

CHƯƠNG 1: LÀM QUEN

Bài 1.1: Android Studio and Hello Word

Bài học trong chương này

Bài 1: Xây dựng ứng dụng đầu tiên của bạn

- 1.1: Android Studio và Hello World
- 1.2A: Giao diện người dùng tương tác đầu tiên của bạn
- 1.2B: Trình chỉnh sửa bố cục
- 1.3: Chế độ xem văn bản và cuộn
- 1.4: Tài nguyên có sẵn

Bài học 2: Hoạt động

- 2.1: Hoạt động và ý định
- 2.2: Vòng đời và trạng thái hoạt động
- 2.3: Ý định ngầm

Bài 3: Kiểm tra, 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

*Giới thiệu

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

*Những điều bạn nên biết

Bạn sẽ có thể:

- Hiểu quy trình phát triển phần mềm chung cho các ứng dụng hướng đối tượng sử dụng 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 số trong số đó tập trung vào ngôn ngữ lập trình Java. (Những thực tế này sẽ không giải thích lập trình hướng đối tượng hoặc ngôn ngữ Java.)

*Những gì bạn cần

- Máy tính chạy Windows hoặc Linux hoặc máy Mac chạy macOS. Xem Android Studio

Tải xuống trang để cập nhật các yêu cầu hệ thống.

- Truy cập Internet hoặc một cách thay thế để tải Android Studio và Java mới nhất cài đặt vào máy tính của bạn.

***Những gì bạn sẽ học**

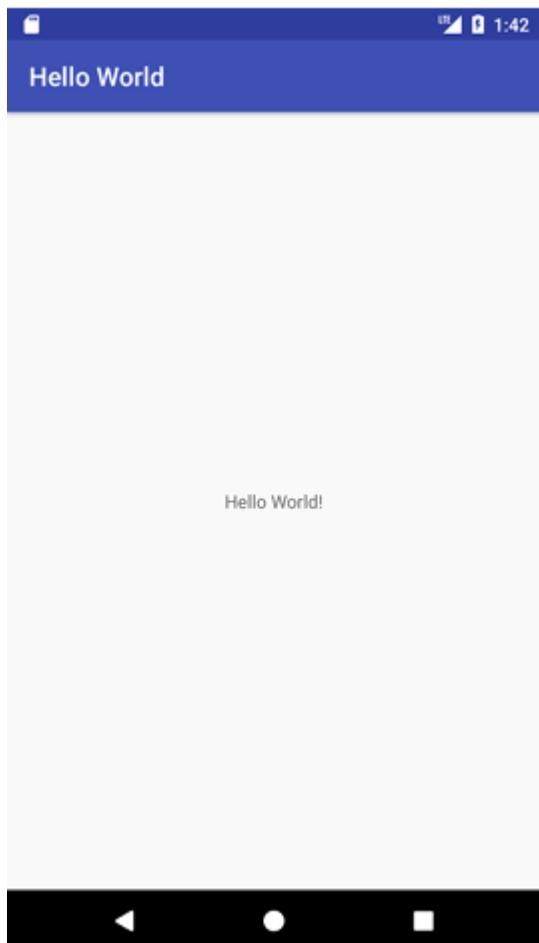
- Cách cài đặt và sử dụng Android Studio IDE.
- Cách sử dụng quy trình phát triển để xây dựng ứng dụng Android.
- Cách tạo dự án Android từ mẫu.
- Cách thêm thông báo nhật ký vào ứng dụng của bạn cho mục đích gỡ lỗi.

***Những gì bạn sẽ làm**

- Cài đặt môi trường phát triển Android Studio.
- Tạo trình giả lập (thiết bị ảo) để chạy ứng dụng trên máy tính.
- Tạo và chạy ứng dụng Hello World trên các thiết bị ảo và vật lý.
- Khám phá bố cục dự án.
- Tạo và xem tin nhắn nhật ký từ ứng dụng của bạn.
- Khám phá tệp AndroidManifest.xml.

***Tổng quan về ứng dụng**

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



Nhiệm vụ 1: Cài đặt Android Studio

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

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

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

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

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

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

Nhiệm vụ 2: Tạo ứng dụng Hello World

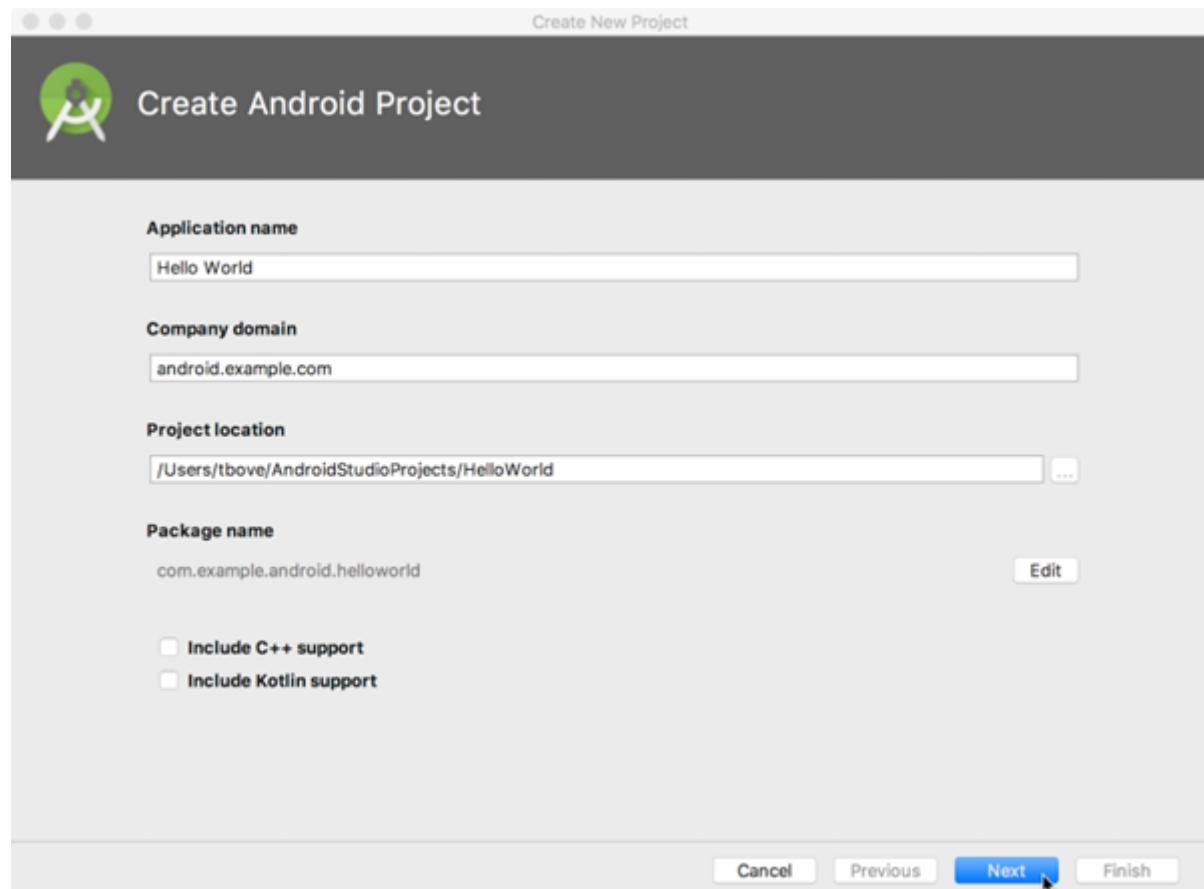
Trong tác vụ này, bạn sẽ tạo một ứng dụng hiển thị "Hello World" để xác minh rằng Android studio đã được cài đặt chính xác và tìm hiểu những kiến thức cơ bản về phát triển bằng Android Studio.

2.1 Tạo dự án ứng dụng

1. Mở Android Studio nếu nó chưa được mở.

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

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



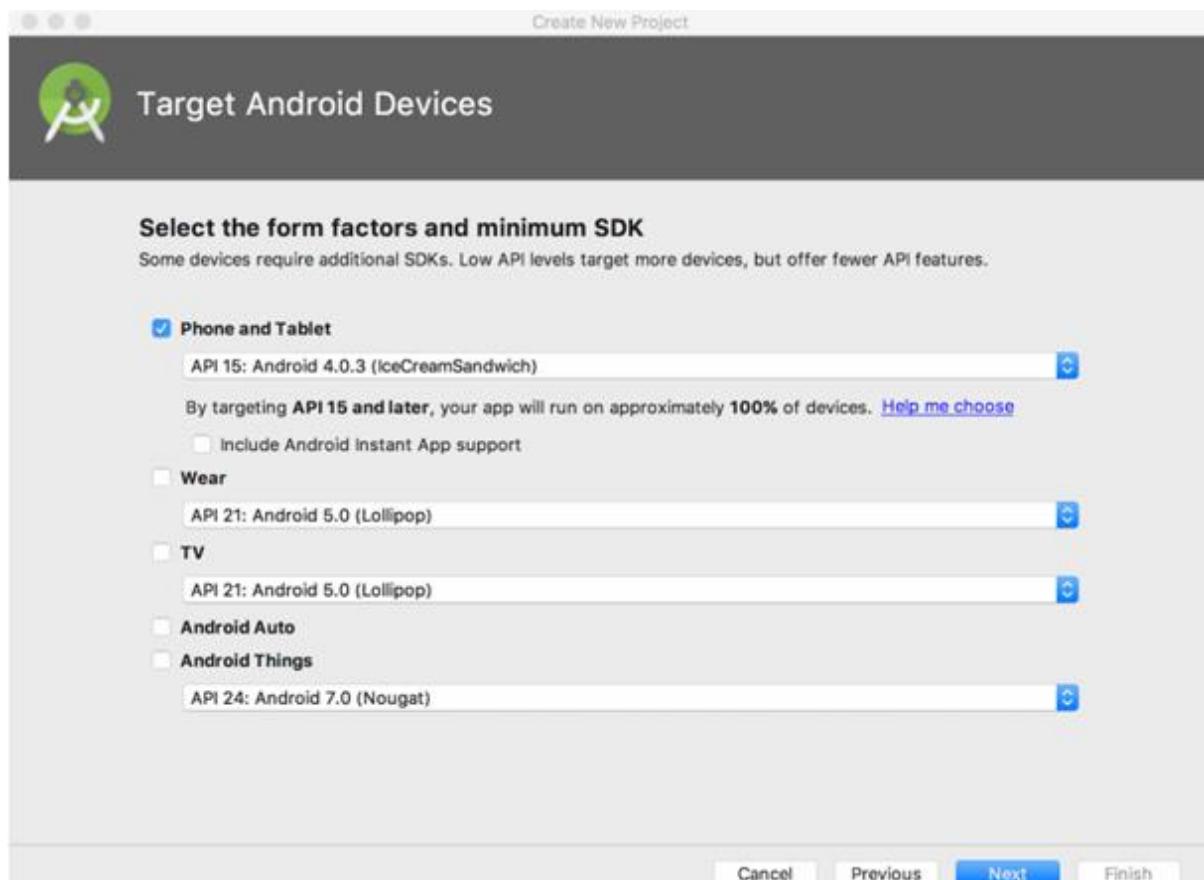
4. Xác minh rằng vị trí Dự án mặc định là nơi bạn muốn lưu trữ ứng dụng Hello World và các dự án Android Studio khác hoặc thay đổi vị trí đó thành thư mục ưa thích của bạn.

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

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

7. Trên màn hình Thiết bị Android mục tiêu, Điện thoại và Máy tính bảng sẽ 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 thiết lập.

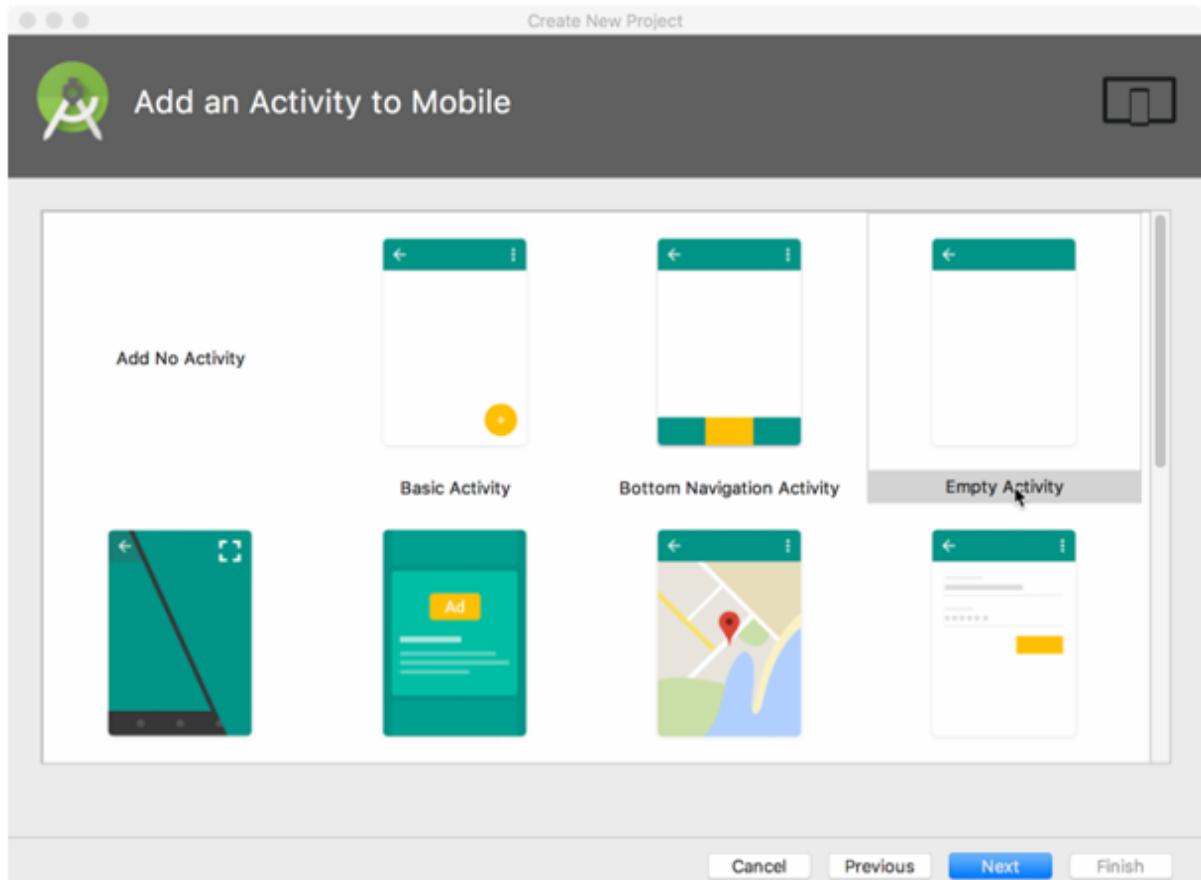


Đây là các cài đặt được sử dụng bởi các ví dụ trong các bài học cho khóa học này. Khi viết bài này, các cài đặt này làm cho ứng dụng Hello World của bạn tương thích với 97% thiết bị Android đang hoạt động trên Cửa hàng Google Play.

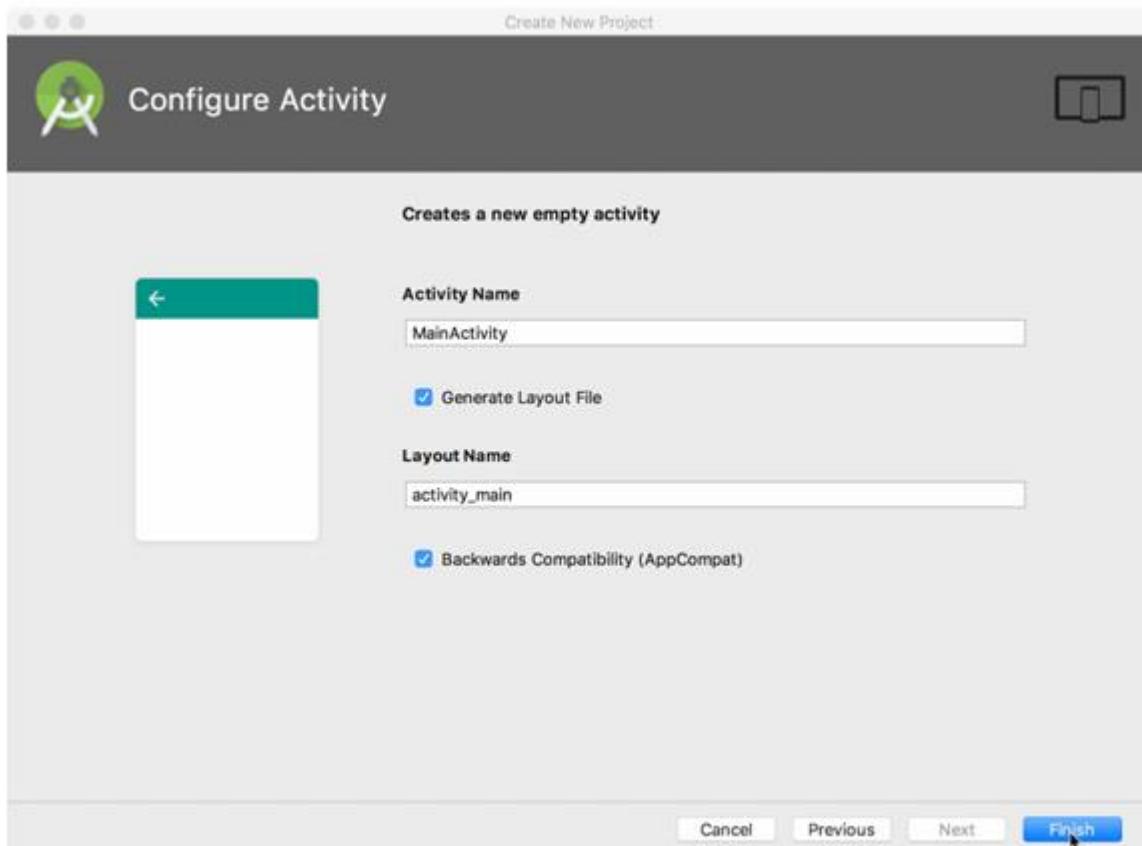
8. Bỏ chọn Bao gồm hỗ trợ ứng dụng tức thì và tất cả các tùy chọn khác. Sau đó nhấp vào Tiếp theo . Nếu dự án của bạn yêu cầu các thành phần bổ sung cho SDK mục tiêu bạn đã chọn, Android Studio sẽ tự động cài đặt các thành phần đó.

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. 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 Hoạt động trống như hình dưới đây và nhấp vào Tiếp theo



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



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

12. Đảm bảo rằng tùy chọn 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.

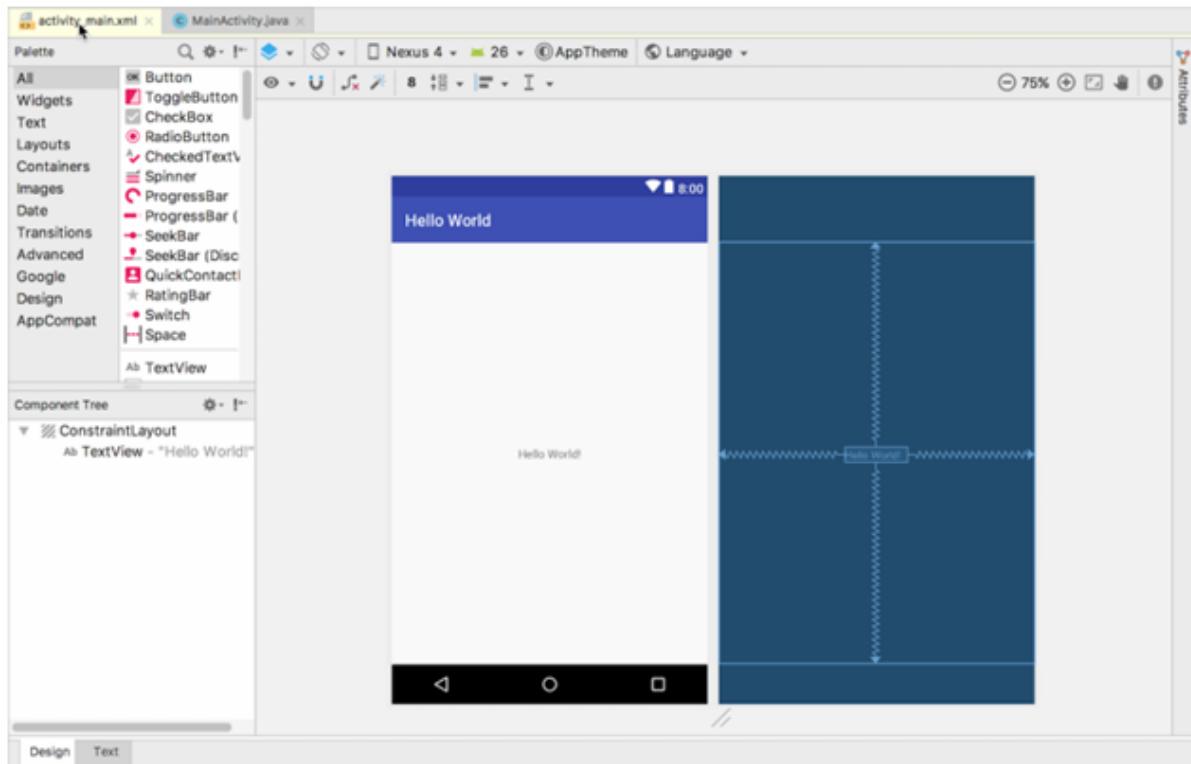
13. Nhấp vào Kết thúc. Android Studio sẽ tạo một thư mục cho các dự án của bạn và xây dựng dự án bằng Gradle (quá trình này có thể mất vài phút).

*Mẹo: Xem trang Định 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 thư. Trình chỉnh sửa Android Studio sẽ xuất hiện.

*Làm theo các bước sau:

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

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



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

The screenshot shows the Android Studio interface with the code editor open. The tab bar at the top has 'activity_main.xml' and 'MainActivity.java'. The code editor displays the following Java code:

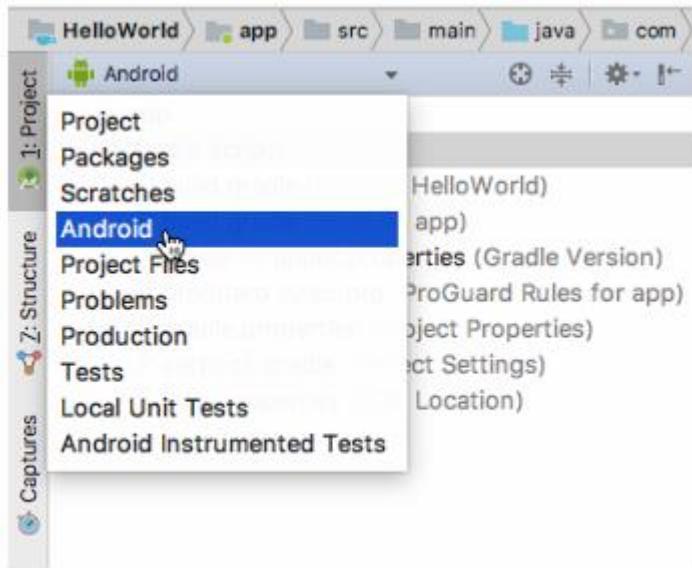
```
1 package com.example.android.helloworld;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_main);
11    }
12
13 }
14
```

The code is color-coded: 'package', 'import', 'public', 'class', 'extends', '@Override', 'protected', 'void', 'super', 'onCreate', 'Bundle', 'setContentView', and 'R.layout.activity_main' are in blue; 'Activity' and 'AppCompatActivity' are in orange; and 'activity_main' is in purple.

2.2 Khám phá ngăn Project > Android

Trong thực tế 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 Dự án xuất hiện.
2. Để xem dự án trong hệ thống phân cấp dự án Android tiêu chuẩn, hãy chọn Android từ menu bật lên ở đầu ngăn Dự án, như hình dưới đây.

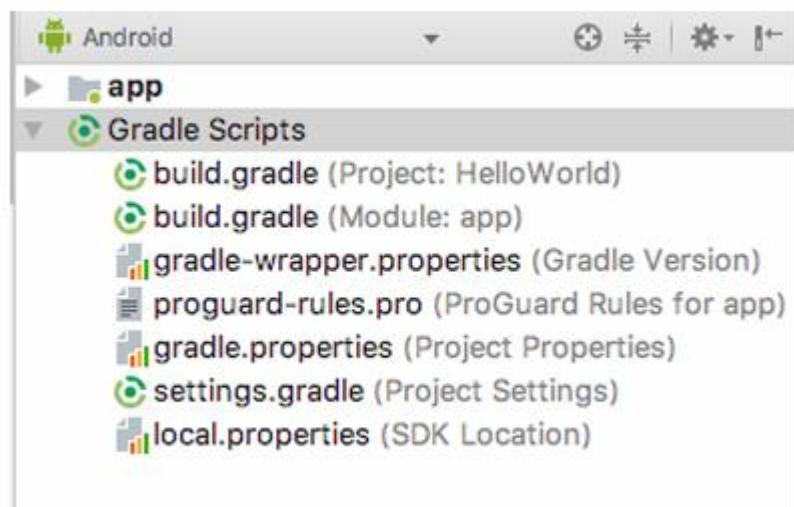


Lưu ý: Chương này và các chương khác đề cập đến ngăn Dự án, khi được đặt thành Android , là ngăn Dự án > Android.

2.3 Khám phá thư mục Tập lệnh Gradle

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

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



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

1. Nếu thư mục Tập lệnh Gradle 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 bản dựng Gradle cấp cao nhất. Hầu hết thời gian, bạn sẽ không cần thực hiện bất kỳ thay đổi nào đối với tệp này, nhưng vẫn hữu ích khi hiểu nội dung của nó. Theo mặc định, tệp bản dựng cấp cao nhất sử dụng khối buildscript để xác định kho lưu trữ Gradle và các phần phụ thuộc chung cho tất cả các mô-đun trong dự án. Khi phần phụ thuộc của bạn không phải là thư viện cục bộ hoặc cây tệp, Gradle sẽ tìm kiếm các tệp trong bất kỳ kho lưu trữ trực tuyến nào được chỉ định trong khối kho lưu trữ của tệp này. Theo mặc định, các dự án Android Studio mới khai báo JCenter và Google (bao gồm kho lưu trữ Google Maven) là vị trí kho lưu trữ:

```
allprojects {  
    repositories {  
        google()  
        jcenter()  
    }  
}
```

3. Tìm tệp build.gradle(Module:app). Ngoài tệp build.gradle cấp dự án, mỗi mô-đun có một tệp build.gradle riêng cho phép bạn định cấu hình cài đặt bản dựng cho từng mô-đun cụ thể (ứng dụng HelloWorld chỉ có một mô-đun). Việc định cấu hình các tùy chọn cài đặt bản dựng này cho phép bạn cung cấp các tùy chọn đóng gói tùy chỉnh, chẳng hạn như các loại bản dựng bổ sung và hương vị sản phẩm. Bạn cũng có thể ghi đè các tùy chọn cài đặt trong tệp AndroidManifest.xml hoặc tệp build.gradle cấp cao nhất. Tệp này thường là tệp cần chỉnh sửa khi thay đổi cấu hình cấp ứng dụng, chẳng hạn như khai báo các phần phụ thuộc trong phần phần phụ thuộc. Bạn có thể khai báo phần phụ thuộc thư viện bằng cách sử dụng một trong một số cấu hình phần phụ thuộc khác nhau. Mỗi cấu hình phần phụ thuộc cung cấp cho Gradle các hướng dẫn khác nhau về cách sử

dụng thư viện. Ví dụ: câu lệnh thực hiện fileTree(dir: 'libs', include: ['*.jar']) thêm phần phụ thuộc của tất cả các tệp ".jar" bên trong thư mục libs. Sau đây là tệp build.gradle(Module:app) cho ứng dụng HelloWorld:

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 26
    defaultConfig {
        applicationId "com.example.android.helloworld"
        minSdkVersion 15
        targetSdkVersion 26
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner
            "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles
                getDefaultProguardFile('proguard-android.txt'),
                'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:26.1.0'
    implementation
        'com.android.support.constraint:constraint-layout:1.0.2'
    testImplementation 'junit:junit:4.12'

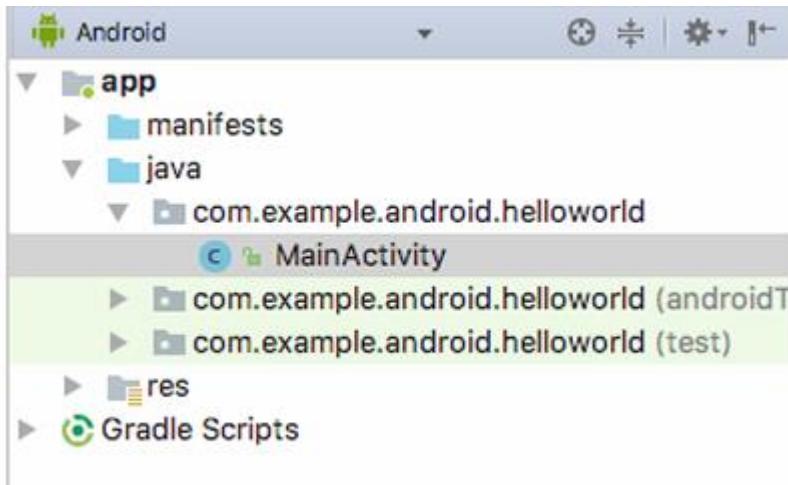
    androidTestImplementation 'com.android.support.test:runner:1.0.1'
    androidTestImplementation
        'com.android.support.test.espresso:espresso-core:3.0.1'
}
```

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

2.4 Khám phá ứng dụng và thư mục res

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

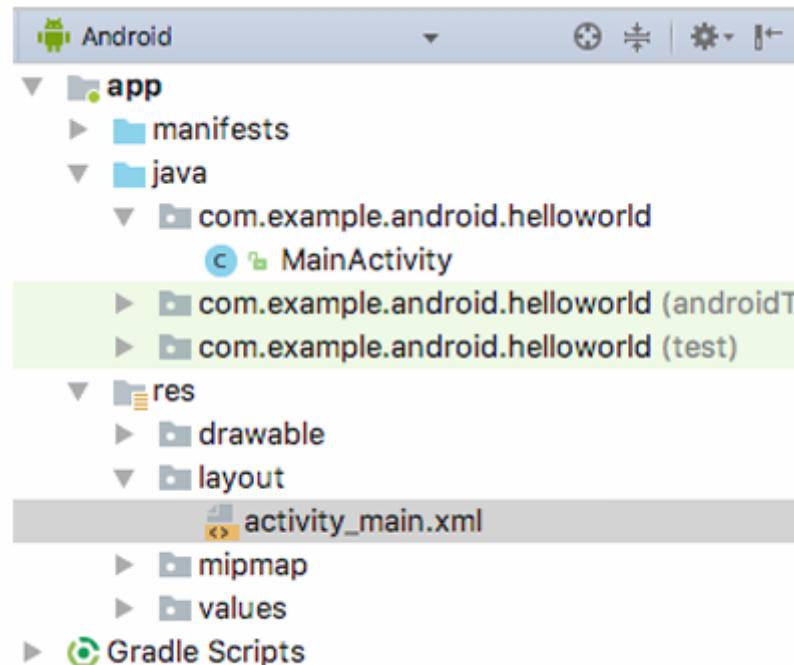
1. Mở rộng thư mục ứng dụng, thư mục java và thư mục com.example.android.helloworld để xem tệp java MainActivity. Nhấp đúp vào tệp sẽ trình soạn thảo mã. Ạn thảo mã. Ở tệp



đó trong

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

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



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

2.5 Khám phá thư mục kê khai

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

1. Mở rộng thư mục tệp kê khai.
2. Mở tệp AndroidManifest.xml.

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

Nhiệm vụ 3: Sử dụng thiết bị ảo (trình mô phỏng)

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

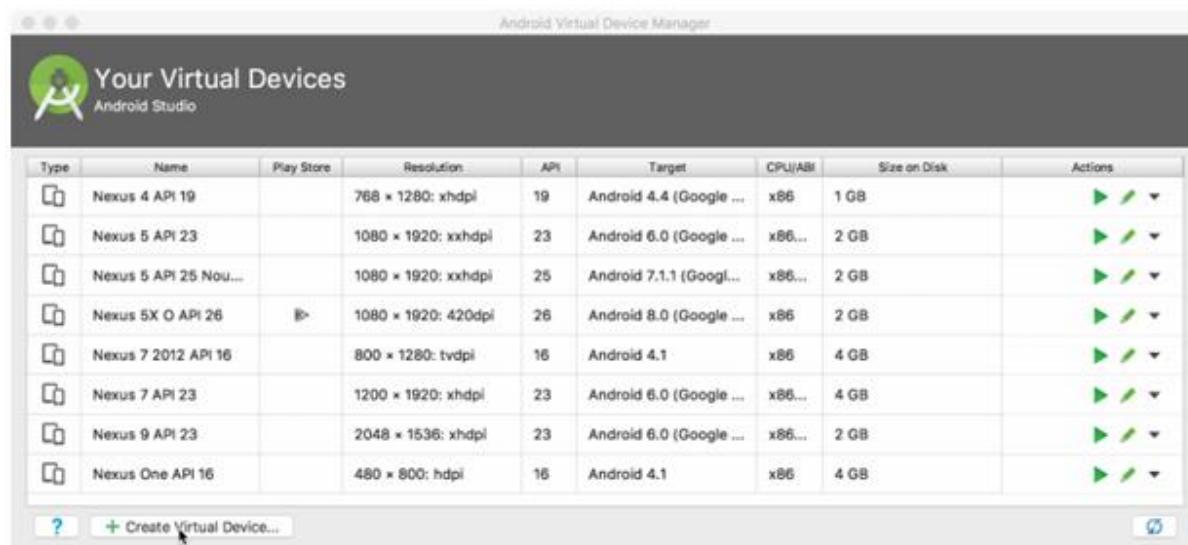
Khi sử dụng Trình quản lý AVD, bạn xác định các đặc điểm phần cứng của thiết bị, cấp độ API, bộ nhớ, giao diện và các thuộc tính khác và lưu thiết bị đó dưới dạng thiết bị ảo. Với thiết bị ảo, bạn có thể kiểm tra ứng dụng trên các cấu hình thiết bị khác nhau (chẳng hạn như máy tính bảng và điện thoại)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

3.1 Tạo thiết bị ảo Android (AVD)

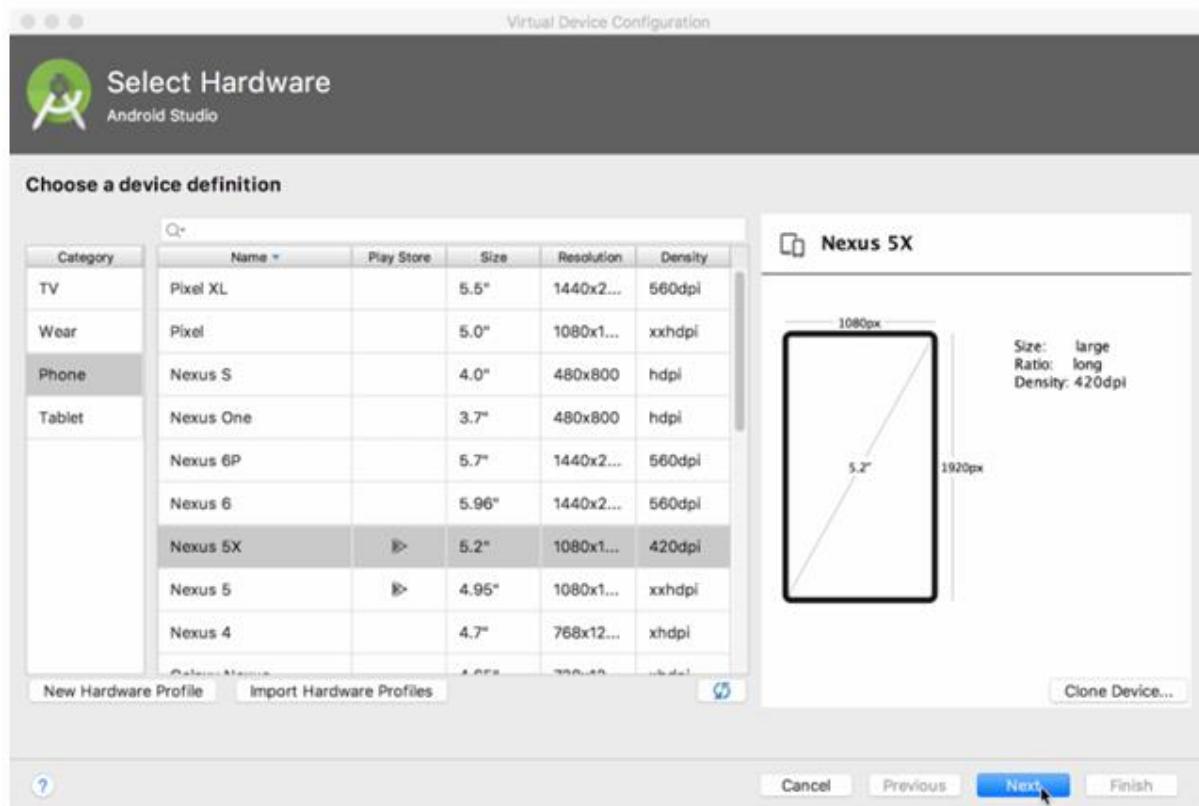
Để chạy trình mô phỏng trên máy tính, bạn phải tạo một cấu hình mô tả thiết bị ảo.

1. Trong Android Studio, chọn Công cụ > Android > Trình quản lý AVD,

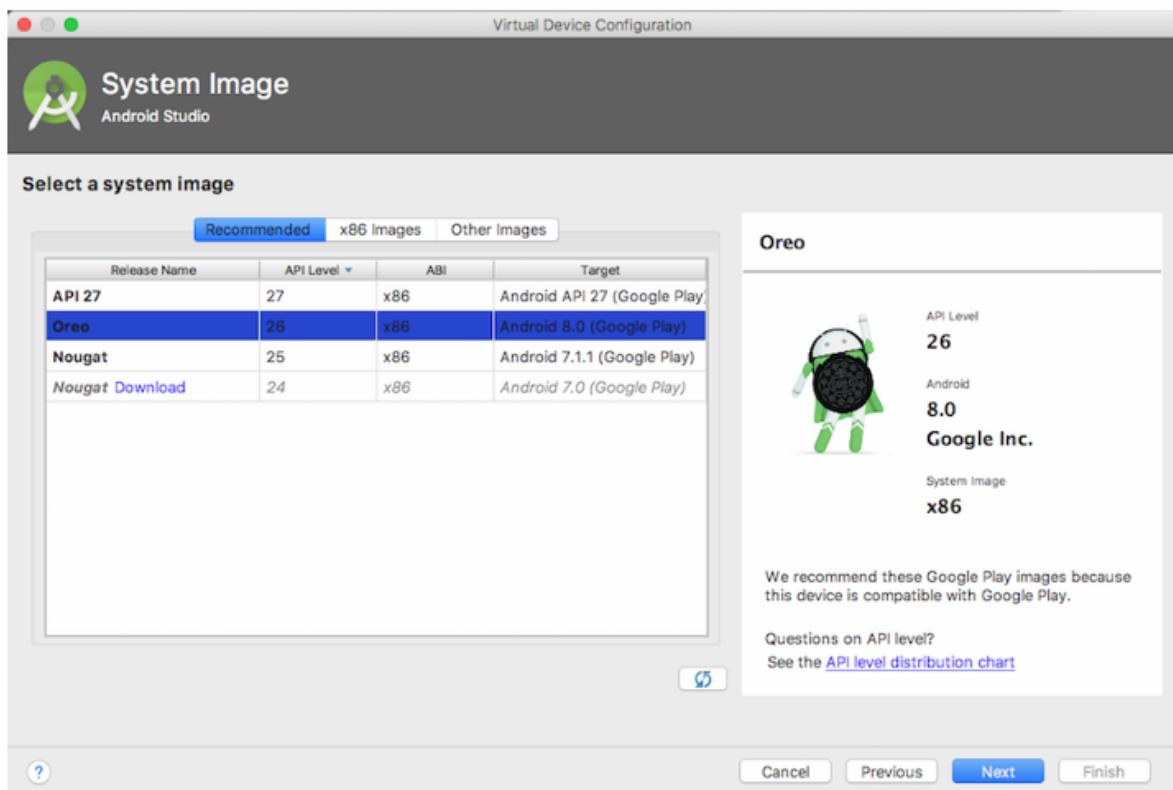
hoặc nhấp vào biểu tượng Trình quản lý AVD  trên thanh công cụ. Màn hình thiết bị ảo của bạn xuất hiện. Nếu bạn đã tạo thiết bị ảo, màn hình sẽ hiển thị chúng (như trong hình bên dưới); nếu không, bạn sẽ thấy một danh sách trống.



2. Nhấp vào +Tạo thiết bị ảo. Cửa sổ Chọn phần cứng xuất hiện hiển thị danh sách các thiết bị phần cứng được định cấu hình sẵn. Đối với mỗi thiết bị, bảng cung cấp một cột cho kích thước hiển thị đường chéo (Kích thước), độ phân giải màn hình tính bằng pixel (Độ phân giải) và mật độ điểm ảnh (Mật độ).



3. Chọn một thiết bị như Nexus 5x hoặc Pixel XL và nhấp vào Tiếp theo .
Màn hình Hình ảnh hệ thống xuất hiện.
4. Nhấp vào tab Đề xuất nếu nó chưa được chọn và chọn phiên bản hệ
thống Android để chạy trên thiết bị ảo (chẳng hạn như Oreo).



Có nhiều phiên bản hơn được hiển thị trong tab Đề xuất. Nhìn vào tab Hình ảnh x86 và Hình ảnh khác để xem chúng.

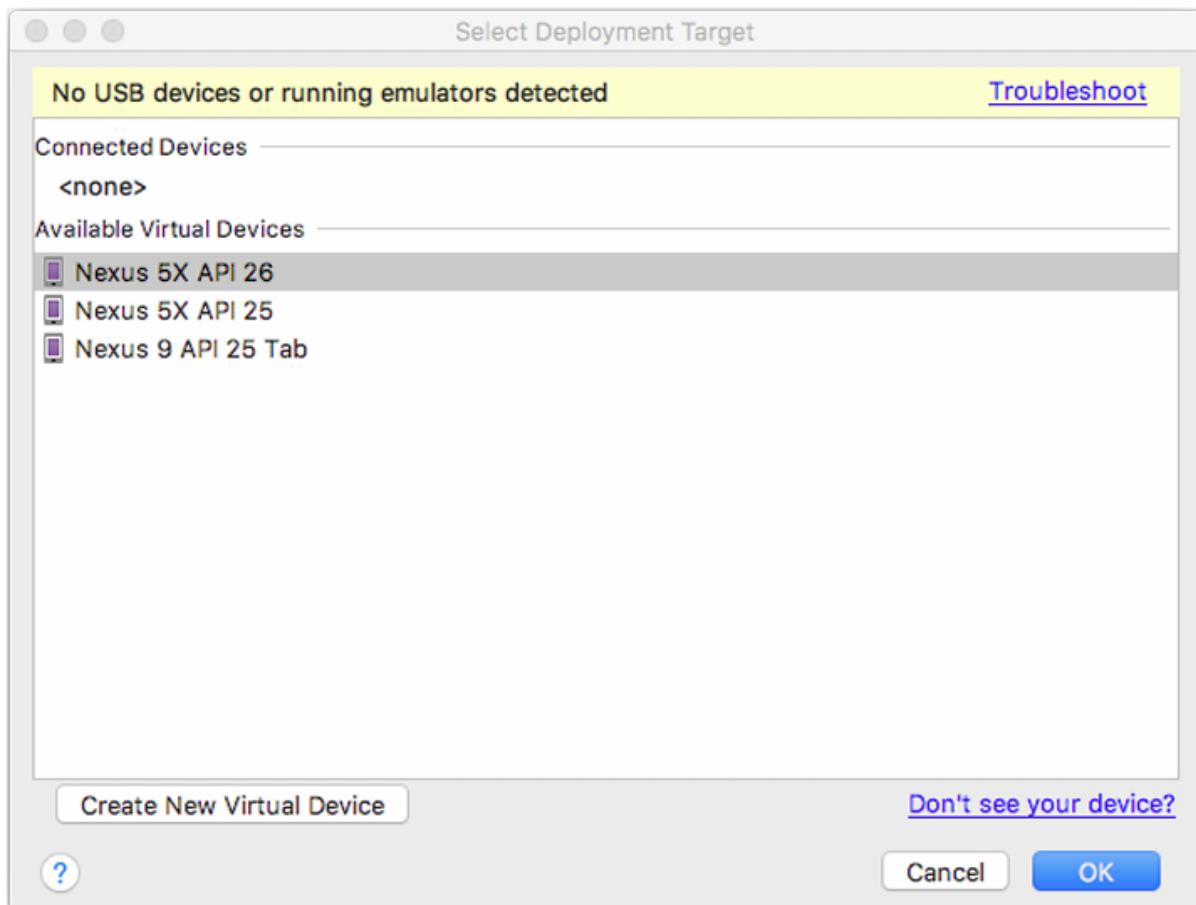
Nếu liên kết Tải xuống hiển thị bên cạnh hình ảnh hệ thống bạn muốn sử dụng, thì liên kết đó chưa được cài đặt. Nhấp vào liên kết để bắt đầu tải xuống và nhấp vào Kết thúc khi hoàn tất.

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

3.2 Chạy ứng dụng trên thiết bị ảo

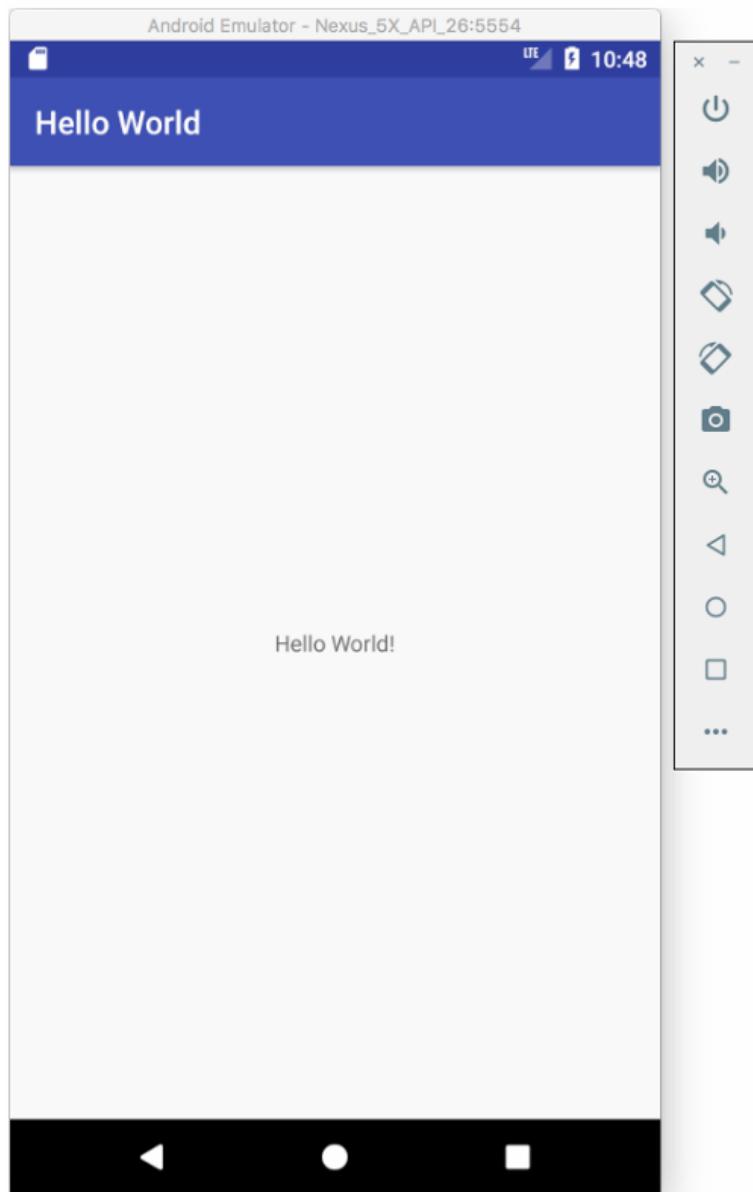
Trong tác vụ này, cuối cùng bạn sẽ chạy ứng dụng Hello World của mình.

1. Trong Android Studio, chọn Ứng dụng Run > Run hoặc nhấp vào biểu tượng Run  trên thanh công cụ.
2. Cửa sổ Chọn mục tiêu triển khai, trong Thiết bị ảo có sẵn , chọn thiết bị ảo mà bạn vừa tạo và nhấp vào OK .



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

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



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

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

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

Những gì bạn cần:

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

4.1 Bật tính năng gỡ lỗi USB

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

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

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

4.2 Chạy ứng dụng trên thiết bị

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

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

2. Nhấp vào nút Run  trên thanh công cụ. Cửa sổ Chọn mục tiêu triển khai sẽ mở ra với danh sách các trình giả lập có sẵn và thiết bị được kết nối.
3. Chọn thiết bị của bạn và nhấp vào OK .

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

Khắc phục sự cố

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

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

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

1. Rút phích cắm của thiết bị.
2. Trên thiết bị, mở Tùy chọn nhà phát triển trong ứng dụng Cài đặt.
3. Nhấn vào Thu hồi ủy quyền gỡ lỗi USB.
4. Kết nối lại thiết bị với máy tính của bạn.
5. Khi được nhắc, cấp ủy quyền. Bạn có thể cần cài đặt trình điều khiển USB thích hợp cho thiết bị của mình. Xem tài liệu Sử dụng thiết bị phần cứng .

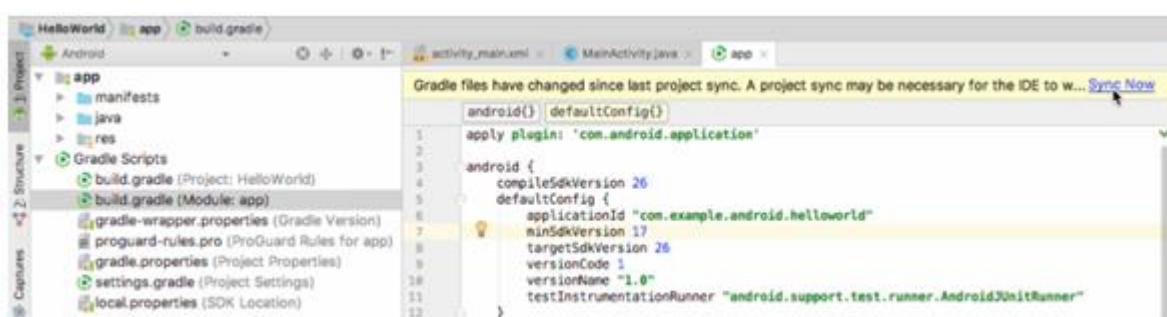
Nhiệm vụ 5: Thay đổi cấu hình Gradle của ứng dụng

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

5.1 Thay đổi phiên bản SDK tối thiểu cho ứng dụng

Làm theo các bước sau:

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



```
Gradle files have changed since last project sync. A project sync may be necessary for the IDE to w... Sync Now
    android() defaultConfig()
    apply plugin: 'com.android.application'

    android {
        compileSdkVersion 26
        defaultConfig {
            applicationId "com.example.android.helloworld"
            minSdkVersion 17
            targetSdkVersion 26
            versionCode 1
            versionName "1.0"
            testInstrumentationRunner "android.support.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 các thay đổi đối với tệp cấu hình bản dựng trong dự án, Android Studio yêu cầu bạn đồng bộ hóa các tệp dự án để có thể nhập các thay đổi về cấu hình bản dựng và chạy một số kiểm tra để đảm bảo cấu hình sẽ không tạo ra lỗi bản dựng.

Để đồng bộ hóa các tệp dự án, hãy nhấp vào Đồng bộ hóa ngay trong thanh thông báo xuất hiện khi thực hiện thay đổi (như trong hình trước)

hoặc nhấp vào biểu tượng Đồng bộ hóa dự án với tệp Gradle  trên thanh công cụ.

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

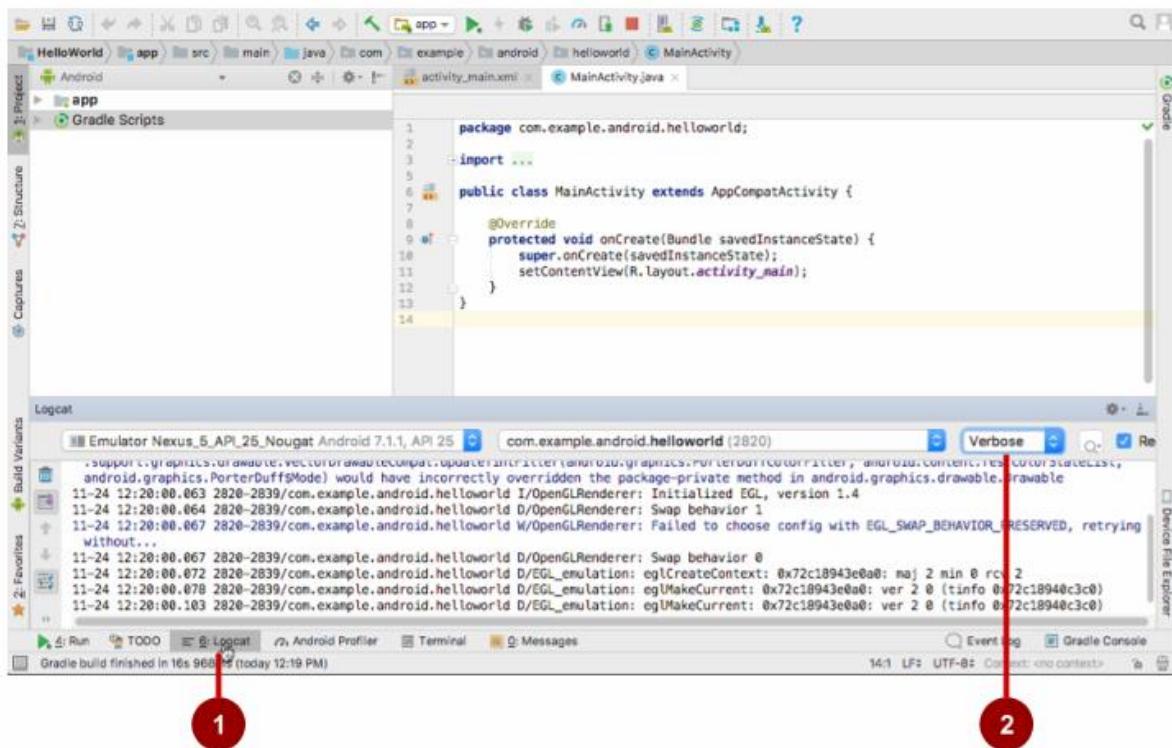
Để tìm hiểu sâu hơn về Gradle, hãy xem tài liệu Tổng quan về hệ thống xây dựng và Định cấu hình bản dựng Gradle.

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

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

6.1 Xem ngăn Logcat

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



Trong hình trên:

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

2. Menu Mức nhật ký được đặt thành Chi tiết (mặc định), hiển thị tất cả các thông báo Nhật ký. Các cài đặt khác bao gồm Gỡ lỗi, Lỗi, Thông tin và Cảnh báo.

6.2 Thêm câu lệnh nhật ký vào ứng dụng

Câu lệnh nhật ký trong mã ứng dụng sẽ hiển thị thông báo trong ngăn Logcat. Chẳng hạn:

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

Các phần của thông điệp là:

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

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

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

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

Làm theo các bước sau:

1. Mở ứng dụng Hello World của bạn trong Android studio và mở MainActivity .

2. Để tự động thêm các mục nhập rõ ràng vào dự án của bạn (chẳng hạn như android.util.Log cần thiết để sử dụng Nhật ký), hãy chọn Cài đặt > tệp trong Windows hoặc Tùy chọn Android Studio > trong macOS.

3. Chọn Trình chỉnh sửa > Chung > Tự động nhập . Chọn tất cả các hộp kiểm và đặt Chèn nhập khi dán thành Tất cả .

4. Nhấp vào Áp dụng và sau đó 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 mục tiêu và tên gói của ứng dụng có chính xác không.

8. Thay đổi mức nhật ký trong ngăn Logcat thành Gõ lỗi (hoặc để nguyên chi tiết vì có rất ít thông báo nhật ký).

9. Chạy ứng dụng của bạn.

Thông báo sau sẽ xuất hiện trong ngăn Logcat:

```
11-24 14:06:59.001 4696-4696/? D/MainActivity: Hello World
```

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.

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

1. Tạo một dự án mới trong Android Studio.
2. Thay đổi lời chào "Hello World" thành "Happy Birthday thành" và tên của một 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 một ngoại lệ với thông báo Nhật ký. Sau đó, viết mã trong ứng dụng của bạn để kích hoạt và ghi lại một ngoại lệ.

Tóm tắt

- Để cài đặt Android Studio, hãy truy cập Android Studio và làm theo hướng dẫn để tải xuống và cài đặt nó.
- 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 tệp build.gradle(Module:app) khi bạn cần thêm thư viện mới vào dự án của mình hoặc thay đổi phiên bản thư viện.
- Tất cả mã và tài nguyên cho ứng dụng đều nằm trong các thư mục ứng dụng và res. Thư mục java bao gồm các hoạt động, kiểm tra và các thành phần khác trong mã nguồn Java. Thư mục res chứa các tài nguyên, chẳng hạn như bố cục, chuỗi và hình ảnh.
- Chỉnh sửa tệp AndroidManifest.xml để thêm các tính năng, thành phần 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 được gọi là trình giả lập) để chạy ứng dụng của bạn.
- Thêm câu lệnh Nhật ký vào ứng dụng của bạn, 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 nhấn vào Số xây dựng bảy lần. Quay lại màn hình trước đó (Cài đặt) và nhấn 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
- Gặp gỡ Android Studio
- Công cụ dòng lệnh Logcat
- Trình quản lý thiết bị ảo Android (AVD)
- Tổng quan về tệp kê khai ứng dụng
- Định cấu hình bản dựng của bạn
- Lớp nhật ký

- Tạo và quản lý thiết bị ảo

Khác:

- Làm cách nào để 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 về nhà

Xây dựng và chạy một ứng dụng

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

Trả lời những câu hỏi

Câu hỏi 1: Tên của file layout cho hoạt động chính là gì?

- MainActivity.java
- AndroidManifest.xml
- activity_main.xml

- build.gradle

Câu hỏi 2: Tên của tài nguyên chuỗi chỉ định tên của ứng dụng là gì?

- app_name
- xmlns:app
- android:name
- applicationId

Câu hỏi 3 Bạn sử dụng công cụ nào để tạo trình giả lập mới?

- Giám sát thiết bị Android
- Trình quản lý AVD
- Trình quản lý SDK
- Trình chỉnh sửa chủ đề

Câu hỏi 4 Giả sử rằng ứng dụng của bạn bao gồm câu lệnh ghi nhật ký sau:

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

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

- Chi tiết
- Gỡ lỗi
- Thông tin
- Cảnh báo
- Lỗi

- Khẳng định

Gửi ứng dụng của bạn để chấm điểm

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

- Hoạt động hiển thị "Hello World" trên màn hình.
- Ghi lại các câu lệnh trong onCreate() trong hoạt động chính.
- Mức nhật ký trong ngăn Logcat chỉ hiển thị các câu lệnh gỡ lỗi hoặc ghi nhật ký lỗi.

Bài 1.2 Phần A: Giao diện người dùng tương tác đầu tiên của bạn

Giới thiệu

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

- TextView để hiển thị văn bản.
- EditText để cho phép người dùng nhập và chỉnh sửa văn bản.
- Nút và các yếu tố có thể nhấp khác (chẳng hạn như RadioButton, CheckBox và Spinner) để cung cấp hành vi tương tác.
- ScrollView và RecyclerView để hiển thị các mục có thể cuộn.
- ImageView để hiển thị hình ảnh.
- ConstraintLayout và LinearLayout để chứa các phần tử View khác và định vị chúng.

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

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

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

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

Những gì bạn nên biết

Bạn nên làm quen:

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

Những gì bạn sẽ học

- Cách tạo một ứng dụng với hành vi tương tác.
- Cách sử dụng trình chỉnh sửa bố cục để thiết kế bố cục.

- Cách chỉnh sửa bố cục trong XML.
- Rất nhiều thuật ngữ mới. Kiểm tra bảng thuật ngữ từ vựng và khái niệm để biết các định nghĩa thân thiện.

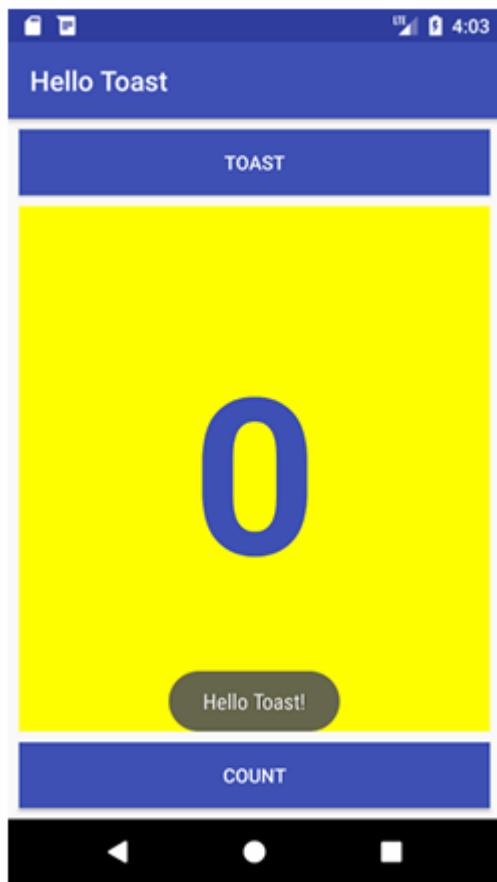
Những gì bạn sẽ làm

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

Tổng quan về ứng dụng

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

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



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

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

1.1 Tạo dự án Android Studio

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

Giá trị	Thuộc tính
Tên ứng dụng	Hello Toast
Tên công ty	Com.example.android(hoặc miền riêng của bạn)
SDK tối thiểu trên điện thoại và máy tính bảng	API15: Android 4.0.3 IceCreamSandwich
Mẫu	Hoạt động rỗng
Tạo hộp tệp bối cảnh	Chọn

Hộp tương thích ngược

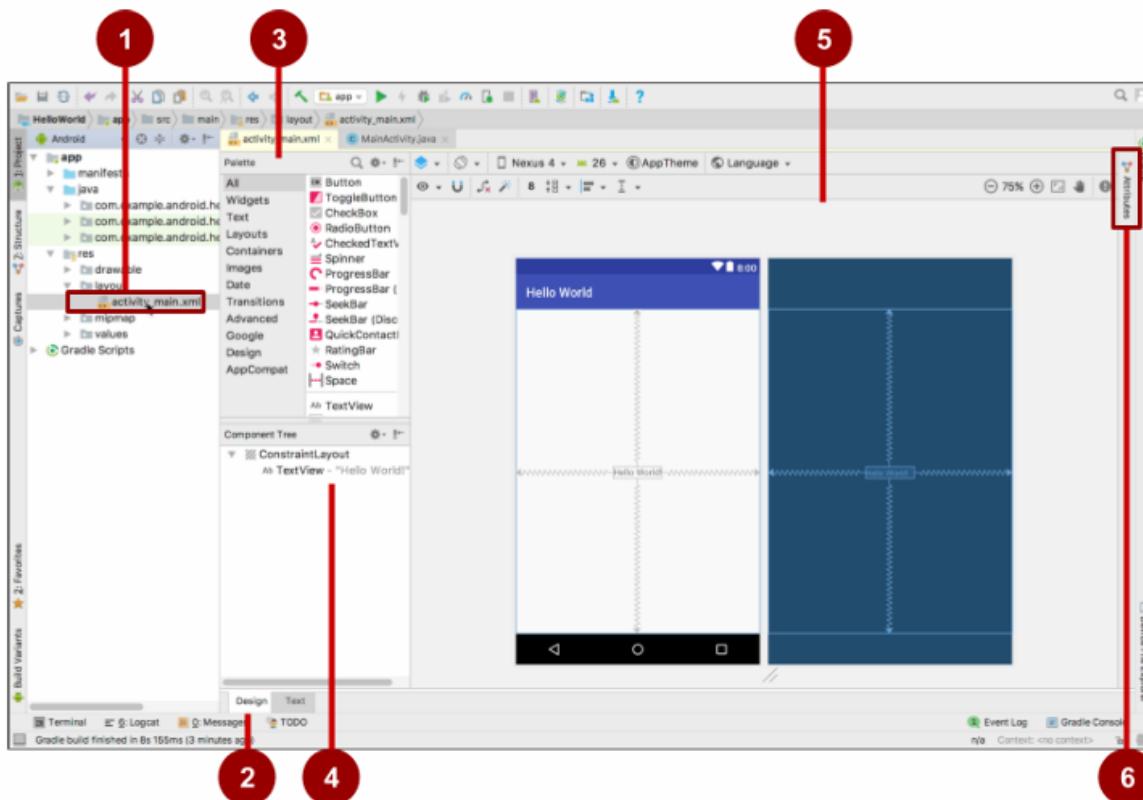
Chọn

15. Chọn Chạy > Chạy ứng dụng hoặc nhấp vào biểu tượng Chạy trên thanh công cụ để tạo và thực thi ứng dụng trên trình mô phỏng 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 tạo bố cục các phần tử giao diện người dùng (UI) của ứng dụng. Nó cho phép bạn kéo các phần tử vào chế độ xem thiết kế trực quan và bản thiết kế, định vị chúng trong bố cục, thêm ràng buộc và đặt thuộc tính. Các ràng buộc xác định vị trí của một phần tử giao diện người dùng trong bố cục. Một ràng buộc đại diện cho một kết nối hoặc căn chỉnh với một chế độ xem khác, bố cục cha hoặc một hướng dẫn vô hình.

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



1. Trong ứng dụng > res > thư mục bố cục trong ngăn Project > Android, nhấp đúp vào tệp activity_main.xml để mở, nếu nó chưa mở.

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

3. Ngăn Bảng màu hiển thị các thành phần giao diện người dùng mà bạn có thể sử dụng trong bố cục của ứng dụng.

4. Ngăn cây thành phần hiển thị hệ thống phân cấp chế độ xem của các phần tử giao diện người dùng. Các phần tử View được tổ chức thành một hệ thống phân cấp cây gồm cha mẹ và con, trong đó con kế thừa các thuộc tính của cha mẹ của nó. Trong hình trên, TextView là con của ConstraintLayout . Các em sẽ học về các yếu tố này ở phần sau của bài học này.

5. Các ngăn thiết kế và bản thiết kế của trình chỉnh sửa bố cục hiển thị các yếu tố giao diện người dùng trong bố cục. Trong hình trên, bố cục chỉ hiển thị một phần tử: TextView hiển thị "Hello World".

6. Tab Thuộc tính hiển thị ngăn Thuộc tính để thiết lập thuộc tính cho phần tử giao diện người dùng.

Mẹo: Xem Tạo giao diện người dùng bằng Trình chỉnh sửa bố cục để biết thông tin chi tiết về cách sử dụng trình chỉnh sửa bố cục và Tìm hiểu về Android Studio để xem tài liệu đầy đủ về Android Studio.

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

Trong tác vụ này, bạn tạo bố cục giao diện người dùng cho ứng dụng HelloToast trong trình chỉnh sửa bố cục bằng cách sử dụng các tính năng ConstraintLayout. Bạn có thể tạo các ràng buộc theo cách thủ công, như được hiển thị sau hoặc tự động bằng cách sử dụng công cụ Tự động kết nối.

2.1 Kiểm tra các ràng buộc của phần tử

Làm theo các bước sau:

1. Mở activity_main.xml từ ngăn Project > Android nếu nó chưa mở. Nếu tab Thiết kế chưa được chọn, hãy nhấp vào tab đó. Nếu không có bản

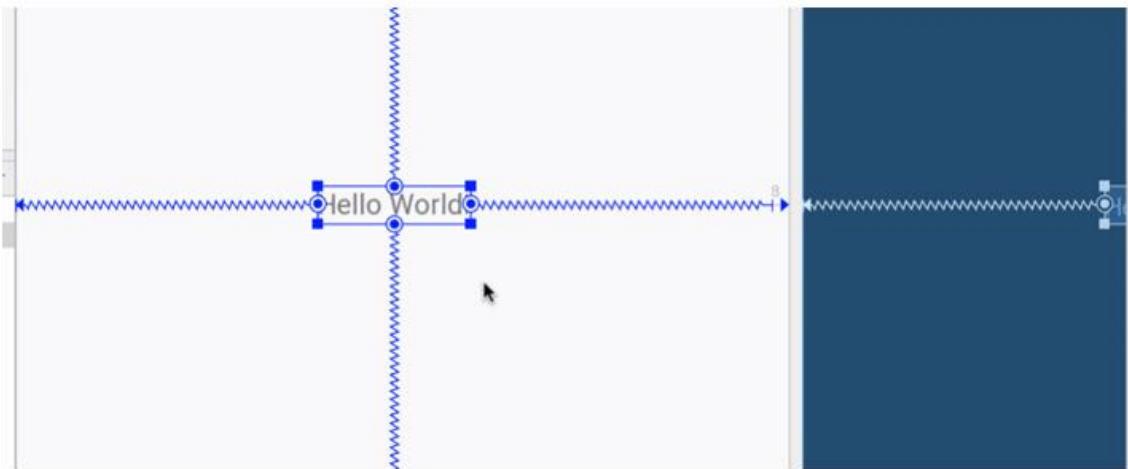
thiết kế, hãy nhấp vào nút Select Design Surface trên thanh công cụ và chọn Design + Blueprint .

2. Công cụ Tự động kết nối cũng nằm trong thanh công cụ. Nó được bật theo mặc định. Đối với bước này, hãy đảm bảo rằng công cụ không bị tắt.

3. Nhấp vào thu phóng trong một cái nhìn cận cảnh. để phóng to các ngăn Thiết kế và Bản thiết kế cho

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

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



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

- Tay cầm ràng buộc: Để tạo một ràng buộc như trong hình động ở trên, hãy nhấp vào một tay cầm ràng buộc, được hiển thị dưới dạng vòng

tròn ở bên cạnh của một phần tử. Sau đó, kéo tay cầm đến một tay cầm ràng buộc khác hoặc đến ranh giới mẹ. Một đường ngoằn ngoèo đại diện cho ràng buộc.



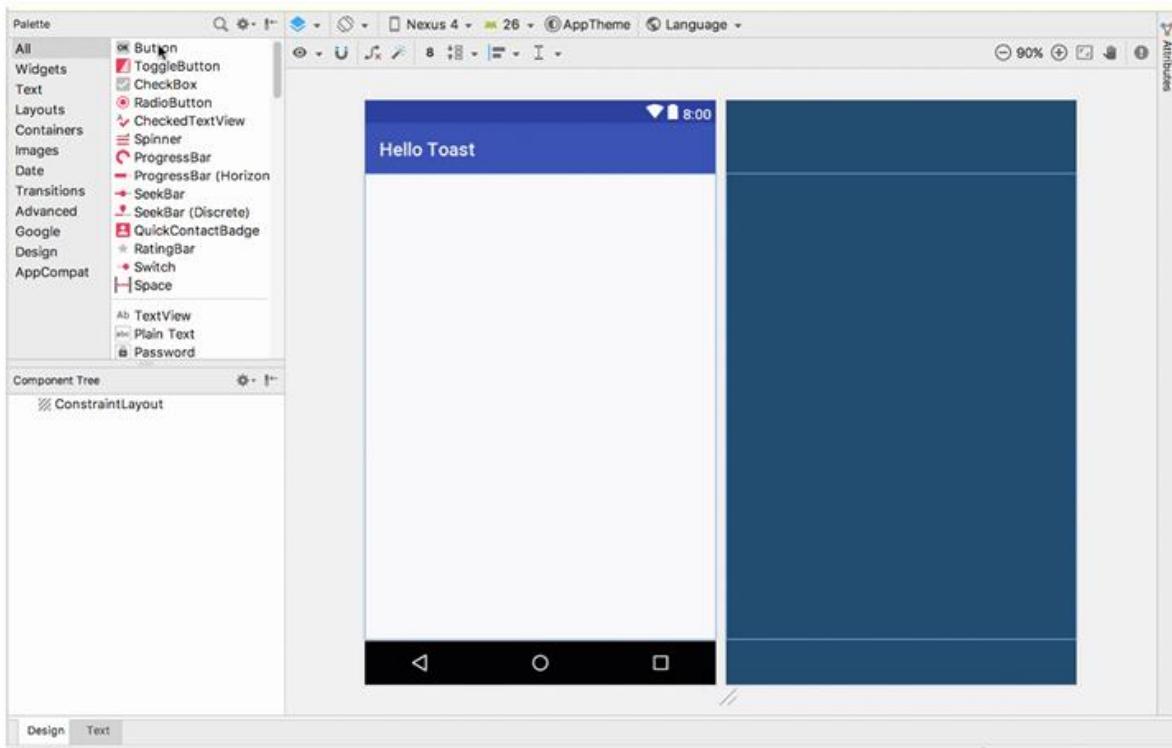
- Thay đổi kích thước tay cầm : Để thay đổi kích thước phần tử, hãy kéo các tay cầm thay đổi kích thước hình vuông. Tay cầm thay đổi thành một góc cạnh trong khi bạn đang kéo nó.



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

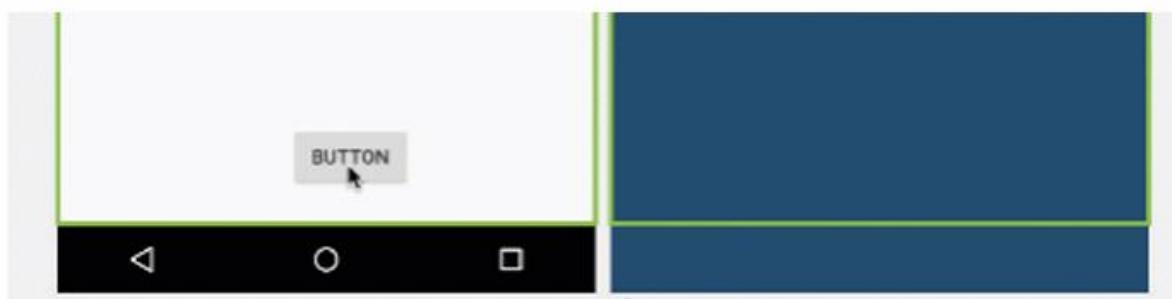
Làm theo các bước sau để thêm Nút:

1. Bắt đầu với một bảng sạch. Phần tử TextView là không cần thiết, vì vậy trong khi nó vẫn được chọn, hãy nhấn phím Delete hoặc chọn Chỉnh sửa > Xóa . Böyle giờ bạn có một bố cục hoàn toàn trống.
2. Kéo một nút từ ngăn Bảng màu đến bất kỳ vị trí nào trong bố cục. Nếu bạn thả Nút ở khu vực giữa trên cùng của bố cục, các ràng buộc có thể tự động xuất hiện. Nếu không, bạn có thể kéo các ràng buộc vào phía trên, bên trái và bên phải của bố cục như trong hình động bên dưới.



2.3 Thêm một nút thứ hai vào bố cục

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



Bạn có thể xóa các ràng buộc khỏi một phần tử bằng cách chọn phần tử và di con trỏ lên phần tử đó để hiển thị nút Xóa ràng buộc. Nhấp vào nút này để loại bỏ tất cả các ràng buộc trên phần tử đã chọn. Để xóa một ràng buộc duy nhất, hãy nhấp vào tay cầm cụ thể đặt ràng buộc. Để xóa tất cả các ràng buộc trong toàn bộ bố cục, hãy nhấp vào công cụ Xóa tất cả các

ràng buộc trên thanh công cụ. Công cụ này rất hữu ích nếu bạn muốn làm lại tất cả các ràng buộc trong bố cục của mình.

Nhiệm vụ 3: Thay đổi thuộc tính phần tử giao diện người dùng

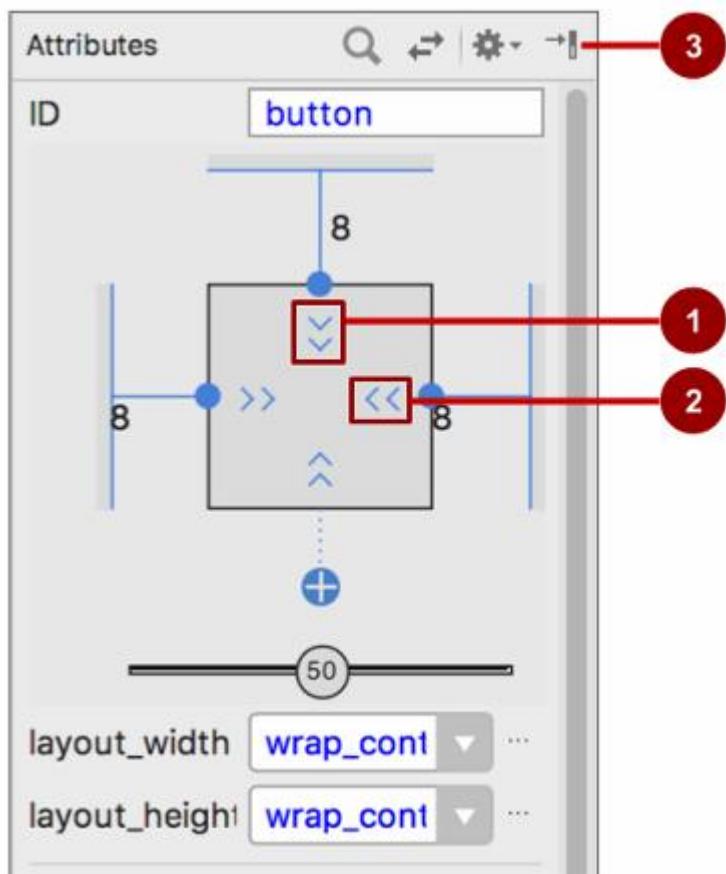
Ngăn Thuộc tính cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho phần tử giao diện người dùng. Bạn có thể tìm thấy các thuộc tính (được gọi là thuộc tính) chung cho tất cả các chế độ xem trong tài liệu lớp View .

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

3.1 Thay đổi kích thước

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

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



Trong hình trên:

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

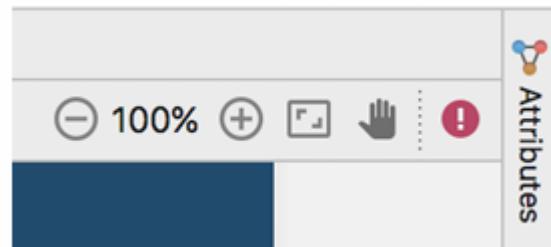
2. Kiểm soát chiều rộng. Điều khiển này chỉ định layout_width và xuất hiện trong hai phân đoạn ở bên trái và bên phải của hình vuông. Các góc cho biết rằng điều khiển này được đặt thành wrap_content có nghĩa là Chế độ xem sẽ mở rộng theo chiều ngang khi cần thiết để phù hợp với nội dung của nó, lên đến biên độ 8dp.

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

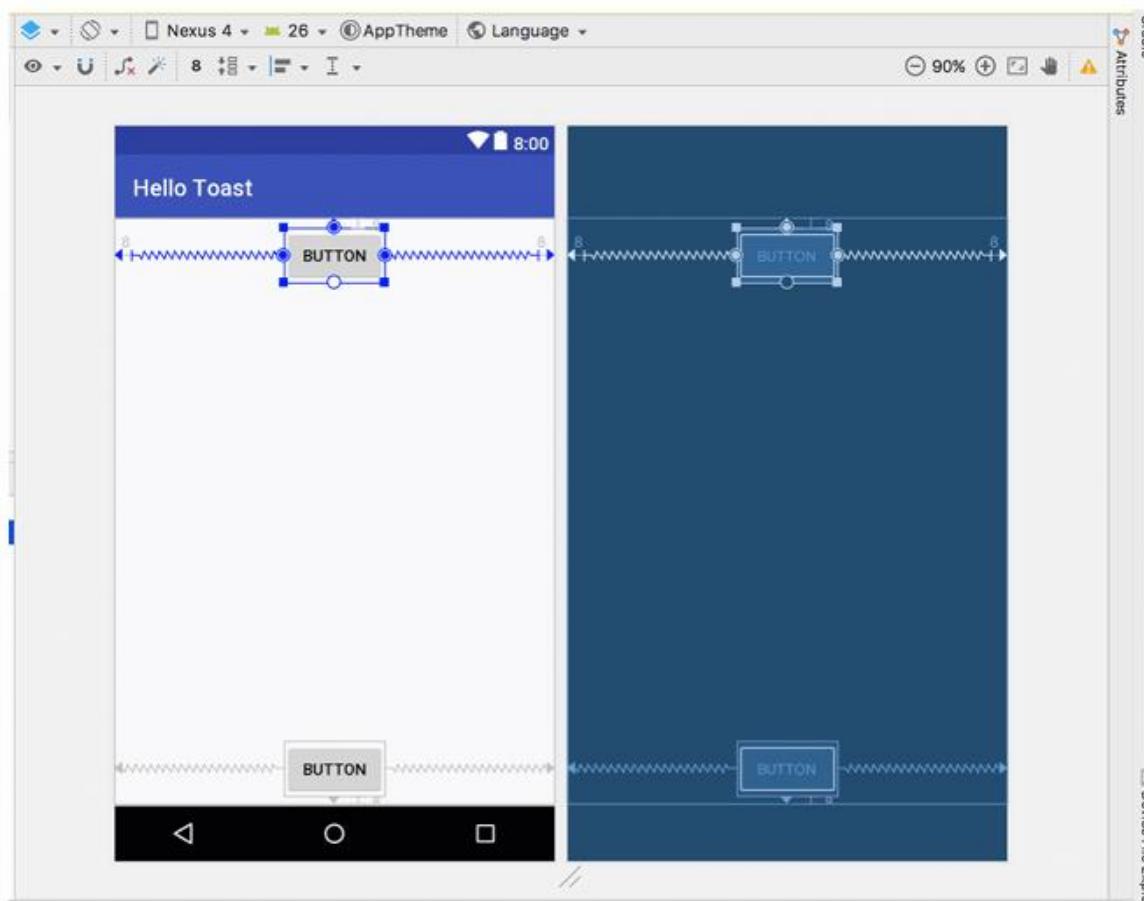
Làm theo các bước sau:

- Chọn nút trên cùng trong ngăn Cây thành phần.

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

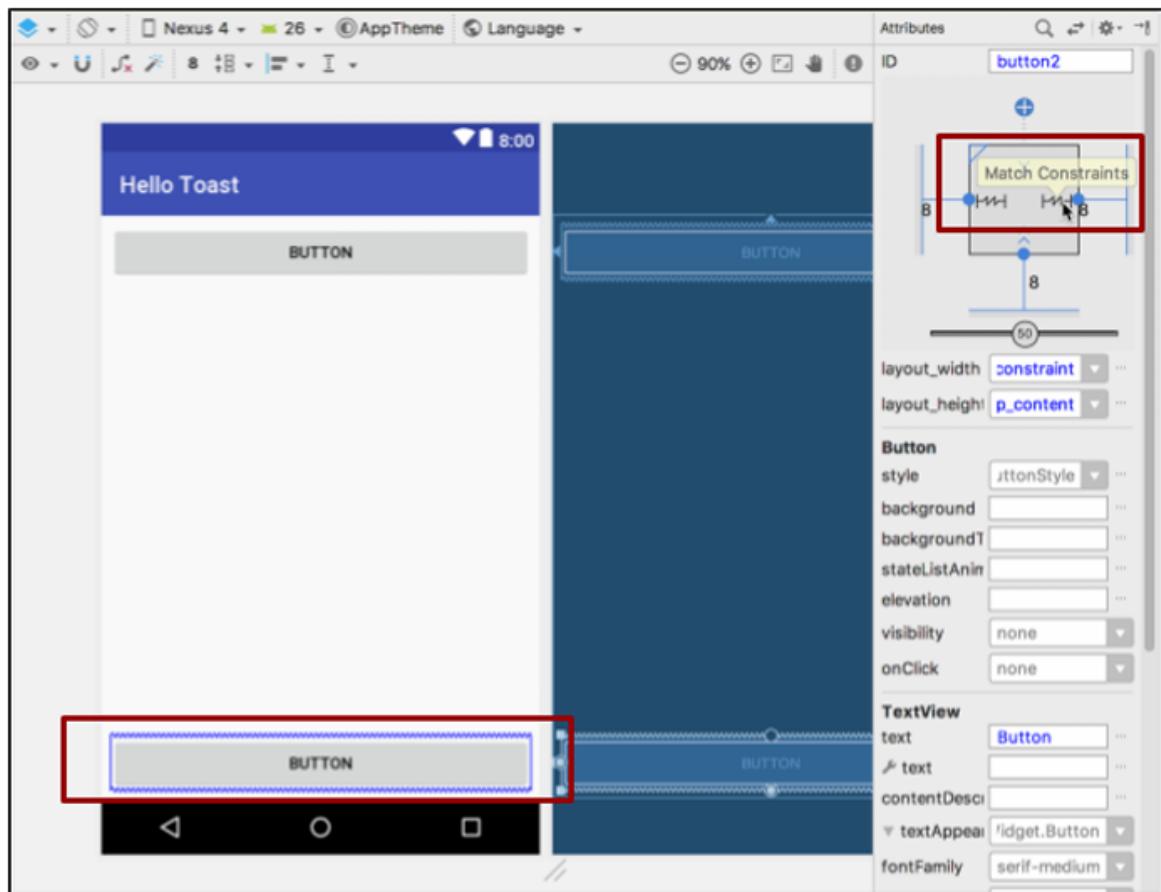


3. Nhấp vào điều khiển chiều rộng hai lần — lần nhấp đầu tiên thay đổi nó thành Cố định với các đường thẳng và lần nhấp thứ hai thay đổi nó thành Kết hợp các ràng buộc với cuộn lò xo, như được hiển thị trong hình ảnh động bên dưới.



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

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



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

- Cài đặt match_constraint mở rộng phần tử View để lấp đầy phần tử cha của nó theo chiều rộng hoặc chiều cao—lên đến lề, nếu được đặt. Cha trong trường hợp này là ConstraintLayout . Bạn tìm hiểu thêm về ConstraintLayout trong nhiệm vụ tiếp theo.
- Cài đặt wrap_content thu nhỏ kích thước của phần tử View để nó vừa đủ lớn để bao bọc nội dung của nó. Nếu không có nội dung, phần tử View sẽ trở nên vô hình.

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

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 kích thước 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.

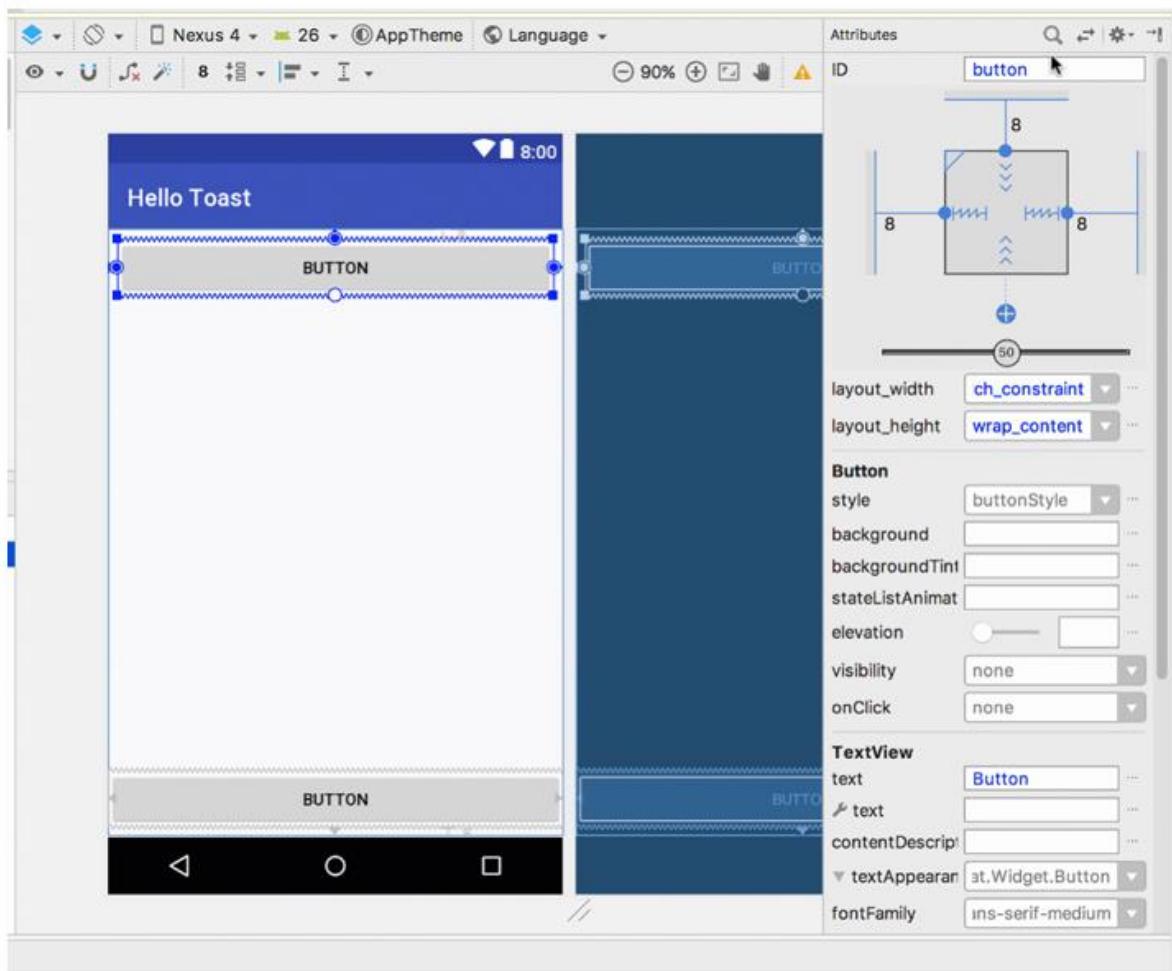
3.2 Thay đổi thuộc tính Nút

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

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

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

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



5. Thực hiện các thay đổi thuộc tính tương tự cho Nút thứ hai , sử dụng button_count làm ID, Đếm cho thuộc tính văn bản và cùng màu cho nền và văn bản như các bước trước.

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

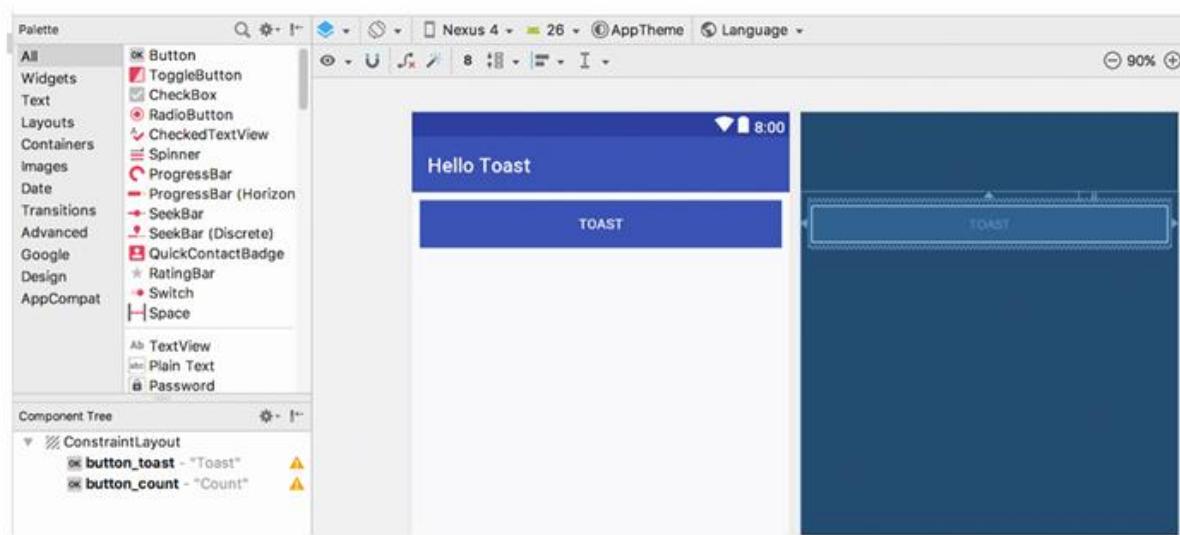
Nhiệm vụ 4: Thêm TextView và đặt các thuộc tính của nó

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

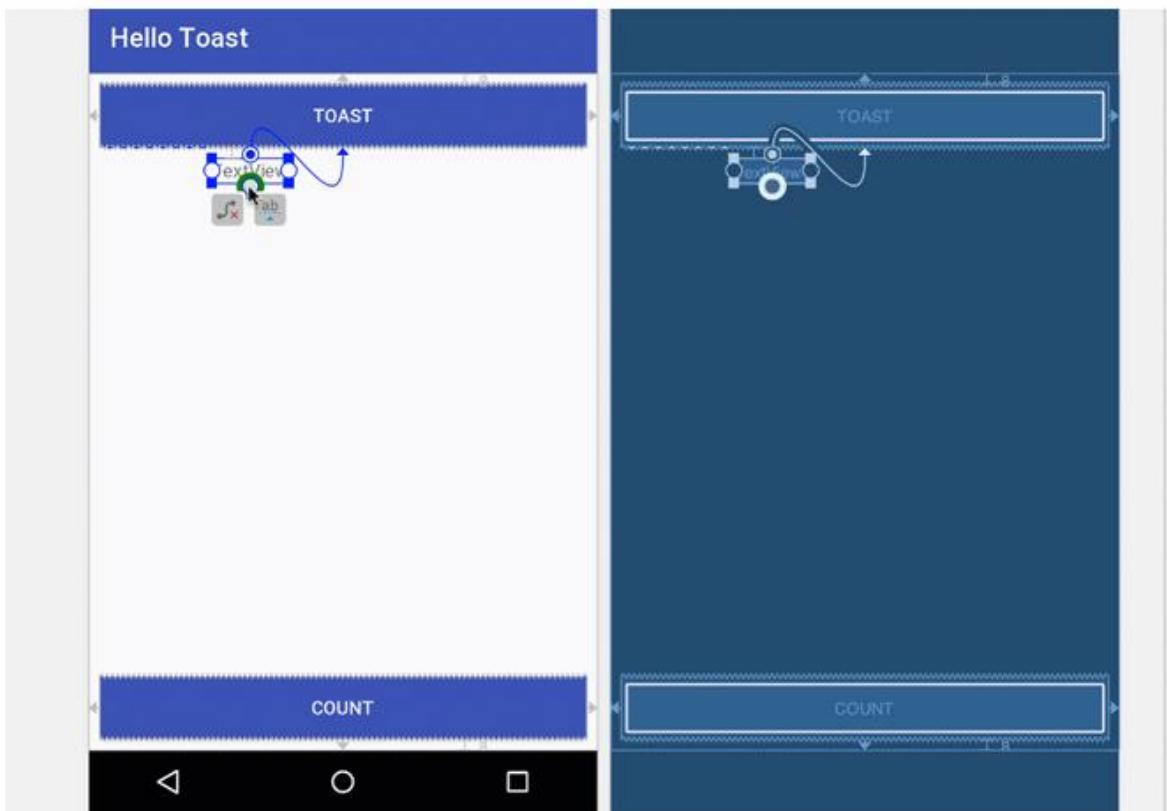
vào lề và theo chiều dọc với hai phần tử Button. Sau đó, bạn sẽ thay đổi các thuộc tính cho TextView trong ngăn Attributes (Thuộc tính).

4.1 Thêm TextView và các ràng buộc

1. Như thể hiện trong hình động 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 .



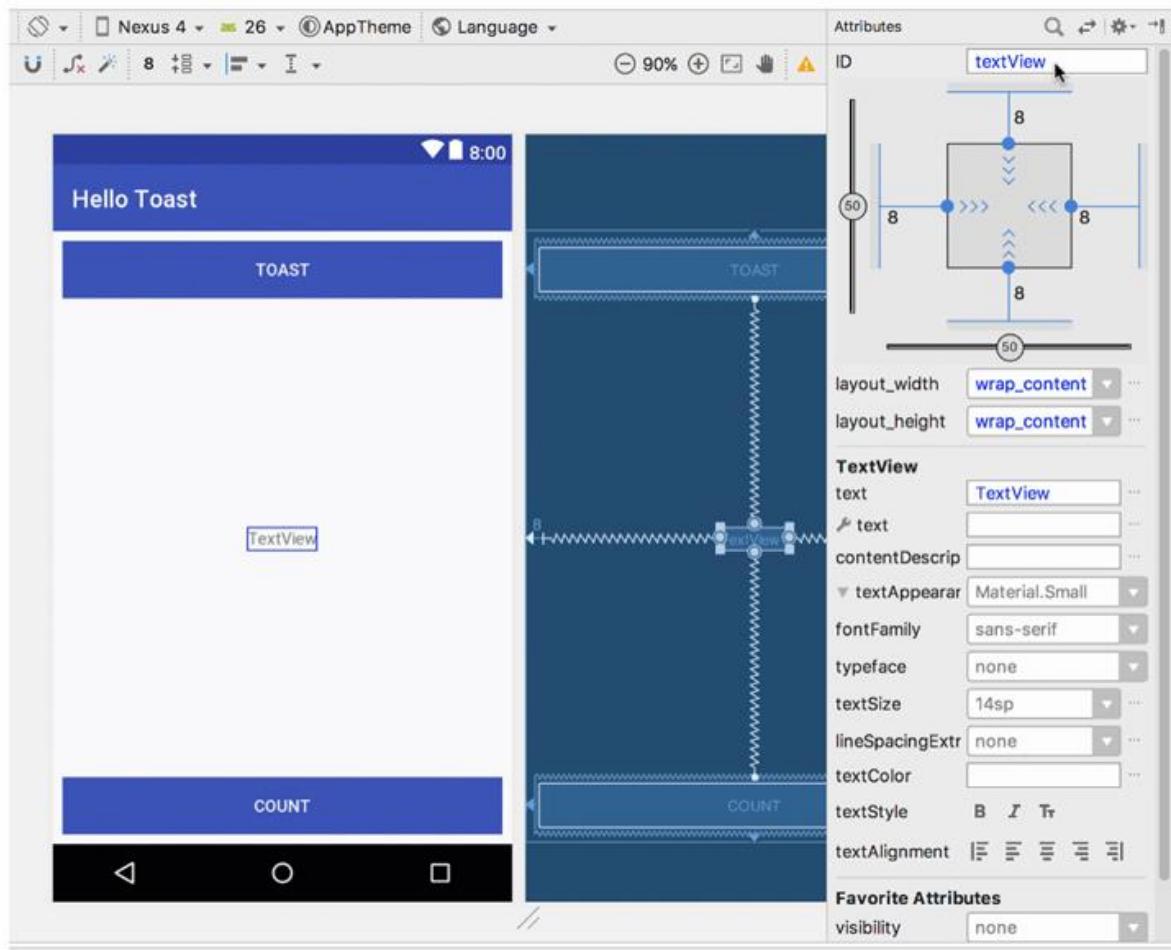
2. Như thể hiện trong hình động bên dưới, kéo một ràng buộc từ dưới cùng của TextView đến tay cầm ở trên cùng của Nút đếm và từ các cạnh của TextView đến các cạnh của bố cục. Điều này hạn chế TextView ở giữa bố cục giữa hai phần tử Button.



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

1. Đặt ID thành `show_count` .
2. Đặt văn bản thành `0`.
3. Đặt `textSize` thành `160sp` .
4. Đặt `textStyle` thành `B` (in đậm) và `textAlignment` thành `ALIGNCENTER` (căn giữa đoạn văn).
5. Thay đổi các điều khiển kích thước chế độ xem ngang và dọc (`layout_width` và `layout_height`) thành `match_constraint`.
6. Đặt `textColor` thành `@color/colorPrimary` .
7. Cuộn xuống ngăn và nhấp vào Xem tất cả các thuộc tính , cuộn xuống trang thứ hai của thuộc tính sang nền , sau đó nhập `#FFF00` cho màu vàng.

8. Cuộn xuống trọng lực , mở rộng trọng lực và chọn center_ver (đối với trung tâm-dọc).



• textSize: Kích thước văn bản của TextView. Đối với bài học này, kích thước được đặt thành 160sp . sp là viết tắt của pixel-independent tỷ lệ và giống như dp , là một đơn vị chia tỷ lệ theo mật độ màn hình và tùy chọn kích thước phông chữ của người dùng. Sử dụng đơn vị dp khi bạn chỉ định kích thước phông chữ để kích thước được điều chỉnh cho cả mật độ màn hình và sở thích của người dùng.

• textStyle và 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).

• trọng lực: Thuộc tính trọng lực chỉ định cách một Chế độ xem được căn chỉnh trong Chế độ xem hoặc ViewGroup gốc của nó. Trong bước này,

bạn căn giữa TextView được căn giữa theo chiều dọc trong ConstraintLayout gốc.

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

Nhiệm vụ 5: Chính sửa bố cục trong XML

Bố cục ứng dụng Hello Toast gần như đã hoàn thành! Tuy nhiên, một dấu chấm than xuất hiện bên cạnh mỗi phần tử giao diện người dùng trong Cây thành phần. Di con trỏ của bạn qua các dấu chấm than này để xem thông báo cảnh báo, như được hiển thị bên dưới. Cảnh báo tương tự xuất hiện cho cả ba phần tử: chuỗi được mã hóa cứng nên sử dụng tài nguyên.



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

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 Văn bản Design Text ở cuối trình chỉnh sửa bố cục.

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

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.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:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.android.hellotoast.MainActivity">

    <Button
        android:id="@+id/button_toast"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:background="@color/colorPrimary"
        android:text="Toast"
        android:textColor="@android:color/white"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button_count"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:background="@color/colorPrimary"
        android:text="Count"
        android:textColor="@android:color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

    <TextView
        android:id="@+id/show_count"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginBottom="8dp"
```

5.2 Trích xuất tài nguyên chuỗi

Thay vì mã hóa các chuỗi cứng, cách tốt nhất là sử dụng 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 nếu bạn sử dụng các chuỗi này nhiều lần. Ngoài ra, tài nguyên chuỗi là bắt buộc để dịch và bản địa hóa ứng dụng của bạn, vì bạn cần tạo tệp tài nguyên chuỗi cho từng ngôn ngữ.

1. Nhập một lần vào từ "Toast" (cảnh báo được đánh dấu đầu tiên).
2. Nhấn Alt-Enter trong Windows hoặc Option-Enter trong macOS và chọn Trích xuất tài nguyên chuỗi từ menu bật lên.
3. Nhập button_label_toast cho Tên tài nguyên .
4. Nhấp vào OK . Tài nguyên chuỗi được tạo trong tệp values/res/string.xml và chuỗi trong mã của bạn được thay thế bằng tham chiếu đến tài nguyên: @string/button_label_toast
5. Trích xuất các chuỗi còn lại: button_label_count cho "Đếm" và count_initial_value cho "0" .

6. Trong ngăn Project > Android, mở rộng các giá trị trong res , sau đó nhấp đúp vào strings.xml để xem tài nguyên chuỗi của bạn trong tệp strings.xml:

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

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

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

Mẹo: Tài nguyên chuỗi bao gồm tên ứng dụng, tên này xuất hiện trong 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 trong MainActivity thực thi khi người dùng nhấn vào Button .

6.1 Thêm thuộc tính và trình xử lý onClick vào mỗi Button

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

1. Với trình chỉnh sửa XML đang mở (tab Văn bản), hãy tìm Nút với android:id được đặt thành button_toast.

```
<Button  
    android:id="@+id/button_toast"  
    android:layout_width="0dp"  
    ...  
    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 Tạo trình xử lý nhấp chuột , chọn MainActivity và nhấp vào OK . Nếu biểu tượng bóng đèn màu đỏ không xuất hiện, hãy nhấp vào tên phương thức ("showToast"). Nhấn Alt-Enter (Option-Enter trên máy Mac), chọn Tạo 'showToast(view)' trong MainActivity và nhấp vào OK . Hành động này tạo sơ khai phương thức giữ chỗ cho phương thức showToast() trong MainActivity , như được hiển thị ở cuối các bước này.

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

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

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

```
<Button
    android:id="@+id/button_toast"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:background="@color/colorPrimary"
    android:text="@string/button_label_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="0dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:background="@color/colorPrimary"
    android:text="@string/button_label_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="0dp"
    android:layout_height="0dp"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:background="#FFFF00"
    android:gravity="center_vertical"
    android:text="@string/count_initial_value"
    android:textAlignment="center"
    android:textColor="@color/colorPrimary"
    android:textSize="160sp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@+id/button_count"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/button_toast" />
```

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

```
package com.example.android.hellotoast;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends AppCompatActivity {

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

    public void showToast(View view) {
    }

    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 nhấn vào Nút hoặc thực hiện một hành động.

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

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

```
public void showToast(View view) {
```

2. Để tạo một thực thể của `Toast`, hãy gọi phương thức nhà máy `makeText()` trên lớp `Toast`.

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

Tuyên bố này không đầy đủ cho đến khi bạn hoàn thành tất cả các bước.

3. Cung cấp ngữ cảnh của ứng dụng Hoạt động. Vì `Toast` hiển thị trên đầu giao diện người dùng Hoạt động, hệ thống cần thông tin về Hoạt động hiện tại. Khi

bạn đã ở trong ngữ cảnh của Hoạt động mà bạn cần, hãy sử dụng nó như một phím tắt.

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

5. Cung cấp thời lượng hiển thị. Ví dụ, Toast.LENGTH_SHORT hiển thị bánh mì nướng trong một thời gian tương đối ngắn.

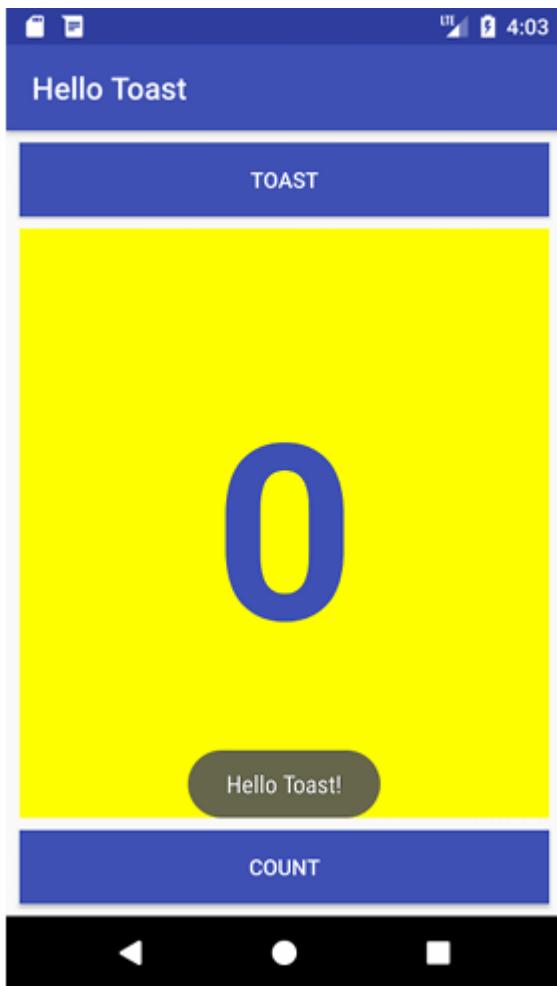
```
Toast toast = Toast.makeText(this, R.string.toast_message,  
Toast.LENGTH_SHORT);
```

Thời lượng của màn hình Toast có thể là Toast.LENGTH_LONG hoặc Toast.LENGTH_SHORT . Độ dài thực tế là khoảng 3.5 giây đối với Bánh mì nướng dài và 2 giây đối với Bánh mì nướng ngắn.

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

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

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



6.3 Chỉnh sửa trình xử lý Nút đếm

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

Mã cho trình xử lý phải:

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

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

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

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

2. Để theo dõi số lượng, bạn cần một biến thành viên riêng. Mỗi lần nhấn vào nút Đếm sẽ tăng giá trị của biến này. Nhập thông tin sau, sẽ được đánh dấu bằng màu đỏ và hiển thị biểu tượng bóng đèn màu đỏ:

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

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

3. Nhấp vào biểu tượng bóng đèn màu đỏ và chọn Tạo trường 'mCount' từ menu bật lên. Thao tác này sẽ tạo ra một biến thành viên 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 { private int mCount;
```

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

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

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

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

6. Nay bạn đã có mShowCount , bạn có thể nhận được tham chiếu đến TextView bằng cách sử dụng ID bạn đã đặt trong tệp bố cục. Để nhận 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 khởi động bố cục, có nghĩa là thiết lập chế độ xem nội dung của màn hình thành bố cục XML. Bạn cũng có thể sử dụng nó để lấy tham chiếu đến các phần tử giao diện người dùng khác trong bố

cục, chẳng hạn như TextView . Xác định vị trí phương thức onCreate() trong MainActivity:

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

7. Thêm câu lệnh findViewById vào cuối phương thức:

```
@Override 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).

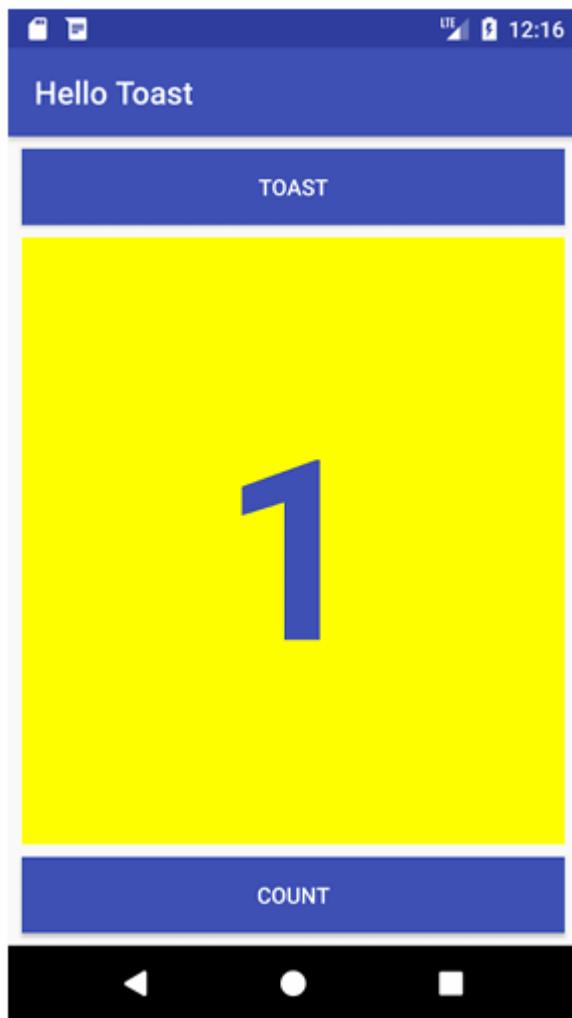
8. Nay bạn đã gán cho mShowCount TextView , bạn có thể sử dụng biến để đặt văn bản trong TextView thành giá trị của biến mCount. Thêm thông tin sau vào phương thức countUp():

```
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 nhấn vào nút Đếm.



Mẹo: Để biết hướng dẫn chuyên sâu về cách sử dụng ConstraintLayout, hãy xem Lớp học lập trình Sử dụng ConstraintLayout để thiết kế chế độ xem .

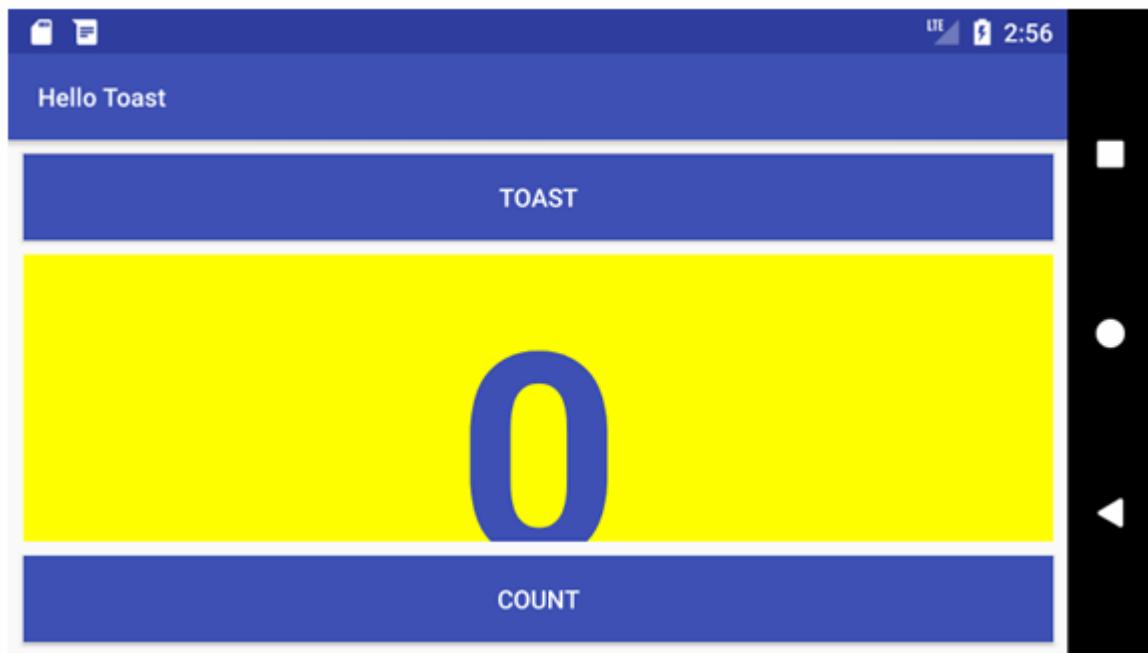
Mã giải pháp dự án

Android Studio: HelloToast

Thử thách mã hóa

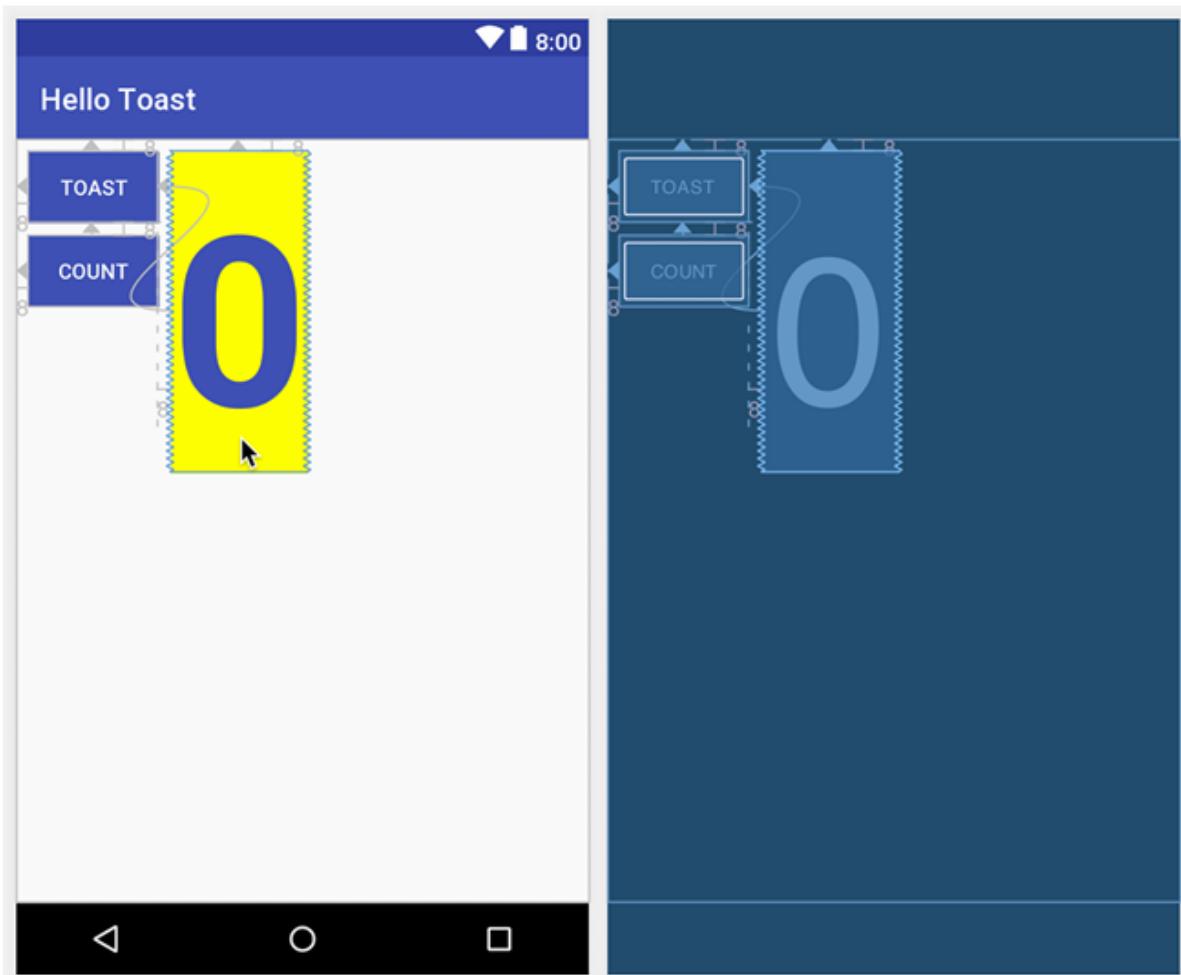
Lưu ý: Tất cả các thử thách mã hóa đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau này.

Ứng dụng HelloToast trông ổn khi thiết bị hoặc trình mô phỏng được định hướng theo chiều dọc. Tuy nhiên, nếu bạn chuyển thiết bị hoặc trình mô phỏng sang hướng ngang, Nút đếm có thể chèn lên TextView dọc theo phía dưới như trong hình bên dưới.



Thách thức : Thay đổi bố cục để nó trông đẹp ở cả hướng ngang và dọc:

1. Trên máy tính của bạn, tạo một bản sao của thư mục dự án HelloToast và đổi tên thành HelloToastChallenge .
2. Mở HelloToastChallenge trong Android Studio và tái cấu trúc nó. (Xem Phụ lục: Tiện ích để biết hướng dẫn sao chép và tái cấu trúc dự án.)
3. Thay đổi bố cục sao cho Nút Toast và Nút đếm xuất hiện ở phía bên trái, như trong hình bên dưới. TextView xuất hiện bên cạnh chúng, nhưng chỉ đủ rộng để hiển thị nội dung của nó. (Gợi ý: Sử dụng wrap_content.)
4. Chạy ứng dụng theo cả hướng ngang và dọc.



Mã giải pháp thử thách dự án

Android Studio: HelloToastChallenge

Tóm tắt

View, ViewGroup và bố cục:

- Tất cả các phần tử giao diện người dùng là các lớp con của lớp View và do đó kế thừa nhiều thuộc tính của lớp siêu View.

- Các phần tử xem có thể được nhóm lại bên trong ViewGroup, hoạt động như một vùng chứa. Mỗi quan hệ là cha-con, trong đó cha là ViewGroup và con là View hoặc ViewGroup khác

- Phương thức onCreate() được sử dụng để khởi tạo bố cục, có nghĩa là đặt chế độ xem nội dung của màn hình thành bố cục XML. Bạn cũng có thể sử dụng nó để lấy tham chiếu đến các thành phần giao diện người dùng khác trong bố cục.

- Chế độ xem, giống như một chuỗi, là một tài nguyên có thể có một id. Lệnh gọi findViewById lấy ID của một chế độ xem làm tham số của nó và trả về View.

Sử dụng trình chỉnh sửa bố cục:

- Nhấp vào tab Thiết kế để thao tác với các phần tử và bố cục, và tab Văn bản để chỉnh sửa mã XML cho bố cục.

- Trong tab Thiết kế, ngăn Bảng màu hiển thị các yếu tố giao diện người dùng mà bạn có thể sử dụng trong bố cục của ứng dụng và ngăn cây thành phần hiển thị hệ thống phân cấp chế độ xem của các thành phần giao diện người dùng.

- Các ngăn thiết kế và bản thiết kế của trình chỉnh sửa bố cục hiển thị các yếu tố giao diện người dùng trong bố cục.

- Tab Thuộc tính hiển thị ngăn Thuộc tính để thiết lập thuộc tính cho phần tử giao diện người dùng.

- Tay cầm ràng buộc: Nhấp vào một tay cầm ràng buộc, được hiển thị dưới dạng vòng tròn ở mỗi bên của một phần tử, sau đó kéo đến một tay cầm ràng buộc khác hoặc đến ranh giới cha để tạo ràng buộc. Ràng buộc được biểu thị bằng đường zigzag.

- Thay đổi kích thước tay cầm: Bạn có thể kéo các tay cầm thay đổi kích thước hình vuông để thay đổi kích thước phần tử. Trong khi kéo, tay cầm thay đổi thành một góc nghịch.

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

- Bạn có thể loại bỏ các ràng buộc khỏi một phần tử bằng cách chọn phần tử và di con trỏ của bạn qua phần tử đó để hiển thị nút Xóa ràng buộc. Nhấp vào nút này để loại bỏ tất cả các ràng buộc trên phần tử đã chọn. Để xóa một ràng buộc duy nhất, hãy nhấp vào tay cầm cụ thể đặt ràng buộc.

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

Đặt chiều rộng và chiều cao bối cảnh:

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 :

- Cài đặt match_constraint mở rộng chế độ xem để lấp đầy khung cảnh chính của nó theo chiều rộng hoặc chiều cao—lên đến lề, nếu được đặt.

- Cài đặt wrap_content thu nhỏ kích thước chế độ xem để chế độ xem vừa đủ lớn để bao bọc nội dung của nó. Nếu không có nội dung, chế độ xem sẽ trở nên vô hình.
- Sử dụng một số dp cố định (pixel không phụ thuộc vào mật độ) để chỉ định kích thước cố định, được điều chỉnh cho phù hợp với kích thước màn hình của thiết bị.

Trích xuất tài nguyên chuỗi:

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

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

Xử lý nhấp chuột:

- Trình xử lý nhấp chuột là một phương thức được gọi khi người dùng nhấp hoặc nhấn vào phần tử giao diện người dùng.
- Chỉ định trình xử lý nhấp chuột cho một phần tử giao diện người dùng chẳng hạn như Nút bằng cách nhập tên của nó vào onClick field trong ngăn Thuộc tính của tab Thiết kế hoặc trong trình chỉnh sửa XML bằng cách thêm thuộc tính android:onClick vào một phần tử giao diện người dùng như Nút.
- Tạo trình xử lý nhấp chuột trong Hoạt động chính bằng cách sử dụng tham số Xem. Ví dụ: public void showToast(Xem chế độ xem) {...} .
- Bạn có thể tìm thấy thông tin về tất cả các thuộc tính Button trong tài liệu lớp Button và tất cả các thuộc tính TextView trong tài liệu lớp TextView .

Hiển thị tin nhắn Toast:

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. Để tạo một phiên bản của Toast , hãy làm theo các bước sau:

1. Gọi phương thức nhà máy makeText() trên lớp Toast.
2. Cung cấp ngữ cảnh của Ứng dụng Hoạt động và thông báo để hiển thị (chẳng hạn như tài nguyên chuỗi).
3. Cung cấp thời lượng hiển thị, ví dụ Toast.LENGTH_SHORT trong một khoảng thời gian ngắn. Thời lượng có thể là Toast.LENGTH_LONG hoặc Toast.LENGTH_SHORT
4. Hiển thị Toast bằng cách gọi show() .

Khái niệm liên quan

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

Tìm hiểu thêm

tài liệu dành cho nhà phát triển Android:

- Android Studio
- Build a UI with Layout Editor
- Build a Responsive UI with ConstraintLayout
- Layouts
- View
- Button
- TextView
- Android resources
- Android standard R.color resources
- Supporting Different Densities
- Android Input Events
- Context

Khác:

- Codelabs: Using ConstraintLayout to design your views
- Vocabulary words and concepts glossary

Lớp học lập trình tiếp theo là Android fundamentals 1.2 Part B: The layout editor .

Bài 1.2 Phần B: Trình soạn thảo bố cục

Giới thiệu

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

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

Thực hành này cũng giới thiệu hai lớp con ViewGroup khác:

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

Những điều bạn nên biết

Bạn sẽ có thể:

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

Những gì bạn sẽ học:

- Cách tạo biến thể bố cục cho hướng ngang (ngang).
- Cách tạo biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Cách sử dụng ràng buộc cơ sở để căn chỉnh các phần tử giao diện người dùng với văn bản.
- Cách sử dụng các nút gói và căn chỉnh để căn chỉnh các yếu tố trong bố cục.

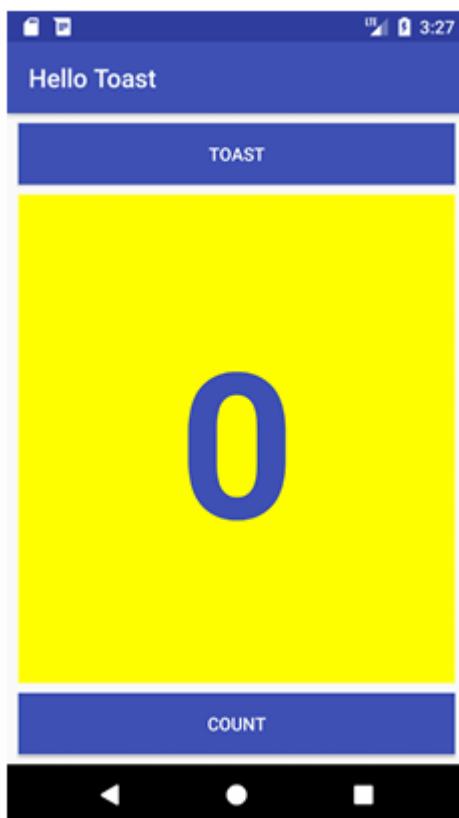
- Cách định vị chế độ xem trong LinearLayout.
- Cách định vị chế độ xem trong RelativeLayout.

Bạn sẽ làm gì

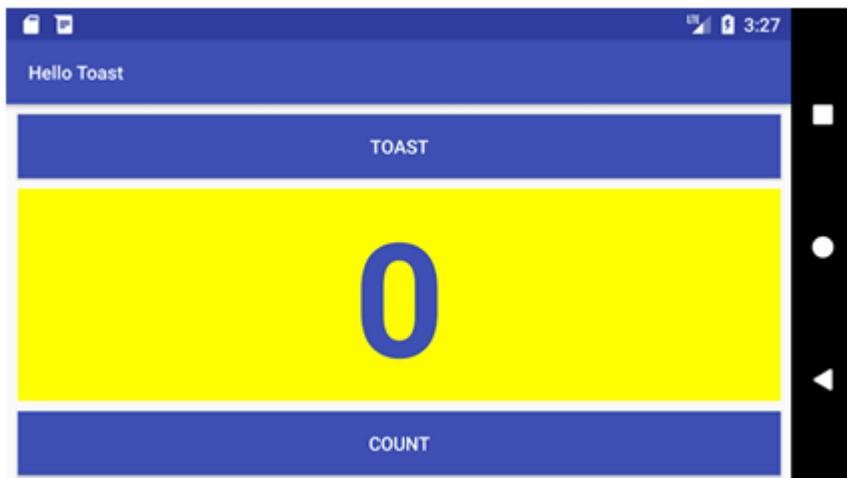
- Tạo biến thể bố cục cho hướng hiển thị ngang.
- Tạo biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Sửa đổi bố cục để thêm ràng buộc vào các phần tử giao diện người dùng.
- Sử dụng các ràng buộc đường cơ sở ConstraintLayout để căn chỉnh các phần tử với văn bản.
- Sử dụng gói ConstraintLayout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

Tổng quan về ứng dụng

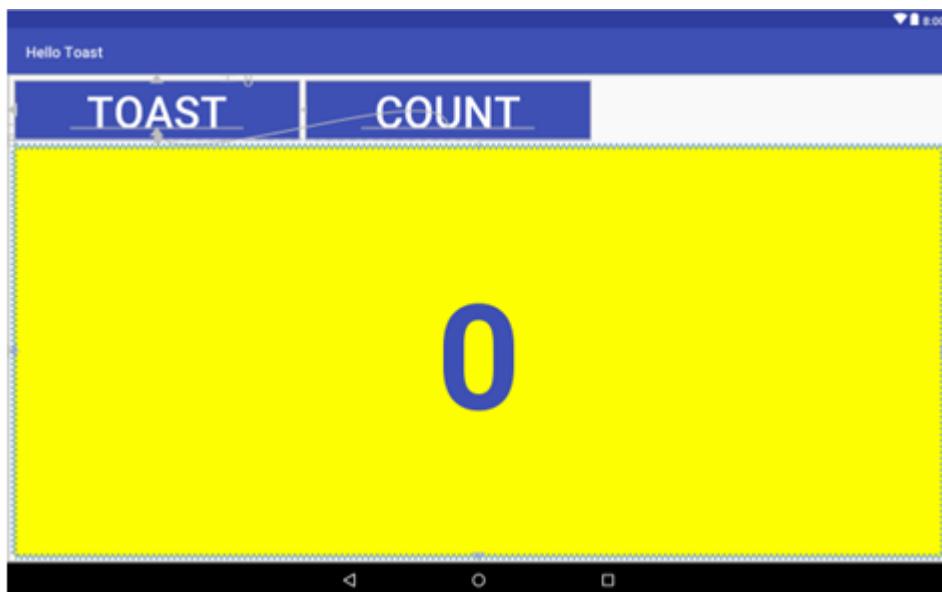
Ứng dụng Hello Toast trong bài học trước sử dụng ConstraintLayout để sắp xếp các phần tử giao diện người dùng trong bố cục Activity, như trong hình bên dưới.



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



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



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

Nhiệm vụ 1: Tạo biến thể bố cục

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

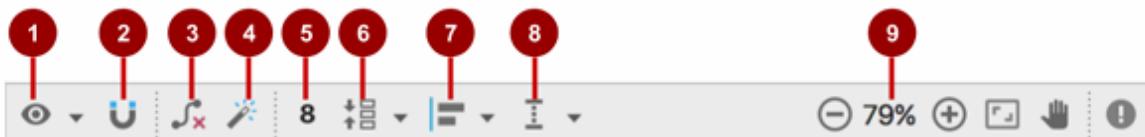
Trong nhiệm vụ này, bạn sẽ sử dụng một số nút trong hai thanh công cụ trên cùng của trình soạn thảo bố cục. Thanh công cụ trên cùng cho phép bạn định cấu hình giao diện của bản xem trước bố cục trong trình chỉnh sửa bố cục:



Trong hình trên:

1. Chọn Bè mặt thiết kế: Chọn Thiết kế để hiển thị bản xem trước màu sắc của bố cục của bạn hoặc Kế hoạch chi tiết để chỉ hiển thị đường viền cho từng phần tử giao diện người dùng. Để xem cả hai ngăn cạnh nhau, hãy chọn Thiết kế + Kế hoạch chi tiết .
2. Định hướng trong Trình chỉnh sửa : Chọn Dọc hoặc Ngang để hiển thị bản xem trước theo hướng dọc hoặc ngang. Điều này rất hữu ích để xem trước bố cục mà không cần phải chạy ứng dụng trên trình mô phỏng hoặc thiết bị. Để tạo bố cục thay thế, hãy chọn Tạo biến thể phong cảnh hoặc các biến thể khác.
3. Thiết bị trong Trình chỉnh sửa: Chọn loại thiết bị (điện thoại / máy tính bảng, Android TV hoặc Android Wear).
4. Phiên bản API trong Trình chỉnh sửa : Chọn phiên bản Android để sử dụng để hiển thị bản xem trước.
5. Chủ đề trong Trình chỉnh sửa : Chọn một chủ đề (chẳng hạn như AppTheme) để áp dụng cho bản xem trước.
6. Ngôn ngữ trong Trình chỉnh sửa : Chọn ngôn ngữ và ngôn ngữ cho bản xem trước. Danh sách này chỉ hiển thị các ngôn ngữ có sẵn trong tài nguyên chuỗi (xem bài học về bản địa hóa để biết chi tiết về cách thêm ngôn ngữ). Bạn cũng có thể chọn Xem trước dưới dạng Từ phải sang trái để xem bố cục như thẻ ngôn ngữ RTL đã được chọn

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



Trong hình trên:

1. Hiển thị: Chọn Hiển thị ràng buộc và Hiển thị lề để hiển thị chúng trong bản xem trước hoặc để ngừng hiển thị chúng.

2. Tự động kết nối: Bật hoặc tắt Tự động kết nối. Khi bật Tự động kết nối, bạn có thể kéo bất kỳ phần tử nào (chẳng hạn như Nút) vào bất kỳ phần nào của bố cục để tạo các ràng buộc đối với bố cục mẹ.

3. Xóa tất cả các ràng buộc: Xóa tất cả các ràng buộc trong toàn bộ bố cục.
4. Suy luận ràng buộc: Tạo ràng buộc bằng suy luận.
5. Lề mặc định: Đặt lề mặc định.
6. Đóng gói: Đóng gói hoặc mở rộng các yếu tố đã chọn.
7. Căn chỉnh: Căn chỉnh các phần tử đã chọn.
8. Hướng dẫn: Thêm hướng dẫn đọc hoặc ngang.
9. Điều khiển thu phóng / xoay: Phóng to hoặc thu nhỏ.\

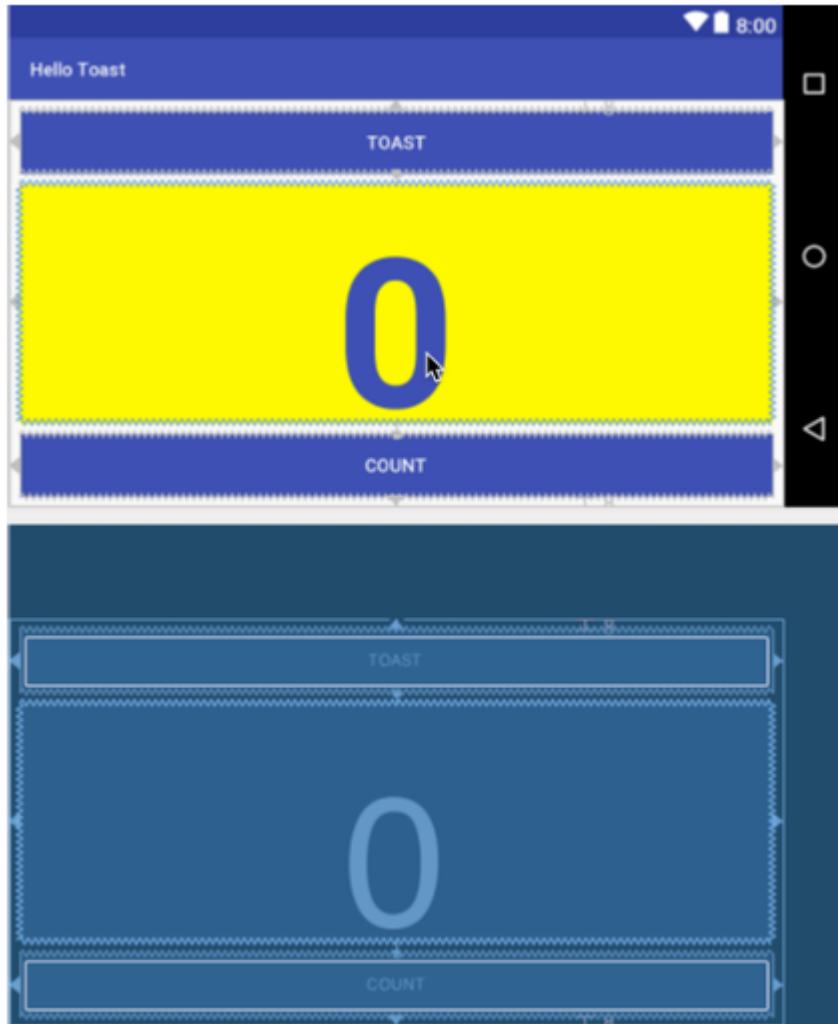
Mẹo: Để tìm hiểu thêm về cách sử dụng trình chỉnh sửa bố cục, hãy xem phần *Tạo giao diện người dùng bằng Trình chỉnh sửa bố cục*. Để tìm hiểu thêm về cách tạo bố cục bằng *ConstraintLayout*, hãy xem bài viết *Tạo giao diện người dùng ứng dụng bằng ConstraintLayout*.

1.1 Xem trước bố cục theo hướng ngang

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

1. Mở ứng dụng Hello Toast từ bài học trước. Lưu ý: Nếu bạn đã tải xuống mã giải pháp cho HelloToast, bạn cần xóa các bố cục ngang và cực lớn đã hoàn thành mà bạn sẽ tạo trong tác vụ này. Chuyển từ Project > Android sang Project > Project Files trong ngăn Project, mở rộng ứng dụng > ứng dụng > src/main > res , chọn cả thư mục layout-land và thư mục layout-xlarge và chọn Edit > Delete . Sau đó, chuyển ngăn Dự án trở lại Dự án > Android .

2. Mở tệp bố cục activity_main.xml. Nhấp vào tab Thiết kế nếu nó chưa được chọn.
3. Nhấp vào nút Định hướng trong Trình chỉnh sửa trên thanh công cụ trên cùng.
4. Chọn Chuyển sang ngang trong menu thả xuống. Bố cục xuất hiện theo hướng ngang như hình dưới đây. Để quay lại hướng dọc, hãy chọn Chuyển sang Dọc .



1.2 Tạo biến thể bối cảnh cho hướng ngang

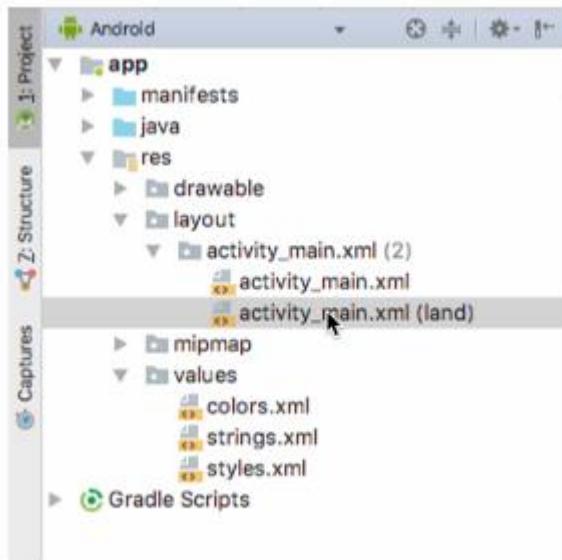
Sự khác biệt trực quan giữa hướng dọc và chiều ngang cho bối cảnh này là chữ số (0) trong phần tử TextView show_count quá thấp so với hướng ngang — quá gần với

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

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

1. Nhấp vào nút Định hướng trong Trình chỉnh sửa trên thanh công cụ trên cùng.
2. Chọn Tạo biến thể phong cảnh. Một cửa sổ trình chỉnh sửa mới mở ra với tab đất liền / activity_main.xml hiển thị bối cảnh cho hướng ngang (ngang). Bạn có thể thay đổi bối cảnh này, dành riêng cho hướng ngang, mà không cần thay đổi hướng dọc (dọc) ban đầu.

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



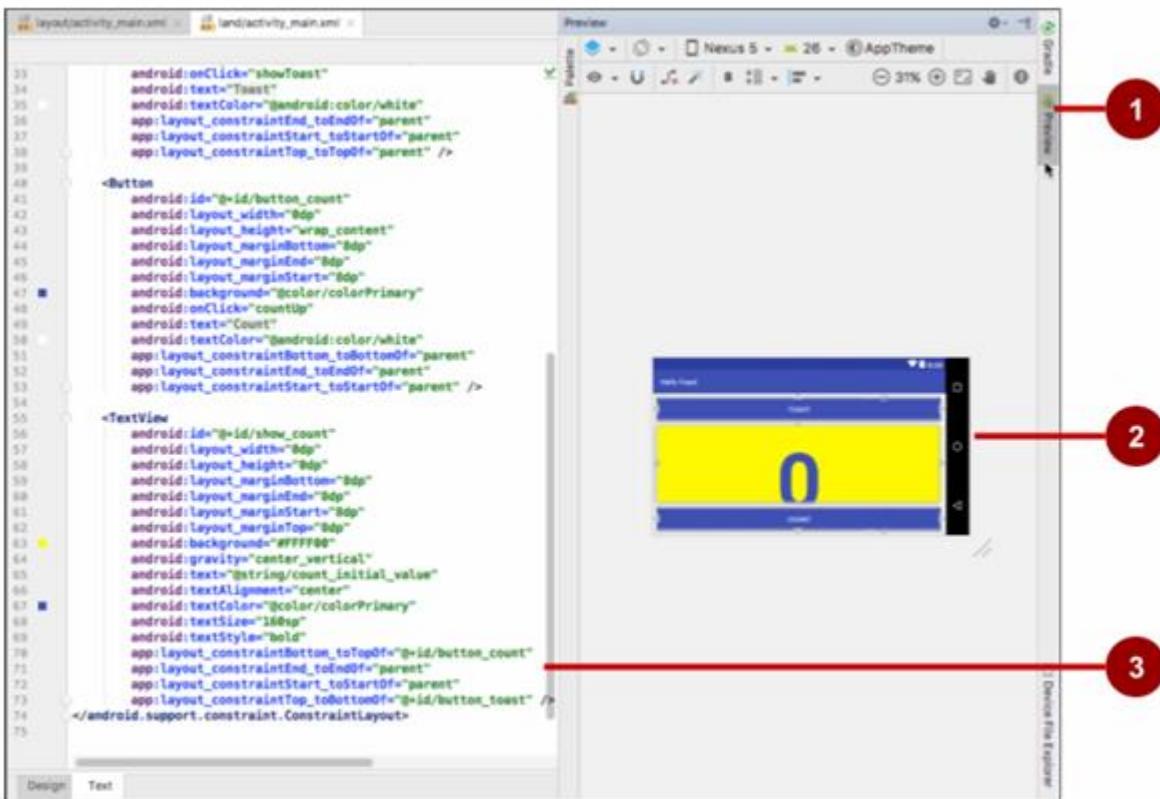
1.3 Xem trước bố cục cho các thiết bị khác nhau

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 mô phỏng. Làm theo các bước sau:

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

1.4 Thay đổi bố cục cho hướng ngang

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

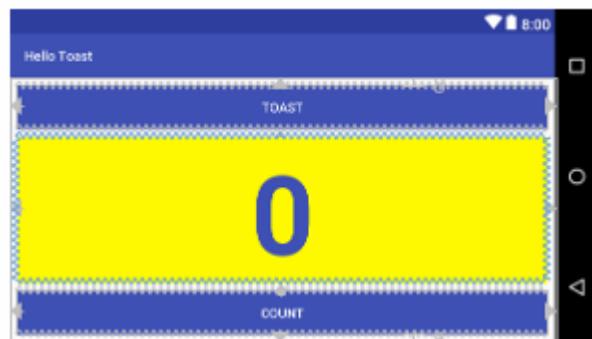


Hình trên cho thấy những điều sau:

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

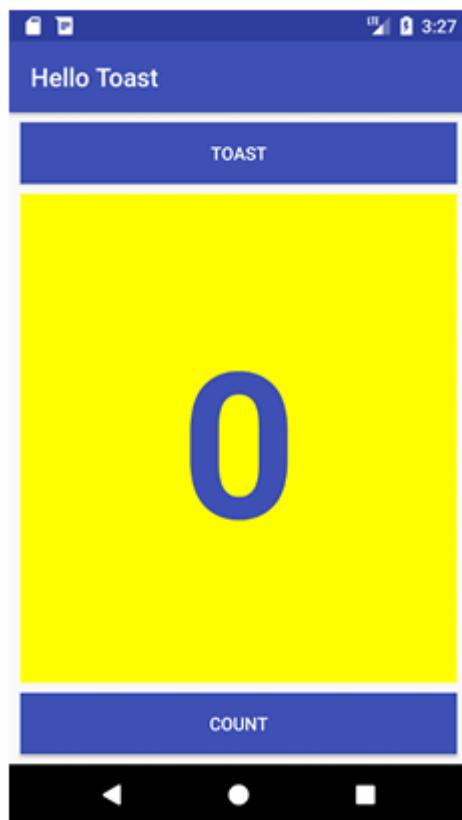
Để thay đổi bố cục, hãy làm theo các bước sau:

1. Tab đât / `activity_main.xml` vẫn sẽ được mở trong trình chỉnh sửa bố cục; Nếu không, hãy nhấp đúp vào tệp `activity_main.xml` (đất) trong thư mục bố cục.
2. Nhập vào tab Văn bản và tab Xem trước (nếu chưa chọn).
3. Tìm phần tử `TextView` trong mã XML.
4. Thay đổi thuộc tính `android:textSize="160sp"` thành `android:textSize="120sp"`. Bản xem trước bố cục hiển thị kết quả:



5. Chọn các thiết bị khác nhau trong menu thả xuống Thiết bị trong Trình chỉnh sửa để xem bố cục trông như thế nào trên các thiết bị khác nhau theo hướng ngang. Trong ngăn trình chỉnh sửa, tab đât/activity_main.xml hiển thị bố cục cho hướng ngang. Tab activity_main.xml hiển thị bố cục không thay đổi cho hướng dọc. Bạn có thể chuyển đổi qua lại bằng cách nhấp vào các tab.

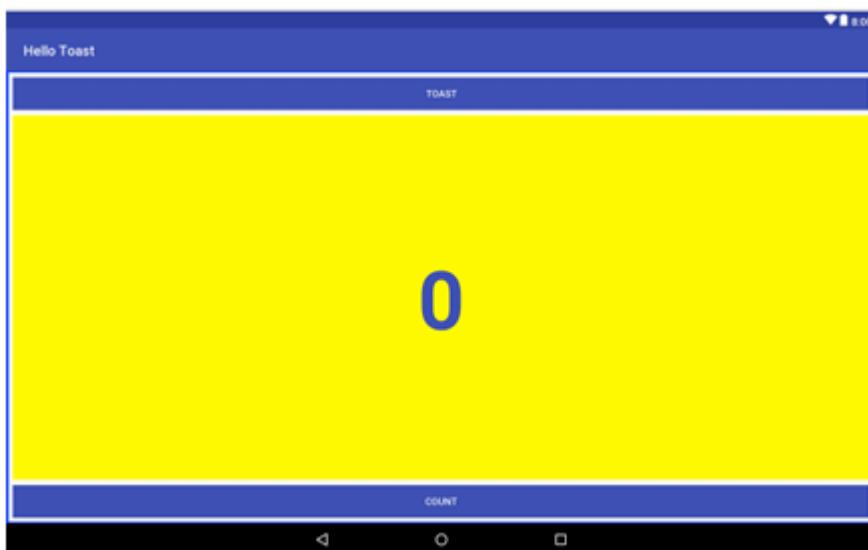
6. Chạy ứng dụng trên trình mô phỏng hoặc thiết bị và chuyển hướng từ dọc sang ngang để xem cả hai bố cục.





1.5 Tạo biến thể bố cục cho máy tính bảng

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



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

1. Nhấp vào tab Thiết kế (nếu chưa chọn) để hiển thị các ngăn thiết kế và bản thiết kế.
2. Nhấp vào nút Orientation in Editor trên thanh công cụ trên cùng.
3. Chọn Tạo bố cục x-large Variation .

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

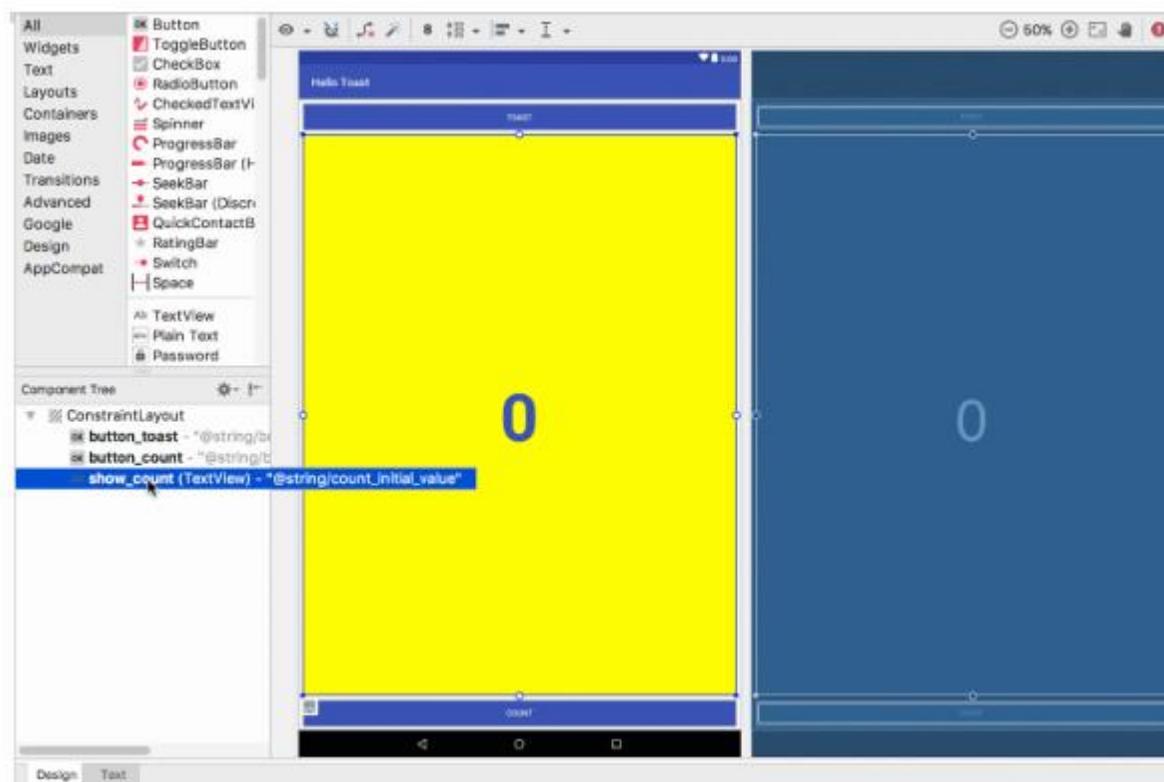
1.6 Thay đổi biến thể bố cục cho máy tính bảng

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

1. Tắt công cụ Tự động kết nối trên thanh công cụ. Đối với bước này, hãy đảm bảo rằng công cụ đã bị tắt:

2. Xóa tất cả các ràng buộc trong bố cục bằng cách nhấp vào nút Xóa tất cả các ràng buộc trên thanh công cụ. Với các ràng buộc được loại bỏ, bạn có thể di chuyển và thay đổi kích thước các phần tử trên bố cục một cách tự do.

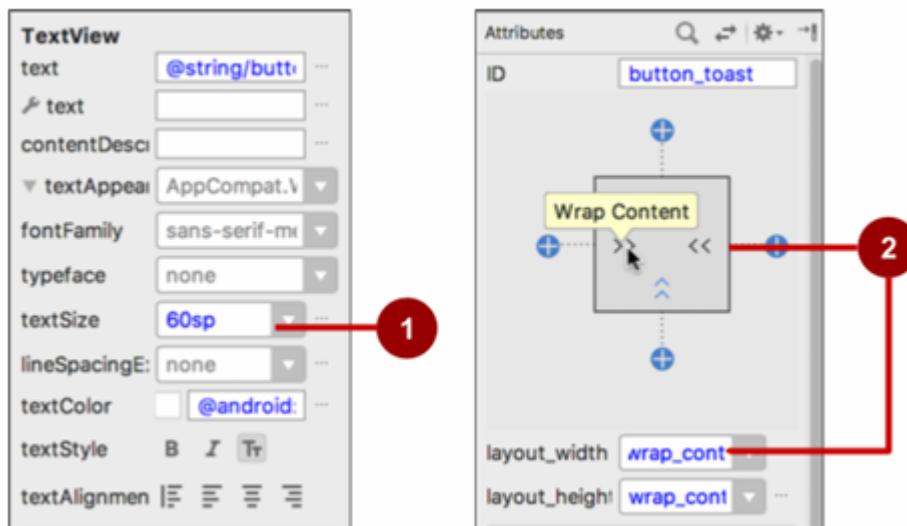
3. Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước trên cả bốn góc của một phần tử để thay đổi kích thước nó. Trong Component Tree, chọn `TextView` được gọi là `show_count`. Để loại bỏ `TextView` để bạn có thể tự do kéo các phần tử `Button`, hãy kéo một góc của nó để thay đổi kích thước,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



Thay đổi kích thước phần tử mã hóa cứng kích thước chiều rộng và chiều cao. Tránh mã hóa cứng kích thước cho hầu hết các yếu tố, vì bạn không thể dự đoán kích thước được mã hóa cứng sẽ trông như thế nào trên màn hình có kích thước và

mật độ khác nhau. Bạn đang làm điều này bây giờ chỉ để di chuyển phần tử ra khỏi đường đi và bạn sẽ thay đổi kích thước trong một bước khác.

4. Chọn nút button_toast trong Cây thành phần , nhấp vào Thuộc tính tab để mở ngăn Thuộc tính và thay đổi textSize thành 60sp (# 1 trong hình bên dưới) và layout_width thành wrap_content (# 2 trong hình bên dưới).



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

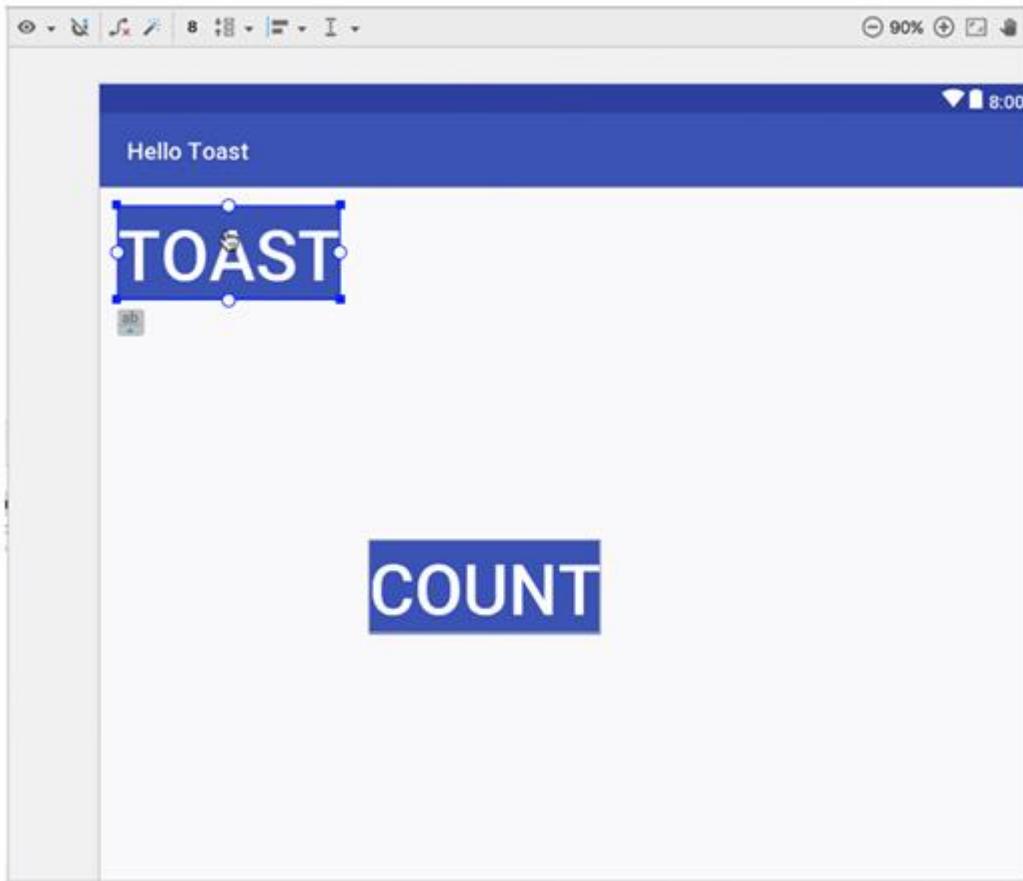
Bạn sử dụng wrap_content để nếu văn bản Button được bắn địa hóa sang một ngôn ngữ khác, Button sẽ xuất hiện rộng hơn hoặc mỏng hơn để phù hợp với từ trong ngôn ngữ khác.

5. Chọn nút button_count trong Cây thành phần , thay đổi textSize thành 60sp và layout_width thành wrap_content và kéo Button phía trên TextView đến một khoảng trống trong bố cục.

1.7 Sử dụng ràng buộc cơ sở

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

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



2. Sử dụng ràng buộc đường cơ sở , bạn có thể hạn chế Nút button_count để đường cơ sở văn bản của nó khớp với đường cơ sở văn bản của nút button_toast . Chọn phần tử button_count, sau đó di con trỏ của bạn lên phần tử cho đến khi nút ràng buộc đường cơ sở xuất hiện bên dưới phần tử.

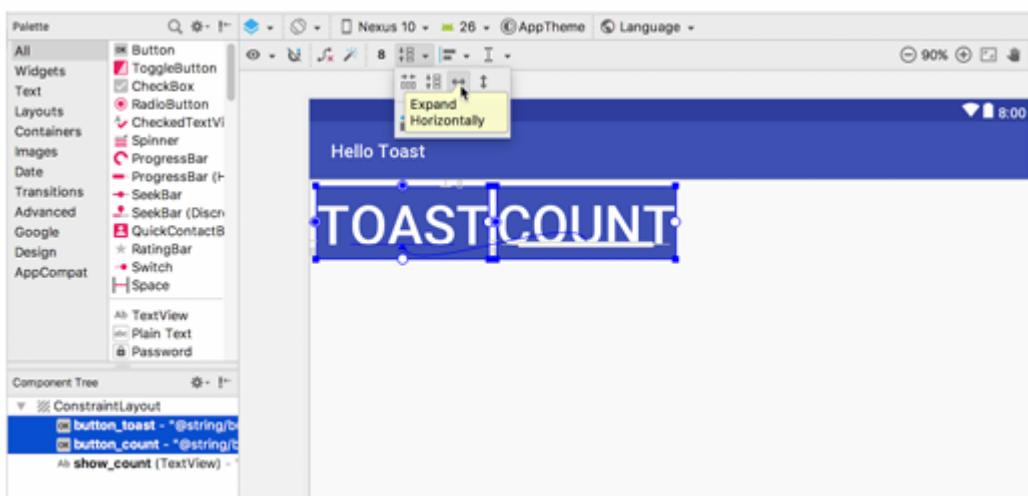
3. Nhấp vào nút ràng buộc đường cơ sở. Tay cầm cơ sở xuất hiện, nhấp nháy màu xanh lục như trong hình động. Nhấp và kéo một đường ràng buộc đường cơ sở vào đường cơ sở của phần tử button_toast.

1.8 Mở rộng các nút theo chiều ngang

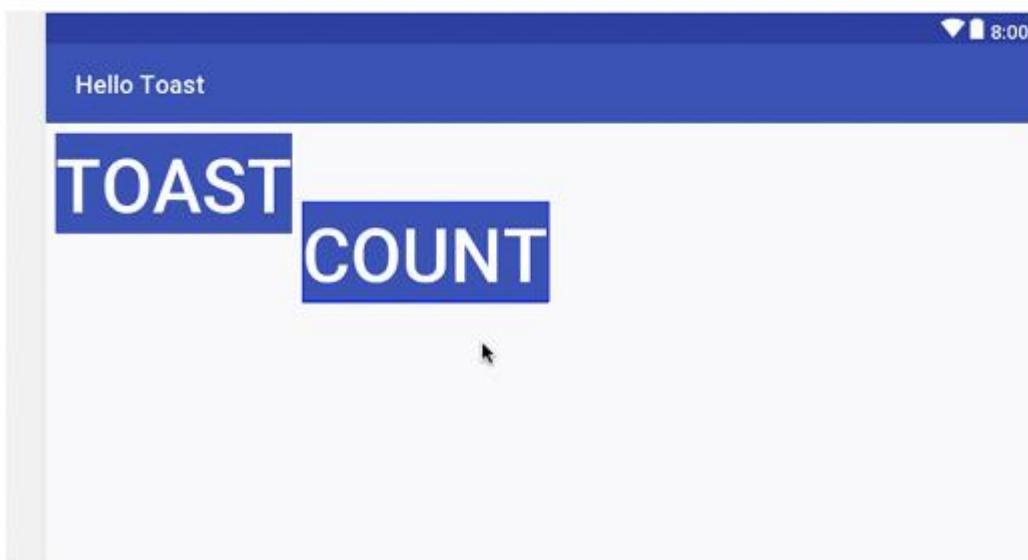
Nút đóng gói trên thanh công cụ cung cấp các tùy chọn để đóng gói hoặc mở rộng các phần tử giao diện người dùng đã chọn. Bạn có thể sử dụng nó để sắp xếp các phần tử Button theo chiều ngang trên bố cục.

1. Chọn Nút button_count trong Cây thành phần và Shift-chọn Nút button_toast để cả hai đều được chọn.

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



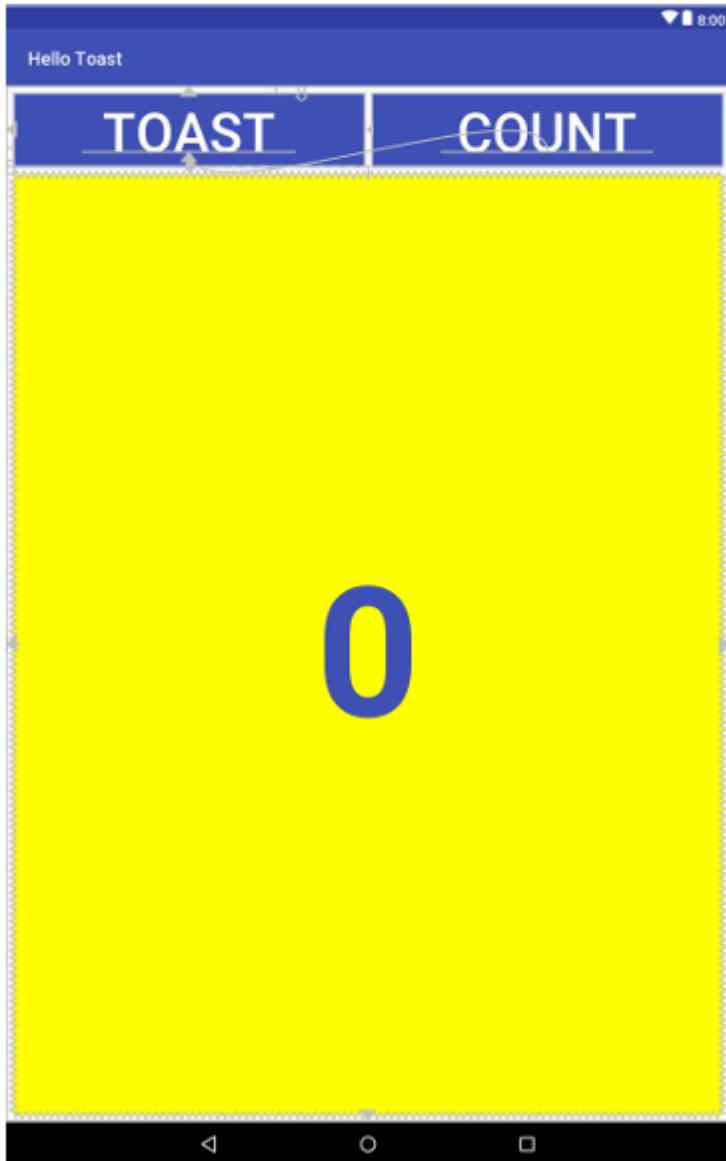
Các phần tử Button mở rộng theo chiều ngang để lấp đầy bố cục như hình dưới đây.



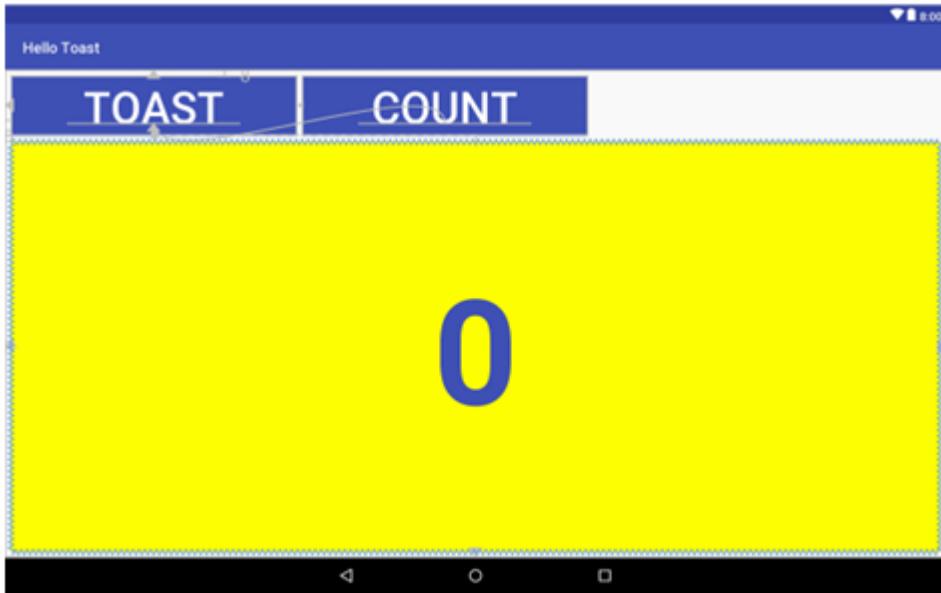
3. Để hoàn thành bố cục, hãy ràng buộc show_count TextView ở dưới cùng của Nút button_toast và hai bên và dưới cùng của bố cục, như trong hình động bên dưới.



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



5. Nhấp vào nút Orientation in Editor ở 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*.

code nhiệm vụ 1

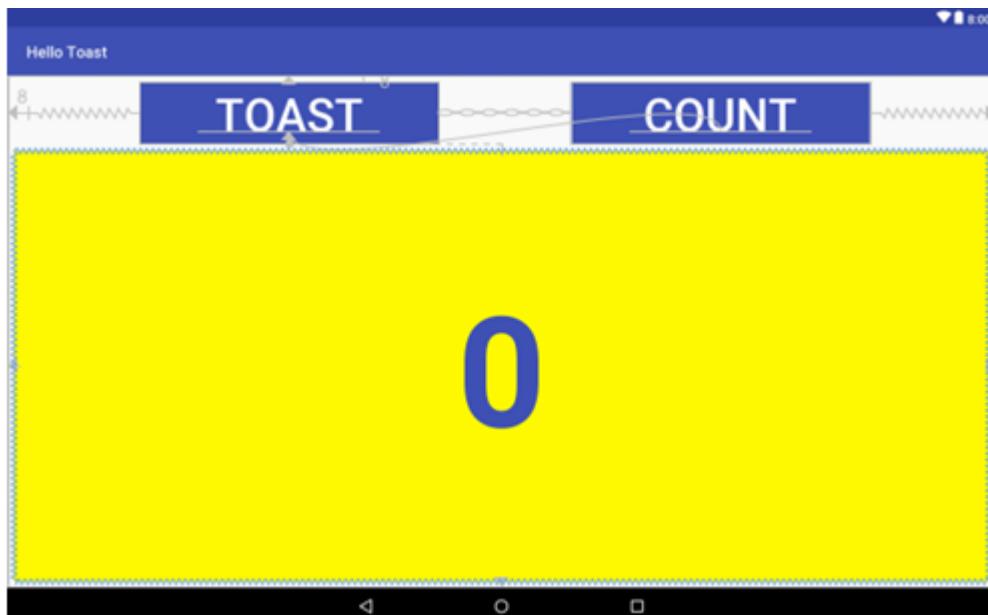
dự án Android Studio: HelloToast

Thử thách lập trình 1

Note: All coding challenges are optional and are not prerequisites for later lessons.

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.



code thử thách 1

dự án Android Studio: HelloToastChallenge2

Nhiệm vụ 2: Thay đổi bố cục thành LinearLayout

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

LinearLayout là bắt buộc phải có các thuộc tính sau:

- layout_width
- layout_height
- định hướng

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

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

Định hướng có thể là:

- ngang: Views được sắp xếp từ trái sang phải.
- dọc: Views được sắp xếp từ trên xuống dưới.

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

2.1 Thay đổi nhóm chế độ xem gốc thành LinearLayout

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở tệp bố cục `activity_main.xml` (nếu nó chưa được mở) và nhấp vào tab Văn bản ở cuối ngăn chỉnh sửa để xem mã XML. Ở trên cùng của mã XML là dòng thẻ sau:

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

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

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

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

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

android:orientation="vertical"

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

2.2 Thay đổi thuộc tính phần tử cho LinearLayout

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

1. Mở ứng dụng Hello Toast từ tác vụ trước.
2. Mở tệp bố cục activity_main.xml (nếu nó chưa được mở) và nhấp vào Văn bản chuyển hướng.
3. Tìm phần tử button_toast Button và thay đổi thuộc tính sau:

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

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

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

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

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

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

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

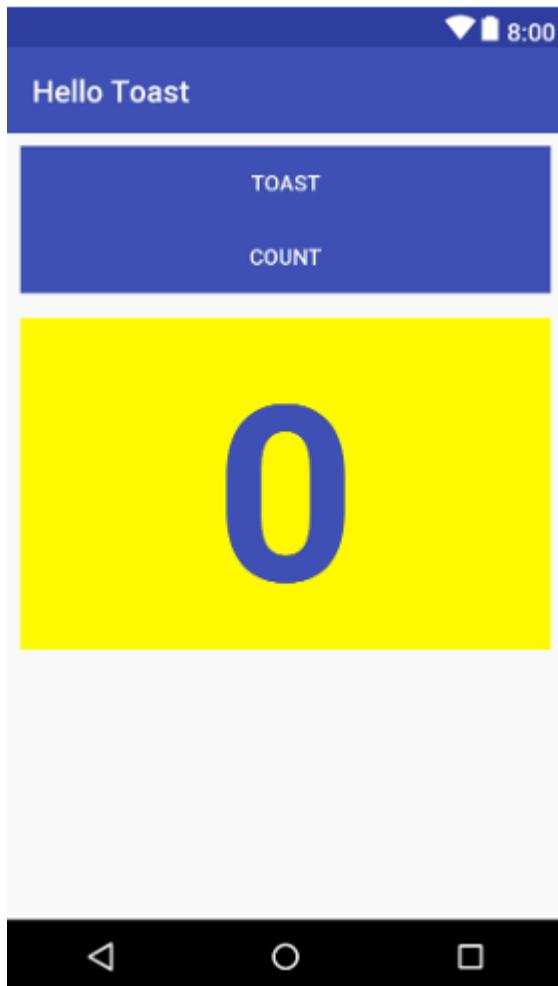
7. Tìm phần tử TextView show_count và thay đổi các thuộc tính sau:

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

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

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

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



2.3 Thay đổi vị trí của các phần tử trong LinearLayout

LinearLayout sắp xếp các phần tử của nó trong một hàng ngang hoặc dọc. Bạn đã thêm thuộc tính `android:orientation="vertical"` cho LinearLayout, vì vậy các phần tử được xếp chồng lên nhau theo chiều dọc như trong hình trước. Để thay đổi vị trí của chúng sao cho nút Đếm ở dưới cùng, hãy làm theo các bước sau:

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở tệp bố cục `activity_main.xml` (nếu nó chưa mở) và nhấp vào Văn bản chuyển hướng.

3. Chọn nút button_count và tất cả các thuộc tính của nó, từ thẻ <Button> đến và bao gồm thẻ /> đóng và chọn **Chỉnh sửa** > **Cắt**.
4. Nhấp sau thẻ đóng /> của phần tử TextView nhưng trước thẻ đóng và </LinearLayout> chọn **Chỉnh sửa** > **Dán**.
5. (Tùy chọn) Để khắc phục bất kỳ vấn đề thụt lề hoặc khoảng cách nào cho mục đích thẩm mỹ, hãy chọn **Mã** > **Định dạng lại Mã** để định dạng lại mã XML với khoảng cách và thụt lề thích hợp.

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

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

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

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

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

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



2.4 Thêm trọng lượng cho phần tử TextView

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

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

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

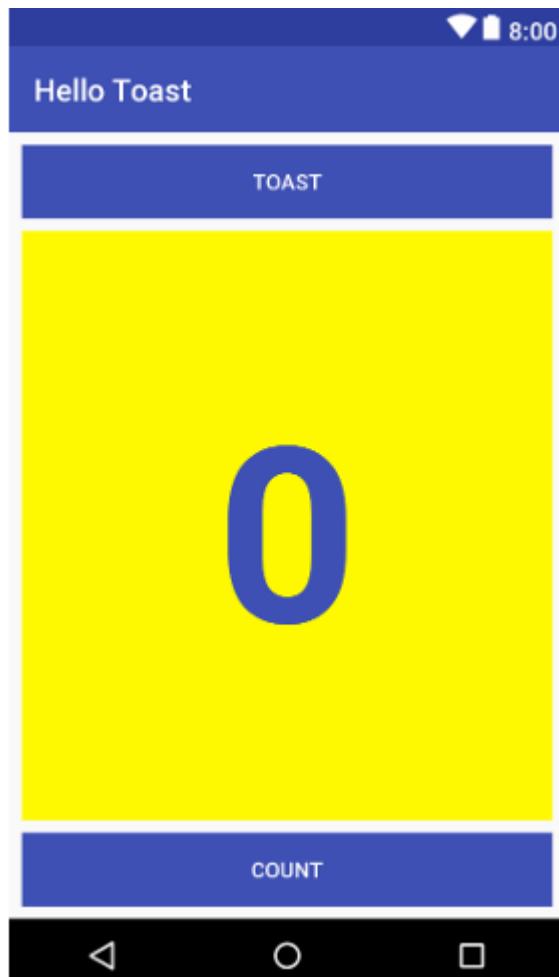
Thuộc tính android:layout_weight cho biết bao nhiêu không gian bổ sung trong LinearLayout sẽ được phân bổ cho View. Nếu chỉ có một Chế độ xem có thuộc tính này, nó sẽ nhận được tất cả không gian màn hình bổ sung. Đối với nhiều phần tử View, không gian được chia theo tỷ lệ. Ví dụ: nếu mỗi phần tử Button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

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

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

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

Bản xem trước bây giờ trông giống như hình sau:



Phần tử TextView show_count chiếm tất cả không gian giữa các nút. Bạn có thể xem trước bối cảnh cho các thiết bị khác nhau, như bạn đã làm trong tác vụ trước bằng cách nhấp vào nút Thiết bị trong Trình chỉnh sửa Nexus 5 ▾ trên thanh công cụ trên cùng của ngăn xem trước và chọn một thiết bị khác. Bất kể bạn chọn thiết bị nào để xem trước, phần tử TextView show_count sẽ chiếm tất cả không gian giữa các nút.

Mã giải pháp nhiệm vụ 2

Mã XML trong activity_main.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context="com.example.android.hellotoast.MainActivity">

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

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

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

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

Nhiệm vụ 3: Thay đổi bố cục thành RelativeLayout

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

3.1 Thay đổi LinearLayout thành RelativeLayout

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

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

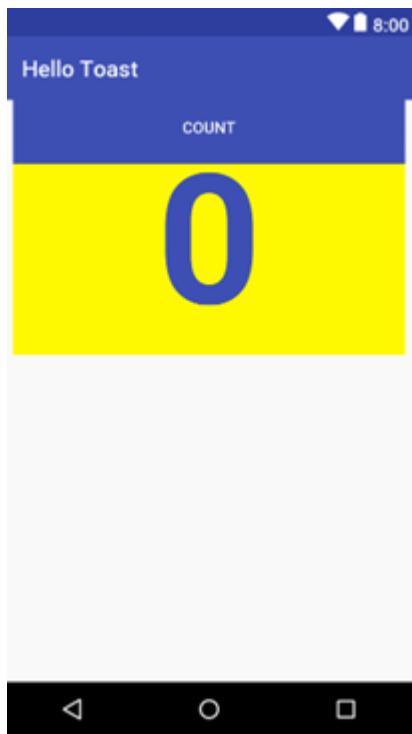
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

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

3.2 Sắp xếp lại các chế độ xem trong RelativeLayout

Một cách dễ dàng để sắp xếp lại và định vị các chế độ xem trong RelativeLayout là thêm các thuộc tính XML vào tab Text.

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



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

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

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

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

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

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

Mã XML đầy đủ cho Nút button_count như sau:

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

4. Thêm các thuộc tính sau vào TextView show_count

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

android:layout_alignParentLeft căn chỉnh chế độ xem ở phía bên trái của nhóm chế độ xem cha RelativeLayout. Mặc dù bản thân thuộc tính này là đủ để căn chỉnh chế độ xem ở phía bên trái, bạn có thể muốn chế độ xem căn chỉnh ở phía bên phải nếu ứng dụng đang chạy trên thiết bị đang sử dụng ngôn ngữ từ phải sang trái. Do đó, thuộc tính android:layout_alignParentStart làm cho cạnh "bắt đầu" của chế độ xem này khớp với cạnh bắt đầu của cha mẹ. Bắt đầu là cạnh trái của màn hình nếu tùy chọn là từ trái sang phải hoặc đó là cạnh phải của màn hình nếu tùy chọn là từ phải sang trái.

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



Mẹo: RelativeLayout giúp bạn dễ dàng nhanh chóng đặt các yếu tố giao diện người dùng trong bố cục. Để tìm hiểu thêm về cách định vị chế độ xem trong RelativeLayout, hãy xem "Định vị chế độ xem" trong chủ đề "Bố cục tương đối" của Hướng dẫn API.

Mã giải pháp nhiệm vụ 3

Mã XML trong activity_main.xml :

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

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

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

```
<Button
    android:id="@+id/button_count"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginBottom="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:background="@color/colorPrimary"
    android:onClick="countUp"
    android:text="@string/button_label_count"
    android:textColor="@android:color/white"
    android:layout_below="@+id/show_count"
    android:layout_centerHorizontal="true"/>
</RelativeLayout>
```

Tóm tắt

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

- Để xem trước bố cục ứng dụng với hướng ngang trong trình chỉnh sửa bố cục, hãy nhấp vào nút Định hướng trong Trình chỉnh sửa  trên thanh công cụ trên cùng và chọn Chuyển sang ngang . Chọn Chuyển sang Đọc để quay lại hướng đọc.
- Để tạo biến thể của bố cục khác nhau cho hướng ngang, hãy nhấp vào nút Định hướng trong Trình chỉnh sửa và chọn Tạo biến thể phong cảnh. Một cửa sổ trình chỉnh sửa mới mở ra với tab đất liền / activity_main.xml hiển thị bố cục cho hướng ngang (ngang).
- Để xem trước bố cục ứng dụng với hướng ngang trong trình chỉnh sửa bố cục, hãy nhấp vào nút Định hướng trong Trình chỉnh sửa  Nexus 5  thiết bị. trên thanh công cụ trên cùng và chọn một
- Để tạo biến thể của bố cục khác cho máy tính bảng (màn hình lớn hơn), nhấp vào nút Định hướng trong Trình chỉnh sửa và chọn Tạo bố cục x-large Biến thể . Một biên tập viên mới cửa sổ mở ra với tab Xlarge/activity_main.xml hiển thị bố cục cho thiết bị có kích thước bảng máy tính bảng.

dụng ConstraintLayout: • Để xóa tất cả các ràng buộc trong bố cục có gốc ConstraintLayout, hãy nhấp vào nút Xóa tất cả các ràng buộc  trên thanh công cụ.

- Bạn có thể căn chỉnh một phần tử giao diện người dùng có chứa văn bản, chẳng hạn như TextView hoặc Nút, với một phần tử giao diện người dùng khác có chứa văn bản. Một ràng buộc đường cơ sở cho phép bạn hạn chế các phần tử để đường cơ sở văn bản khớp nhau.
- Để tạo ràng buộc đường cơ sở, hãy di con trỏ của bạn qua phần tử giao diện người dùng cho đến khi nút ràng buộc đường cơ sở  xuất hiện bên dưới phần tử.

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

Sử dụng LinearLayout :

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

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

Sử dụng RelativeLayout:

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

Các khái niệm liên quan:

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

Tìm hiểu thêm

Tài liệu về Android Studio:

- Gặp gỡ Android Studio

- Tạo biểu tượng ứng dụng bằng Image Asset Studio

Tài liệu dành cho nhà phát triển Android:

- Bố cục

- Xây dựng giao diện người dùng bằng Trình chỉnh sửa bố cục

- Xây dựng giao diện người dùng đáp ứng với ConstraintLayout

- Bố cục

- Bố cục tuyến tính

- RelativeLayout

- Chế độ xem

- Nút

- TextView

- Hỗ trợ các mật độ pixel khác nhau

Khác:

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

- Bảng thuật ngữ từ vựng và khái niệm

Bài tập về nhà

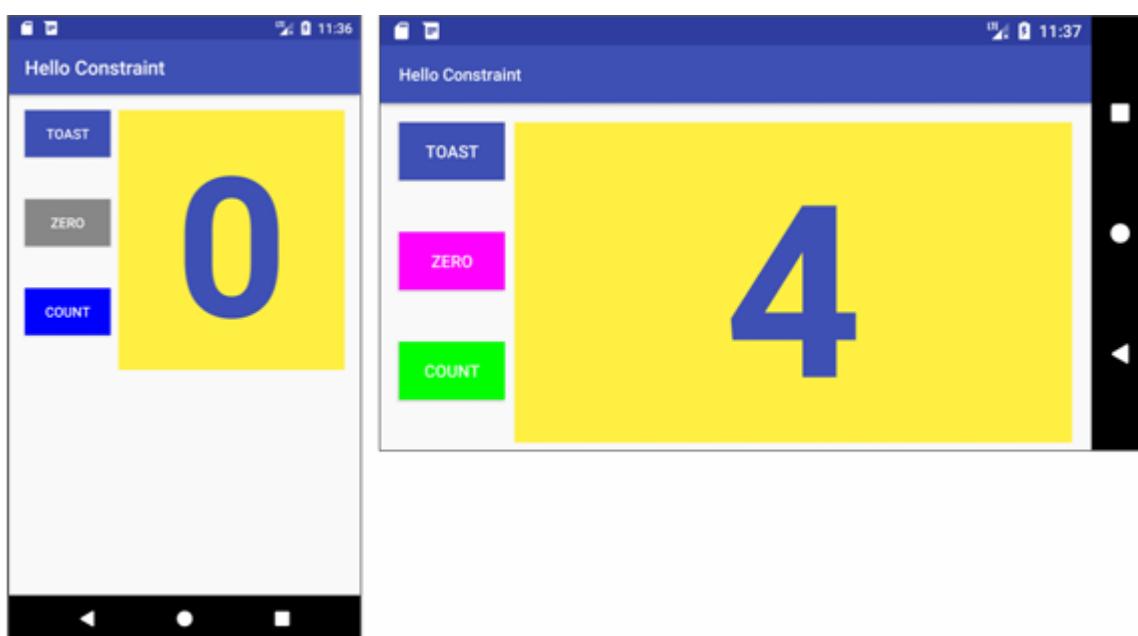
Thay đổi ứng dụng

Mở ứng dụng HelloToast.

1. Thay đổi tên của dự án thành HelloConstraint và tái cấu trúc dự án thành Hello Constraint . (F hoặc hướng dẫn về cách sao chép và tái cấu trúc dự án, xem Phụ lục: Tiện ích .)

2. Sửa đổi bố cục activity_main.xml để căn chỉnh các phần tử Toast và Count Button dọc theo phía bên trái của show_count TextView hiển thị "0".
Tham khảo các hình dưới đây để biết bố cục.

3. Bao gồm một nút thứ ba có tên là Zero xuất hiện giữa các phần tử Toast và Count Button.
4. Phân phối các phần tử Button theo chiều dọc giữa trên cùng và dưới cùng của TextView show_count .
5. Đặt Nút Zero ban đầu có nền màu xám.
6. Đảm bảo rằng bạn bao gồm Nút Zero cho hướng ngang trong activity_main.xml (đất liền) và cả cho màn hình có kích thước bằng máy tính bảng ở activity_main (xlarge).
7. Làm cho Nút Zero thay đổi giá trị trong TextView show_count thành 0.
8. Cập nhật trình xử lý nhấp chuột cho Nút đếm để nó thay đổi màu nền của riêng nó, tùy thuộc vào việc số mới là lẻ hay chẵn. Gợi ý: Không sử dụng findViewById để tìm nút đếm . Bạn có thể sử dụng thứ gì khác không? Hãy thoải mái sử dụng các hằng số trong lớp Color cho hai màu nền khác nhau.
9. Cập nhật trình xử lý nhấp chuột cho Nút đếm để đặt màu nền cho Nút Zero thành màu khác ngoài màu xám để cho biết 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 thành màu xám, sao cho màu xám khi số đếm bằng không.



Trả lời những câu hỏi này

Câu hỏi 1

Hai thuộc tính ràng buộc bối cục nào trên Zero Button đặt nó theo chiều dọc bằng khoảng cách giữa hai phần tử Button còn lại? (Chọn 2 câu trả lời.)

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

Câu hỏi 2: Thuộc tính ràng buộc bối cục nào trên Zero Button định vị nó theo chiều ngang phù hợp với hai phần tử Button còn lại?

- ứng dụng: layout_constraintLeft_toLeftOf = "cha mẹ"
- ứng dụng: layout_constraintBottom_toTopOf = "@ + id / button_count"
- android: layout_marginBottom = "8dp"
- ứng dụng: 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(Xem chế độ xem)
- private void callMethod(Xem chế độ xem)
- public boolean callMethod(Xem chế độ xem)

Câu hỏi 4 Trình xử lý nhấp chuột cho Nút đếm 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 đếm. 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 (đất) và activity_main.xml (xlarge).
- Ứng dụng bao gồm việc triển khai phương pháp 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 pháp xử lý nhấp chuột cho nút Đếm 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.

Bài 1.3: Chế độ xem văn bản và cuộn

Giới thiệu

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

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

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

Lớp ScrollView cung cấp bố cục cho chế độ xem cuộn. ScrollView là một lớp con của FrameLayout. Chỉ đặt một chế độ xem làm dạng xem con trong đó — một chế độ xem con chứa toàn bộ nội dung cần cuộn. Bản thân chế độ xem con này có thể là một ViewGroup (chẳng hạn như LinearLayout) chứa các phần tử giao diện người dùng.

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

Với ScrollView, tất cả các thành phần giao diện người dùng đều nằm trong bộ nhớ và trong hệ thống phân cấp chế độ xem ngay cả khi chúng không được hiển thị trên màn hình. Điều này làm cho ScrollView trở nên lý tưởng để cuộn các trang văn bản dạng tự do một cách mượt mà, vì văn bản đã có trong bộ nhớ. Tuy nhiên, ScrollView có thể sử dụng nhiều bộ nhớ, điều này

có thể ảnh hưởng đến hiệu suất của phần còn lại của ứng dụng. Để hiển thị danh sách dài các mục mà người dùng có thể thêm, xóa hoặc chỉnh sửa, hãy cân nhắc sử dụng RecyclerView , được mô tả trong một bài học riêng.

Những điều bạn nên biết

Bạn sẽ có thể:

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

Những gì bạn sẽ học

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

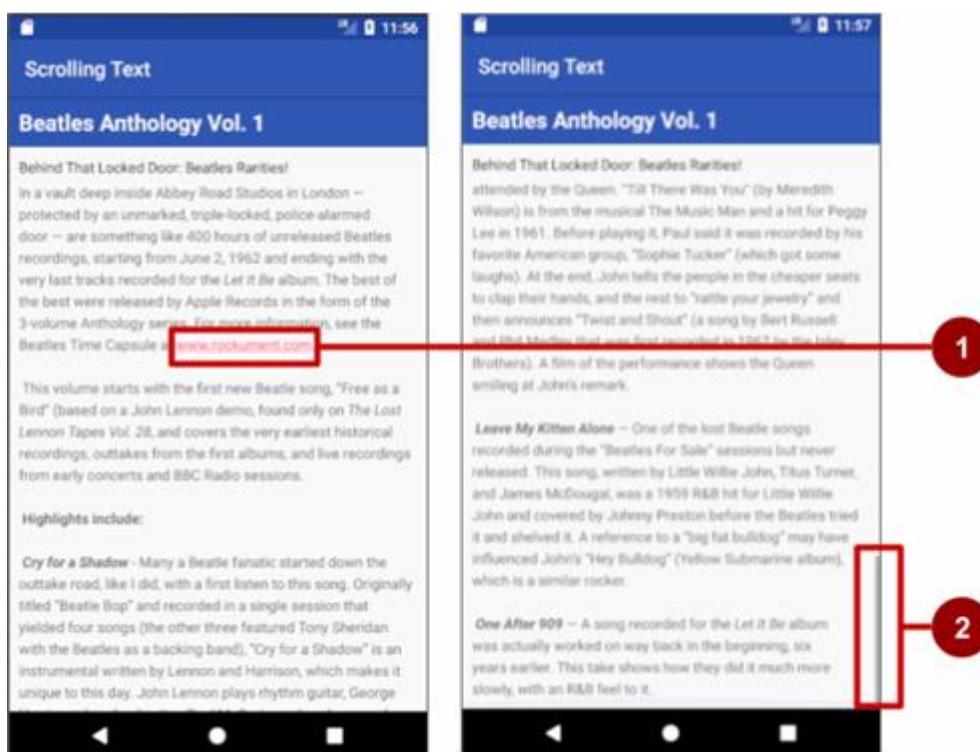
Những gì bạn sẽ làm

- Tạo ứng dụng ScrollingText.
- Thay đổi ConstraintLayout ViewGroup thành RelativeLayout .
- Thêm hai phần tử TextView cho tiêu đề và tiêu đề phụ của bài viết.
- Sử dụng kiểu và màu TextAppearance cho tiêu đề và tiêu đề phụ của bài viết.
- Sử dụng thẻ HTML trong chuỗi văn bản để kiểm soát định dạng.

- Sử dụng thuộc tính lineSpacingExtra để thêm khoảng cách dòng để dễ đọc.
- Thêm ScrollView vào bố cục để cho phép cuộn phần tử TextView.
- Thêm thuộc tính autoLink để cho phép các URL trong văn bản hoạt động và có thể nhấp vào.

Tổng quan về ứng dụng

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

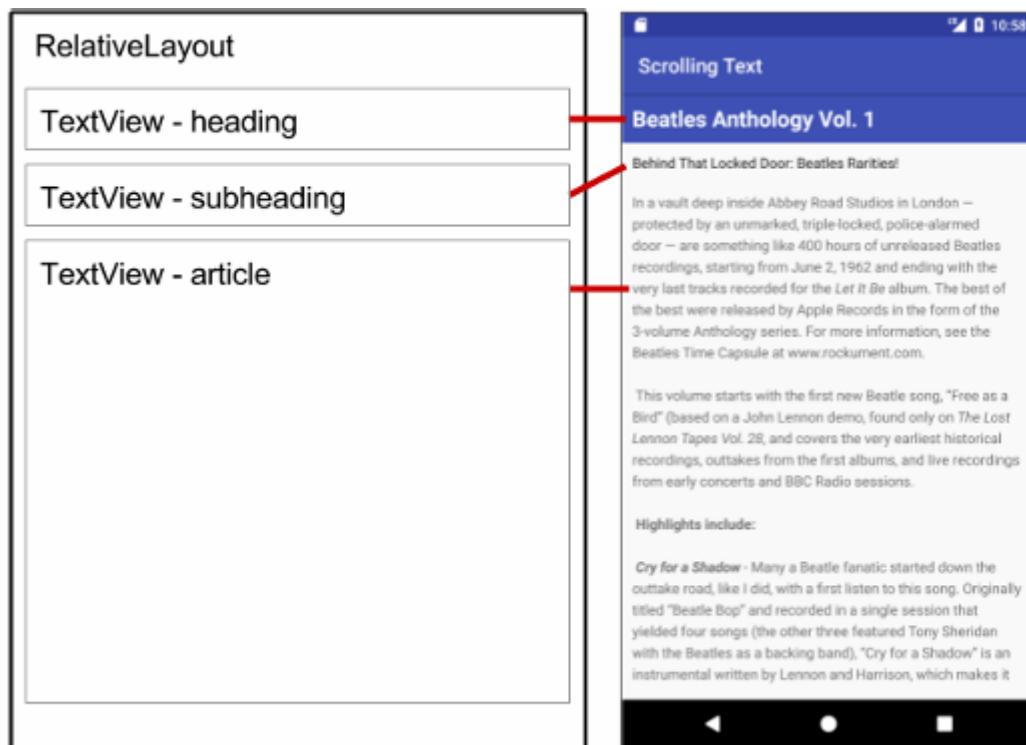


Trong hình trên, xuất hiện như sau:

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

Nhiệm vụ 1: Thêm và chỉnh sửa các phần tử TextView

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



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

1.1 Tạo các phần tử dự án và TextView

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

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

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

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

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

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

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

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

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

RelativeLayout cho phép bạn đặt các phần tử giao diện người dùng tương đối với nhau hoặc tương đối với chính RelativeLayout gốc. Phần tử TextView "Hello World" mặc định được tạo bởi mẫu Bố cục trống vẫn có các thuộc tính ràng buộc (chẳng hạn như app:layout_constraintBottom_toBottomOf="parent"). Đừng lo lắng—you sẽ xóa chúng trong bước tiếp theo.

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

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

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

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.android.scrollingtext.MainActivity">
```

5. Thêm phần tử TextView phía trên TextView "Hello World" bằng cách nhập <TextView. Một khối TextView xuất hiện kết thúc bằng /> và hiển thị các thuộc tính layout_width và layout_height, cần thiết cho TextView.

6. Nhập các thuộc tính sau cho TextView . Khi bạn nhập từng thuộc tính và giá trị, các đề xuất sẽ xuất hiện để hoàn thành tên hoặc giá trị thuộc tính.

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

7. Trích xuất tài nguyên chuỗi cho chuỗi được mã hóa cứng "Tiêu đề bài viết" của thuộc tính android:text trong TextView để tạo một mục nhập cho nó trong strings.xml.

Đặt con trỏ vào chuỗi được mã hóa cứng, nhấn Alt-Enter (Option-Enter trên máy Mac) và chọn Trích xuất tài nguyên chuỗi . Đảm bảo rằng tùy chọn Tạo tài nguyên trong thư mục được chọn, sau đó chỉnh sửa tên tài nguyên cho giá trị chuỗi thành article_title . Tài nguyên chuỗi được mô tả chi tiết trong Tài nguyên chuỗi .

8. Trích xuất tài nguyên thứ nguyên cho chuỗi được mã hóa cứng "10dp" của thuộc tính android:padding trong TextView để tạo dimens.xml và thêm một mục nhập vào đó. Đặt con trỏ vào chuỗi được mã hóa cứng, nhấn Alt-Enter (Option-Enter trên máy Mac) và chọn Trích xuất tài nguyên thứ nguyên . Đảm bảo rằng tùy chọn Tạo tài nguyên trong thư mục được chọn, sau đó chỉnh sửa Tên tài nguyên thành padding_regular .

9. Thêm một phần tử TextView khác phía trên TextView "Hello World" và bên dưới TextView bạn đã tạo trong các bước trước. Thêm các thuộc tính sau vào TextView

TextView #2 Attribute	Value
layout_width	"match_parent"

layout_height	"wrap_content"
android:id	"@+id/article_subheading"
android:layout_below	"@+id/article_heading"
android:padding	"@dimen/padding_regular"
android:textAppearance	"@+id/article_subtitle"
android:text	"Article Subtitle"

Vì bạn đã trích xuất tài nguyên thứ nguyên cho chuỗi "10dp" để padding_regular trong TextView đã tạo trước đó nên bạn có thể sử dụng "@dimen/padding_regular" cho thuộc tính android:padding trong TextView này

10. Trích xuất tài nguyên chuỗi cho chuỗi được mã hóa cứng "Article Subtitle" của thuộc tính android:text trong TextView để article_subtitle .

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

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

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

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

13. Trích xuất tài nguyên chuỗi cho "Văn bản bài viết" để article_text và trích xuất tài nguyên thứ nguyên cho "5sp" để line_spacing .

14. Định dạng lại và căn chỉnh mã bằng cách chọn Mã > Định dạng lại mã. Bạn nên định dạng lại và căn chỉnh mã của mình để bạn và những người khác dễ hiểu hơn.

1.2 Thêm văn bản của bài viết

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

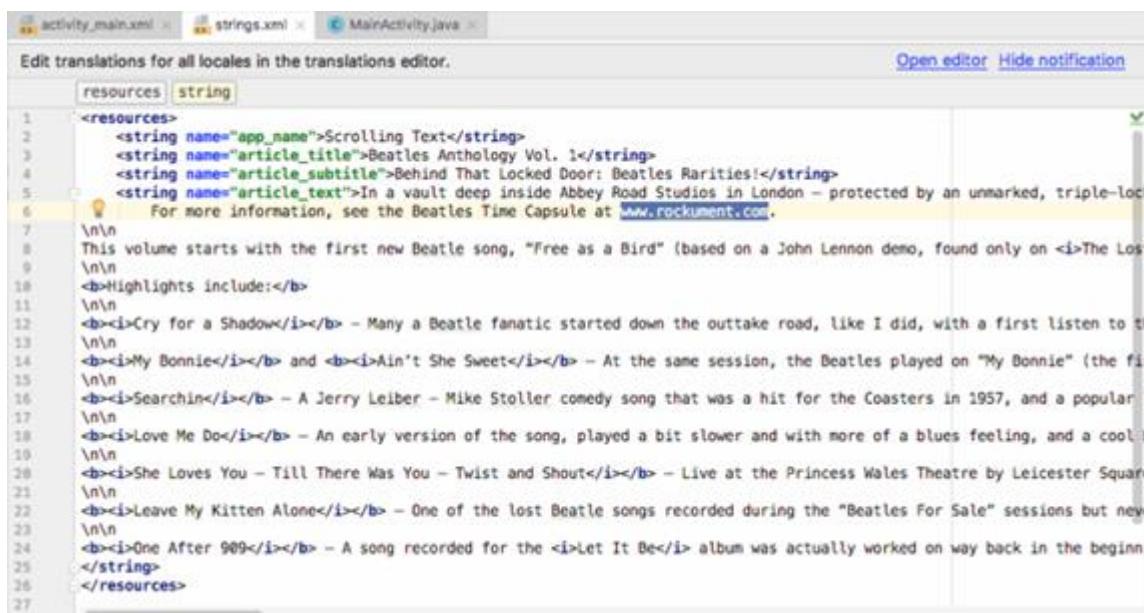
Đối với thực tế này, bạn sẽ tạo bài viết dưới dạng một chuỗi dài duy nhất trong tài nguyên strings.xml.

1. Trong ứng dụng > thư mục res > values, hãy mở strings.xml.
2. Mở bất kỳ tệp văn bản nào có lượng lớn văn bản hoặc mở tệp strings.xml của ứng dụng ScrollingText đã hoàn thành.
3. Nhập các giá trị cho các chuỗi article_title và article_subtitle với tiêu đề và phụ đề được tạo ra hoặc sử dụng các giá trị trong tệp strings.xml của ứng dụng ScrollingText đã hoàn thành. Đặt giá trị chuỗi thành văn bản một dòng 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 mình hoặc sử dụng văn bản được cung cấp cho chuỗi article_text trong tệp strings.xml của ứng dụng ScrollingText đã hoàn thành. Yêu cầu duy nhất cho tác vụ này là văn bản phải đủ dài để không vừa với màn hình.

- Khi bạn nhập hoặc dán văn bản vào tệp strings.xml, các dòng văn bản không ngắt quãng đến dòng tiếp theo — chúng kéo dài ra ngoài lề bên phải. Đây là hành vi chính xác—mỗi dòng văn bản mới bắt đầu từ lề trái đại diện cho toàn bộ đoạn văn. Nếu bạn muốn văn bản trong strings.xml được bao bọc, bạn có thể nhấn Return để nhập kết thúc dòng cứng hoặc định dạng văn bản trước tiên trong trình soạn thảo văn bản có kết thúc dòng cứng.
- Nhập n để biểu thị phần cuối của một dòng và một n khác để biểu thị một dòng trống. Bạn cần thêm các ký tự cuối dòng để giữ cho các đoạn văn không chạy vào nhau.
- Nếu bạn có dấu nháy đơn (') trong văn bản của mình, bạn phải thoát nó bằng cách trước nó bằng dấu gạch chéo ngược ('). Nếu bạn có dấu ngoặc kép trong văn bản của mình, bạn cũng phải thoát nó ("). Bạn cũng phải thoát khỏi bất kỳ ký tự không phải ASCII nào khác. Xem phần Định dạng và tạo kiểu của Tài nguyên chuỗi để biết thêm chi tiết.
- Nhập HTML và thẻ xung quanh các từ nên in đậm.

- Nhập HTML *<i>* và *</i>* thẻ xung quanh các từ nên in nghiêng. Nếu bạn sử dụng dấu nháy đơn cong trong cụm từ 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 bị bỏ qua.
- Bao gồm toàn bộ văn bản trong *<string name="article_text"> </string>* trong tệp 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.) Không sử dụng thẻ HTML, vì bất kỳ thẻ HTML nào ngoại trừ thẻ in đậm và in nghiêng đều bị bỏ qua và được hiển thị dưới dạng văn bản, đây không phải là điều bạn muốn.



```

1 activity_main.xml
2 strings.xml
3 MainActivity.java

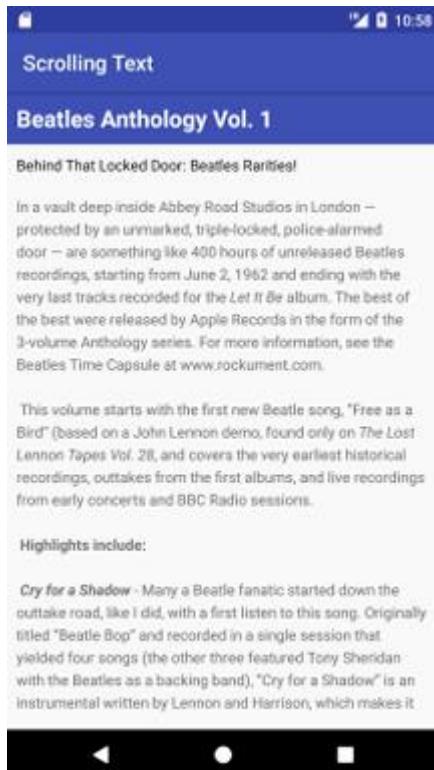
Edit translations for all locales in the translations editor.

resources string
<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 door. For more information, see the Beatles Time Capsule at www.rockument.com.
    \n\n
    This volume starts with the first new Beatle song, "Free as a Bird" (based on a John Lennon demo, found only on <i>The Lost Beatles Demo</i>). Highlights include:</b>
    \n\n
    <b><i>Cry for a Shadow</i></b> – Many a Beatle fanatic started down the outtake 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 song recorded by the band).
    <b><i>Searchin'</i></b> – A Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular Beatles cover.
    <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 title.
    <b><i>She Loves You – Till There Was You – Twist and Shout</i></b> – Live at the Princess Wales Theatre by Leicester Square.
    <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 909</i></b> – A song recorded for the <i>Let It Be</i> album was actually worked on way back in the beginning.
</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 ScrollView (bạn sẽ thực hiện trong tác vụ tiếp theo). Cũng lưu ý rằng nhấn vào liên kết web hiện không làm được gì. Bạn cũng sẽ khắc phục điều đó trong nhiệm vụ tiếp theo.



Mã giải pháp nhiệm vụ 1

Tệp bố cục 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:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    tools:context="com.example.android.scrollingtext.MainActivity">

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

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

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

</RelativeLayout>
```

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

Trong tác 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 chưa hoạt động. Bạn sẽ thêm mã để làm cho nó hoạt động. Ngoài ra, bản thân TextView không thể cho phép người dùng cuộn văn bản bài viết để xem tất cả. Bạn sẽ thêm một ViewGroup mới có tên là ScrollView vào bố cục XML để làm cho TextView có thể cuộn được.

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

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

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

2.2 Thêm ScrollView vào bộ cục

Để làm cho Chế độ xem (chẳng hạn như TextView) có thể cuộn được, hãy nhúng Chế độ xem bên trong ScrollView .

1. Thêm ScrollView giữa TextView article_subheading và TextView bài viết. Khi bạn nhập <ScrollView, Android Studio sẽ tự động thêm vào </ScrollView> cuối và hiển thị các thuộc tính android:layout_width và android:layout_height cùng với các đề xuất.
2. Chọn wrap_content từ các đề xuất cho cả hai thuộc tính.

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

```
<TextView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/article_subheading"  
    android:layout_below="@+id/article_heading"  
    android:padding="@dimen/padding_regular"  
    android:text="@string/article_subtitle"  
    android:textAppearance=  
        "@android:style/TextAppearance.DeviceDefault"/>  
  
<ScrollView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"></ScrollView>  
  
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:id="@+id/article"  
    android:autoLink="web"  
    android:layout_below="@+id/article_subheading"  
    android:lineSpacingExtra="@dimen/line_spacing"  
    android:padding="@dimen/padding_regular"  
    android:text="@string/article_text" />
```

3. Di chuyển mã kết thúc </ScrollView> sau TextView của bài viết để các thuộc tính TextView của bài viết hoàn toàn nằm trong ScrollView .

4. Xóa thuộc tính sau khỏi bài viết TextView và thêm nó vào ScrollView:

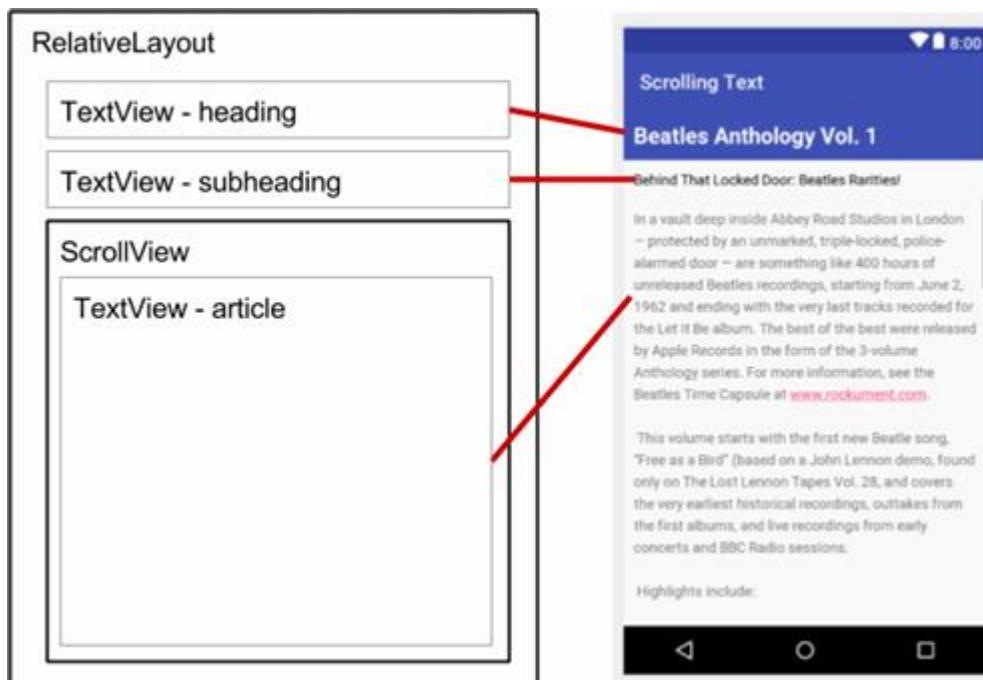
```
    android:layout_below="@+id/article_subheading"
```

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 Mã > Định dạng lại Mã để định dạng lại mã XML sao cho TextView của bài viết hiện xuất hiện thực lề bên trong mã <ScrollView>.

6. Nhấp vào Xem trước tab ở phía bên phải của trình chỉnh sửa bố cục để xem trước bố cục.

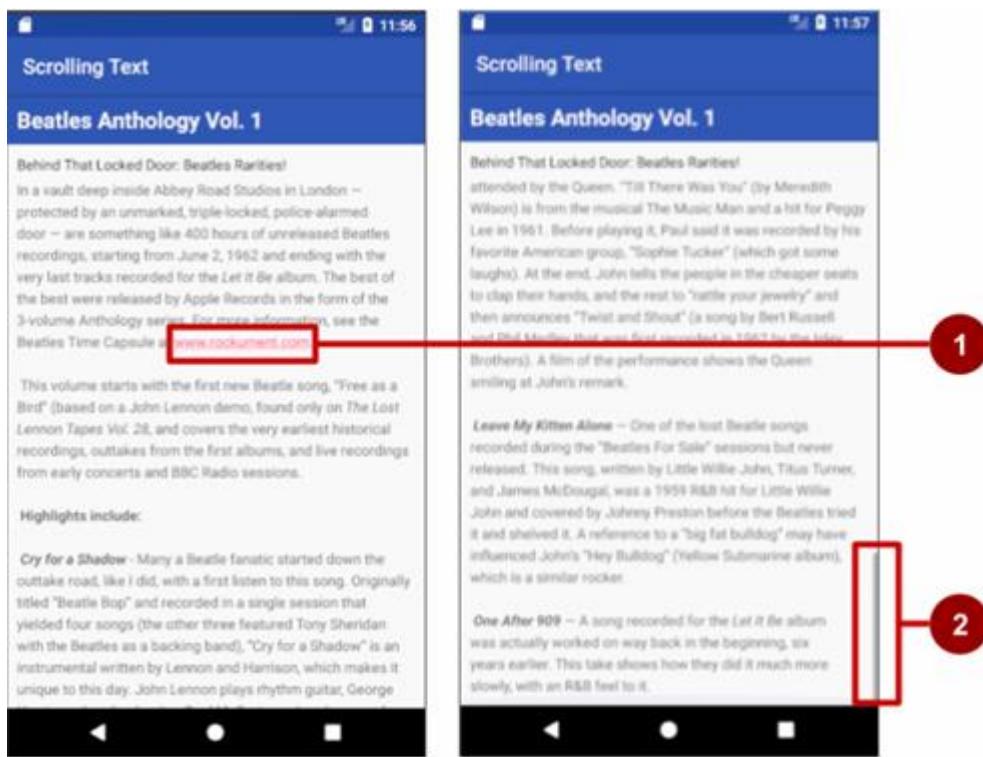
Bố cục bây giờ trông giống như phía bên phải của hình sau:



2.3 Chạy ứng dụng

Để kiểm tra các văn bản cuộn:

1. Chạy ứng dụng trên thiết bị hoặc trình giả lập. Vuốt lên và xuống để cuộn bài viết. Thanh cuộn xuất hiện ở lề phải khi bạn cuộn. Nhấn vào web liên kết để chuyển đến web trang. Thuộc tính android:autoLink biến mọi URL có thể nhận biết được trong TextView (chẳng hạn như www.rockument.com) thành liên kết web.
2. Xoay thiết bị hoặc trình giả lập của bạn trong khi chạy ứng dụng. Lưu ý cách chế độ xem cuộn mở rộng để sử dụng toàn bộ màn hình và vẫn cuộn đúng cách.
3. Chạy ứng dụng trên máy tính bảng hoặc trình giả lập máy tính bảng. Lưu ý cách chế độ xem cuộn mở rộng để sử dụng toàn bộ màn hình và vẫn cuộn đúng cách.



Trong hình trên, xuất hiện như sau:

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

Mã giải pháp nhiệm vụ 2

Mã XML cho bố cục với chế độ xem cuộn như sau:

```

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.android.scrollingtext.MainActivity">

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

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

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

        <TextView
            android:id="@+id/article"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:autoLink="web"

            android:lineSpacingExtra="@dimen/line_spacing"
            android:padding="@dimen/padding_regular"
            android:text="@string/article_text" />

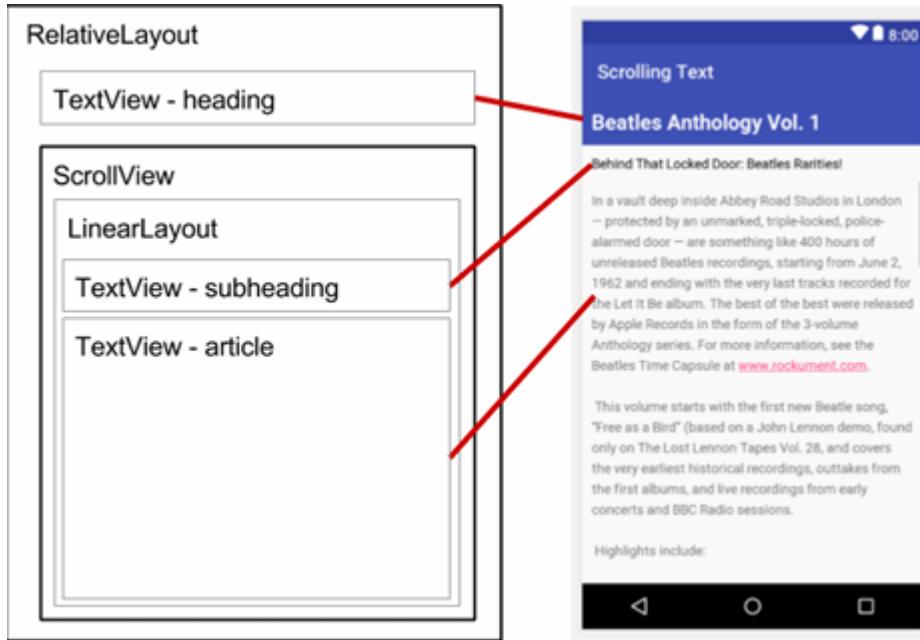
    </ScrollView>
</RelativeLayout>

```

Nhiệm vụ 3: Cuộn nhiều phần tử

Như đã lưu ý trước đây, ScrollView chỉ có thể chứa một Chế độ xem con (chẳng hạn như bài viết TextView 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 lồng một ViewGroup chẳng hạn như LinearLayout trong ScrollView , do đó cuộn mọi thứ bên trong LinearLayout .

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



3.1 Thêm LinearLayout vào ScrollView

1. Mở tệp activity_main.xml của dự án ứng dụng ScrollingText và chọn tab Văn bản để chỉnh sửa mã XML (nếu chưa được chọn).
2. Thêm một LinearLayout phía trên TextView bài viết trong ScrollView . Khi bạn nhập <LinearLayout, Android Studio sẽ tự động thêm vào </LinearLayout> cuối và hiển thị các thuộc tính android:layout_width và android:layout_height kèm theo các đề xuất. Chọn match_parent và wrap_content từ các đề xuất cho chiều rộng và chiều cao của nó, tương ứng. Mã ở đầu ScrollView bây giờ trông như thế này:

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

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

<TextView
    android:id="@+id/article"
```

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

3. Di chuyển mã kết thúc </LinearLayout> sau bài viết TextView nhưng trước khi đóng </ScrollView> . LinearLayout hiện bao gồm bài viết TextView và hoàn toàn nằm bên trong ScrollView .
4. Thêm thuộc tính android:orientation="vertical" vào LinearLayout để đặt hướng của nó thành dọc.
5. Chọn MÃ > Định dạng lại MÃ để thuần lè mã một cách chính xác.

LinearLayout bây giờ trông như thế này:

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

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

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

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

</ScrollView>
```

3.2 Di chuyển các phần tử giao diện người dùng trong LinearLayout

LinearLayout hiện chỉ có một phần tử giao diện người dùng — bài viết TextView . Bạn muốn bao gồm TextView article_subheading trong LinearLayout để cả hai sẽ cuộn.

1. Để di chuyển article_subheading TextView , chọn mã, chọn **Chỉnh sửa > Cắt** , nhấp vào phía trên bài viết TextView bên trong LinearLayout và chọn **Chỉnh sửa > Dán** .
2. Xóa thuộc tính android:layout_below="@+id/article_heading" khỏi TextView article_subheading. Bởi vì TextView này hiện nằm trong LinearLayout , thuộc tính này sẽ xung đột với các thuộc tính LinearLayout. 3. Thay đổi thuộc tính bố cục ScrollView từ android:layout_below="@+id/article_subheading" thành android:layout_below="@+id/article_heading" . Bây giờ tiêu đề phụ là một phần của LinearLayout , ScrollView phải được đặt bên dưới tiêu đề, không phải tiêu đề phụ. Mã XML cho ScrollView bây giờ là gas sau:

```
<ScrollView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_below="@+id/article_heading">  
  
    <LinearLayout  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:orientation="vertical">  
  
        <TextView  
            android:id="@+id/article_subheading"  
            android:layout_width="match_parent"  
            android:layout_height="wrap_content"
```

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

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

</ScrollView>
```

4. Chạy ứng dụng.

Vuốt lên và xuống để cuộn bài viết và lưu ý rằng tiêu đề phụ bấy giờ cuộn cùng với bài viết trong khi tiêu đề vẫn giữ nguyên.

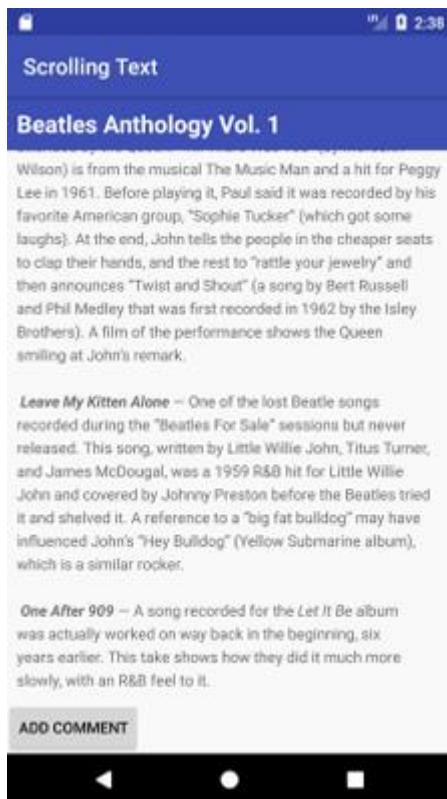
Mã giải pháp dự án Android Studio: ScrollingText

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.

Thử thách: Thêm một phần tử giao diện người dùng khác — Button — vào LinearLayout bên trong ScrollView để nó cuộn cùng với văn bản.

- Làm cho Nút xuất hiện bên dưới bài viết. Người dùng cuộn đến cuối bài viết để xem nút .
- Sử dụng văn bản Thêm nhận xét cho nút. Đối với thử thách này, không cần phải tạo phương pháp xử lý nút; tất cả những gì bạn phải làm là đặt phần tử Button vào vị trí thích hợp trong bố cục.



Mã giải pháp thử thách

dự án Android Studio: ScrollingTextChallenge

Tóm tắt

- Sử dụng ScrollView để cuộn một Chế độ xem con duy nhất (chẳng hạn như TextView). ScrollView chỉ có thể chứa một View hoặc ViewGroup con.
- Sử dụng ViewGroup như LinearLayout làm View con trong ScrollView để cuộn nhiều phần tử View. Bao gồm các phần tử trong LinearLayout .
- Hiển thị văn bản dạng tự do trong TextView với các thẻ định dạng HTML cho in đậm và in nghiêng.
- Sử dụng n làm ký tự cuối dòng trong văn bản dạng tự do để giữ cho một đoạn văn không chạy vào đoạn tiếp theo.
- Sử dụng thuộc tính android:autoLink="web" để làm cho các liên kết web trong văn bản có thể nhấp vào.

Các khái niệm liên quan

Tài liệu khái niệm liên quan có trong 1.3 Chế độ xem văn bản và cuộn .

Tìm hiểu thêm

tài liệu về Android Studio:

- Trang tải xuống Android Studio
- Gặp gỡ tài liệu

dành cho nhà phát triển Android Studio:

- ScrollView
- LinearLayout
- RelativeLayout
- View
- Nút
- TextView
- Tài nguyên chuỗi
- Bố cục tương đối

Khác:

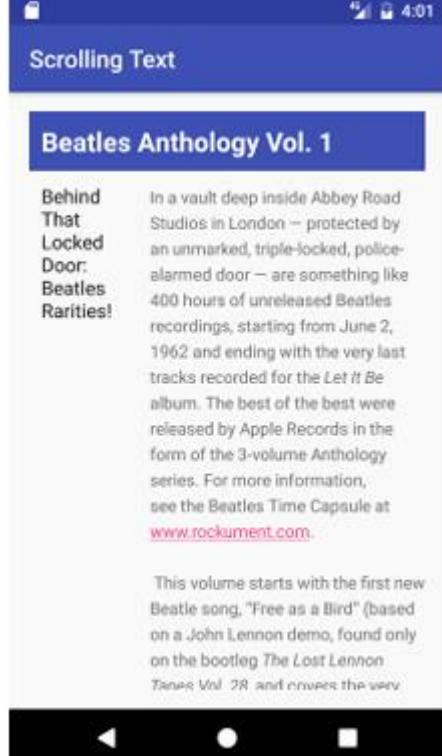
- Blog dành cho các nhà phát triển Android: Liên kết văn bản của bạn!
- Pathway mã: Làm việc với TextView

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 để nó bao bọc trong một cột ở bên trái rộng 100 dp, như hình dưới đây.
2. Đặt văn bản của bài viết ở bên phải của tiêu đề phụ như hình dưới đây.



Trả lời các câu hỏi sau

Câu hỏi 1: Bạn có thể sử dụng bao nhiêu chế độ xem trong một ScrollView? Chọn một:

- Chỉ một chế độ xem
- Một chế độ xem hoặc một nhóm dạng xem
- Bao nhiêu tùy thích

Câu hỏi 2 Bạn sử dụng thuộc tính XML nào trong LinearLayout để hiển thị các dạng xem cạnh nhau? Chọn một:

- android:orientation="horizontal"
- android:orientation="vertical"
- android:layout_width="wrap_content"

Câu hỏi 3 Bạn sử dụng thuộc tính XML nào để xác định chiều rộng của LinearLayout bên trong chế độ xem cuộn? Chọn một:

- android:layout_width="wrap_content"
- android:layout_width="match_parent"
- android:layout_width="200dp"

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:

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

Bài 1.4: Học cách giúp đỡ bản thân

Giới thiệu

Những gì bạn nên biết

Bạn sẽ có thể:

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

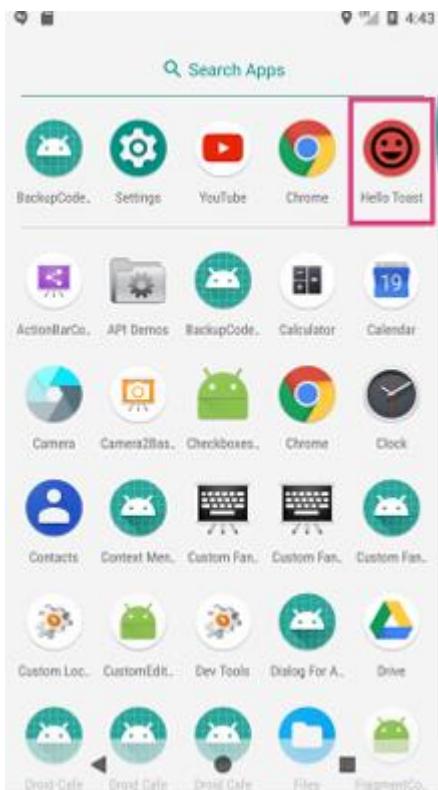
Những gì bạn sẽ học

- Tìm thông tin và tài nguyên dành cho nhà phát triển ở đâu.

- Cách thêm biểu tượng trình khởi chạy vào ứng dụng của bạn.
- Cách tìm kiếm trợ giúp khi bạn đang phát triển ứng dụng Android của mình. 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 trình khởi chạy cho ứng dụng của bạn.

Tổng quan về ứng dụng:

Bạn sẽ thêm biểu tượng trình khởi chạy vào ứng dụng HelloToast mà bạn đã tạo trước đó hoặc vào ứng dụng mới.

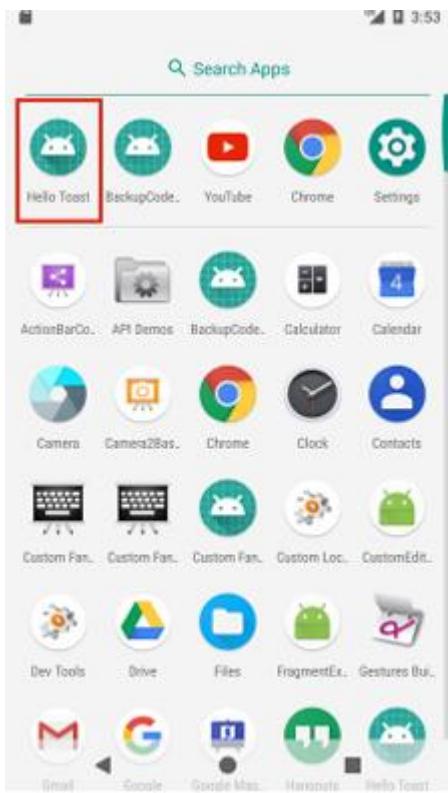


Nhiệm vụ 1: Thay đổi biểu tượng trình chạy

Mỗi ứng dụng mới bạn tạo bằng Android Studio đều bắt đầu bằng một biểu tượng trình chạy tiêu chuẩn đại diện cho ứng dụng. Biểu tượng trình

chạy xuất hiện trong danh sách cửa hàng Google Play. Khi người dùng tìm kiếm trên cửa hàng Google Play, biểu tượng cho ứng dụng của bạn sẽ xuất hiện trong kết quả tìm kiếm.

Khi người dùng đã cài đặt ứng dụng, biểu tượng trình khởi chạy sẽ xuất hiện trên thiết bị ở nhiều nơi 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 mô phỏng với biểu tượng tiêu chuẩn cho các dự án ứng dụng mới, như minh họa bên dưới.



Thay đổi biểu tượng trình chạy là một quy trình từng bước đơn giản giới thiệu cho bạn các tính năng nội dung hình ảnh của Android Studio. Trong nhiệm vụ này, bạn cũng 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 chính thức dành cho nhà phát triển Android tại developer.android.com . Tài liệu này chứa rất nhiều thông tin được Google cập nhật.

16. Đi đến developer.android.com/design/ .

Phần này nói về Material Design, là một triết lý thiết kế khái niệm phác thảo cách các ứng dụng nên trông và hoạt động trên thiết bị di động. Điều hướng các liên kết để tìm hiểu thêm về Material Design. Ví dụ: truy cập phần Phong cách để tìm hiểu thêm về cách sử dụng màu sắc và các chủ đề khác.

17. Truy cập developer.android.com/docs/ để tìm thông tin API, tài liệu tham khảo, hướng dẫn, hướng dẫn công cụ và mẫu mã.

18. Truy cập developer.android.com/distribute/ để tìm thông tin về việc đưa ứng dụng lên Google Play, hệ thống phân phối kỹ thuật số của Google dành cho các ứng dụng được phát triển bằng Android SDK. Sử dụng Google Play Console để phát triển cơ sở người dùng của bạn và bắt đầu kiếm tiền.

1.2 Thêm nội dung hình ảnh cho biểu tượng trình khởi chạy

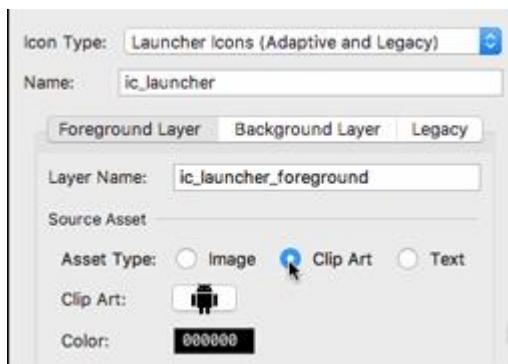
Để thêm hình ảnh clip-art làm biểu tượng trình khởi chạy, hãy làm theo các bước sau:

1. Mở dự án ứng dụng HelloToast từ bài học trước về cách sử dụng trình chỉnh sửa bối cảnh hoặc tạo dự án ứng dụng mới.
2. Trong ngăn Project > Android, nhấp chuột phải (hoặc Control khi nhấp vào) thư mục res và chọn New > Image Asset . Cửa sổ Định cấu hình nội dung hình ảnh xuất hiện.



3. Trong trường Icon Type, chọn Launcher Icons (Adaptive & Legacy) nếu nó chưa được chọn.

4. Nhấp vào Foreground Layer tab, chọn Clip Art cho Loại nội dung .

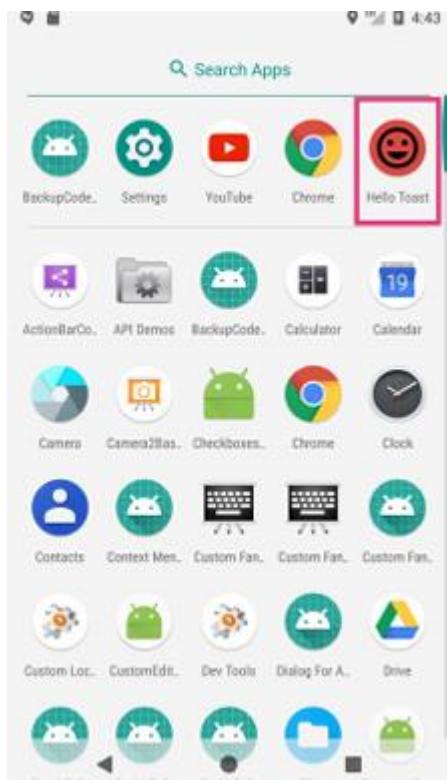


5. Nhấp vào biểu tượng trong Clip Art trường. Các biểu tượng xuất hiện từ bộ biểu tượng thiết kế vật liệu. 6. Duyệt qua cửa sổ Chọn biểu tượng, chọn một biểu tượng thích hợp (chẳng hạn như biểu tượng tâm trạng để gợi ý tâm trạng tốt), sau đó nhấp vào OK .



7. Nhấp vào tab Background Layer, chọn Color làm Asset Type , sau đó nhấp vào chip màu để chọn màu để sử dụng làm layer nền.
8. Nhấp vào tab Legacy và xem lại cài đặt mặc định. Xác nhận rằng bạn muốn tạo các biểu tượng cũ, hình tròn và Cửa hàng Google Play. Nhấp vào Tiếp theo khi hoàn tất.
9. Chạy ứng dụng. Android Studio tự động thêm hình ảnh trình khởi chạy vào thư mục mipmap cho các mật độ khác nhau.

Do đó, biểu tượng khởi chạy ứng dụng sẽ thay đổi thành biểu tượng mới sau khi bạn chạy ứng dụng, như hình dưới đây.



Mẹo: Xem Biểu tượng trình khởi chạy để tìm hiểu thêm về cách thiết kế các biểu tượng trình khởi chạy 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 đề xuất. Sử dụng các mẫu tích hợp giúp tiết kiệm thời gian và giúp bạn tuân theo các phương pháp thiết kế tốt nhất.

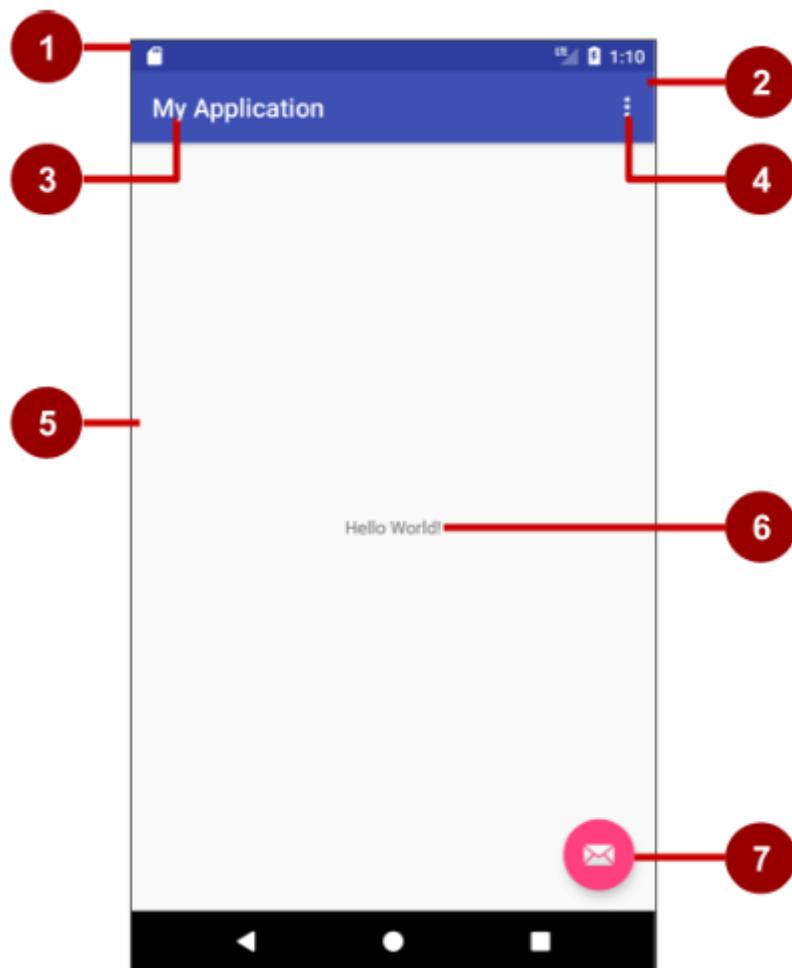
Mỗi mẫu kết hợp một hoạt động khung và giao diện người dùng. Bạn đã sử dụng mẫu Hoạt động trống. Mẫu Hoạt động cơ bản có nhiều tính năng hơn và kết hợp các tính năng ứng dụng được đề xuất, chẳng hạn như menu tùy chọn xuất hiện trong thanh ứng dụng.

2.1 Khám phá kiến trúc Hoạt động cơ bản

Mẫu Hoạt động cơ bản là một mẫu linh hoạt do Android Studio cung cấp để hỗ trợ bạn bắt đầu phát triển ứng dụng của mình.

- Trong Android Studio, tạo một dự án mới bằng mẫu Basic Activity.
- Xây dựng và chạy ứng dụng.
- Xác định các bộ phận được dán nhãn trong hình và bảng bên dưới. Tìm các ứng dụng tương đương trên màn hình thiết bị hoặc trình mô phỏng của bạn. Kiểm tra mã Java tương ứng và các tệp XML được mô tả trong bảng.

Làm quen với mã nguồn Java và các tệp XML sẽ giúp bạn mở rộng và tùy chỉnh mẫu này cho nhu cầu của riêng bạn.



Kiến trúc của mẫu Hoạt động cơ bản:

#	Tổ chức giao diện người dùng	Tham khảo mã
1	Thanh trạng thái Hệ thống Android cung cấp và kiểm soát thanh trạng thái.	Không hiển thị trong mã mẫu. Bạn có thể

		truy cập nó từ hoạt động của bạn. Ví dụ: bạn có thể ẩn thanh trạng thái , nếu cần.
2	AppBarLayout > Thanh công cụ Thanh ứng dụng (còn được gọi là thanh hành động) cung cấp cấu trúc trực quan, các yếu tố trực quan được chuẩn hóa và điều hướng. Đối ngược tương thích, AppBarLayout trong mẫu nhúng Thanh công cụ có chức năng tương tự như ActionBar	Trong activity_main.xml , hãy tìm android.support.v7.widget.Toolbar bên trong android.support.design.widget.AppBarLayout . Thay đổi thanh công cụ để thay đổi giao diện của thanh ứng dụng. Để biết ví dụ, hãy xem Hướng dẫn thanh ứng dụng
3	Tên ứng dụng này bắt nguồn từ tên gói của bạn, nhưng có thể là bất cứ thứ gì bạn chọn.	Trong AndroidManifest.xml: android:label="@string/app_name"
4	Nút tràn menu Tùy chọn Các mục menu cho hoạt động, cũng như các tùy chọn chung, chẳng hạn như Tìm kiếm và Cài đặt cho ứng dụng. Các mục menu ứng dụng của bạn sẽ đi vào menu này.	Trong MainActivity.java : onOptionsItemSelected() thực hiện những gì xảy ra khi một mục menu được chọn. res > menu > menu_main.xml Tài nguyên chỉ định các mục menu cho menu tùy chọn.
5	Nhóm chế độ xem bố cục CoordinatorLayout ViewGroup là một bố cục giàu tính năng cung cấp cơ chế cho các phần tử Chế độ xem (UI) tương tác. Giao diện người dùng của ứng dụng đi vào bên	Trong activity_main.xml : Không có chế độ xem nào được chỉ định trong bố cục này; thay vào đó, nó bao gồm

	trong tệp content_main.xml có trong ViewGroup này	một bối cảnh khác với một lệnh bối cảnh bao gồm để bao gồm @layout / content_main nơi các chế độ xem được chỉ định. Điều này tách chế độ xem hệ thống khỏi chế độ xem duy nhất cho ứng dụng của bạn.
6	TextView Trong ví dụ, được sử dụng để hiển thị "Hello World". Thay thế điều này bằng các thành phần giao diện người dùng cho ứng dụng của bạn.	Trong content_main.xml : Tất cả các thành phần giao diện người dùng của ứng dụng được xác định trong tệp này.
7	Nút hành động (FAB)	Trong activity_main.xml như một phần tử giao diện người dùng sử dụng biểu tượng clip-art. MainActivity.java bao gồm một sơ khai trong onCreate() để đặt trình nghe onClick() cho FAB.

2.2 Tùy chỉnh ứng dụng do mẫu tạo ra

Thay đổi giao diện của ứng dụng do mẫu Hoạt động cơ bản tạo ra. Ví dụ: bạn có thể thay đổi màu của thanh ứng dụng để phù hợp với thanh trạng thái (trên một số thiết bị có màu tối hơn của