị di động. Điều hướng các liên kết để tìm hiểu thêm về Material Design. Ví dụ, hãy truy cập vào phần Style để tìm hiểu thêm về việc sử dụng màu sắc và các chủ đề khác.

2. Truy cập vào developer.android.com/docs/ để tìm thông tin về API, tài liệu tham khảo, hướng dẫn, công cụ và mẫu mã.
3. Truy cập vào developer.android.com/distribute/ để tìm thông tin về cách đưa ứng dụng lên Google Play, hệ thống phân phối kỹ thuật số của Google cho các ứng dụng phát triển với Android SDK. Sử dụng Google Play Console để phát triển người dùng và bắt đầu kiếm tiền.

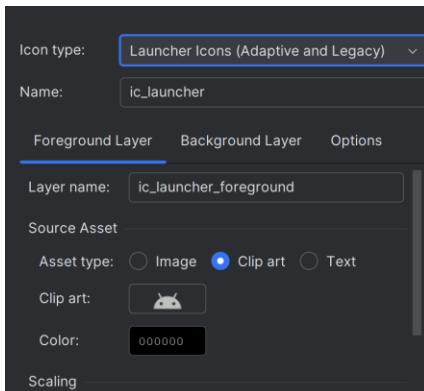
1.2 Thêm tài nguyên hình ảnh cho biểu tượng ứng dụng

Để thêm hình ảnh clip-art làm biểu tượng ứng dụng, hãy làm theo các bước sau:

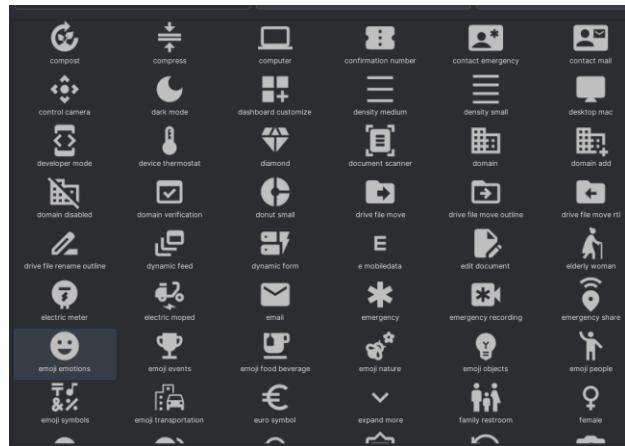
1. Mở dự án ứng dụng Hello Toast từ bài học trước về việc sử dụng trình chỉnh sửa giao diện, hoặc tạo một dự án ứng dụng mới.
2. Trong bảng Project > Android, nhấp chuột phải (hoặc Control-click) vào thư mục res và chọn New > Image Asset. Cửa sổ Configure Image Asset sẽ xuất hiện.



3. Trong trường Icon Type, chọn Launcher Icon (Adaptive & Legacy) nếu nó chưa được chọn.
4. Nhấp vào tab Foreground Layer, chọn Clip Art cho Asset Type



5. Nhấp vào biểu tượng trong trường Clip Art. Các biểu tượng sẽ xuất hiện từ bộ sưu tập biểu tượng Material Design.
6. Duyệt qua cửa sổ Select Icon, chọn biểu tượng phù hợp (chẳng hạn như biểu tượng tâm trạng để gợi ý một tâm trạng tốt), sau đó nhấp OK.



7. Nhấp vào tab Background Layer, chọn Color làm Asset Type, sau đó nhấp vào ô màu để chọn màu làm lớp nền.
8. Nhấp vào tab Legacy à kiểm tra các cài đặt mặc định. Xác nhận rằng bạn muốn tạo các biểu tượng legacy, tròn và biểu tượng cho Google Play Store. Nhấp Next khi hoàn tất.
9. Chạy ứng dụng



Mẹo: Xem Launcher Icons để tìm hiểu thêm về cách thiết kế biểu tượng ứng dụng hiệu quả

Nhiệm vụ 2: Sử dụng mẫu dự án

Android Studio cung cấp các mẫu cho các thiết kế ứng dụng và hoạt động phổ biến và được khuyến nghị. Sử dụng các mẫu có sẵn giúp tiết kiệm thời gian và giúp bạn tuân theo các phương pháp thiết kế tốt nhất.

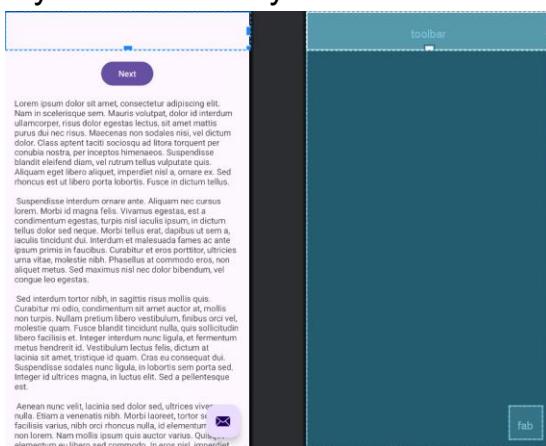
Mỗi mẫu đều bao gồm một hoạt động và giao diện người dùng cơ bản. Bạn đã sử dụng mẫu Empty Activity. Mẫu Basic Activity có nhiều tính năng hơn và bao gồm các tính năng ứng dụng được khuyến nghị, chẳng hạn như menu tùy chọn xuất hiện trên thanh công cụ ứng dụng.

2.1 Khám phá kiến trúc của Basic Activity

Mẫu Basic Activity là một mẫu linh hoạt được cung cấp bởi Android Studio để giúp bạn bắt đầu phát triển ứng dụng.

1. Trong Android Studio, tạo một dự án mới với mẫu Basic Activity
2. Xây dựng và chạy ứng dụng
3. Xác định các phần được ghi nhãn trong hình ảnh và bảng dưới đây. Tìm các phần tương ứng trên màn hình thiết bị hoặc trình giả lập của bạn.
Kiểm tra mã nguồn Java và các tệp XML tương ứng được mô tả trong bảng.

Việc làm quen với mã nguồn Java và các tệp XML sẽ giúp bạn mở rộng và tùy chỉnh mẫu này theo nhu cầu của mình.



2.2 Tùy chỉnh ứng dụng được tạo bởi mẫu

Thay đổi giao diện của ứng dụng được tạo bởi mẫu Basic Activity. Ví dụ, bạn có thể thay đổi màu sắc của thanh công cụ (app bar) để phù hợp với trạng thái (màu trên một số thiết bị có thể là màu tối hơn của cùng một màu chủ đạo). Bạn cũng có thể muốn xóa nút hành động nổi (floating action button) nếu không có kế hoạch sử dụng nó.

1. Thay đổi màu sắc của thanh công cụ (Toolbar) trong activity_main.xml bằng cách thay đổi thuộc tính android:background thành ?attr/colorPrimaryDark, điều này sẽ thiết lập màu của thanh công cụ thành màu chủ đạo tối hơn, phù hợp với màu của thanh trạng thái.

```
        android:background="?attr/colorPrimaryDark"
```

2. Để xóa nút hành động nổi , bắt đầu bằng cách xóa mã mẫu trong phương thức `onCreate()` thiết lập trình nghe sự kiện `onClick()` cho nút. Mở `MainActivity` và xóa đoạn mã sau:

```
binding.fab.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH_LONG)
            .setAnchorView(R.id.fab)
            .setAction("Action", listener: null).show();
    }
})
```

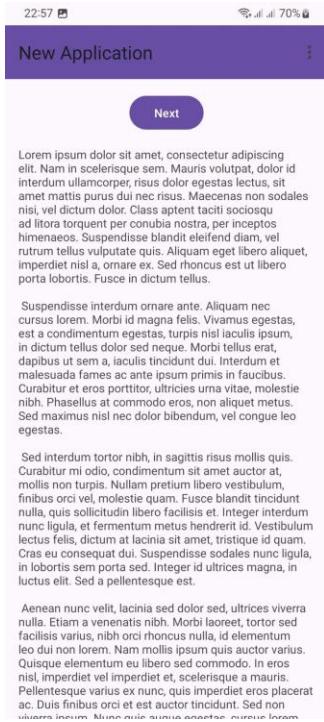
3. Để xóa nút hành động nổi (floating action button) khỏi giao diện, hãy xóa đoạn mã XML sau trong `activity_main.xml`:

```
<com.google.android.material.floatingactionbutton.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|end"
    android:layout_marginEnd="@dimen/fab_margin"
    android:layout_marginBottom="16dp"
    app:srcCompat="@android:drawable/ic_dialog_email" />
```

4. Đổi tên ứng dụng hiển thị trên thanh công cụ (app bar) bằng cách thay đổi tài nguyên chuỗi `app_name` trong `strings.xml` thành:

```
<string name="app_name">New Application</string>
```

5. Chạy ứng dụng. Nút hành động nổi không còn xuất hiện, tên ứng dụng đã thay đổi, và màu nền của thanh công cụ cũng đã được thay đổi.



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) Bộ 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**