**TRƯỜNG ĐẠI HỌC THỦY LỢI**

**KHOA CÔNG NGHỆ THÔNG TIN**



**GIÁO TRÌNH**

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

Hà Nội, 2.2025

MỤC LỤC

[CHƯƠNG 1. Làm quen 4](#_Toc190855147)

[Bài 1) Tạo ứng dụng đầu tiên 4](#_Toc190855148)

[1.1) Android Studio và Hello World 4](#_Toc190855149)

[1.2) Giao diện người dùng tương tác đầu tiên 5](#_Toc190855150)

[1.3) Trình chỉnh sửa bố cục 5](#_Toc190855151)

[1.4) Văn bản và các chế độ cuộn 5](#_Toc190855152)

[1.5) Tài nguyên có sẵn 5](#_Toc190855153)

[Bài 2) Activities 5](#_Toc190855154)

[2.1) Activity và Intent 5](#_Toc190855155)

[2.2) Vòng đời của Activity và trạng thái 5](#_Toc190855156)

[2.3) Intent ngầm định 5](#_Toc190855157)

[Bài 3) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ 5](#_Toc190855158)

[3.1) Trình gỡ lỗi 5](#_Toc190855159)

[3.2) Kiểm thử đơn vị 5](#_Toc190855160)

[3.3) Thư viện hỗ trợ 5](#_Toc190855161)

[CHƯƠNG 2. Trải nghiệm người dùng 6](#_Toc190855162)

[Bài 1) Tương tác người dùng 6](#_Toc190855163)

[1.1) Hình ảnh có thể chọn 6](#_Toc190855164)

[1.2) Các điều khiển nhập liệu 6](#_Toc190855165)

[1.3) Menu và bộ chọn 6](#_Toc190855166)

[1.4) Điều hướng người dùng 6](#_Toc190855167)

[1.5) RecycleView 6](#_Toc190855168)

[Bài 2) Trải nghiệm người dùng thú vị 6](#_Toc190855169)

[2.1) Hình vẽ, định kiểu và chủ đề 6](#_Toc190855170)

[2.2) Thẻ và màu sắc 6](#_Toc190855171)

[2.3) Bố cục thích ứng 6](#_Toc190855172)

[Bài 3) Kiểm thử giao diện người dùng 6](#_Toc190855173)

[3.1) Espresso cho việc kiểm tra UI 6](#_Toc190855174)

[CHƯƠNG 3. Làm việc trong nền 6](#_Toc190855175)

[Bài 1) Các tác vụ nền 6](#_Toc190855176)

[1.1) AsyncTask 6](#_Toc190855177)

[1.2) AsyncTask và AsyncTaskLoader 6](#_Toc190855178)

[1.3) Broadcast receivers 6](#_Toc190855179)

[Bài 2) Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền 6](#_Toc190855180)

[2.1) Thông báo 6](#_Toc190855181)

[2.2) Trình quản lý cảnh báo 6](#_Toc190855182)

[2.3) JobScheduler 6](#_Toc190855183)

[CHƯƠNG 4. Lưu dữ liệu người dùng 7](#_Toc190855184)

[Bài 1) Tùy chọn và cài đặt 7](#_Toc190855185)

[1.1) Shared preferences 7](#_Toc190855186)

[1.2) Cài đặt ứng dụng 7](#_Toc190855187)

[Bài 2) Lưu trữ dữ liệu với Room 7](#_Toc190855188)

[2.1) Room, LiveData và ViewModel 7](#_Toc190855189)

[2.2) Room, LiveData và ViewModel 7](#_Toc190855190)

# LÀM QUEN

## Tạo ứng dụng đầu tiên

### Android Studio và Hello World

Giới thiệu

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

Những gì Bạn nên biết

Bạn nên có khả năng:

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

Những gì Bạn sẽ cần:

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

Những gì bạn sẽ học

* Cách cài đặt và sử dụng IDE Android Studio.
* Cách sử dụng quy trình phát triển để xây dựng ứng dụng Android.
* Cách tạo một dự án Android từ một template có sẵn.
* Cách thêm các thông báo log vào ứng dụng để hỗ trợ việc gỡ lỗi.

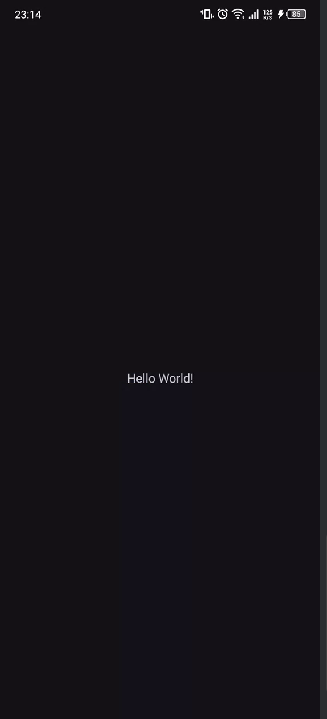
Những gì bạn sẽ làm

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

Tổng quan về ứng dụng

Sau khi cài đặt thành công Android Studio, bạn sẽ tạo một dự án mới từ mẫu có sẵn cho ứng dụng Hello World. Ứng dụng đơn giản này hiển thị dòng chữ "Hello World" trên màn hình của thiết bị Android (ảo hoặc thật).

Đây là giao diện của ứng dụng khi hoàn thành:

****

Nhiệm vụ 1: Cài đặt Android Studio

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

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

Android Studio có sẵn cho các máy tính chạy Windows hoặc Linux, và cho máy Mac chạy macOS. Phiên bản OpenJDK (Java Development Kit) mới nhất được tích hợp sẵn với Android Studio. Để bắt đầu sử dụng Android Studio, trước tiên hãy kiểm tra yêu cầu hệ thống để đảm bảo rằng hệ thống của bạn đáp ứng được chúng. Quy trình cài đặt tương tự cho tất cả các nền tảng. Mọi khác biệt sẽ được ghi chú bên dưới.

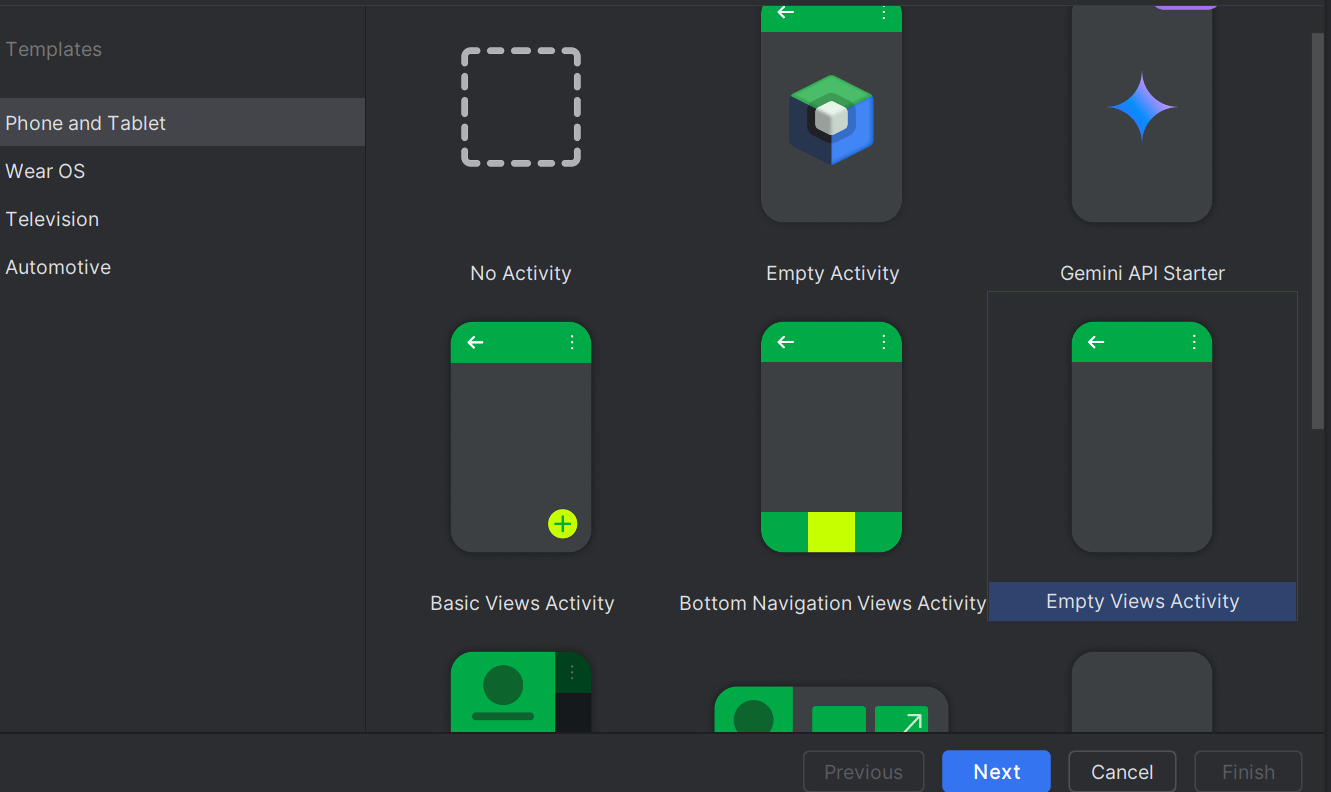
1. Truy cập vào trang web dành cho nhà phát triển Android và làm theo hướng dẫn để tải xuống và cài đặt Android Studio.
2. Chấp nhận cấu hình mặc định cho tất cả các bước, và đảm bảo rằng tất cả các thành phần được chọn để cài đặt.
3. Sau khi hoàn tất cài đặt, Trình hướng dẫn Thiết lập sẽ tải xuống và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

Nhiệm vụ 2: Tạo ứng dụng Hello World

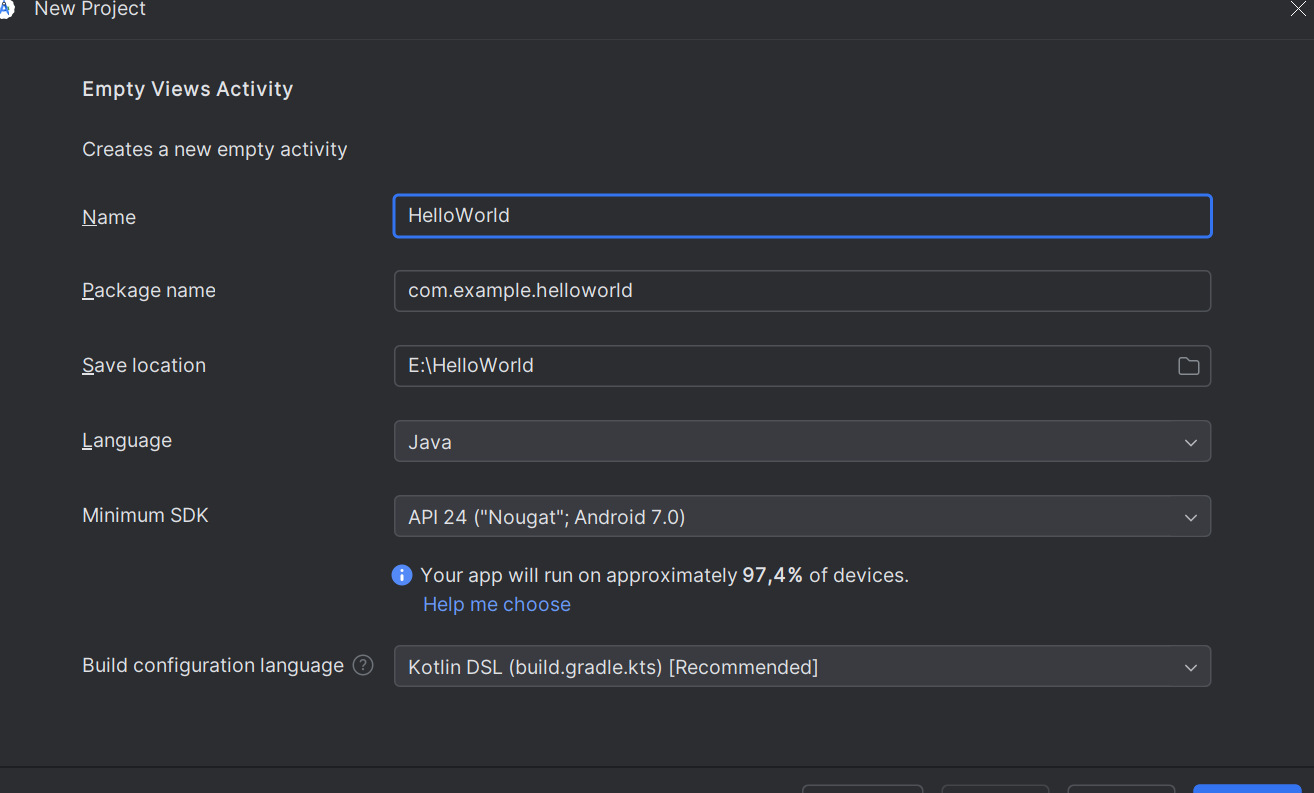
Trong nhiệm vụ này, bạn sẽ tạo một ứng dụng hiển thị "Hello World" để xác minh rằng Android Studio đã được cài đặt đúng cách, và để học các kiến thức cơ bản về phát triển với Android Studio.

**2.1 Tạo dự án ứng dụng**

1. Mở Android Studio nếu chưa được mở.
2. Trong cửa sổ chính Welcome to Android Studio, nhấp vào Start a new Android Studio project.
3. Cửa sổ Add an Activity xuất hiện. Activity là một thao tác đơn lẻ, tập trung mà người dùng có thể thực hiện. Đây là một thành phần quan trọng của bất kỳ ứng dụng Android nào. Một Activity thường có một layout đi kèm định nghĩa cách các thành phần UI xuất hiện trên màn hình. Android Studio cung cấp các mẫu Activity để giúp bạn bắt đầu. Đối với dự án Hello World, chọn Empty Views Activity như hiển thị bên dưới, và nhấp Next.



1. Trong cửa sổ Create Android Project, nhập HelloWorld cho tên ứng dụng (Application name).
2. Xác nhận rằng vị trí dự án mặc định (Project location) là nơi bạn muốn lưu trữ ứng dụng Hello World và các dự án Android Studio khác, hoặc thay đổi nó thành thư mục ưa thích của bạn.
3. Chấp nhận android.example.com mặc định cho Company Domain (Tên miền công ty), hoặc tạo một tên miền công ty riêng.Nếu bạn không có kế hoạch xuất bản ứng dụng của mình, bạn có thể chấp nhận mặc định. Lưu ý rằng việc thay đổi tên gói của ứng dụng sau này sẽ tốn thêm công sức.
4. Không đánh dấu vào các tùy chọn Include C++ support và Include Kotlin support, và nhấp Next.
5. Trên màn hình Target Android Devices, Phone and Tablet (Điện thoại và Máy tính bảng) nên được chọn. Đảm bảo rằng API 24: được đặt làm Minimum SDK; nếu không, hãy sử dụng menu popup để đặt nó.
6. Không đánh dấu vào Include Instant App support và tất cả các tùy chọn khác. Sau đó nhấp Next. Nếu dự án của bạn yêu cầu các thành phần bổ sung cho SDK đích đã chọn, Android Studio sẽ tự động cài đặt chúng.



1. Nhấp Finish

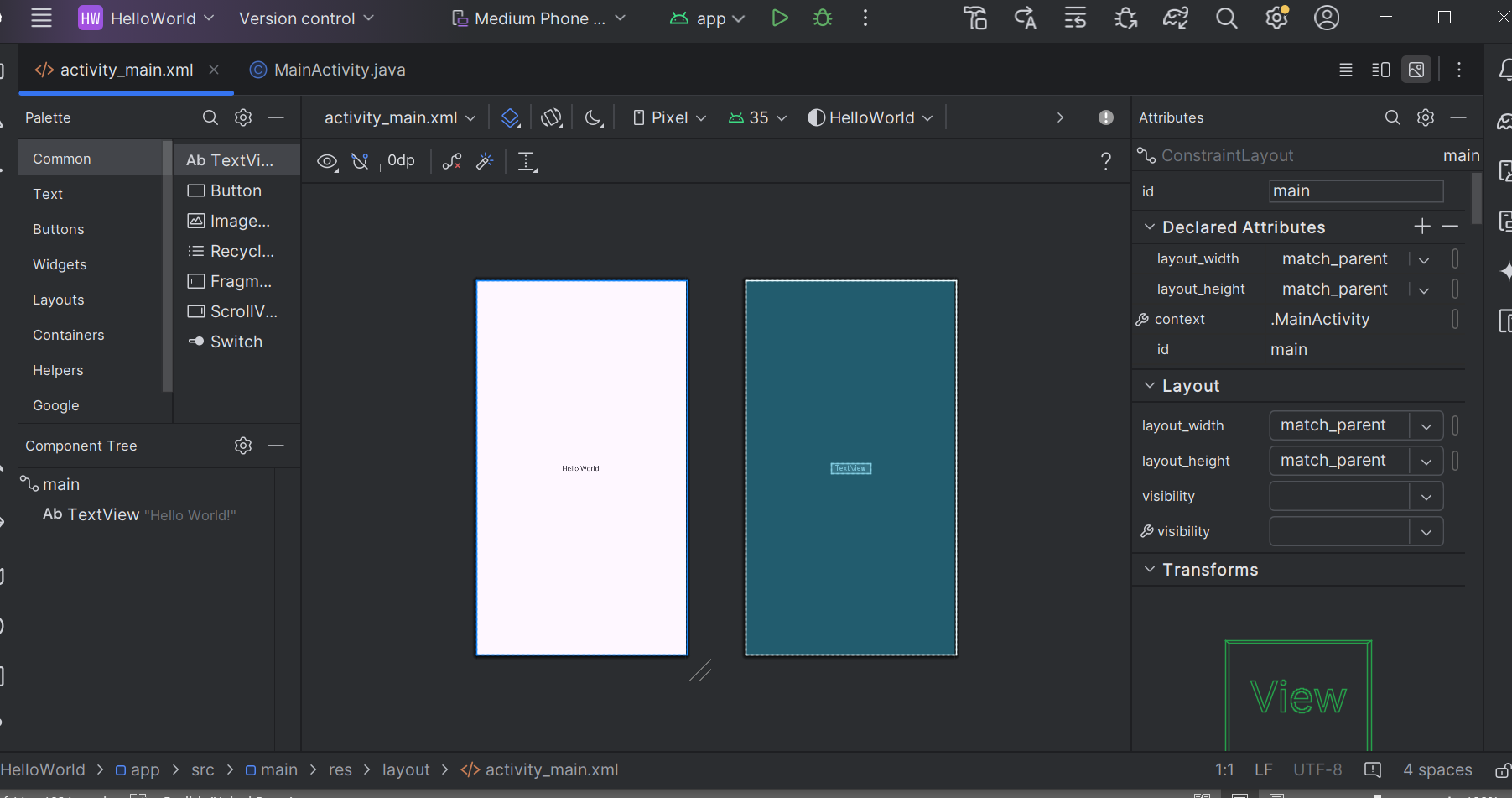
Android Studio tạo một thư mục cho các dự án của bạn, và xây dựng dự án bằng Gradle (điều này có thể mất một vài phút).

Mẹo: Xem trang Configure your build dành cho nhà phát triển để biết thông tin chi tiết.

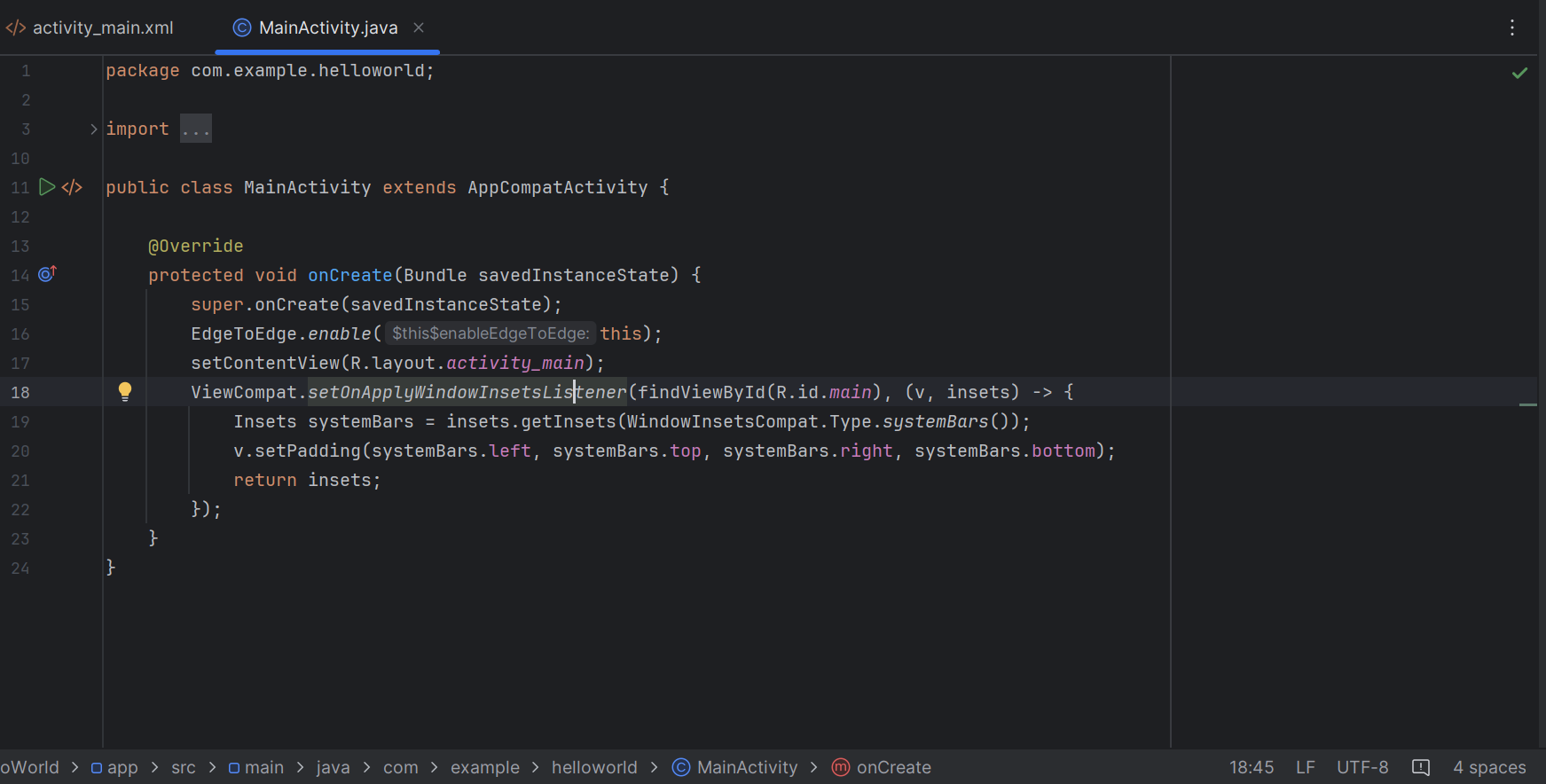
Bạn cũng có thể thấy thông báo "Tip of the day" với các phím tắt và các mẹo hữu ích khác. Nhấp Close để đóng thông báo.

Trình soạn thảo Android Studio xuất hiện. Thực hiện theo các bước sau:

1. Nhấp vào tab **activity\_main.xml** để xem trình soạn thảo layout.
2. Nhấp vào tab Design của trình soạn thảo layout, nếu chưa được chọn, để hiển thị bản dựng đồ họa của layout như hiển thị bên dưới.



1. Nhấp vào tab **MainActivity.java** để xem trình soạn thảo mã như hình dưới đây

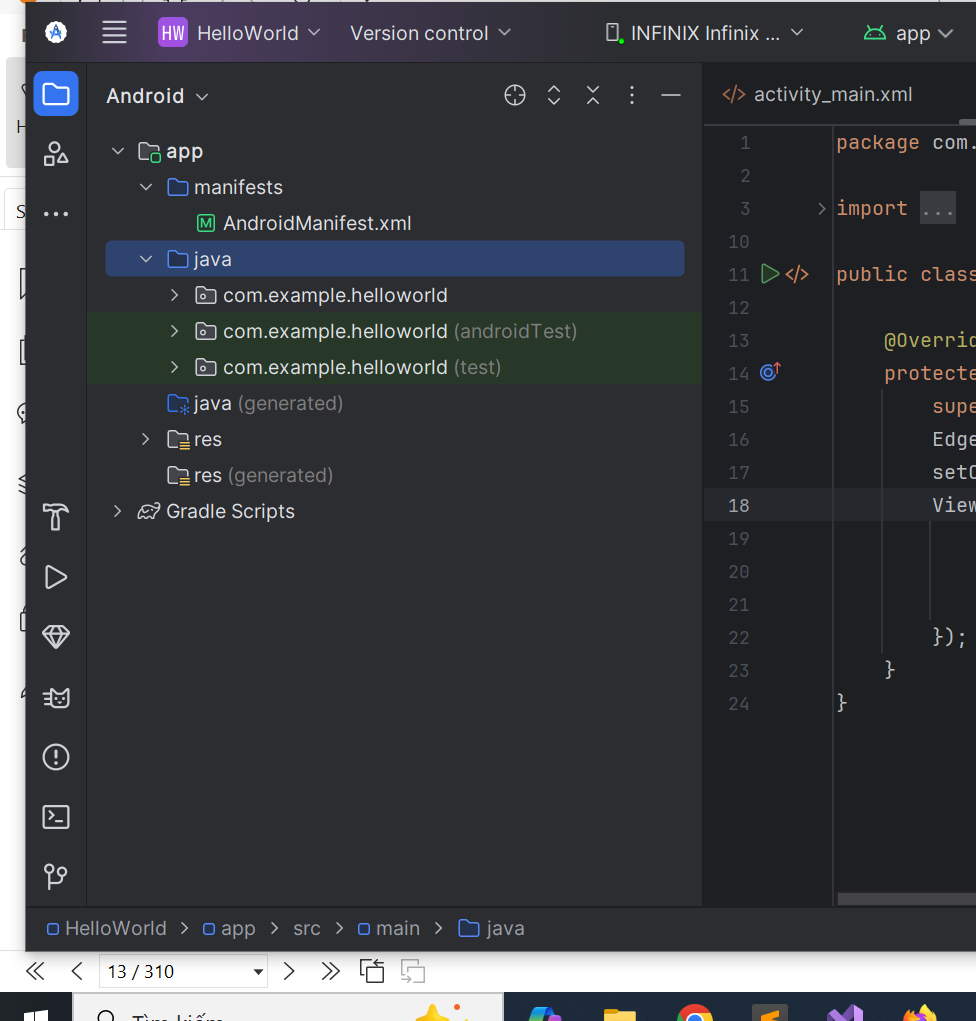


****2.2 Khám phá Project > Bảng điều khiển Android****

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

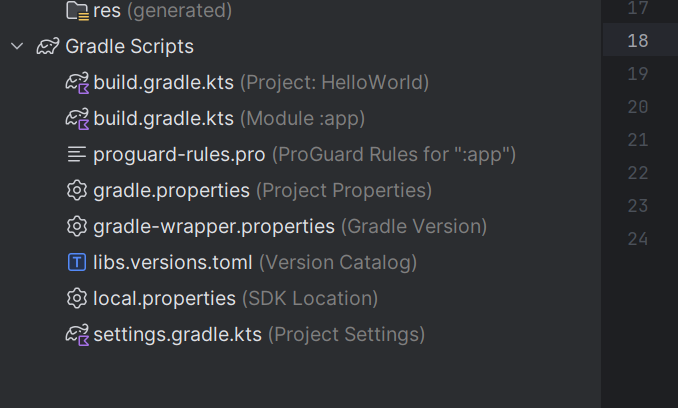
1. **Nếu chưa được chọn, hãy nhấp vào tab Project trong cột tab dọc ở bên trái cửa sổ Android Studio. Bảng điều khiển Project sẽ xuất hiện.**

**2. Để xem dự án theo hệ thống phân cấp dự án Android tiêu chuẩn, hãy chọn Android từ menu popup ở phía trên cùng của bảng điều khiển Project, như hình dưới đây.**



2.3 Khám phá thư mục Gradle Scripts

1. Hệ thống build Gradle trong Android Studio giúp dễ dàng thêm các tệp nhị phân bên ngoài hoặc các module thư viện khác vào bản build của bạn dưới dạng các phụ thuộc (dependencies).
2. Khi bạn lần đầu tạo một dự án ứng dụng, bảng Project > Android xuất hiện với thư mục Gradle Scripts được mở rộng như hình dưới đây.



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

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

Thư mục này chứa tất cả các file cần thiết cho hệ thống build.

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

Đây là nơi bạn sẽ tìm thấy các tùy chọn cấu hình chung cho tất cả các module tạo nên dự án của bạn. Mỗi dự án Android Studio đều chứa một file build Gradle cấp cao nhất duy nhất. Hầu hết thời gian, bạn sẽ không cần thay đổi file này, nhưng việc hiểu nội dung của nó vẫn rất hữu ích.

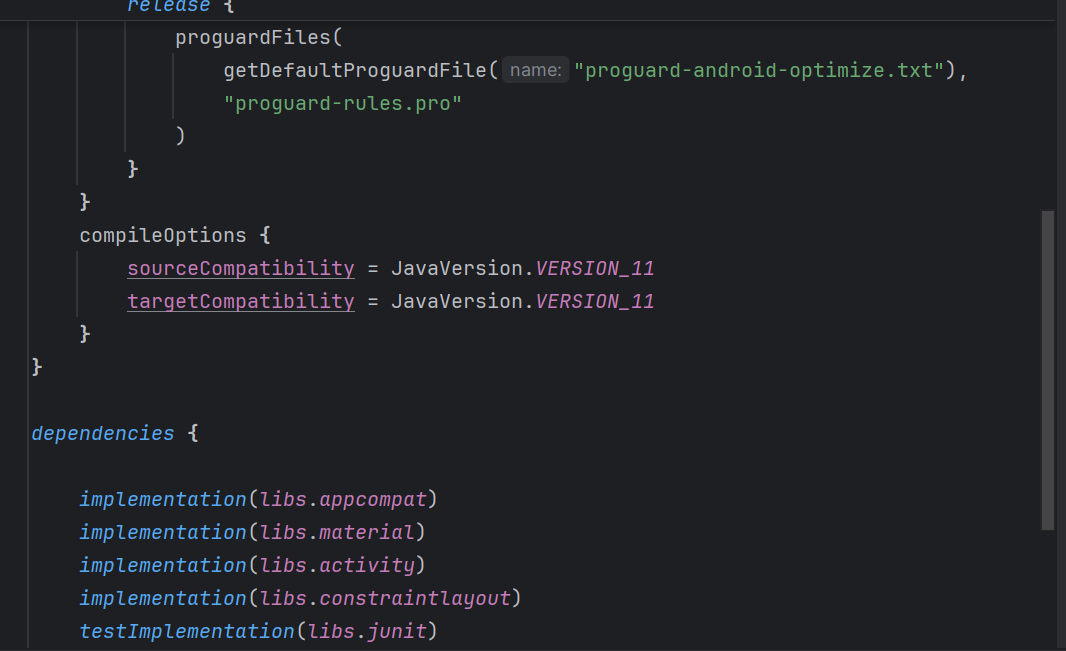
Theo mặc định, file build cấp cao nhất sử dụng khối buildscript để xác định các kho lưu trữ (repositories) và phụ thuộc (dependencies) Gradle chung cho tất cả các module trong dự án. Khi dependency của bạn không phải là thư viện cục bộ hoặc cây thư mục, Gradle sẽ tìm kiếm các file trong các kho lưu trữ trực tuyến được chỉ định trong khối repositories của file này. Theo mặc định, các dự án Android Studio mới khai báo JCenter và Google (bao gồm cả kho Google Maven) làm vị trí kho lưu trữ:

3. Ngoài tệp build.gradle ở cấp độ dự án, mỗi module đều có tệp build.gradle riêng, cho phép bạn cấu hình các thiết lập build cho từng module cụ thể (ứng dụng HelloWorld chỉ có một module). Việc cấu hình các thiết lập build này cho phép bạn cung cấp các tùy chọn đóng gói bổ sung, như các kiểu build và product flavor bổ sung. Bạn cũng có thể ghi đè các thiết lập trong tệp AndroidManifest.xml hoặc tệp build.gradle cấp cao nhất.

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

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



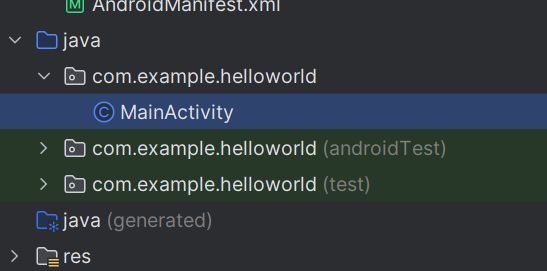


4. Nhấp vào biểu tượng tam giác để đóng phần Gradle Scripts

2.4 Khám phá thư mục app và res

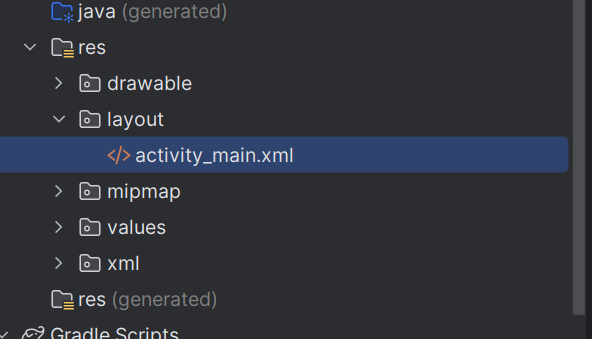
Tất cả mã nguồn và tài nguyên cho ứng dụng đều được đặt trong các thư mục app và res.

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



Thư mục java bao gồm các tệp class Java trong ba thư mục con, như được hiển thị trong hình trên. Thư mục com.example.hello.helloworld (hoặc tên miền mà bạn đã chỉ định) chứa tất cả các tệp cho một gói ứng dụng. Hai thư mục khác được sử dụng cho việc kiểm thử và sẽ được mô tả trong bài học khác. Đối với ứng dụng Hello World, chỉ có một gói và nó chứa MainActivity.java. Tên của Activity (màn hình) đầu tiên mà người dùng nhìn thấy, cũng là nơi khởi tạo các tài nguyên toàn ứng dụng, theo thông lệ được gọi là MainActivity (phần mở rộng tệp được bỏ qua trong khung Project > Android).

2. Mở rộng thư mục res và thư mục layout, và nhấp đúp vào tệp activity\_main.xml để mở nó trong trình soạn thảo layout.



Thư mục res chứa các tài nguyên như layout, chuỗi và hình ảnh. Một Activity thường được liên kết với một layout của các view UI được định nghĩa dưới dạng tệp XML. Tệp này thường được đặt tên theo Activity của nó.

2.5 Khám phá thư mục manifests

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

1. Mở rộng thư mục manifests.
2. Mở tệp AndroidManifest.xml.

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

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

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

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

//CHƯA XONG

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

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

Những thứ bạn cần:

● Một thiết bị Android như điện thoại hoặc máy tính bảng.

● Một cáp dữ liệu để kết nối thiết bị Android của bạn với máy tính thông qua cổng USB.

● Nếu bạn đang sử dụng hệ thống Linux hoặc Windows, bạn có thể cần thực hiện các bước bổ sung để chạy trên thiết bị phần cứng. Hãy 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 trên Windows, hãy xem phần OEM USB Drivers.

4.1 Bật chế độ gỡ lỗi USB

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

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

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

Tôi sẽ dịch đoạn văn này sang tiếng Việt:

4.2 Chạy ứng dụng của bạn trên thiết bị

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

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

2. Nhấp vào nút Run trong thanh công cụ. Cửa sổ Select Deployment Target (Chọn Mục tiêu Triển khai) mở ra với danh sách các trình giả lập khả dụng và các thiết bị đã kết nối.

3. Chọn thiết bị của bạn và nhấp OK.

Tôi sẽ dịch đoạn văn này sang tiếng Việt:

**Xử lý sự cố**

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

1. Rút và cắm lại thiết bị của bạn.

2. Khởi động lại Android Studio.

Nếu máy tính của bạn vẫn không tìm thấy thiết bị hoặc thông báo nó "không được ủy quyền", hãy làm theo các bước sau:

1. Rút thiết bị ra.

2. Trên thiết bị, mở Tùy chọn nhà phát triển trong ứng dụng Cài đặt.

3. Nhấn Thu hồi ủy quyền Gỡ lỗi USB.

4. Kết nối lại thiết bị với máy tính của bạn.

5. Khi được nhắc, hãy cấp ủy quyền.

Task 5: Thay đổi cấu hình Gradle của ứng dụng

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

5.1 Thay đổi phiên bản SDK tối thiểu cho ứng dụng

Làm theo các bước sau:

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

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

Trong khối defaultConfig, thay đổi giá trị của minSdkVersion thành 24 như bên dưới



5.2 Đồng bộ hóa cấu hình Gradle mới

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

Để đồng bộ hóa các tệp dự án, nhấp vào Sync Now trong thanh thông báo xuất hiện khi thực hiện thay đổi (như hiển thị trong hình trước), hoặc nhấp vào biểu tượng Sync Project with Gradle Files trong thanh công cụ.

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

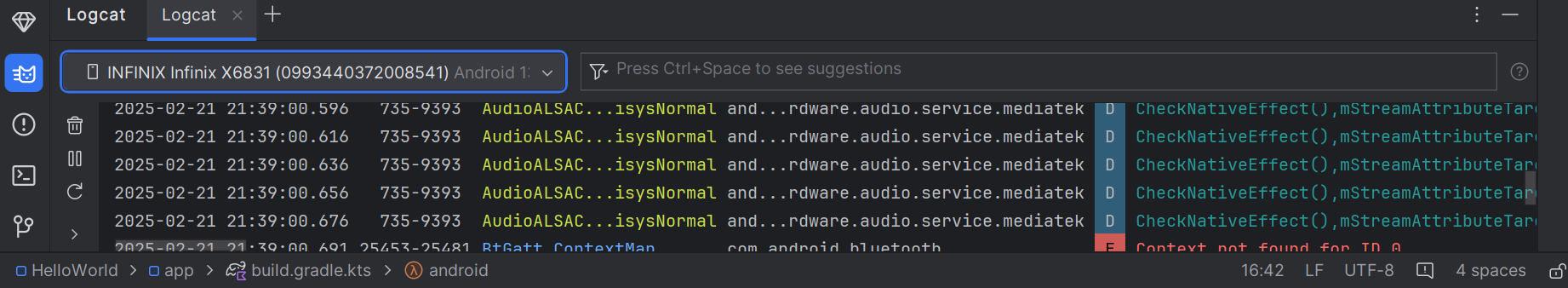
Để tìm hiểu sâu hơn về Gradle, hãy xem tài liệu Tổng quan về Hệ thống Build và Cấu hình Build Gradle.

Task 6: Thêm câu lệnh log vào ứng dụng của bạn

Trong task này, bạn sẽ thêm các câu lệnh Log vào ứng dụng của mình, để hiển thị các thông báo trong cửa sổ Logcat. Các thông báo log là một công cụ gỡ lỗi mạnh mẽ mà bạn có thể sử dụng để kiểm tra giá trị, đường dẫn thực thi và báo cáo các ngoại lệ.

6.1 Xem cửa sổ Logcat

Để xem cửa sổ Logcat, nhấp vào tab Logcat ở dưới cùng của cửa sổ Android Studio như được hiển thị trong hình bên dưới.



### Giao diện người dùng tương tác đầu tiên

Giới thiệu

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

1. **TextView** để hiển thị văn bản.
2. **EditText** cho phép người dùng nhập và chỉnh sửa văn bản.
3. **Button** và các phần tử có thể nhấp khác (chẳng hạn như RadioButton, CheckBox và Spinner) để cung cấp hành vi tương tác.
4. **ScrollView** và **RecyclerView** để hiển thị các mục có thể cuộn.
5. **ImageView** để hiển thị hình ảnh.
6. **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 kế thừa từ Activity. Một Activity thường được liên kết với một bố cục của các view UI được định nghĩa dưới dạng tệp XML (Ngôn ngữ đánh dấu mở rộng). Tệp XML này thường được đặt tên theo Activity của nó và định nghĩa bố cục của các phần tử View trên màn hình.

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

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

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

Những điều bạn cần biết trước

Bạn cần làm quen với:

Cách cài đặt và mở Android Studio.

Cách tạo ứng dụng HelloWorld.

Cách chạy ứng dụng HelloWorld.

Những điều bạn sẽ học

Cách tạo một ứng dụng có hành vi tương tác.

Cách sử dụng trình chỉnh sửa bố cục để thiết kế bố cục.

Cách chỉnh sửa bố cục trong XML.

Rất nhiều thuật ngữ mới. Hãy xem bảng chú giải các từ và khái niệm trong phần Từ vựng để biết định nghĩa thân thiện.

Những điều bạn sẽ làm

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

Thao tác từng phần tử trong ConstraintLayout để ràng buộc chúng với lề và các phần tử khác.

Thay đổi thuộc tính của phần tử UI.

Chỉnh sửa bố cục của ứng dụng trong XML.

Khai thác các chuỗi được mã hóa cứng vào tài nguyên chuỗi.

Triển khai các phương thức xử lý nhấp chuột để hiển thị thông báo trên màn hình khi người dùng chạm vào mỗi Button.

Tổng quan về ứng dụng

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

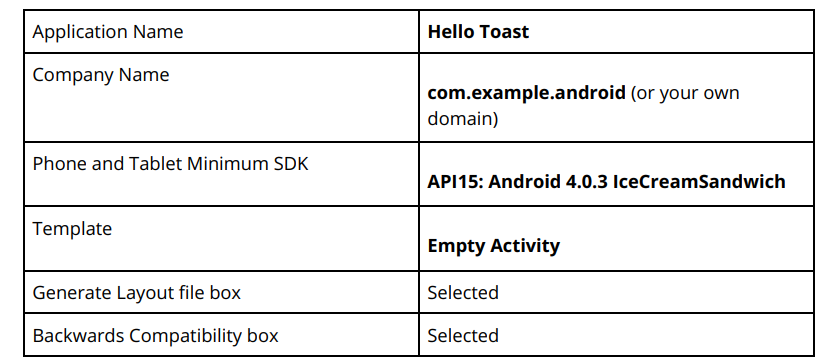
Đây là giao diện của ứng dụng khi hoàn thành:

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

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

1.1 Tạo dự án Android Studio

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



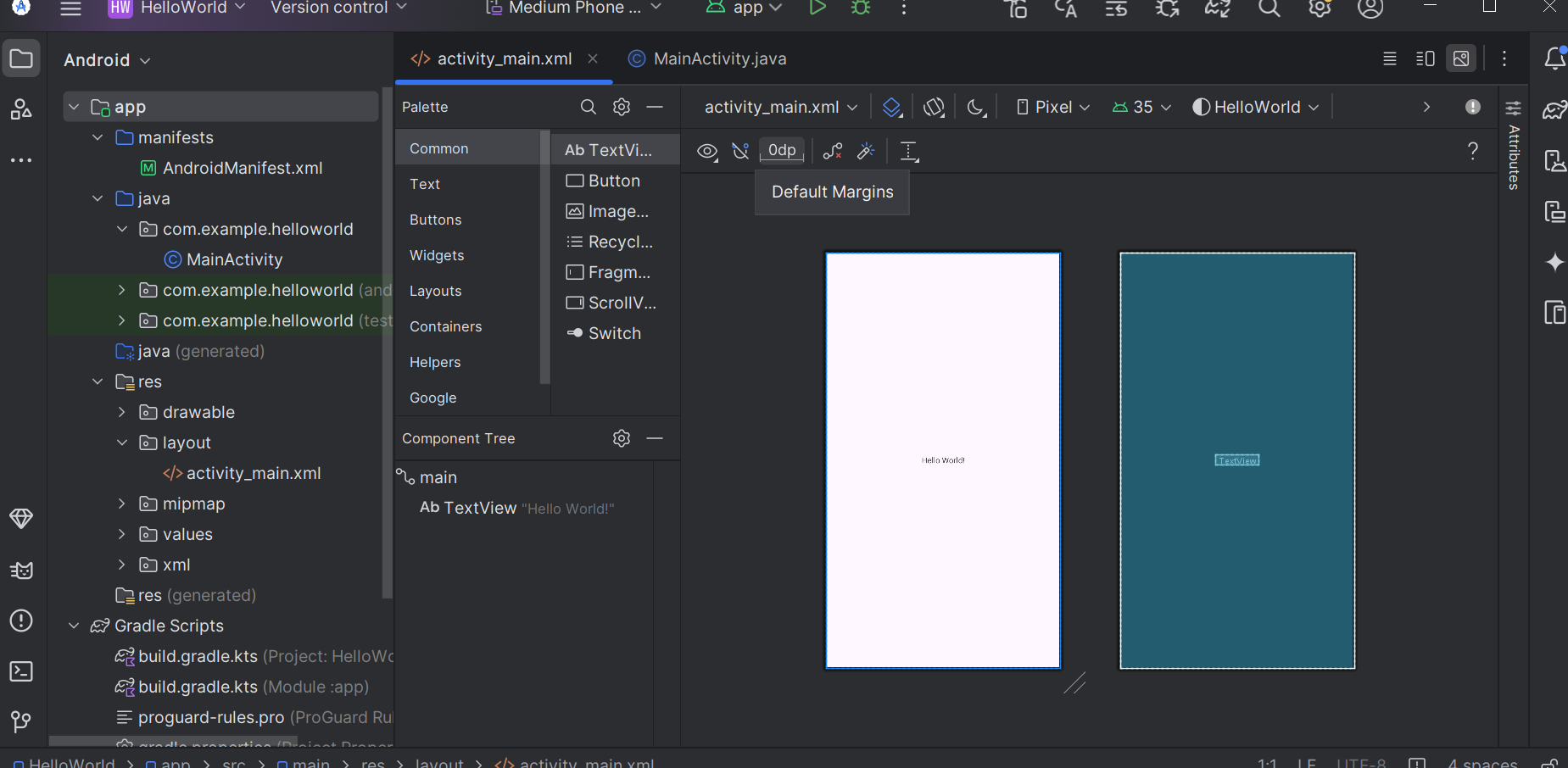
Chọn Run > Run app hoặc nhấp vào biểu tượng Run trên thanh công cụ để biên dịch và chạy ứng dụng trên trình giả lập hoặc thiết bị của bạn.

1.2 Khám phá trình soạn thảo layout

Android Studio cung cấp trình soạn thảo layout để xây dựng nhanh giao diện người dùng (UI) của ứng dụng. Nó cho phép bạn kéo thả các phần tử vào màn hình thiết kế và bản thiết kế, định vị chúng trong layout, thêm các ràng buộc và thiết lập thuộc tính. Các ràng buộc (constraints) xác định vị trí của một phần tử UI trong layout. Một ràng buộc thể hiện kết nối hoặc căn chỉnh với view khác, layout cha, hoặc một đường dẫn ẩn.

Hãy khám phá trình soạn thảo layout và tham khảo hình bên dưới khi bạn làm theo các bước được đánh số:

1 3 2



4 5 6

1. Trong thư mục app > res > layout trong cửa sổ Project > Android, nhấp đúp vào tệp activity\_main.xml để mở nó (nếu chưa được mở).
2. Nhấp vào tab Design nếu nó chưa được chọn. Bạn sử dụng tab Design để thao tác với các phần tử và layout, và tab Text để chỉnh sửa mã XML cho layout.
3. Cửa sổ Palettes hiển thị các phần tử UI mà bạn có thể sử dụng trong layout của ứng dụng.
4. Cửa sổ Component tree hiển thị cấu trúc phân cấp của các phần tử UI. Các phần tử View được tổ chức theo cấu trúc cây gồm các phần tử cha và con, trong đó phần tử con kế thừa các thuộc tính của phần tử cha. Trong hình trên, TextView là phần tử con của ConstraintLayout. Bạn sẽ tìm hiểu về các phần tử này sau trong bài học.
5. Các cửa sổ thiết kế và bản thiết kế của trình soạn thảo layout hiển thị các phần tử UI trong layout. Trong hình trên, layout chỉ hiển thị một phần tử: TextView hiển thị dòng chữ "Hello World".
6. Tab Attributes hiển thị cửa sổ Attributes để thiết lập các thuộc tính cho phần tử UI.

Nhiệm vụ 2: Thêm các phần tử View trong trình soạn thảo layout

Trong nhiệm vụ này, bạn sẽ tạo layout UI cho ứng dụng HelloToast trong trình soạn thảo layout sử dụng các tính năng của ConstraintLayout. Bạn có thể tạo các ràng buộc thủ công, như được hiển thị sau này, hoặc tự động bằng công cụ Autoconnect.

2.1 Kiểm tra các ràng buộc phần tử

Thực hiện theo các bước sau:

1. Mở activity\_main.xml từ cửa sổ Project > Android nếu nó chưa được mở. Nếu tab Design chưa được chọn, hãy nhấp vào nó.

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

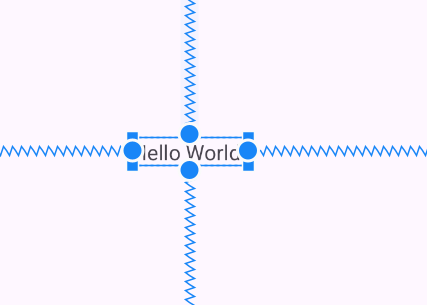
2. Công cụ Autoconnect cũng nằm trên thanh công cụ. Nó được bật theo mặc định. Trong bước này, hãy đảm bảo rằng công cụ không bị tắt.

3. Nhấp vào nút phóng to để phóng to các cửa sổ thiết kế và bản thiết kế để xem chi tiết hơn.

4. Chọn TextView trong cửa sổ Component Tree. TextView "Hello World" sẽ được làm nổi bật trong cửa sổ thiết kế và bản thiết kế và các ràng buộc cho phần tử sẽ hiển thị.

5. Tham khảo hình động bên dưới cho bước này. Nhấp vào điểm điều khiển hình tròn ở bên phải của TextView để xóa ràng buộc ngang gắn view với cạnh phải của layout. TextView sẽ nhảy sang bên trái vì nó không còn bị ràng buộc ở bên phải nữa. Để thêm lại ràng buộc ngang, nhấp vào cùng điểm điều khiển đó và kéo một đường đến cạnh phải của layout.

Trong cửa sổ bản thiết kế hoặc thiết kế, các điểm điều khiển sau xuất hiện trên phần tử TextView:



● Điểm điều khiển ràng buộc (Constraint handle): Để tạo một ràng buộc như hiển thị trong hình động ở trên, nhấp vào điểm điều khiển ràng buộc, được hiển thị dưới dạng hình tròn ở cạnh của phần tử. Sau đó kéo điểm điều khiển đến một điểm điều khiển ràng buộc khác, hoặc đến ranh giới của phần tử cha. Một đường ziczac thể hiện ràng buộc.



● Điểm điều khiển kích thước (Resizing handle): Để thay đổi kích thước phần tử, kéo các điểm điều khiển kích thước hình vuông. Điểm điều khiển sẽ chuyển thành góc nghiêng khi bạn đang kéo nó.

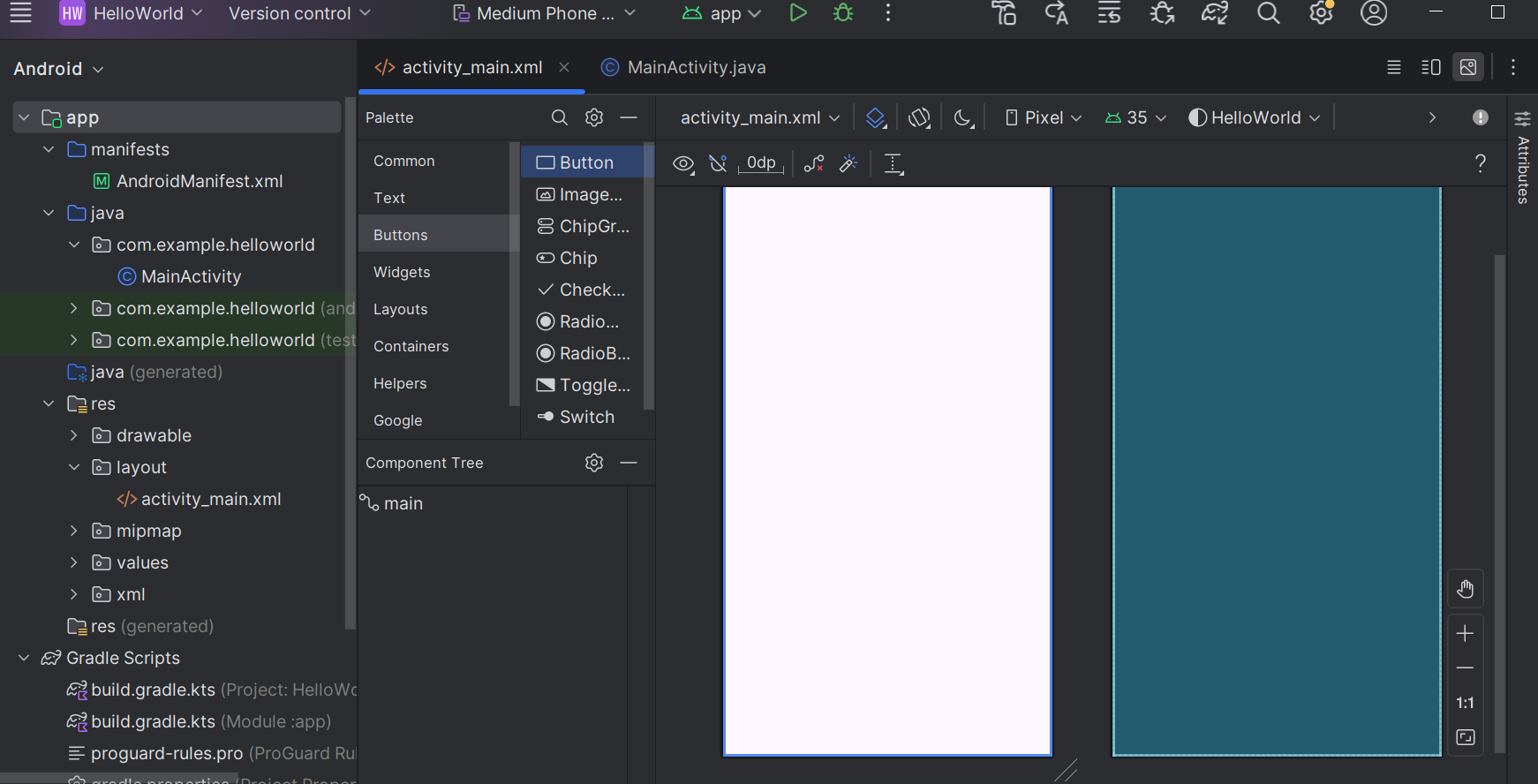


2.2 Thêm một Button vào layout

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

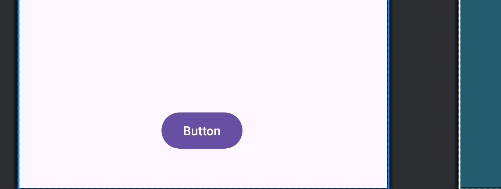
Thực hiện theo các bước sau để thêm một Button:

1. Bắt đầu với một layout trống. Không cần phần tử TextView nữa, vì vậy trong khi nó vẫn được chọn, nhấn phím Delete hoặc chọn Edit > Delete. Bây giờ bạn đã có một layout hoàn toàn trống.
2. Kéo một Button từ cửa sổ Palette đến bất kỳ vị trí nào trong layout. Nếu bạn thả Button ở khu vực giữa phía trên của layout, các ràng buộc có thể tự động xuất hiện. Nếu không, bạn có thể kéo các ràng buộc đến phía trên, cạnh trái và cạnh phải của layout như hiển thị trong hình động bên dưới



2.3 Thêm Button thứ hai vào layout

1. Kéo một Button khác từ cửa sổ Palette vào giữa layout như hiển thị trong hình động bên dưới. Autoconnect có thể tự động tạo các ràng buộc ngang cho bạn (nếu không, bạn có thể tự kéo chúng).
2. Kéo một ràng buộc dọc đến phía dưới của layout (tham khảo hình bên dưới).

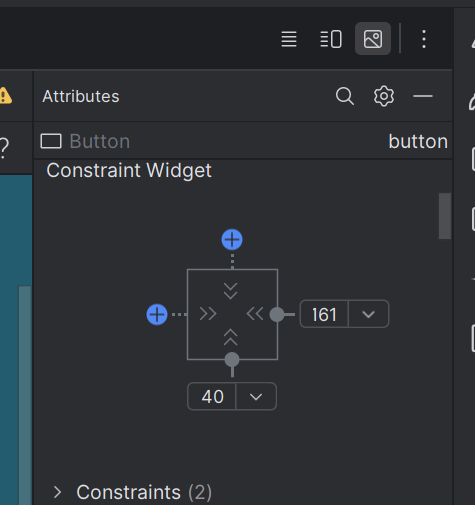


Bạn có thể xóa các ràng buộc khỏi một phần tử bằng cách chọn phần tử đó và di chuột qua nó để hiển thị nút Clear Constraints. Nhấp vào nút này để xóa tất cả các ràng buộc trên phần tử đã chọn. Để xóa một ràng buộc đơn lẻ, nhấp vào điểm điều khiển cụ thể tạo ra ràng buộc đó. Để xóa tất cả các ràng buộc trong toàn bộ layout, nhấp vào công cụ Clear All Constraints trên thanh công cụ. Công cụ này rất hữu ích nếu bạn muốn tạo lại tất cả các ràng buộc trong layout của mình.

3.1 Thay đổi kích thước Button

Trình soạn thảo layout cung cấp các điểm điều khiển kích thước ở cả bốn góc của View để bạn có thể nhanh chóng điều chỉnh kích thước View. Bạn có thể kéo các điểm điều khiển ở mỗi góc của View để thay đổi kích thước, nhưng làm vậy sẽ cố định các kích thước chiều rộng và chiều cao. Nên tránh cố định kích thước cho hầu hết các phần tử View, vì các kích thước cố định không thể thích ứng với các nội dung và kích thước màn hình khác nhau.

Thay vào đó, hãy sử dụng cửa sổ Attributes ở bên phải của trình soạn thảo layout để chọn chế độ kích thước không sử dụng kích thước cố định. Cửa sổ Attributes bao gồm một bảng điều khiển kích thước hình vuông gọi là view inspector ở phía trên. Các ký hiệu bên trong hình vuông thể hiện các thiết lập chiều cao và chiều rộng như sau:

 2

1

Trong hình trên:

1. **Điều khiển chiều cao**. Điều khiển này xác định thuộc tính layout\_height và xuất hiện ở hai phần ở cạnh trên và dưới của hình vuông. Các góc cho biết điều khiển này được đặt thành wrap\_content, nghĩa là View sẽ mở rộng theo chiều dọc khi cần thiết để vừa với nội dung của nó. Số "8" chỉ lề chuẩn được đặt là 8dp.

2. **Điều khiển chiều rộng**. Điều khiển này xác định layout\_width và xuất hiện ở hai phần ở cạnh trái và phải của hình vuông. Các góc cho biết điều khiển này được đặt thành wrap\_content.

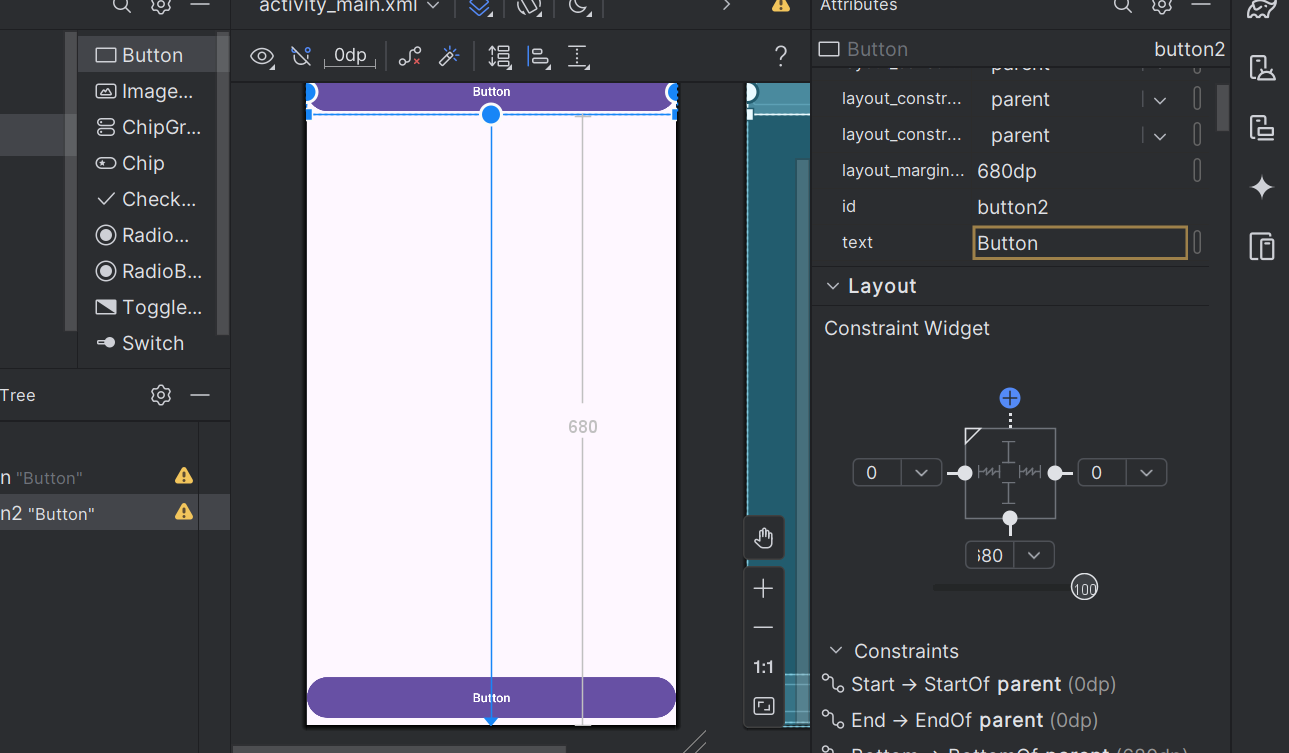
Thực hiện theo các bước sau:

1. Chọn Button phía trên trong cửa sổ Component Tree.

2. Nhấp vào tab Attributes ở bên phải cửa sổ trình soạn thảo layout.

3. Nhấp vào điều khiển chiều rộng hai lần—nhấp lần đầu sẽ thay đổi nó thành Fixed với các đường thẳng, và nhấp lần thứ hai sẽ thay đổi nó thành Match Constraints với các đường xoắn ốc, như hiển thị trong hình động bên dưới.

4. Chọn Button thứ hai, và thực hiện các thay đổi tương tự với layout\_width như trong bước trước đó, như hiển thị trong hình bên dưới.



Kết quả của việc thay đổi điều khiển chiều rộng là thuộc tính layout\_width trong cửa sổ Attributes sẽ hiển thị giá trị match\_constraint và phần tử Button sẽ giãn theo chiều ngang để lấp đầy khoảng trống giữa cạnh trái và cạnh phải của layout.

Tôi sẽ giải thích các bước thay đổi thuộc tính của Button:

3.2 Thay đổi các thuộc tính Button

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

Cửa sổ Attributes cho phép truy cập tất cả các thuộc tính mà bạn có thể gán cho một phần tử View. Bạn có thể nhập giá trị cho từng thuộc tính, như android:id, background, textColor, và text.

Thực hiện theo các bước sau:

1. Sau khi chọn Button đầu tiên, chỉnh sửa trường ID ở đầu cửa sổ Attributes thành button\_toast cho thuộc tính android:id, được sử dụng để xác định phần tử trong layout.

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

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

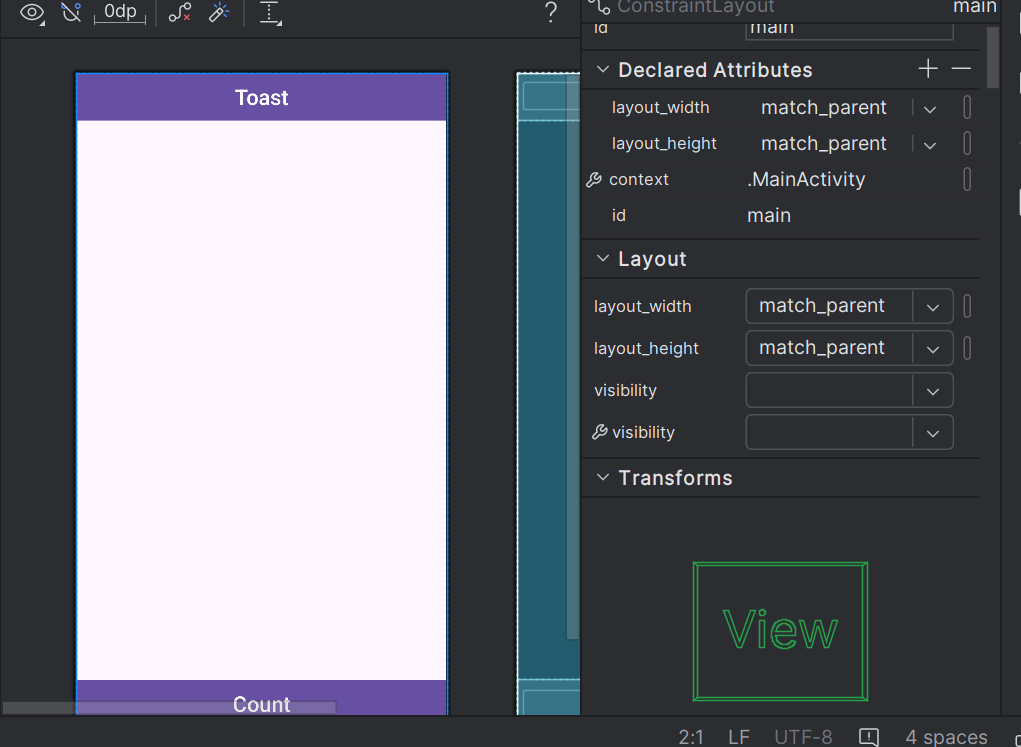
4. Chỉnh sửa thuộc tính text thành Toast.

5. Thực hiện các thay đổi thuộc tính tương tự cho Button thứ hai:

- Sử dụng button\_count làm ID

- Đặt Count làm thuộc tính text

- Sử dụng cùng màu sắc cho background và text như các bước trước



colorPrimary là màu chính của chủ đề, một trong các màu cơ bản được định nghĩa sẵn trong tệp tài nguyên colors.xml. Nó được sử dụng cho thanh ứng dụng (app bar). Việc sử dụng các màu cơ bản cho các phần tử UI khác tạo ra một giao diện đồng nhất. Bạn sẽ tìm hiểu thêm về chủ đề ứng dụng và Material Design trong một bài học khác.

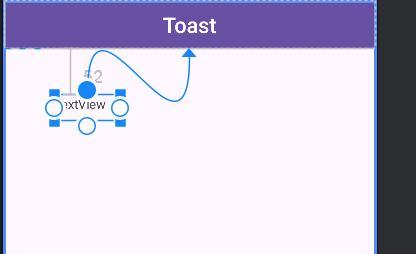
Nhiệm vụ 4: Thêm một TextView và thiết lập thuộc tính của nó

Một trong những lợi ích của ConstraintLayout là khả năng căn chỉnh hoặc ràng buộc các phần tử so với các phần tử khác. Trong nhiệm vụ này, bạn sẽ thêm một TextView vào giữa bố cục và ràng buộc nó theo chiều ngang với lề, cũng như theo chiều dọc với hai nút Button. Sau đó, bạn sẽ thay đổi các thuộc tính của TextView trong bảng Attributes.

4.1 Thêm một TextView và thiết lập ràng buộc

Như minh họa trong hình động bên dưới, kéo một TextView từ Palette pane vào phần trên của bố cục.

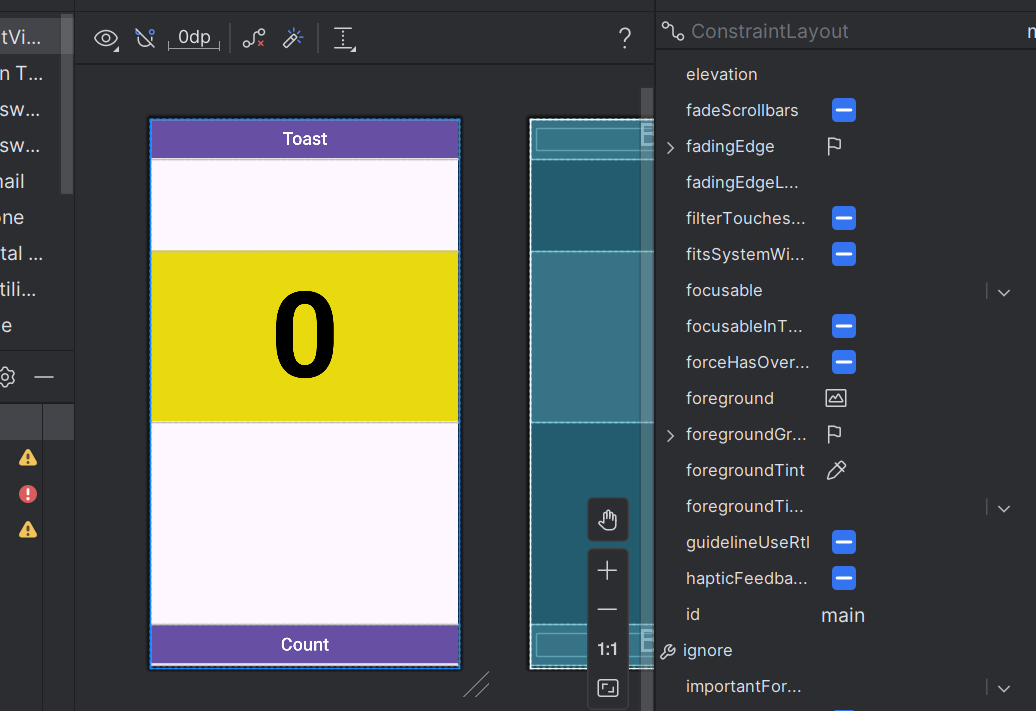
Kéo một ràng buộc từ phía trên của TextView đến tay cầm ở phía dưới của nút Toast Button. Điều này sẽ ràng buộc TextView nằm bên dưới nút Button.



4.2 Thiết lập thuộc tính cho TextView

Khi TextView được chọn, hãy mở bảng Attributes (nếu chưa mở). Thiết lập các thuộc tính cho TextView như trong hình động bên dưới. Những thuộc tính mới mà bạn chưa gặp sẽ được giải thích sau hình minh họa:

1. Đặt ID thành show\_count.
2. Đặt text thành 0.
3. Đặt textSize thành 160sp.
4. Đặt textStyle thành B (bold) và textAlignment thành ALIGNCENTER (căn giữa đoạn văn bản).
5. Thay đổi kích thước ngang và dọc (layout\_width và layout\_height) thành match\_constraint.
6. Đặt textColor thành @color/colorPrimary.
7. Cuộn xuống trong bảng Attributes, nhấp vào View all attributes, tiếp tục cuộn xuống trang thứ hai để tìm thuộc tính background, sau đó nhập #FFF00 để đặt màu nền thành một sắc vàng.
8. Cuộn xuống gravity, mở rộng tùy chọn này và chọn center\_ver (căn giữa theo chiều dọc).



Nhiệm vụ 5: Chỉnh sửa bố cục trong XML

Ứng dụng Hello Toast gần như hoàn thành! Tuy nhiên, trong Component Tree, mỗi phần tử giao diện người dùng (UI element) xuất hiện một dấu chấm than. Khi di chuột qua dấu này, bạn sẽ thấy thông báo cảnh báo.

Tất cả các phần tử đều gặp chung một cảnh báo: Các chuỗi văn bản không nên được mã hóa cứng (hardcoded), mà cần sử dụng tài nguyên chuỗi (string resources).

Mặc dù layout editor rất mạnh mẽ, nhưng một số chỉnh sửa sẽ dễ dàng hơn khi thực hiện trực tiếp trong mã XML.

5.1 Mở mã XML của bố cục

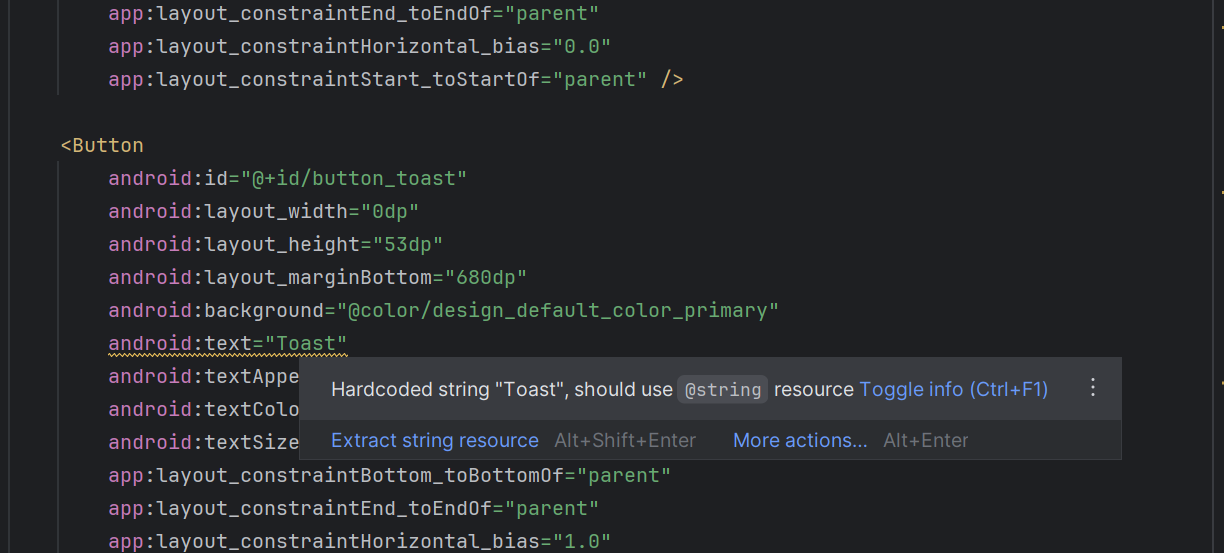
Mở tệp activity\_main.xml (nếu chưa mở).

Nhấp vào tab Text ở phía dưới layout editor để chuyển sang chế độ chỉnh sửa XML.

Sau khi nhấp vào tab Text, trình chỉnh sửa XML sẽ mở ra, thay thế các ngăn Design và Blueprint.

Như trong hình minh họa bên dưới, một phần của mã XML cho bố cục sẽ được hiển thị, với các cảnh báo được đánh dấu. Các chuỗi văn bản được mã hóa cứng (hardcoded strings) như "Toast" và "Count" bị gạch chân để cảnh báo. (Chuỗi "0" cũng bị đánh dấu nhưng không xuất hiện trong hình minh họa.)

Di chuột qua chuỗi hardcoded "Toast" để xem thông báo cảnh báo.



Thay vì hard-code (mã hóa cứng) các chuỗi văn bản, cách tốt nhất là sử dụng string resources. Điều này giúp:

Dễ dàng quản lý các chuỗi văn bản, đặc biệt khi chúng được sử dụng nhiều lần.

Bắt buộc đối với việc dịch thuật và bản địa hóa ứng dụng, vì bạn cần tạo một tệp tài nguyên chuỗi cho từng ngôn ngữ.

1. Nhấp một lần vào từ "Toast" (cảnh báo đầu tiên được đánh dấu).
2. Windows: Nhấn Alt + Enter hoặc macOS: Nhấn Option + Enter, sau đó chọn Extract string resource từ menu bật lên.
3. Đặt Resource name thành button\_label\_toast.
4. Nhấp OK. Chuỗi "Toast" sẽ được tạo thành một tài nguyên trong tệp res/values/strings.xml, và trong mã XML, nó được thay thế bằng:

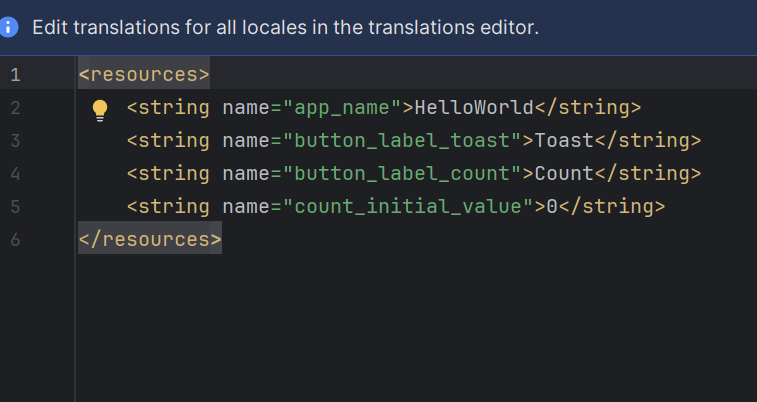
@string/button\_label\_toast

1. Thực hiện tương tự cho các chuỗi còn lại:

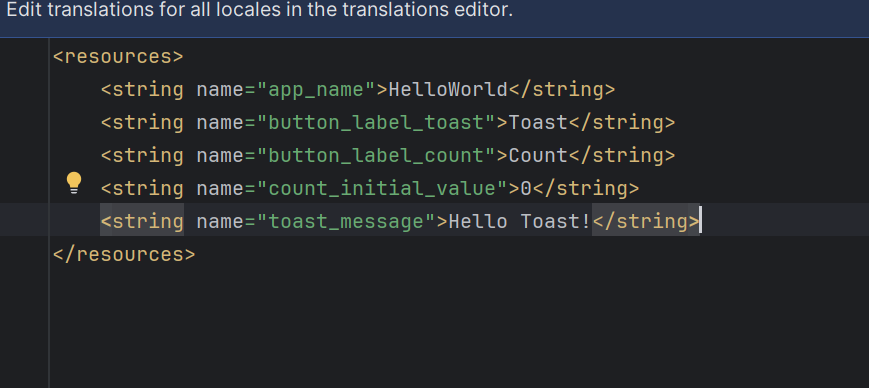
"Count" → button\_label\_count

"0" → count\_initial\_value

1. Mở Project > Android pane, mở rộng thư mục res > values, sau đó double-click vào tệp strings.xml để xem các tài nguyên chuỗi trong đó.



1. Bạn cần một chuỗi khác để sử dụng trong nhiệm vụ tiếp theo hiển thị thông báo.Thêm vào tệp strings.xml một tài nguyên chuỗi khác có tên toast\_message với nội dung "Hello Toast!":



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

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

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

Trình xử lý click (click handler) là một phương thức được gọi khi người dùng nhấp hoặc chạm vào một phần tử UI có thể nhấp được. Trong Android Studio, bạn có thể chỉ định tên của phương thức trong trường onClick trong cửa sổ 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 Button. Bạn sẽ sử dụng phương pháp sau vì bạn chưa tạo các phương thức xử lý, và trình soạn thảo XML cung cấp cách tự động để tạo các phương thức đó.

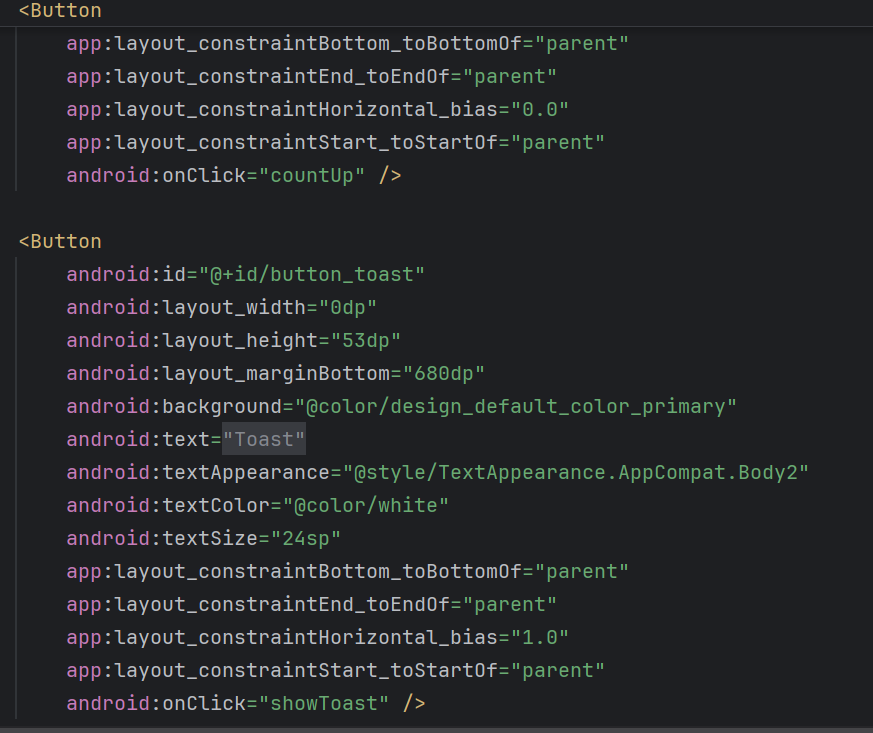
Với trình soạn thảo XML đang mở (tab Text), tìm Button có android:id được đặt thành button\_toast:

1. Thêm thuộc tính android:onClick vào cuối phần tử button\_toast sau thuộc tính cuối cùng và trước dấu kết thúc />:

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

1. Nhấp vào biểu tượng bóng đèn đỏ xuất hiện bên cạnh thuộc tính. Chọn Create click handler, chọn MainActivity, và nhấp OK.
2. Nếu biểu tượng bóng đèn đỏ không xuất hiện, nhấp vào tên phương thức ("showToast"). Nhấn Alt-Enter (Option-Enter trên Mac), chọn Create 'showToast(view)' trong MainActivity, và nhấp OK.
3. Hành động này tạo một phương thức stub placeholder cho phương thức showToast() trong MainActivity, như được hiển thị ở cuối các bước này.
4. Lặp lại hai bước cuối cùng với Button button\_count: Thêm thuộc tính android:onClick vào cuối, và thêm trình xử lý click:

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

******

1. Nếu MainActivity.java chưa được mở, hãy mở rộng phần java trong giao diện Project > Android, mở rộng com.example.android.hellotoast, và sau đó nhấp đúp vào MainActivity. Trình soạn thảo mã sẽ xuất hiện với mã trong MainActivity:



Để giúp bạn tạo một Toast message, hãy làm theo các bước sau:

6.2 Chỉnh sửa trình xử lý nút Toast

Bây giờ bạn sẽ chỉnh sửa phương thức showToast() - trình xử lý click cho nút Toast trong MainActivity - để nó hiển thị một thông báo. Toast cung cấp cách để hiển thị một thông điệp đơn giản trong cửa sổ popup nhỏ. Nó chỉ chiếm không gian cần thiết cho thông điệp. Activity hiện tại vẫn hiển thị và có thể tương tác được. 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 của việc nhấn Button hoặc thực hiện một hành động.

Thực hiện theo các bước sau để chỉnh sửa trình xử lý click cho nút Toast:

1. Tìm phương thức showToast() vừa được tạo:

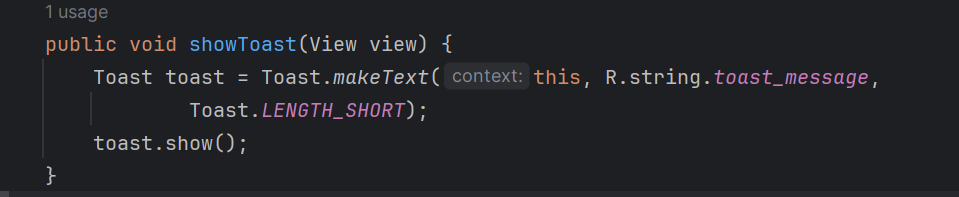
***public void showToast(View view) {***

***}***

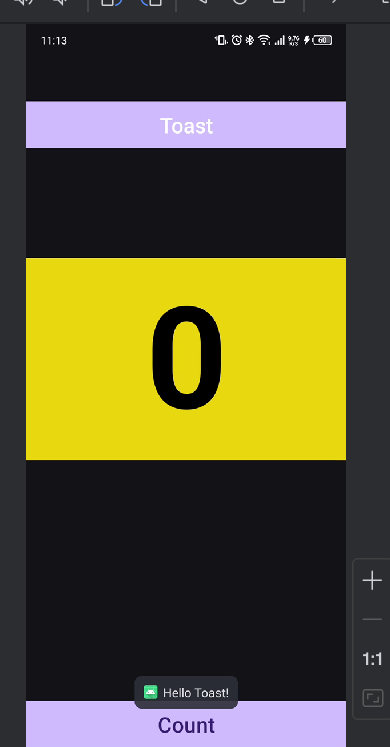
1. Để tạo một thể hiện của Toast, hãy gọi phương thức makeText() trên lớp Toast
2. Cung cấp context của Activity trong ứng dụng. Vì Toast hiển thị phía trên giao diện Activity, hệ thống cần thông tin về Activity hiện tại. Khi bạn đã ở trong context của Activity mà bạn cần, sử dụng this như một cách viết tắt.
3. Cung cấp thông điệp để hiển thị, như một tài nguyên chuỗi (toast\_message mà bạn đã tạo ở bước trước). Tài nguyên chuỗi toast\_message được xác định bởi R.string.
4. Cung cấp thời gian hiển thị. Ví dụ, Toast.LENGTH\_SHORT hiển thị toast trong một khoảng thời gian tương đối ngắn.

*Thời gian hiển thị của Toast có thể là Toast.LENGTH\_LONG hoặc Toast.LENGTH\_SHORT. Thời gian thực tế khoảng 3.5 giây cho Toast dài và 2 giây cho Toast ngắn.*

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



Run the app and verify that the Toast message appears when the Toast button is tapped



6.3 Chỉnh sửa trình xử lý nút Count

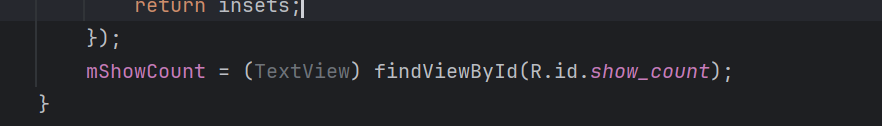
Bây giờ bạn sẽ chỉnh sửa phương thức countUp() - trình xử lý click cho nút Count trong MainActivity - để hiển thị số đếm hiện tại sau khi nhấn Count. Mỗi lần nhấn sẽ tăng số đếm lên một.

Mã cho trình xử lý phải:

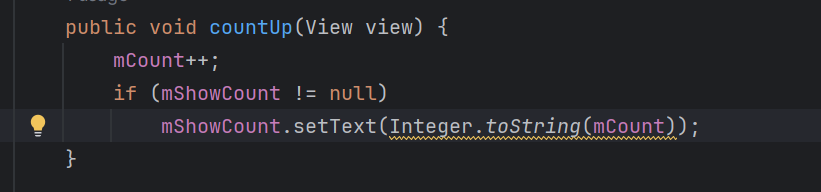
* Theo dõi số đếm khi nó thay đổi.
* Gửi số đếm đã cập nhật đến TextView để hiển thị.

Thực hiện theo các bước sau để chỉnh sửa trình xử lý click cho nút Count:

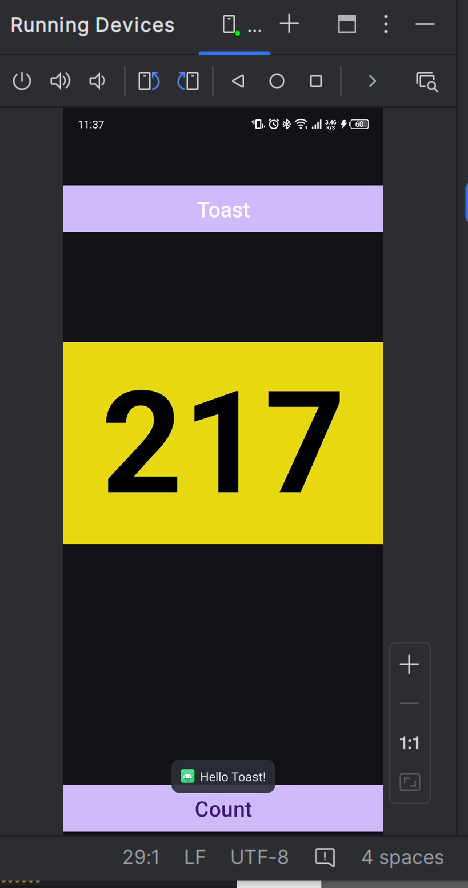
1. Tìm phương thức countUp() vừa được tạo:
2. Để theo dõi số đếm, bạn cần một biến thành viên private. Mỗi lần nhấn nút Count sẽ tăng giá trị của biến này.
3. Cùng với biến ở trên, bạn cũng cần một biến thành viên private để tham chiếu đến TextView show\_count, biến này sẽ được thêm vào trình xử lý click. Đặt tên biến này là mShowCount
4. Thêm câu lệnh findViewById vào cuối phương thức:



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



1. Chạy ứng dụng để kiểm tra xem số đếm có tăng lên khi bạn nhấn nút Count hay không.



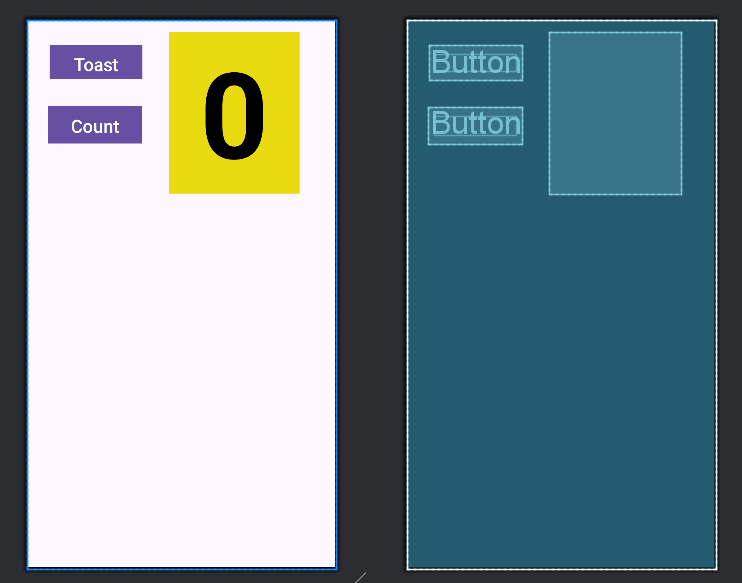
Thử thách lập trình

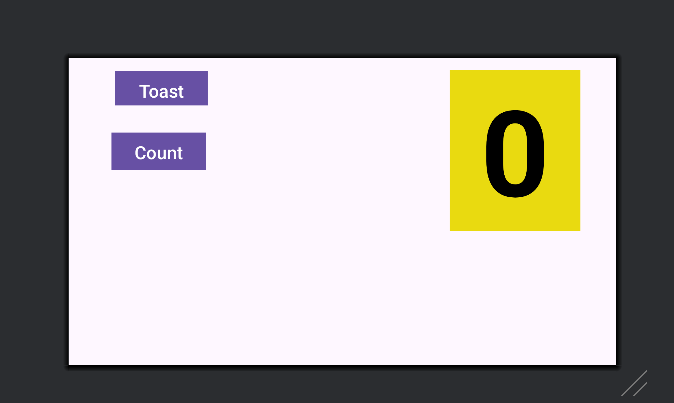
Lưu ý: Tất cả các thử thách lập trình đều không bắt buộc và không ảnh hưởng đến các bài học sau.

Ứng dụng HelloToast hiển thị tốt khi thiết bị hoặc trình giả lập ở chế độ dọc. Tuy nhiên, khi chuyển sang chế độ ngang, nút Count Button có thể chồng lên TextView ở phía dưới

Thử thách: Thay đổi bố cục sao cho hiển thị tốt ở cả chế độ ngang và dọc:

1. Trên máy tính, sao chép thư mục dự án HelloToast và đổi tên thành HelloToastChallenge.
2. Mở HelloToastChallenge trong Android Studio và refactor nó. (Xem Phụ lục: Tiện ích để biết hướng dẫn sao chép và refactor dự án.)
3. Thay đổi bố cục sao cho nút Toast và nút Count xuất hiện ở bên trái, như trong hình minh họa. TextView xuất hiện bên cạnh chúng, nhưng chỉ đủ rộng để hiển thị nội dung. (Gợi ý: Sử dụng wrap\_content.)
4. Chạy ứng dụng ở cả chế độ ngang và dọc.





Tóm tắt

View, ViewGroup và layout:

- Tất cả các phần tử giao diện người dùng (UI elements) đều là lớp con của lớp View, do đó kế thừa nhiều thuộc tính của lớp View cha.

- Các phần tử View có thể được nhóm bên trong một ViewGroup, đóng vai trò như một vùng chứa. Mối quan hệ giữa chúng là cha-con, trong đó ViewGroup là cha và View hoặc một ViewGroup khác là con.

- Phương thức onCreate() được sử dụng để inflate layout, tức là đặt nội dung hiển thị của màn hình thành tệp XML layout. Bạn cũng có thể sử dụng nó để lấy tham chiếu đến các phần tử UI khác trong layout.

- Một View, giống như một chuỗi (string), là một tài nguyên có thể có ID. Lệnh findViewById sử dụng ID của một View làm tham số và trả về View đó.

Sử dụng trình chỉnh sửa layout:

- Nhấn vào tab Design để thao tác với các phần tử và layout, hoặc tab Text để chỉnh sửa mã XML của layout.

- Trong tab Design, bảng Palettes hiển thị các phần tử UI mà bạn có thể sử dụng trong layout, còn bảng Component Tree hiển thị hệ thống phân cấp của các phần tử UI.

- Các bảng thiết kế và sơ đồ (blueprint) trong trình chỉnh sửa layout hiển thị các phần tử UI trong layout.

- Tab Attributes hiển thị bảng thuộc tính để đặt các thuộc tính cho một phần tử UI.

- **Constraint handle**: Nhấn vào tay cầm ràng buộc (hình tròn ở mỗi cạnh của phần tử), sau đó kéo đến một tay cầm khác hoặc đến ranh giới cha để tạo ràng buộc. Ràng buộc được biểu thị bằng một đường gấp khúc.

- **Resizing handle**: Kéo tay cầm vuông để thay đổi kích thước phần tử. Khi kéo, tay cầm sẽ chuyển thành một góc nghiêng.

- Khi được bật, công cụ \*\*Autoconnect\*\* sẽ tự động tạo hai hoặc nhiều ràng buộc cho một phần tử UI với layout cha. Khi bạn kéo phần tử vào layout, nó sẽ tạo ràng buộc dựa trên vị trí của phần tử.

- Bạn có thể \*\*xóa ràng buộc\*\* bằng cách chọn phần tử, di chuột lên nó để hiển thị nút Clear Constraints, rồi nhấn vào nút này để xóa tất cả ràng buộc của phần tử đã chọn. Để xóa một ràng buộc cụ thể, nhấn vào tay cầm đặt ràng buộc đó.

- Bảng Attributes cung cấp quyền truy cập vào tất cả các thuộc tính XML có thể gán cho một phần tử UI. Nó cũng bao gồm một bảng kiểm tra kích thước (view inspector) ở trên cùng, với các ký hiệu bên trong thể hiện cài đặt chiều cao và chiều rộng.

Cài đặt chiều rộng và chiều cao layout:

Các thuộc tính layout\_width và layout\_height thay đổi khi bạn thay đổi kích thước trong bảng kiểm tra kích thước. Đối với ConstraintLayout, các giá trị này có thể là:

- match\_constraint: Mở rộng phần tử để lấp đầy vùng chứa theo chiều rộng hoặc chiều cao (giới hạn trong margin nếu có).

- wrap\_content: Giữ kích thước phần tử vừa đủ để chứa nội dung. Nếu không có nội dung, phần tử sẽ trở nên vô hình.

- Sử dụng giá trị dp (density-independent pixels) để đặt kích thước cố định, được điều chỉnh theo kích thước màn hình thiết bị.

Trích xuất tài nguyên chuỗi (string resources):

Thay vì viết cứng chuỗi trong mã nguồn, nên sử dụng tài nguyên chuỗi để dễ quản lý và hỗ trợ đa ngôn ngữ. Các bước thực hiện:

1. Nhấn vào chuỗi cần trích xuất, bấm Alt-Enter (hoặc Option-Enter trên Mac), và chọn **Extract string resource** từ menu xuất hiện.

2. Đặt tên cho tài nguyên chuỗi.

3. Nhấn OK để tạo một tài nguyên chuỗi trong tệp values/res/strings.xml, đồng thời thay thế chuỗi trong mã nguồn bằng tham chiếu đến tài nguyên: @string/button\_label\_toast.

Xử lý sự kiện nhấn (click handling):

- Một click handler là phương thức được gọi khi người dùng nhấn hoặc chạm vào một phần tử UI.

- Để chỉ định một click handler cho phần tử UI (như Button), bạn có thể nhập tên phương thức vào trường onClick trong bảng thuộc tính (Attributes) của tab Design, hoặc thêm thuộc tính android:onClick vào mã XML của phần tử UI.

- Tạo click handler trong Activity chính bằng cách sử dụng tham số View. Ví dụ:

public void showToast(View view) {

// Xử lý sự kiện nhấn

}

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

Hiển thị thông báo Toast:

Toast cung cấp cách hiển thị một thông báo đơn giản trong một cửa sổ pop-up nhỏ. Nó chỉ chiếm không gian cần thiết để hiển thị thông điệp. Để tạo một Toast, thực hiện các bước sau:

1. Gọi phương thức makeText() của lớp Toast.

2. Cung cấp ngữ cảnh của Activity và thông điệp cần hiển thị (ví dụ: một chuỗi tài nguyên).

3. Xác định khoảng thời gian hiển thị, chẳng hạn Toast.LENGTH\_SHORT (ngắn) hoặc Toast.LENGTH\_LONG (dài).

4. Gọi phương thức show() để hiển thị Toast.

Khái niệm liên quan:

Tài liệu khái niệm liên quan có trong bài 1.2: \*\*Layouts và tài nguyên cho UI\*\*.

### Trình chỉnh sửa bố cục

Giới thiệu

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

ConstraintLayout là một **ViewGroup**, một loại View đặc biệt có thể chứa các đối tượng View khác (gọi là child views). Bài thực hành này sẽ giới thiệu thêm nhiều tính năng của ConstraintLayout và trình chỉnh sửa layout.

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

- **LinearLayout**: Một nhóm sắp xếp các phần tử con theo chiều ngang hoặc chiều dọc.

- **RelativeLayout**: Một nhóm trong đó các phần tử con được định vị và căn chỉnh dựa trên các phần tử khác trong cùng ViewGroup. Vị trí của các phần tử con được mô tả theo quan hệ với nhau hoặc với ViewGroup cha.

Những gì bạn cần biết trước

Bạn nên có khả năng:

- Tạo một ứng dụng Hello World bằng Android Studio.

- Chạy ứng dụng trên trình giả lập hoặc thiết bị thực.

- Tạo một layout đơn giản bằng ConstraintLayout.

- Trích xuất và sử dụng tài nguyên chuỗi (string resources).

Những gì bạn sẽ học

- Cách tạo một biến thể layout cho chế độ ngang (landscape).

- Cách tạo một biến thể layout cho máy tính bảng và màn hình lớn hơn.

- Cách sử dụng baseline constraint để căn chỉnh phần tử UI với văn bản.

- Cách sử dụng nút Pack và Align để căn chỉnh phần tử trong layout.

- Cách định vị các View trong LinearLayout.

- Cách định vị các View trong RelativeLayout.

Những gì bạn sẽ làm

- Tạo một biến thể layout cho màn hình ngang.

- Tạo một biến thể layout cho máy tính bảng và màn hình lớn.

- Chỉnh sửa layout để thêm các ràng buộc (constraints) cho phần tử UI.

- Sử dụng baseline constraint trong ConstraintLayout để căn chỉnh phần tử với văn bản.

- Sử dụng các nút Pack và Align trong ConstraintLayout để căn chỉnh phần tử.

- Thay đổi layout để sử dụng LinearLayout.

- Định vị các phần tử trong LinearLayout.

- Thay đổi layout để sử dụng RelativeLayout.

- Sắp xếp lại các View trong layout chính sao cho chúng liên quan đến 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 của Activity, như trong hình dưới đây.

Để thực hành thêm với ConstraintLayout, bạn sẽ tạo một phiên bản khác của bố cục này dành cho chế độ hiển thị ngang, như hình dưới đây.

Bạn cũng sẽ học cách sử dụng ràng buộc đường cơ sở (baseline constraints) và một số tính năng căn chỉnh của ConstraintLayout bằng cách tạo một bố cục khác dành cho màn hình máy tính bảng.

Ngoài ra, bạn sẽ tìm hiểu về các lớp con khác của ViewGroup như LinearLayout và RelativeLayout, đồng thời thay đổi bố cục của ứng dụng Hello Toast để sử dụng chúng.

Nhiệm vụ 1: Tạo các biến thể bố cục

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

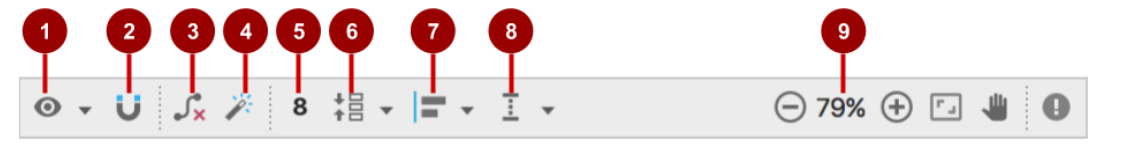
Trong nhiệm vụ này, bạn sẽ sử dụng một số nút trong hai thanh công cụ trên cùng của trình chỉnh sửa bố cục. Thanh công cụ trên cùng cho phép bạn cấu hình giao diện của bản xem trước bố cục trong trình chỉnh sửa:



Trong hình trên:

1. **Chọn bề mặt thiết kế**: Chọn "Design" để hiển thị bản xem trước có màu của bố cục hoặc "Blueprint" để chỉ hiển thị đường viền của từng phần tử UI. Để xem cả hai cùng lúc, chọn "Design + Blueprint".
2. **Hướng trong trình chỉnh sửa**: Chọn "Portrait" hoặc "Landscape" để xem trước bố cục ở chế độ dọc hoặc ngang. Điều này hữu ích để xem trước bố cục mà không cần chạy ứng dụng trên trình giả lập hoặc thiết bị. Để tạo bố cục thay thế, chọn "Create Landscape Variation" hoặc các biến thể khác.
3. **System UI mode** : Giúp kiểm tra xem ứng dụng có hỗ trợ Dark Theme hay không và có cần chỉnh sửa giao diện cho chế độ này không.
4. **Thiết bị trong trình chỉnh sửa**: Chọn loại thiết bị (điện thoại/máy tính bảng, Android TV hoặc Android Wear).
5. **Phiên bản API trong trình chỉnh sửa:** Chọn phiên bản Android để hiển thị bản xem trước.
6. **Chủ đề trong trình chỉnh sửa**: Chọn chủ đề (chẳng hạn như AppTheme) để áp dụng vào bản xem trước.
7. **Ngôn ngữ trong trình chỉnh sửa**: Chọn ngôn ngữ và khu vực 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 cách thêm ngôn ngữ). Bạn cũng có thể chọn "Preview as Right To Left" để xem bố cục như thể đang sử dụng một ngôn ngữ từ phải sang trái (RTL).

Thanh công cụ thứ hai cho phép bạn cấu hình giao diện của các phần tử UI trong ConstraintLayout, cũng như phóng to và di chuyển bản xem trước.



Mô tả các tùy chọn trong thanh công cụ của Layout Editor:

1. **Show**: Bật/tắt hiển thị Constraints (ràng buộc) và Margins (khoảng cách lề) trong bản xem trước.

2. **Autoconnect**: Bật/tắt Autoconnect. Khi bật, Android Studio sẽ tự động tạo ràng buộc (constraint) khi bạn kéo một phần tử vào layout.

3. **Clear All Constraints**: Xóa tất cả các ràng buộc trên layout.

4. **Infer Constraints**: Dự đoán và tự động tạo ràng buộc dựa trên vị trí hiện tại của các phần tử.

5**. Default Margins**: Thiết lập khoảng cách lề mặc định giữa các phần tử khi đặt ràng buộc.

6. **Pack:** Gom nhóm hoặc mở rộng các phần tử đã chọn lại với nhau.

7. **Align**: Căn chỉnh các phần tử được chọn theo hàng ngang/dọc.

8. **Guidelines**: Thêm đường hướng dẫn (Guideline) theo chiều dọc hoặc chiều ngang để dễ dàng căn chỉnh phần tử.

9. **Zoom/Pan Controls**: Phóng to, thu nhỏ hoặc di chuyển bản xem trước layout.

Mẹo: Nếu muốn tìm hiểu sâu hơn, bạn có thể tham khảo tài liệu chính thức: Build a UI with Layout Editor, Build a Responsive UI with ConstraintLayout.

1.1 Xem trước bố cục ở chế độ ngang

Để xem trước bố cục của ứng dụng Hello Toast ở chế độ 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 nguồn hoàn chỉnh của HelloToast, bạn cần xóa bố cục landscape và extra-large đã hoàn thành mà bạn sẽ tạo trong bước này. Chuyển từ Project > Android sang Project > Project Files trong khung Project, mở rộng đường dẫn app > src/main > res, chọn cả hai thư mục layout-land và layout-xlarge, sau đó chọn Edit > Delete. Sau đó, chuyển khung Project về Project > Android.

1. Mở tệp activity\_main.xml. Nhấp vào tab Design nếu nó chưa được chọn.
2. Nhấp vào nút Orientation in Editor trên thanh công cụ trên cùng.
3. Chọn **Switch to Landscape** trong menu thả xuống. Bố cục sẽ hiển thị ở chế độ ngang như hình bên dưới. Để quay lại chế độ dọc, chọn **Switch to Portrait**.

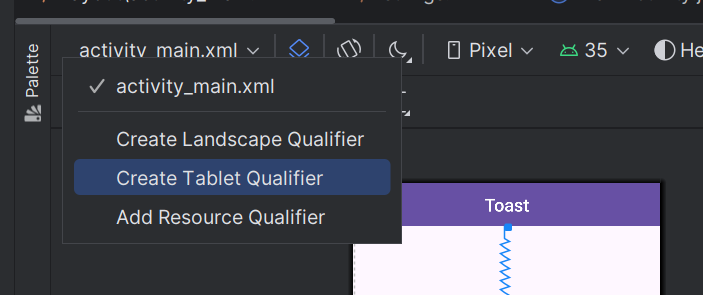
**1.2 Tạo một biến thể bố cục cho chế độ ngang**

Sự khác biệt giữa chế độ dọc và chế độ ngang trong bố cục này là số 0 trong TextView show\_count quá thấp trong chế độ ngang—quá gần với nút Count. Tùy thuộc vào thiết bị hoặc trình giả lập bạn sử dụng, phần tử TextView có thể hiển thị quá lớn hoặc không căn giữa vì kích thước văn bản được đặt cố định ở 160sp.

Để khắc phục vấn đề này trong chế độ ngang mà không ảnh hưởng đến chế độ dọc, bạn có thể tạo một biến thể bố cục riêng cho chế độ ngang của ứng dụng Hello Toast. Làm theo các bước sau:

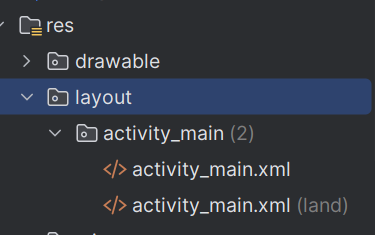
Nhấn vào nút activity\_main.xml trên thanh công cụ trên cùng.

Chọn Create Landscape Qualifier.



Một cửa sổ chỉnh sửa mới sẽ mở ra với tab land/activity\_main.xml, hiển thị bố cục dành riêng cho chế độ ngang. Bạn có thể thay đổi bố cục này mà không làm thay đổi bố cục gốc của chế độ dọc.

Trong Project > Android, mở thư mục res > layout, bạn sẽ thấy Android Studio tự động tạo biến thể bố cục có tên activity\_main.xml (land).



1.3 Xem trước bố cục trên các thiết bị khác nhau

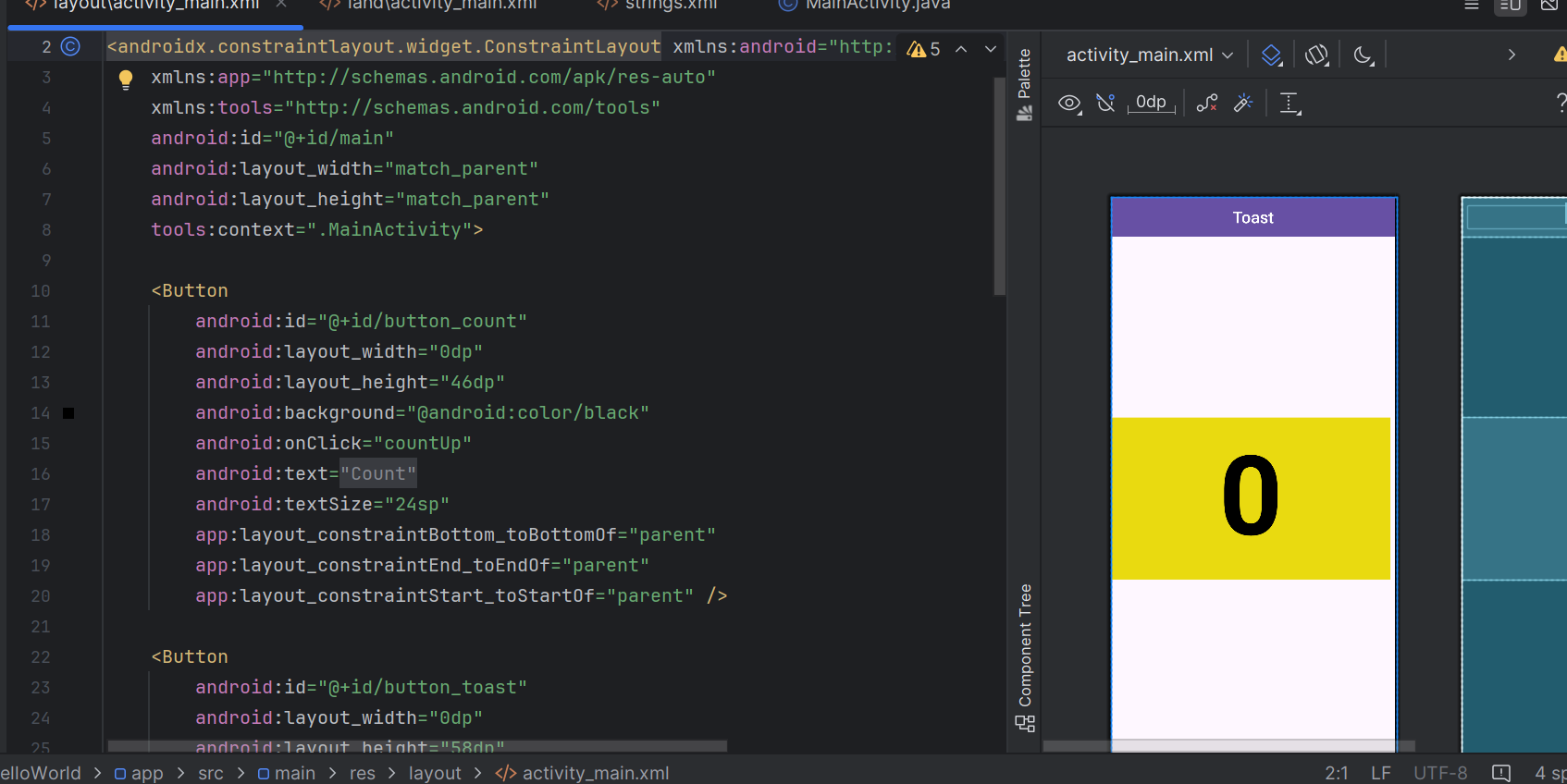
Bạn có thể xem trước bố cục trên nhiều loại thiết bị mà không cần chạy ứng dụng trên thiết bị thật hoặc trình giả lập. Làm theo các bước sau:

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

1.4 Thay đổi bố cục cho chế độ ngang

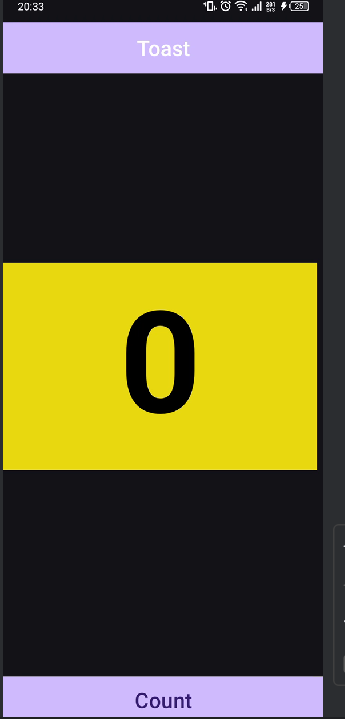
Bạn có thể sử dụng bảng Attributes trong tab Design để thiết lập hoặc thay đổi thuộc tính, nhưng đôi khi sử dụng tab Text để chỉnh sửa trực tiếp mã XML sẽ nhanh hơn.

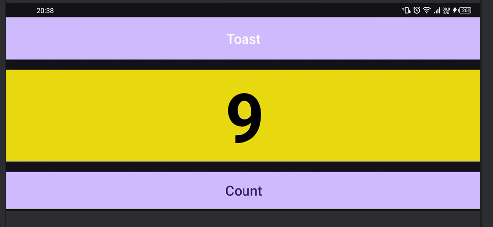
Tab Text hiển thị mã XML và cung cấp tab Preview ở bên phải cửa sổ để xem trước bố cục, như minh họa trong hình dưới đây.



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

1. Tab **land/activity\_main.xml** vẫn nên được mở trong trình chỉnh sửa bố cục; nếu chưa, nhấp đúp vào tệp **activity\_main.xml (land)** trong thư mục **layout**.
2. Nhấp vào tab **Text** và tab **Preview** (nếu chưa được chọn).
3. Tìm phần tử **TextView** trong mã XML.
4. Thay đổi thuộc tính android:textSize="160sp" thành android:textSize="120sp". Bản xem trước bố cục sẽ hiển thị kết quả thay đổi.
5. Chọn các thiết bị khác trong menu thả xuống **Device in Editor** để xem bố cục trông như thế nào trên các thiết bị khác nhau khi ở chế độ ngang.
   * Trong cửa sổ chỉnh sửa, tab **land/activity\_main.xml** hiển thị bố cục cho chế độ ngang.
   * Tab **activity\_main.xml** hiển thị bố cục không thay đổi cho chế độ dọc.
   * Bạn có thể chuyển đổi qua lại bằng cách nhấp vào các tab.
6. Chạy ứng dụng trên trình giả lập hoặc thiết bị thực, sau đó thay đổi hướng từ dọc sang ngang để xem cả hai bố cục.





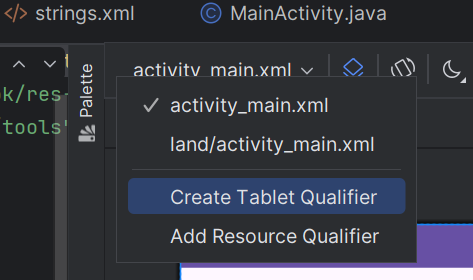
1.5 Tạo một biến thể bố cục cho máy tính bảng

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

Để khắc phục điều này trong khi vẫn giữ nguyên bố cục cho điện thoại ở cả chế độ ngang và dọc, bạn có thể tạo một biến thể bố cục hoàn toàn khác dành riêng cho máy tính bảng. Làm theo các bước sau:

Nhấp vào tab Design (nếu chưa được chọn) để hiển thị các khung thiết kế và bản vẽ.

Chọn **Create Tablet Qualifier**



Một cửa sổ chỉnh sửa mới sẽ mở ra với tab xlarge/activity\_main.xml, hiển thị bố cục dành cho thiết bị có kích thước màn hình máy tính bảng. Trình chỉnh sửa cũng sẽ tự động chọn một thiết bị máy tính bảng như Nexus 9 hoặc Nexus 10 để xem trước. Bạn có thể thay đổi bố cục này để phù hợp với máy tính bảng mà không ảnh hưởng đến các bố cục khác.

1.6 Thay đổi bố cục biến thể cho máy tính bảng

Bạn có thể sử dụng Attributes pane trong tab Design để thay đổi các thuộc tính của bố cục này.

Tắt công cụ Autoconnect trên thanh công cụ. Đảm bảo rằng công cụ này đã bị vô hiệu hóa.

Xóa tất cả các ràng buộc (constraints) trong bố cục bằng cách nhấp vào nút Clear All Constraints trên thanh công cụ.

Khi tất cả ràng buộc bị xóa, bạn có thể di chuyển và thay đổi kích thước các phần tử trong bố cục một cách tự do.

Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước (resizing handles) ở bốn góc của một phần tử để thay đổi kích thước của nó.

Trong Component Tree, chọn TextView có id là show\_count

Để di chuyển TextView ra khỏi khu vực làm việc và dễ dàng di chuyển các Button, hãy kéo một góc của nó để thay đổi kích thước như trong hình minh họa bên dưới.

**Lưu ý khi thay đổi kích thước phần tử**

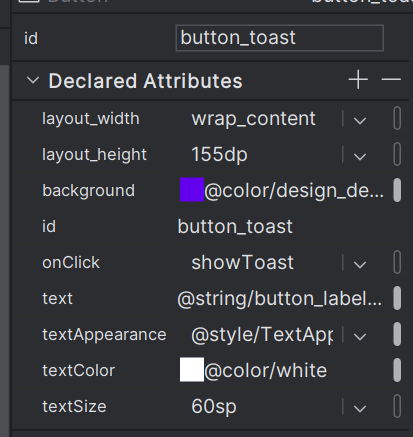
Khi thay đổi kích thước một phần tử bằng cách kéo thả, hệ thống sẽ mặc định gán cứng (hardcode) các giá trị width và height. Tuy nhiên, bạn nên tránh việc này trong hầu hết các trường hợp, vì kích thước cứng có thể khiến giao diện không hiển thị đúng trên các màn hình có kích thước và mật độ khác nhau. Ở bước này, bạn chỉ đang tạm thời thay đổi kích thước để sắp xếp lại bố cục, sau đó sẽ chỉnh lại kích thước hợp lý hơn.

Thực hiện thay đổi thuộc tính của Button

Chọn button\_toast trong Component Tree, sau đó:

Mở tab Attributes để truy cập Attributes pane.

Thay đổi textSize thành 60sp (Mục #1 trong hình minh họa).



Thay đổi layout\_width thành wrap\_content (Mục #2 trong hình minh họa).

Chọn **button\_count** trong Component Tree, sau đó:

Thay đổi textSize thành 60sp.

Thay đổi layout\_width thành wrap\_content.

Kéo và thả Button lên một khoảng trống phía trên TextView trong bố cục.

**Giải thích về wrap\_content**

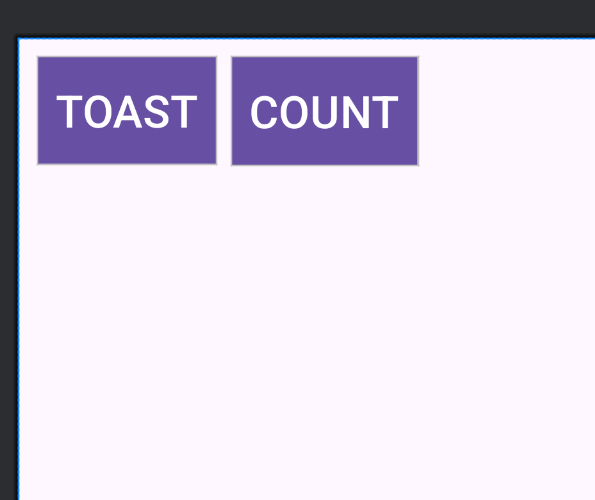
Bạn sử dụng wrap\_content để đảm bảo rằng nút Button có thể tự động điều chỉnh kích thước theo nội dung văn bản bên trong nó. Điều này rất quan trọng khi ứng dụng hỗ trợ đa ngôn ngữ, vì một số ngôn ngữ có thể có từ dài hơn hoặc ngắn hơn. Việc sử dụng wrap\_content giúp bố cục giao diện linh hoạt hơn, tránh lỗi hiển thị khi nội dung thay đổi

1.7 Sử dụng ràng buộc đường cơ sở (Baseline Constraint)

Bạn có thể căn chỉnh một phần tử UI chứa văn bản, chẳng hạn như TextView hoặc Button, với một phần tử UI khác cũng chứa văn bản. Baseline Constraint giúp đảm bảo rằng dòng cơ sở (baseline) của văn bản trong các phần tử này khớp với nhau.

Các bước thực hiện:

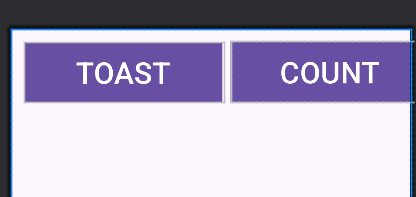
1. Ràng buộc Button button\_toast với cạnh trên và cạnh trái của layout, kéo Button button\_count đến một vị trí gần Button button\_toast, và ràng buộc Button button\_count với cạnh trái của Button button\_toast, như được hiển thị trong hình dưới:



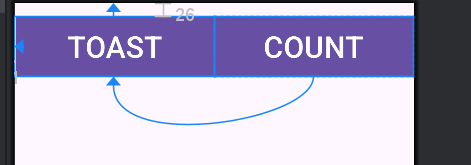
1. Sử dụng ràng buộc baseline, bạn có thể ràng buộc Button button\_count để đường cơ sở văn bản của nó khớp với đường cơ sở văn bản của Button button\_toast. Chọn phần tử button\_count, sau đó di chuột qua phần tử đó cho đến khi nút ràng buộc baseline xuất hiện bên dưới phần tử.
2. Nhấp vào nút ràng buộc baseline. Điểm điều khiển baseline sẽ xuất hiện, nhấp nháy màu xanh lá như hiển thị trong hình động. Nhấp và kéo một đường ràng buộc baseline đến baseline của phần tử button\_toast.

1.8 Mở rộng các nút theo chiều ngang

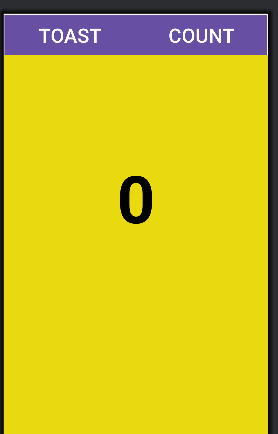
1. Nút pack (đóng gói) trên thanh công cụ cung cấp các tùy chọn để đóng gói hoặc mở rộng các phần tử UI đã chọn. Bạn có thể sử dụng nó để sắp xếp đều các phần tử Button theo chiều ngang trên layout.
2. Chọn Button button\_count trong Component Tree, và giữ phím Shift khi chọn Button button\_toast để cả hai đều được chọn.
3. Nhấp vào nút pack trên thanh công cụ, và chọn Expand Horizontally (Mở rộng theo chiều ngang) như hiển thị trong hình bên dưới.



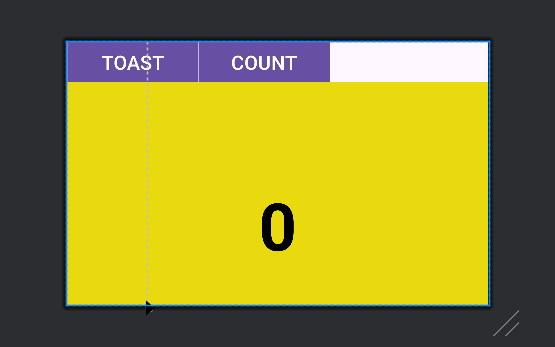
1. Để hoàn thành layout, hãy ràng buộc TextView show\_count với phía dưới của Button button\_toast và với các cạnh bên và cạnh dưới của layout, như hiển thị trong hình động bên dưới.



1. Các bước cuối cùng là thay đổi layout\_width và layout\_height của TextView show\_count thành Match Constraints và textSize thành 200sp. Layout cuối cùng trông như hình bên dưới.



1. Nhấp vào nút Orientation in Editor (Hướng trong Trình soạn thảo) trên thanh công cụ phía trên và chọn Switch to Landscape (Chuyển sang Ngang). Layout máy tính bảng xuất hiện theo hướng ngang như hiển thị bên dưới. (Bạn có thể chọn Switch to Portrait để trở lại hướng dọc.)

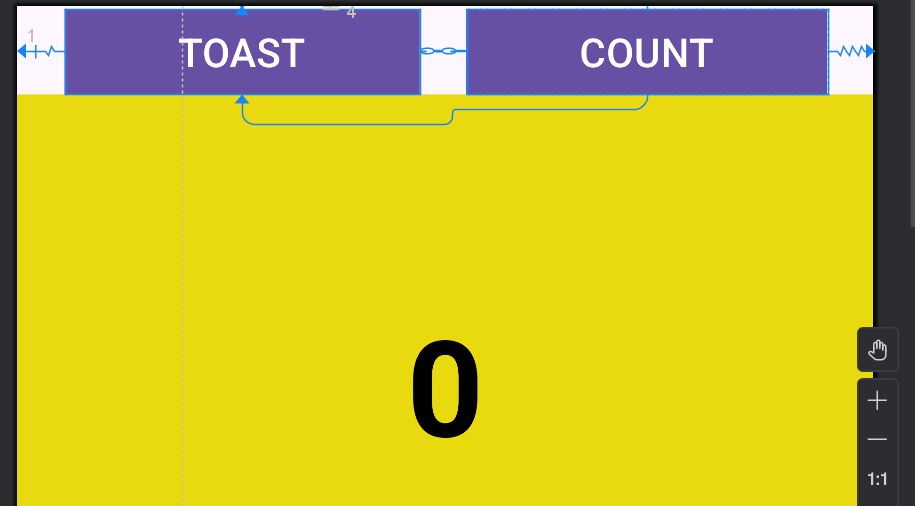


1. Chạy ứng dụng trên các trình giả lập khác nhau, và thay đổi hướng sau khi chạy ứng dụng, để xem nó trông như thế nào trên các loại thiết bị khác nhau. Bạn đã tạo thành công một ứng dụng có thể chạy với giao diện người dùng phù hợp trên điện thoại và máy tính bảng có kích thước và mật độ màn hình khác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

Thử Thách 1

Thử thách: Để thích ứng với hướng ngang (landscape) cho máy tính bảng, bạn có thể căn giữa các phần tử Button trong activity\_main.xml (xlarge) để chúng xuất hiện như hiển thị trong hình bên dưới.

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



Nhiệm vụ 2: Thay đổi layout thành LinearLayout

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

Một LinearLayout bắt buộc phải có các thuộc tính sau:

● layout\_width

● layout\_height

● orientation

Thuộc tính layout\_width và layout\_height có thể nhận một trong các giá trị sau:

● match\_parent: Mở rộng view để lấp đầy chiều rộng hoặc chiều cao của view cha. Khi LinearLayout là view gốc, nó mở rộng đến kích thước của màn hình (view cha).

● wrap\_content: Thu nhỏ kích thước view sao cho view vừa đủ lớn để chứa nội dung của nó. Nếu không có nội dung, view sẽ trở nên vô hình.

● Số cố định dp (điểm ảnh độc lập với mật độ): Chỉ định kích thước cố định, được điều chỉnh theo mật độ màn hình của thiết bị. Ví dụ, 16dp nghĩa là 16 điểm ảnh độc lập với mật độ.

Orientation có thể là:

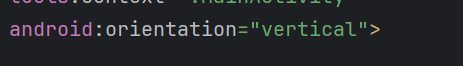
● horizontal: Các view được sắp xếp từ trái sang phải.

● vertical: Các view được sắp xếp từ trên xuống dưới.

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

2.1 Thay đổi nhóm view gốc thành LinearLayout

1. Mở ứng dụng Hello Toast từ nhiệm vụ trước.
2. Mở file layout activity\_main.xml (nếu chưa mở), và nhấp vào tab Text ở dưới cùng của cửa sổ chỉnh sửa để xem mã XML. Ở phần đầu của mã XML là dòng tag sau:
3. Thay đổi thẻ <android.support.constraint.ConstraintLayout thành <LinearLayout để mã trông như thế này: <LinearLayout xmlns:android="http:...
4. Đảm bảo thẻ đóng ở cuối mã đã được thay đổi thành </LinearLayout> (AndroidStudio tự động thay đổi thẻ đóng nếu bạn thay đổi thẻ mở). Nếu nó chưa tự động thay đổi, hãy thay đổi thủ công.
5. Dưới dòng thẻ <LinearLayout, thêm thuộc tính sau đây sau thuộc tính android:layout\_height:

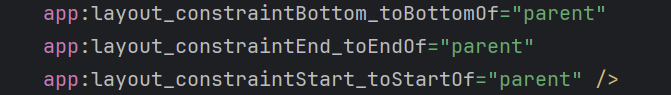


1. Sau khi thực hiện những thay đổi này, một số thuộc tính XML cho các phần tử khác được gạch chân màu đỏ vì chúng được sử dụng với ConstraintLayout và không liên quan đến LinearLayout
   1. Thay đổi thuộc tính phần tử cho LinearLayout
2. Làm theo các bước sau để thay đổi thuộc tính phần tử giao diện người dùng để chúng hoạt động với LinearLayout:
3. Mở ứng dụng Hello Toast từ nhiệm vụ trước.
4. Mở tệp bố cục activity\_main.xml (nếu chưa mở), và nhấp vào tab Text.
5. Tìm phần tử Button button\_toast, và thay đổi thuộc tính sau:

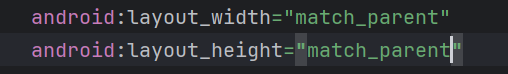




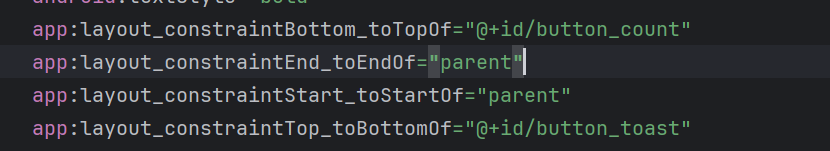
1. Xóa các thuộc tính sau khỏi phần tử button\_toast:



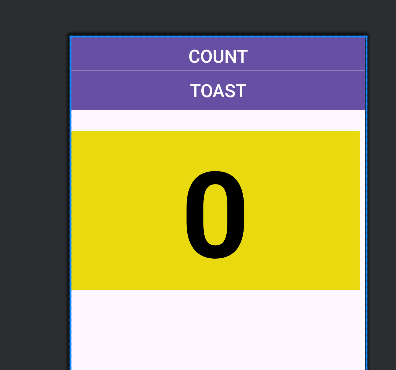
1. Tìm phần tử Button button\_count, và thay đổi thuộc tính y như button\_Toast
2. Tìm phần tử TextView show\_count, và thay đổi các thuộc tính sau:



1. Xóa các thuộc tính sau khỏi phần tử show\_count:



1. Nhấp vào tab Preview ở bên phải cửa sổ Android Studio (nếu chưa được chọn) để xem bản xem trước của 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 theo 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 hiển thị trong hình trước.

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

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

2. Mở tệp bố cục activity\_main.xml (nếu chưa mở), và nhấp vào tab Text.

### Văn bản và các chế độ cuộn

### Tài nguyên có sẵn

## Activities

### Activity và Intent

### Vòng đời của Activity và trạng thái

### Intent ngầm định

## Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ

### Trình gỡ lỗi

### Kiểm thử đơn vị

### Thư viện hỗ trợ

# TRẢI NGHIỆM NGƯỜI DÙNG

## Tương tác người dùng

### Hình ảnh có thể chọn

### Các điều khiển nhập liệu

### Menu và bộ chọn

### Điều hướng người dùng

### RecycleView

## Trải nghiệm người dùng thú vị

### Hình vẽ, định kiểu và chủ đề

### Thẻ và màu sắc

### Bố cục thích ứng

## Kiểm thử giao diện người dùng

### Espresso cho việc kiểm tra UI

# LÀM VIỆC TRONG NỀN

## Các tác vụ nền

### AsyncTask

### AsyncTask và AsyncTaskLoader

### Broadcast receivers

## Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền

### Thông báo

### Trình quản lý cảnh báo

### JobScheduler

# LƯU DỮ LIỆU NGƯỜI DÙNG

## Tùy chọn và cài đặt

### Shared preferences

### Cài đặt ứng dụng

## Lưu trữ dữ liệu với Room

### Room, LiveData và ViewModel

### Room, LiveData và ViewModel