

Unit 1: Bắt đầu

Các bài học trong bài này

Bài học 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: Văn bản và chế độ xem cuộn
- 1.4: Tài nguyên có sẵn

Bài học 2: Hoạt động

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

Bài học 3: Kiểm tra, gỡ lỗi và sử dụng các thư viện hỗ trợ

- 3.1: Trình gỡ lỗi
- 3.2: Kiểm tra đơn vị
- 3.3: Thư viện hỗ trợ

Bài học 1: Xây dựng ứng dụng đầu tiên của bạn

1.1: Android Studio và Hello World

Giới thiệu Trong bài thực hành này, bạn sẽ học 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 trình giả lập và trên 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 hướng đối tượng sử dụng IDE (môi trường phát triển tích hợp) như Android Studio.
- Chứng minh rằng bạn đã có ít nhất 1-3 năm kinh nghiệm lập trình hướng đối tượng, với một số trong đó tập trung vào ngôn ngữ lập trình Java. (Bài thực hành này sẽ không giải thích lập trình hướng đối tượng hoặc ngôn ngữ Java.)

Những gì bạn sẽ cần

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

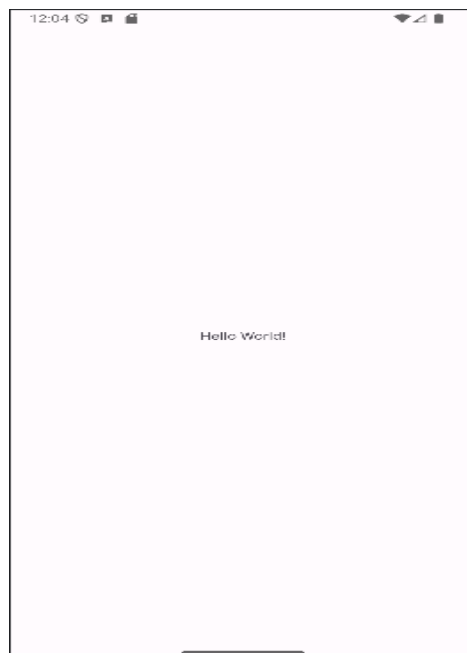
Những gì bạn sẽ học

- Cách cài đặt và sử dụng Android Studio IDE.
- Cách sử dụng quy trình phát triển để xây dựng ứng dụng Android.
- Cách tạo một dự án Android từ một mẫu.
- Cách thêm thông báo nhật ký vào ứng dụng của bạn để 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 bố cục dự án.
- Tạo và xem các thông báo 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 từ một mẫu một dự án mới cho ứng dụng Hello World. Ứng dụng đơn giản này hiển thị chuỗi “Hello World” trên màn hình của thiết bị Android ảo hoặc vật lý.



Nhiệm vụ 1: Cài đặt Android Studio

Android Studio cung cấp một môi trường phát triển tích hợp hoàn chỉnh (IDE) bao gồm một trình chỉnh sửa mã nâng cao và một tập hợp các mẫu ứng dụng. Ngoài ra, nó chứa các công cụ cho phát triển, gỡ lỗi, kiểm tra và hiệu suất giúp phát triển ứng dụng nhanh hơn và dễ dàng hơn. Bạn có thể kiểm tra ứng dụng của mình với một loạt các 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.

Android Studio có sẵn cho các máy tính chạy Windows hoặc Linux, và cho Mac chạy macOS. Bộ công cụ phát triển Java (Java Development Kit) mới nhất được bao gồm với Android Studio.

Để bắt đầu với Android Studio, trước tiên hãy kiểm tra các yêu cầu hệ thống để đảm bảo rằng hệ thống của bạn đáp ứng đủ các yêu cầu này. Quá trình cài đặt là giống nhau cho tất cả các nền tảng. Bất kỳ sự khác biệt nào được lưu ý dưới đây:

1. Điều hướng đến trang web của 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 đã 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, điều này có thể mất một thời gian tùy thuộc vào tốc độ Internet của bạn và một số bước có thể có vẻ dư thừa.
4. Khi quá trình tải xuống hoàn tất, Android Studio sẽ bắt đầu và bạn đã sẵn sàng để tạo dự án đầu tiên của mình.

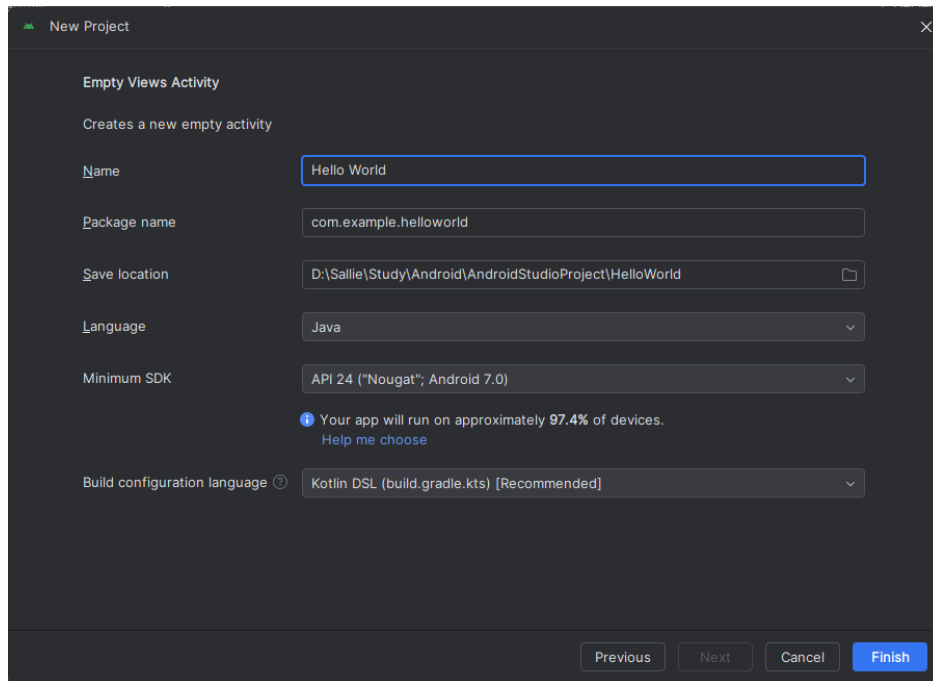
Khắc phục sự cố: Nếu bạn gặp phải các vấn đề với cài đặt của mình, hãy kiểm tra các ghi chú phát hành của Android Studio hoặc nhận trợ giúp từ các giảng viên của bạn.

Nhiệm vụ 2: Tạo ứng dụng Hello World

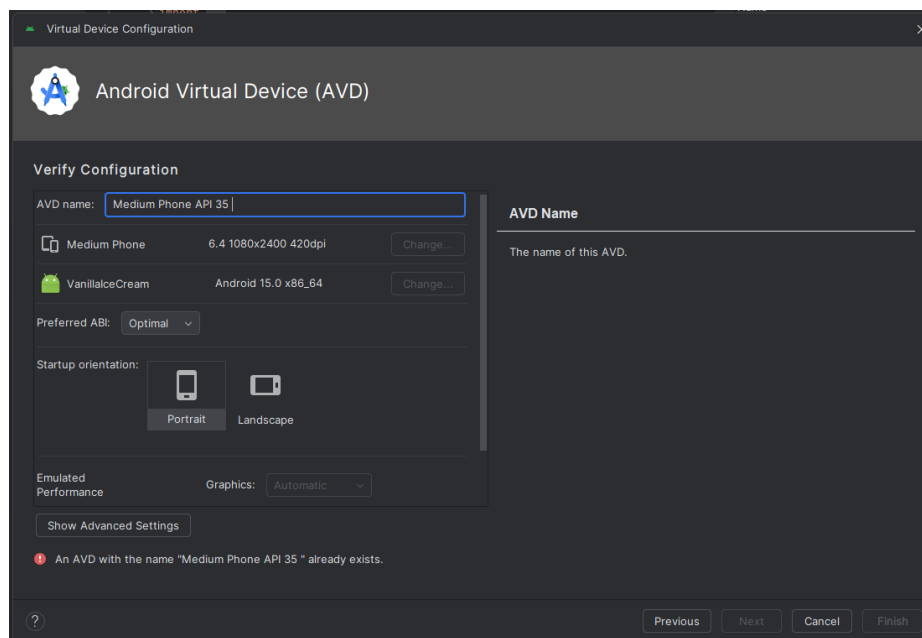
Trong nhiệm vụ này, bạn sẽ tạo một ứng dụng hiển thị "Hello World" để xác minh rằng Android Studio đã được cài đặt đúng cách, và học các kiến thức cơ bản về phát triển với Android Studio.

2.1 Tạo dự án ứng dụng

1. Mở Android Studio nếu nó chưa được mở.
2. Trong cửa sổ chính **Chào mừng đến với Android Studio**, nhấp vào **Bắt đầu một dự án Android Studio mới**.
3. Trong cửa sổ **Tạo Dự án Android**, nhập **Hello World** cho **Tên ứng dụng**.

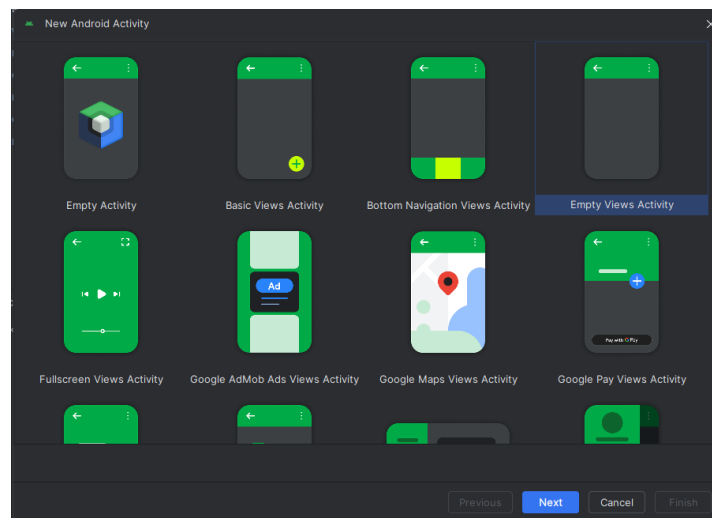


4. Xác minh rằng Vị trí Dự án mặc định là nơi bạn muốn lưu ứng dụng Hello World và các dự án Android Studio khác, hoặc thay đổi nó thành thư mục bạn ưa thích.
5. Chấp nhận mặc định android.example.com cho Miền Công ty, hoặc tạo một miền công ty duy nhất. Nếu bạn không có kế hoạch xuất bản ứng dụng của mình, bạn có thể chấp nhận mặc định. Hãy lưu ý rằng việc thay đổi tên gói của ứng dụng sau này sẽ tốn thêm công sức.
6. Để trống các tùy chọn bao gồm Hỗ trợ C++ và Hỗ trợ Kotlin, sau đó nhấp Tiếp theo.
7. Trên màn hình Mục tiêu Thiết bị Android, Điện thoại và Máy tính bảng nên được chọn. Đảm bảo rằng API 15: Android 4.0.3 IceCreamSandwich được đặt làm SDK Tối thiểu; nếu không, hãy sử dụng menu bật lên để thiết lập nó.

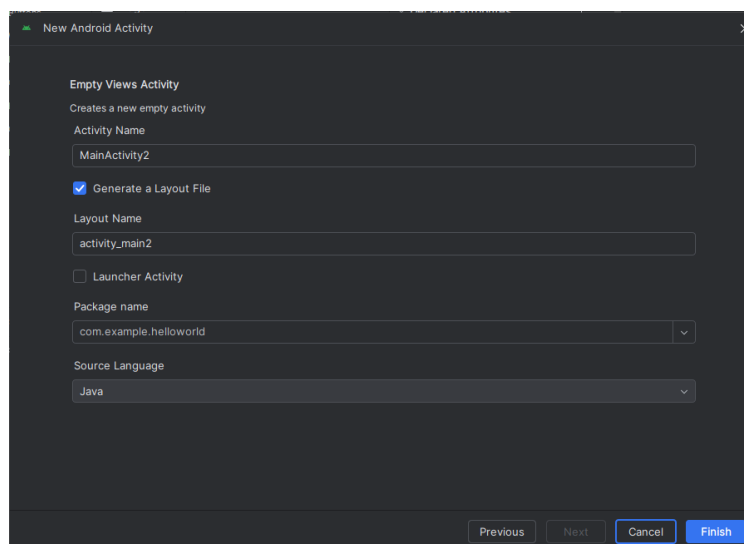


Đây là các thiết lập được sử dụng bởi các ví dụ trong các bài học của khóa học này. Tính đến thời điểm viết bài này, các thiết lập này làm cho ứng dụng Hello World của bạn tương thích với 97% thiết bị Android đang hoạt động trên Google Play Store.

8. Để trống các tùy chọn bao gồm Hỗ trợ Instant App và tất cả các tùy chọn khác. Sau đó nhấp Tiếp theo. Nếu dự án của bạn yêu cầu các thành phần bổ sung cho SDK mục tiêu đã chọn, Android Studio sẽ tự động cài đặt chúng.
9. Cửa sổ Thêm một Hoạt động xuất hiện. Một Hoạt động là một thứ duy nhất, tập trung mà người dùng có thể làm. Nó là một thành phần quan trọng của bất kỳ ứng dụng Android nào. Một Hoạt động thường có một bố cục liên quan đến nó xác định cách các yếu tố 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, hãy chọn Hoạt động Trống như hình bên dưới và nhấp Tiếp theo.



10. Màn hình Cấu hình Hoạt động xuất hiện (có thể khác nhau tùy thuộc vào mẫu bạn đã chọn ở bước trước). Theo mặc định, Hoạt động trống được cung cấp bởi mẫu được đặt 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 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 (App Compat) được chọn. Điều này đảm bảo rằng ứng dụng của bạn sẽ tương thích ngược với các phiên bản Android trước đó.
13. Nhấp Hoàn tất.

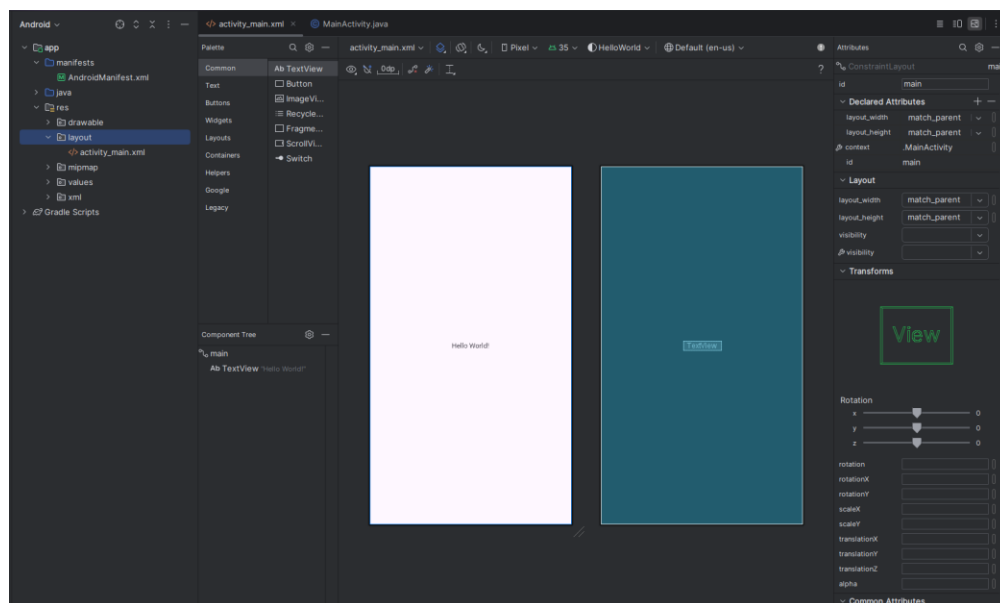
Android Studio sẽ tạo một thư mục cho các dự án của bạn và xây dựng dự án với Gradle (điều này có thể mất một vài phút).

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

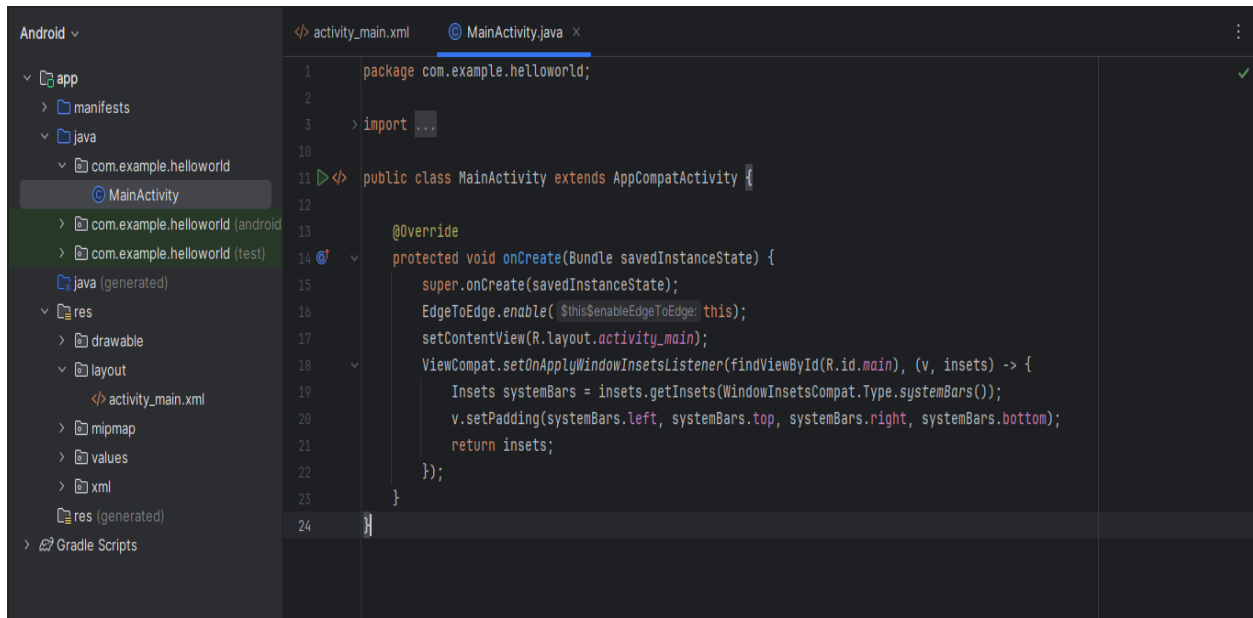
Bạn cũng có thể thấy một thông báo "Mẹo của ngày" với các phím tắt và các mẹo hữu ích khác. Nhấp Đóng để đóng thông báo.

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

1. Nhấp vào tab `activity_main.xml` để xem trình chỉnh sửa bố cục.
2. Nhấp vào tab Thiết kế của trình chỉnh sửa bố cục, nếu chưa được chọn, để hiển thị bản dựng đồ họa của bố cục như hình dưới đây.

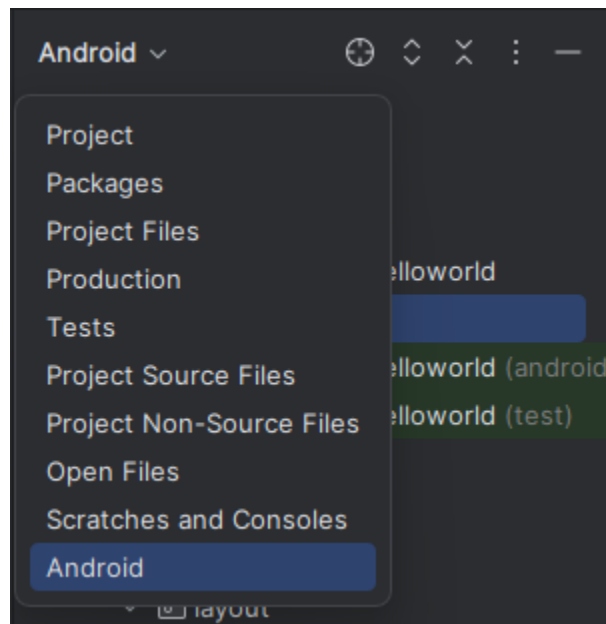


3. Nhấp vào tab `MainActivity.java` để xem trình chỉnh sửa mã như hình dưới đây.



2.2 Khám phá khung dự án > Android Trong bài thực hành này, bạn sẽ khám phá cách tổ chức dự án trong Android Studio.

1. Nếu chưa được chọn, nhấp vào tab Dự án trong cột tab dọc ở bên trái cửa sổ Android Studio. Khung Dự án sẽ xuất hiện.
2. Để xem dự án trong cấu trúc phân cấp dự án Android chuẩn, chọn Android từ menu bật lên ở đầu khung Dự án, như hình dưới đây.

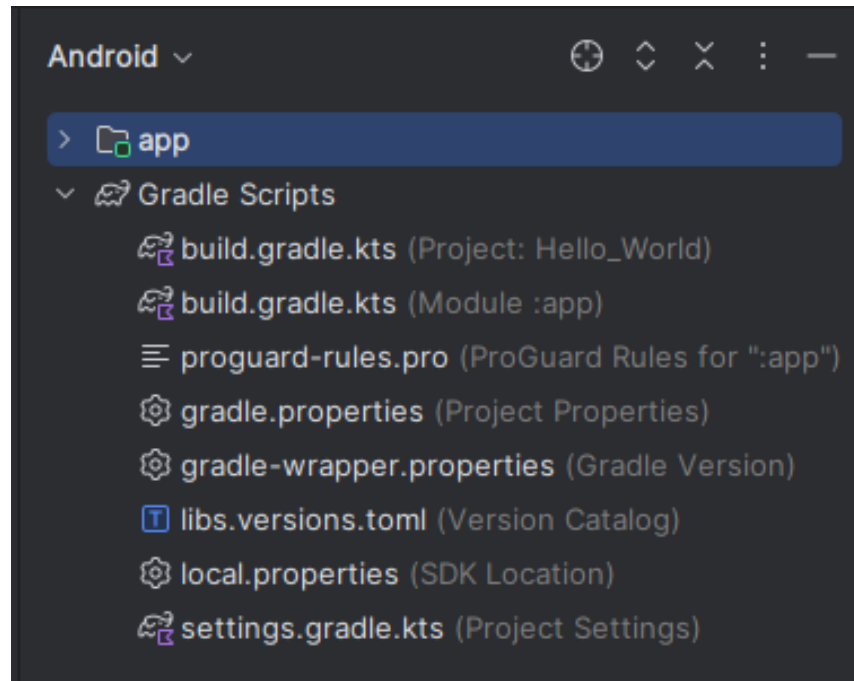


Lưu ý: Chương này và các chương khác tham chiếu đến khung Dự án, khi được đặt thành Android, là khung Dự án > Android.

2.3 Khám phá thư mục Gradle Scripts

Hệ thống xây dựng Gradle trong Android Studio giúp dễ dàng bao gồm các tệp nhị phân bên ngoài hoặc các mô-đun thư viện khác vào quá trình xây dựng của bạn như là các phụ thuộc.

Khi bạn lần đầu tiên tạo một dự án ứng dụng, khung Dự án > Android xuất hiện với thư mục Gradle Scripts được mở rộng như hình dưới đây.



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

1. Nếu thư mục Gradle Scripts không được mở rộng, nhấp vào tam giác để mở rộng nó. 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 xây dựng Gradle cấp cao duy nhất. Hầu hết thời gian, bạn sẽ không cần thực hiện bất kỳ thay đổi nào đối với tệp này, nhưng vẫn hữu ích để hiểu nội dung của nó.

Theo mặc định, tệp xây dựng cấp cao nhất sử dụng khối buildscript để xác định các kho lưu trữ Gradle và các phụ thuộc chung cho tất cả các mô-đun trong dự án. Khi phụ thuộc của bạn là một thư viện địa phương hoặc cây thư mục, Gradle sẽ tìm kiếm các tệp trong bất kỳ kho lưu trữ trực tuyến nào được chỉ định trong khối repositories của tệp này. Theo mặc định, các dự án Android Studio mới khai báo JCenter và Google (bao gồm kho lưu trữ Google Maven) làm 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 cấu hình các thiết lập xây dựng cho mỗi mô-đun cụ thể (ứng dụng HelloWorld chỉ có một mô-đun). Cấu hình các thiết lập xây dựng này cho phép bạn cung cấp các tùy chọn đóng gói tùy chỉnh, chẳng hạn như các loại xây dựng bổ sung và các hương vị sản phẩm. Bạn cũng có thể ghi đè các thiết lập trong tệp AndroidManifest.xml hoặc tệp build.gradle cấp cao.

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

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

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

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

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

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

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

```

}

dependencies {

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

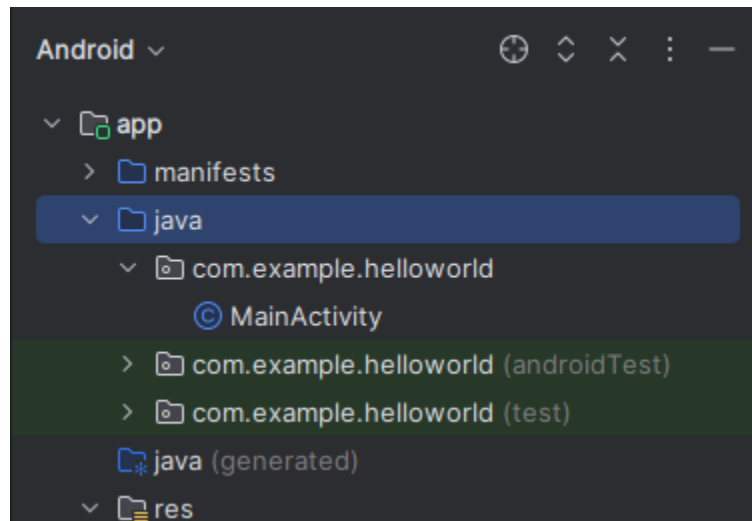
```

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

2.4 Khám phá các thư mục app và res

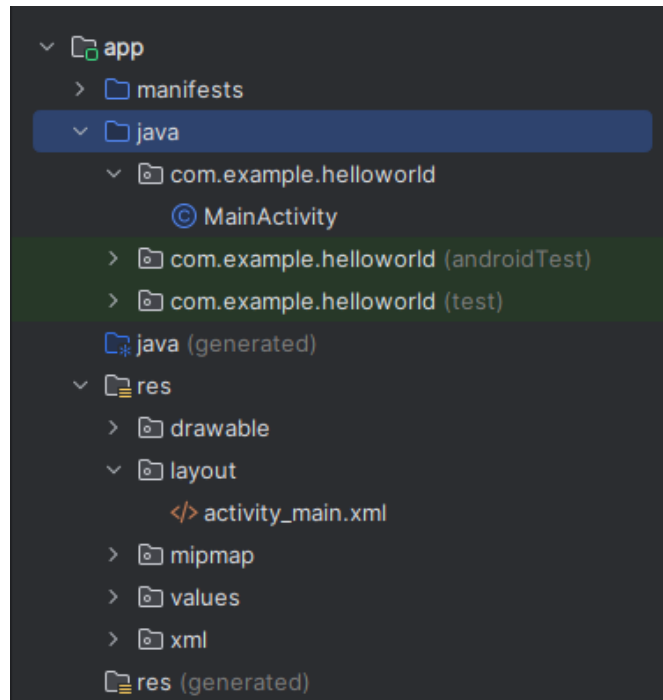
Tất cả mã và tài nguyên cho ứng dụng đều nằm trong các thư mục app và res.

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



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

2. Mở rộng thư mục res và thư mục layout, và 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. Một Hoạt động thường được liên kết với một bố cục các chế độ xem giao diện người dùng được định nghĩa là một tệp XML. Tệp này thường được đặt tên theo Hoạt động của nó.

2.5 Khám phá thư mục manifests

Thư mục manifests chứa các tệp cung cấp thông tin cần thiết về ứng dụng của bạn cho hệ thống Android, mà hệ thống cần 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 cho một ứng dụng, chẳng hạn như mỗi Hoạt động, phải được khai báo trong tệp XML này. Trong các bài học khác của khóa học, bạn sẽ sửa đổi tệp này để thêm các tính năng và quyền tính năng. Để có phần giới thiệu, hãy xem Tổng quan về App Manifest.


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

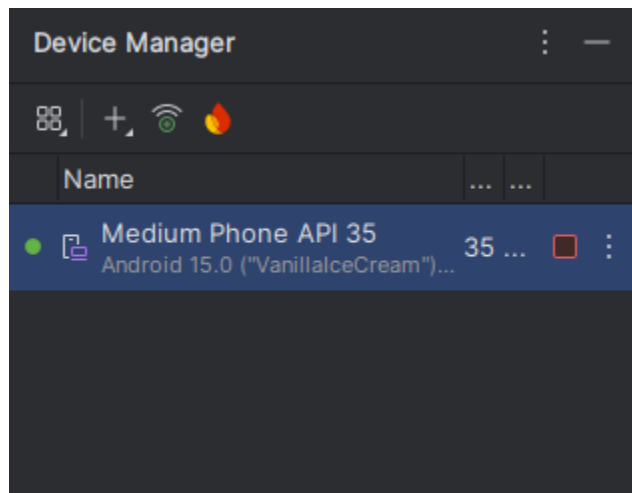
Trong nhiệm vụ này, bạn sẽ sử dụng trình quản lý Thiết bị ảo Android (AVD) để tạo một thiết bị ảo (còn được gọi là trình giả lập) mô phỏng cấu hình cho một loại thiết bị Android cụ thể và sử dụng thiết bị ảo đó để chạy ứng dụng. Lưu ý rằng Trình giả lập Android có các yêu cầu bổ sung ngoài các yêu cầu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

Sử dụng Trình quản lý AVD, bạn xác định các đặc điểm phần cứng của thiết bị, mức API, lưu trữ, giao diện và các thuộc tính khác và lưu nó dưới dạng thiết bị ảo. Với các 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 mức API khác nhau mà không cần sử dụng thiết bị vật lý.

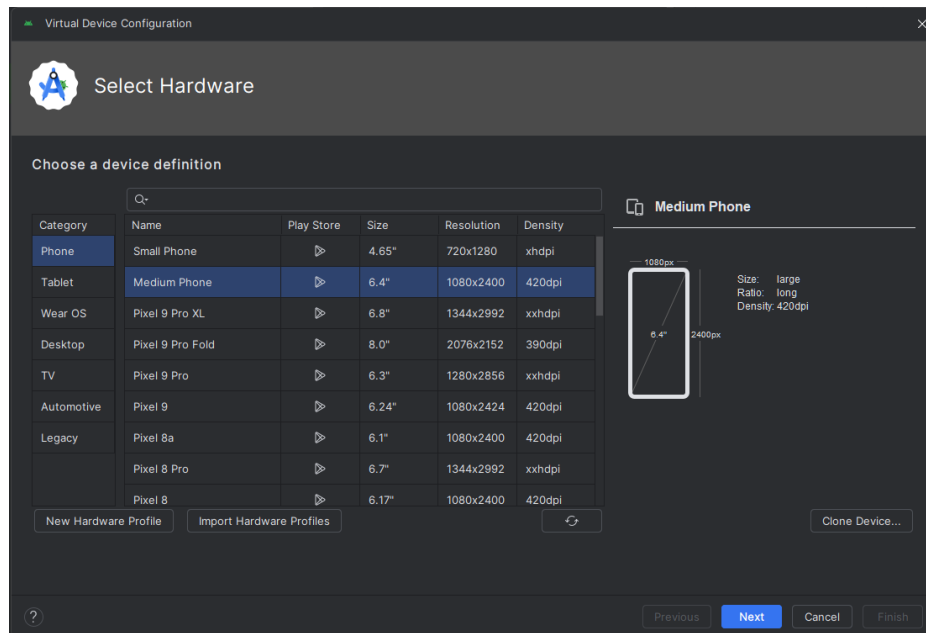
3.1 Tạo một thiết bị ảo Android (AVD) Để chạy trình giả lập trên máy tính của bạn, bạn phải tạo một cấu hình mô tả thiết bị ảo.

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 các thiết bị ảo, màn hình sẽ hiển thị chúng (như hình bên dưới); nếu không, bạn sẽ thấy một danh sách trống.

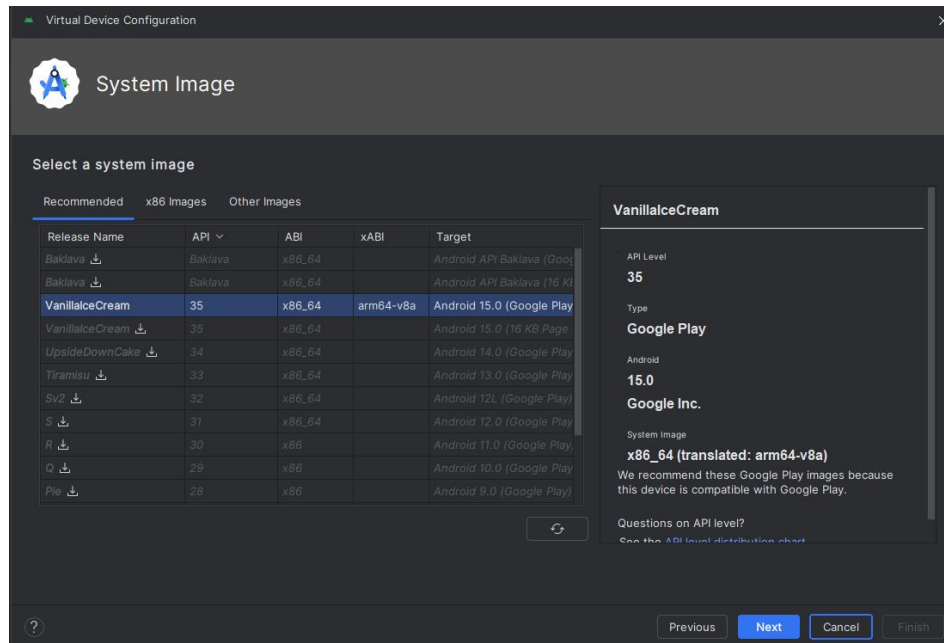


2. Nhấp vào +Tạo Thiết bị Ảo. Cửa sổ Chọn Phần cứng xuất hiện hiển thị danh sách các thiết bị phần cứng đã được cấu hình sẵn. Đối với mỗi thiết bị, bảng cung cấp một cột cho kích thước màn hình đường chéo của nó (Kích thước), độ phân giải màn hình bằng pixel (Độ phân giải) và mật độ pixel (Mật độ).



3. Chọn một thiết bị như Nexus 5x hoặc Pixel XL, và nhấp Tiếp theo. Màn hình Hình ảnh Hệ thống xuất hiện.

4. Nhấp vào tab Được đề xuất nếu nó chưa được chọn, và chọn phiên bản hệ thống Android để chạy trên thiết bị ảo (chẳng hạn như Oreo).



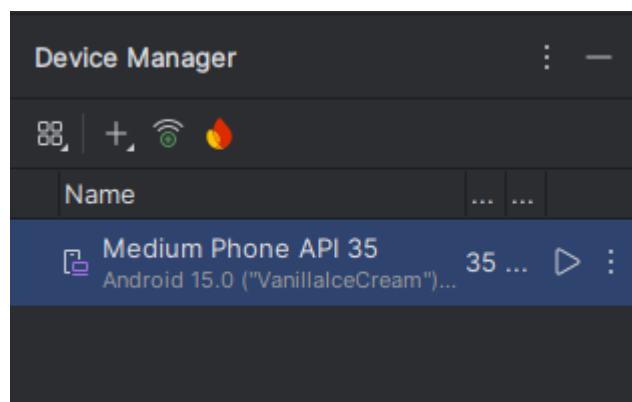
Có nhiều phiên bản hơn được hiển thị trong tab Được đề xuất. Hãy xem tab Hình ảnh x86 và Hình ảnh Khác để xem chúng. Nếu một liên kết Tải xuống hiển thị bên cạnh một hình ảnh hệ thống bạn muốn sử dụng, có nghĩa là nó chưa được cài đặt. Nhấp vào liên kết để bắt đầu tải xuống và nhấp Hoàn tất khi hoàn tất.

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

3.2 Chạy ứng dụng trên thiết bị ảo

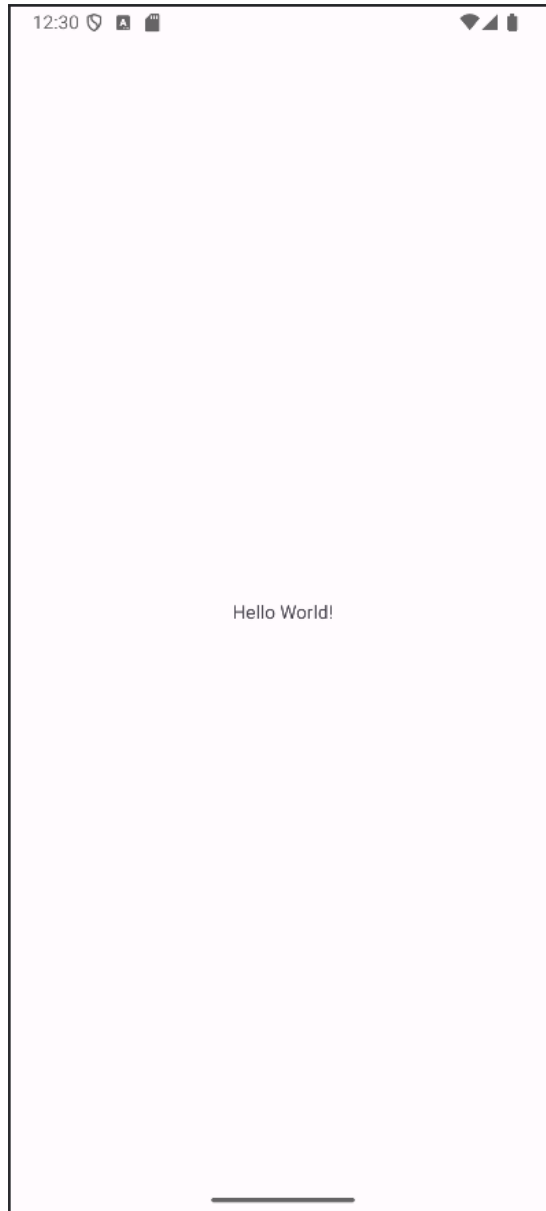
Trong nhiệm vụ này, cuối cùng bạn sẽ chạy ứng dụng Hello World của mình.

1. Trong Android Studio, chọn Chạy > Chạy ứng dụng hoặc nhấp vào biểu tượng Chạy trên thanh công cụ.
2. Trong cửa sổ Chọn Mục tiêu Triển khai, dưới Thiết bị Ảo Có sẵn, chọn thiết bị ảo mà bạn vừa tạo và nhấp OK.



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

Bạn nên thấy ứng dụng Hello World như hình dưới đây.



Mẹo: Khi kiểm tra trên thiết bị ảo, thực hành tốt là khởi động nó một lần, ngay từ đầu phiên của bạn. Bạn không nên đóng nó cho đến khi bạn hoàn thành việc kiểm tra ứng dụng của mình, để ứng dụng của bạn không phải trải qua quá trình khởi động thiết bị lại. Để đóng thiết bị ảo, 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 một thiết bị di động vật lý như điện thoại hoặc máy tính bảng. Bạn nên luôn kiểm tra ứng dụng của mình trên cả thiết bị ảo và thiết bị vật lý.

Những gì 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 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 một 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 phù hợp cho thiết bị của mình. Đối với trình điều khiển USB dựa trên Windows, xem Trình điều khiển USB của OEM.

4.1 Bật gỡ lỗi USB Để cho Android Studio giao tiếp với thiết bị của bạn, bạn phải bật Gỡ lỗi USB trên thiết bị Android của bạn. Điều này được kích hoạt trong cài đặt Tùy chọn nhà phát triển của 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 tùy chọn 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 chạm vào Số bản dựng bảy lần.
2. Quay lại màn hình trước (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

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

1. Kết nối thiết bị của bạn với máy phát triển bằng cáp USB.
2. Nhấp vào nút Chạy trên thanh công cụ. Cửa sổ Chọn Mục tiêu Triển khai mở ra với danh sách các trình giả lập và thiết bị đã kết nối có sẵn.
3. Chọn thiết bị của bạn và nhấp OK.

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

Khắc phục sự cố Nếu Android Studio của bạn không nhận ra thiết bị của bạn, hãy thử các bước 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 tuyên bố nó là "không được phép", hãy thực hiện các bước sau:

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

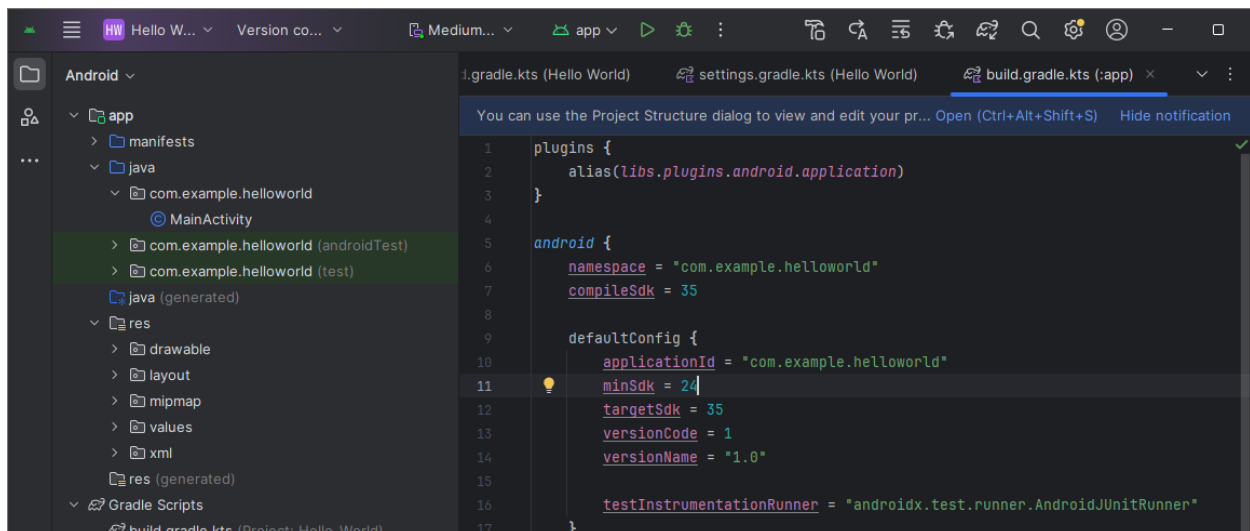
Nhiệm vụ 5: Thay đổi cấu hình Gradle của ứng dụng

Trong nhiệm vụ này, bạn sẽ thay đổi một số thông tin cấu hình của ứng dụng trong tệp build.gradle(Module:app) để học cách thực hiện các thay đổi và đồng bộ 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 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 chỉnh sửa mã.
2. Trong khối defaultConfig, thay đổi giá trị của minSdkVersion thành 17 như hiển thị dưới đây (nó ban đầu được đặt là 15).



Trình chỉnh sửa mã hiển thị một thanh thông báo ở đầu với liên kết Đồng bộ hóa ngay.

5.2 Đồng bộ hóa cấu hình Gradle mới

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

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

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

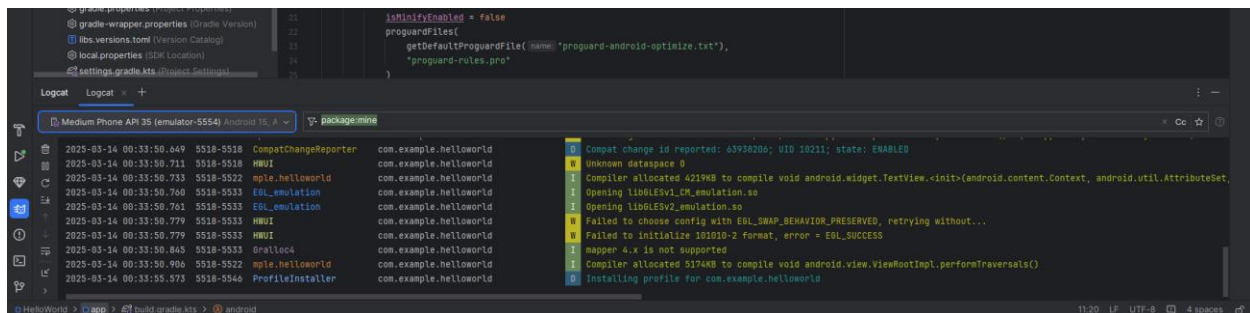
Để tìm hiểu sâu hơn về Gradle, hãy xem Tổng quan về Hệ thống Xây dựng và tài liệu Cấu hình Bản dựng Gradle.

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

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

6.1 Xem khung Logcat

Để xem khung Logcat, nhấp vào tab Logcat ở dưới cùng của cửa sổ Android Studio như hình dưới đây.



Trong hình trên:

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

6.2 Thêm câu lệnh log vào ứng dụng của bạn Các câu lệnh Log trong mã ứng dụng của bạn hiển thị các thông báo trong khung Logcat. Ví dụ:

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

Các phần của thông báo là:

- **Log:** Lớp Log để gửi thông báo nhật ký tới khung Logcat.

- **d**: Cài đặt mức nhật ký Debug để lọc hiển thị thông báo nhật ký trong khung Logcat. Các mức nhật ký khác bao gồm e cho Error, w cho Warn, và i cho Info.
- **"MainActivity"**: Đối số đầu tiên là một thẻ có thể được sử dụng để lọc các thông báo trong khung Logcat. Đây thường là tên của Hoạt động mà từ đó thông báo bắt nguồn. Tuy nhiên, bạn có thể làm cho điều này trở thành bất cứ điều gì hữu ích cho bạn khi gỡ lỗi. **Theo quy ước, các thẻ nhật ký được định nghĩa là các hằng số cho Hoạt động:**

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

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

Làm theo các bước sau:

1. Mở ứng dụng Hello World của bạn trong Android Studio và mở MainActivity.
2. Để thêm các imports rõ ràng tự động vào dự án của bạn (chẳng hạn như android.util.Log cần thiết để sử dụng Log), chọn Tệp > Cài đặt trong Windows hoặc Android Studio > Tùy chọn trong macOS.
3. Chọn Trình chỉnh sửa > Chung > Tự động nhập. Chọn tất cả các hộp kiểm và đặt Chèn nhập vào 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 trông giống như đoạn mã sau:

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

6. Nếu khung Logcat chưa được mở, nhấp vào tab Logcat ở dưới cùng của Android Studio để mở nó.
7. Kiểm tra rằng tên mục tiêu và tên gói của ứng dụng là chính xác.
8. Thay đổi mức Log trong khung Logcat thành Debug (hoặc để là Verbose vì có rất ít thông báo log).

9. Chạy ứng dụng của bạn.

Thông báo sau đây sẽ xuất hiện trong khung Logcat:

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

Tóm tắt

- Để cài đặt Android Studio, truy cập Android Studio và làm theo các hướng dẫn để tải xuống và cài đặt.
- Khi tạo ứng dụng mới, 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 khung Dự án, 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 (Mô-đun: ứng dụng) khi bạn cần thêm thư viện mới vào dự án hoặc thay đổi phiên bản thư viện.
- Tất cả mã và tài nguyên cho ứng dụng nằm trong thư mục app 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 thành phần và quyền tính năng vào ứng dụng Android của bạn. Tất cả các thành phần cho một ứng dụng, chẳng hạn như nhiều hoạt động, phải được khai báo trong tệp XML này.
- Sử dụng trình quản lý Thiết bị ảo Android (AVD) để tạo một thiết bị ảo (còn được gọi là trình giả lập) để chạy ứng dụng của bạn.
- Thêm các câu lệnh Log vào ứng dụng của bạn, hiển thị các thông báo trong khung Logcat như một công cụ cơ bản để gỡ lỗi.
- Để chạy ứng dụng của bạn trên một thiết bị Android vật lý bằng Android Studio, bật Gỡ lỗi USB trên thiết bị. Mở Cài đặt > Giới thiệu về điện thoại và chạm vào Số bản dựng bảy lần. Quay lại màn hình trước (Cài đặt) và nhấn vào Tùy chọn nhà phát triển. Chọn Gỡ lỗi USB.

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 một thiết bị Android bao gồm một hệ thống phân cấp các đối tượng được gọi là views — mỗi yếu tố của màn hình là một View. Lớp View đại diện cho khối xây dựng cơ bản cho tất cả các thành phần UI và là lớp cơ bản cho các lớp cung cấp các thành phần UI tương tác như nút, hộp kiểm và trường nhập văn bản. Các lớp con View thường được sử dụng được mô tả trong nhiều bài học bao gồm:

- **TextView** để hiển thị văn bản.
- **EditText** để cho phép người dùng nhập và chỉnh sửa văn bản.
- **Button** và các 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 yếu tố 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. Một Activity thường được liên kết với một bố cục các chế độ xem giao diện người dùng được định nghĩa là một tệp XML (eXtended Markup Language). Tệp XML này thường được đặt tên theo Activity của nó và định nghĩa bố cục của các yếu tố View trên màn hình.

Ví dụ, mã MainActivity trong ứng dụng Hello World hiển thị một 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 nhấn của người dùng, vẽ nội dung đồ họa, hoặc yêu cầu dữ liệu từ cơ sở dữ liệu hoặc internet. Bạn sẽ học thêm về lớp Activity trong bài học khác.

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

Những gì bạn nên biết trước

- 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 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.
- Nhiều thuật ngữ mới. Kiểm tra từ điển thuật ngữ và khái niệm cho các định nghĩa thân thiện.

Những gì bạn sẽ làm

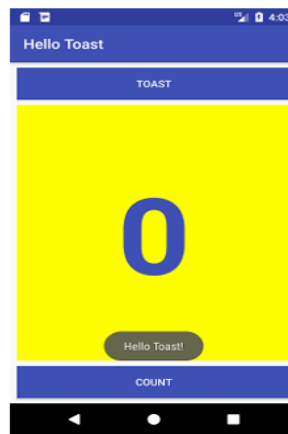
- Tạo một ứng dụng và thêm hai phần tử Button và một TextView vào bố cục.
- Thao tác từng phần tử trong ConstraintLayout để hạn chế chúng vào các lề và các phần tử khác.

- Thay đổi thuộc tính phần tử giao diện người dùng.
- Chỉnh sửa bố cục của ứng dụng trong XML.
- Trích xuất các chuỗi cứng vào tài nguyên chuỗi.
- Thực hiện các phương pháp xử lý click 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 (một Toast) trên màn hình. Nhấn vào nút thứ hai sẽ tăng bộ đếm "click" hiển thị trong TextView, bắt đầu từ số không.

Đây là hình ảnh của ứng dụng đã 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. Một liên kết đến mã giải pháp sẽ được cung cấp ở cuối.

1.1 Tạo dự án Android Studio

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

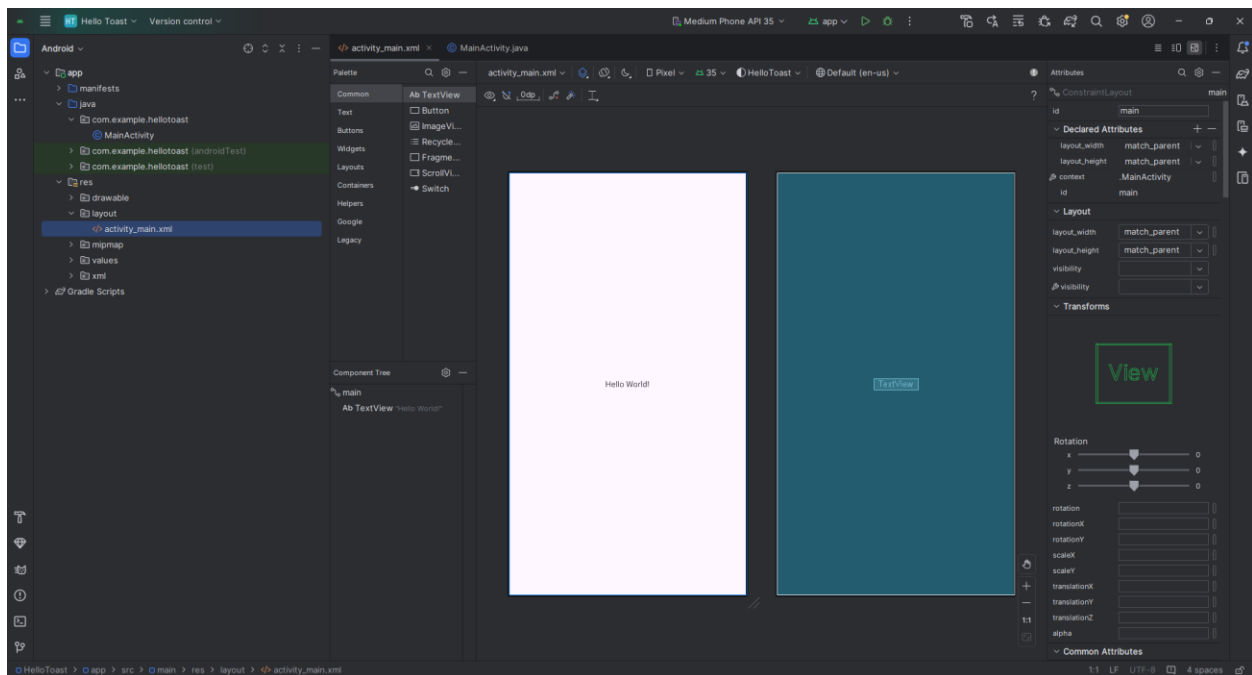
Thuộc tính	Giá trị
Tên ứng dụng	Hello Toast
Tên công ty	com.example.android (hoặc miền của riêng bạn)
SDK tối thiểu cho điện thoại và máy tính bảng	API15: Android 4.0.3 IceCreamSandwich
Mẫu	Empty Activity
Hộp tệp bố cục tạo	Đã chọn

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

1.2 Khám phá trình chỉnh sửa bố cục

Android Studio cung cấp trình chỉnh sửa bố cục để nhanh chóng xây dựng bố cục giao diện người dùng (UI) của một ứng dụng. Nó cho phép bạn kéo các yếu tố vào thiết kế trực quan và chế độ xem bản vẽ, định vị chúng trong bố cục, thêm các ràng buộc và đặt thuộc tính. Các ràng buộc xác định vị trí của một yếu tố UI trong bố cục. Một ràng buộc đại diện cho một kết nối hoặc căn chỉnh đến một view 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 đánh số:



1. Trong thư mục app > res > layout trong khung Dự án > Android, nhấp đúp vào tệp activity_main.xml để mở nó, nếu nó chưa được mở.
2. Nhấp vào tab Design nếu nó chưa được chọn. Bạn sử dụng tab Design để thao tác các yếu tố và bố cục, và tab Text để chỉnh sửa mã XML cho bố cục.
3. Khung Palettes hiển thị các yếu tố UI mà bạn có thể sử dụng trong bố cục của ứng dụng.
4. Khung Component tree hiển thị hệ thống phân cấp các yếu tố UI. Các yếu tố View được tổ chức thành một hệ thống phân cấp cây của các phần tử cha và con, trong đó một phần tử con kế thừa các thuộc tính của phần tử cha của nó. Trong hình trên, TextView là phần tử con của ConstraintLayout. Bạn sẽ tìm hiểu về những yếu tố này sau trong bài học này.

5. Các khung thiết kế và bản vẽ của trình chỉnh sửa bố cục hiển thị các yếu tố UI trong bố cục. Trong hình trên, bố cục chỉ hiển thị một yếu tố: một TextView hiển thị "Hello World".
6. Tab Attributes hiển thị khung Attributes để đặt các thuộc tính cho một yếu tố UI.

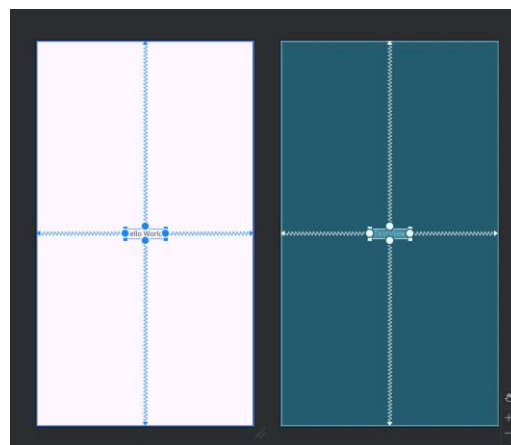
Mẹo: Xem Building a UI with Layout Editor để biết chi tiết về cách sử dụng trình chỉnh sửa bố cục, và Meet Android Studio để xem đầy đủ tài liệu về Android Studio.

Nhiệm vụ 2: Thêm các yếu tố View trong trình chỉnh sửa bố cục

Trong nhiệm vụ này, bạn sẽ 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 chỉ ra sau, hoặc tự động sử dụng công cụ Autoconnect.

2.1 Kiểm tra các ràng buộc của yếu tố Làm theo các bước sau:

1. Mở tệp activity_main.xml từ khung Dự án > Android nếu nó chưa được mở. Nếu tab Design chưa được chọn, hãy nhấp vào nó. Nếu không có bản vẽ, nhấp vào nút Chọn Bề mặt Thiết kế trên thanh công cụ và chọn Thiết kế + Bản vẽ.
2. Công cụ Autoconnect cũng nằm trên thanh công cụ. Nó được bật theo mặc định. Đối với bước này, đảm bảo rằng công cụ không bị tắt.
3. Nhấp vào nút phóng to để phóng to các khung thiết kế và bản vẽ để xem cận cảnh.
4. Chọn TextView trong khung Component Tree. TextView "Hello World" được tô sáng trong các khung thiết kế và bản vẽ và các ràng buộc cho yếu tố là có thể nhìn thấy.
5. Tham khảo hình động dưới đây cho bước này. Nhấp vào tay cầm tròn ở bên phải của TextView để xóa ràng buộc ngang ràng buộc view vào phía bên phải của bố cục. TextView nhảy sang bên trái vì nó không còn bị ràng buộc vào phía bên phải. Để thêm lại ràng buộc ngang, nhấp vào cùng một tay cầm và kéo một đường sang phía bên phải của bố cục.



Trong các khung thiết kế hoặc bản vẽ, các tay cầm sau xuất hiện trên yếu tố TextView:

- **Constraint handle:** Để tạo một ràng buộc như được chỉ ra trong hình động trên, nhấp vào một tay cầm ràng buộc, được hiển thị dưới dạng vòng tròn ở bên của một yếu tố. Sau đó kéo tay cầm đến một tay cầm ràng buộc khác hoặc đến một giới hạn cha. Một đường gấp khúc đại diện cho ràng buộc.



- **Resizing handle:** Để thay đổi kích thước của yếu tố, kéo các tay cầm thay đổi kích thước hình vuông. Tay cầm chuyển đổi thành một góc nghiêng khi bạn đang kéo nó.

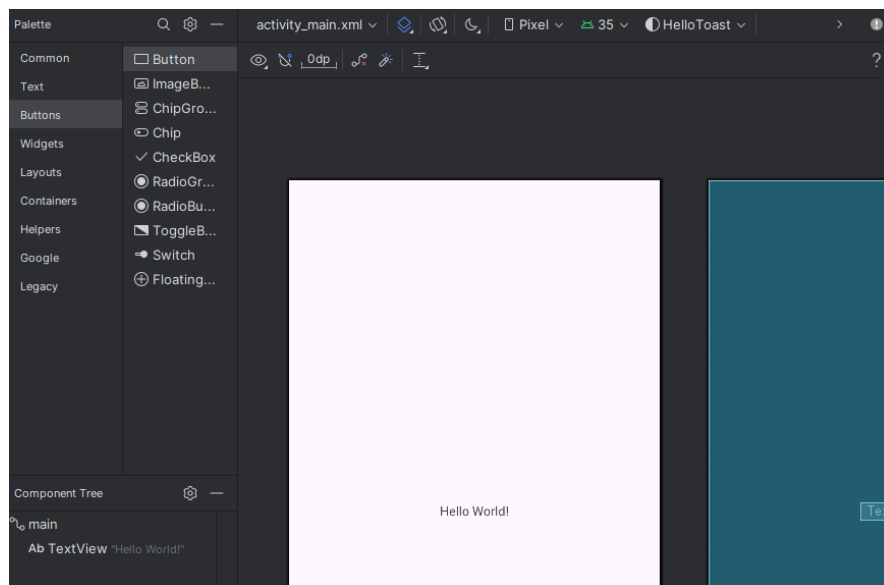


2.2 Thêm nút vào bố cục

Khi được bật, công cụ Autoconnect tự động tạo hai hoặc nhiều ràng buộc cho một yếu tố giao diện người dùng với bố cục cha. Sau khi bạn kéo yếu tố vào bố cục, nó tạo ra các ràng buộc dựa trên vị trí của yếu tố.

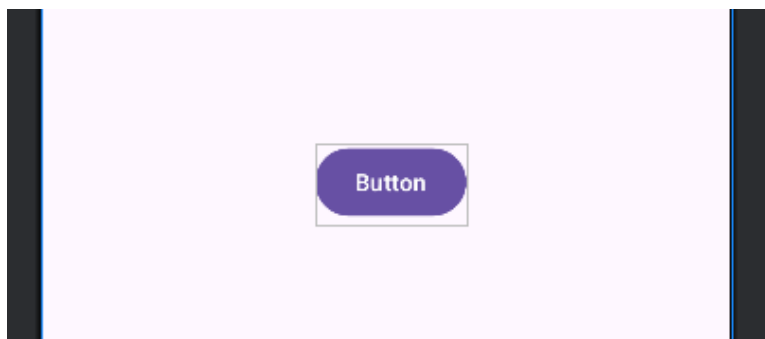
Làm theo các bước sau để thêm một nút:

1. Bắt đầu với một trang trống. Yếu tố TextView không cần thiết, vì vậy trong khi nó vẫn được chọn, nhấn phím Delete hoặc chọn Chỉnh sửa > Xóa. Bây giờ bạn có một bố cục hoàn toàn trống.
2. Kéo một nút từ khung Palette đến bất kỳ vị trí nào trong bố cục. Nếu bạn thả nút vào 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 lên trên, bên trái và bên phải của bố cục như trong hình động dưới đây.



2.3 Thêm nút thứ hai vào bố cục

1. Kéo một nút khác từ khung Palette vào giữa bố cục như hiển thị trong hình động dưới đây. Autoconnect có thể cung cấp các ràng buộc ngang cho bạn (nếu không, bạn có thể kéo chúng tự mình).
2. Kéo một ràng buộc dọc xuống dưới bố cục (tham khảo hình dưới đây).



Bạn có thể xóa các ràng buộc từ một yếu tố bằng cách chọn yếu tố đó và di chuột qua nó để hiển thị nút Xóa các Ràng buộc. Nhấp vào nút này để xóa tất cả các ràng buộc trên yếu tố đã chọn. Để xóa một ràng buộc duy nhất, nhấp vào tay cầm cụ thể đặt ràng buộc.

Để xóa tất cả các ràng buộc trong toàn bộ bố cục, nhấp vào công cụ Xóa Tất cả các Ràng buộc trên thanh công cụ. Công cụ này hữu ích nếu bạn muốn làm lại tất cả các ràng buộc trong bố cục của mình.

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

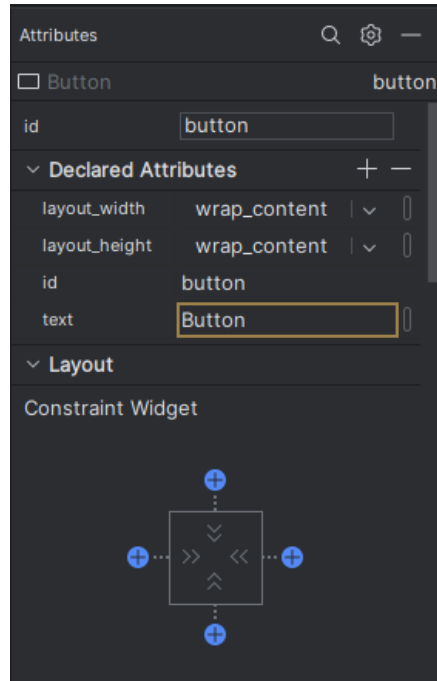
Khung Attributes cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một phần tử giao diện người dùng. Bạn có thể tìm thấy các thuộc tính (được gọi là properties) chung cho tất cả các views trong tài liệu của lớp View.

Trong nhiệm vụ này, bạn sẽ nhập các giá trị mới và thay đổi các giá trị cho các thuộc tính quan trọng của Button, mà áp dụng cho hầu hết các loại View.

3.1 Thay đổi kích thước nút

Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước trên bốn góc của một View để bạn có thể nhanh chóng thay đổi kích thước của View. Bạn có thể kéo các tay cầm trên mỗi góc của View để thay đổi kích thước của nó, nhưng làm như vậy sẽ cố định các kích thước chiều rộng và chiều cao. Tránh cố định kích thước cho hầu hết các yếu tố 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, sử dụng khung Attributes ở bên phải của trình chỉnh sửa bố cục để chọn một chế độ kích thước không sử dụng các kích thước cố định. Khung Attributes bao gồm một bảng điều khiển kích thước vuông được gọi là trình kiểm tra view ở trên cùng. Các biểu tượng bên trong hình vuông đại diện cho các cài đặt chiều cao và chiều rộng như sau:



Trong hình trên:

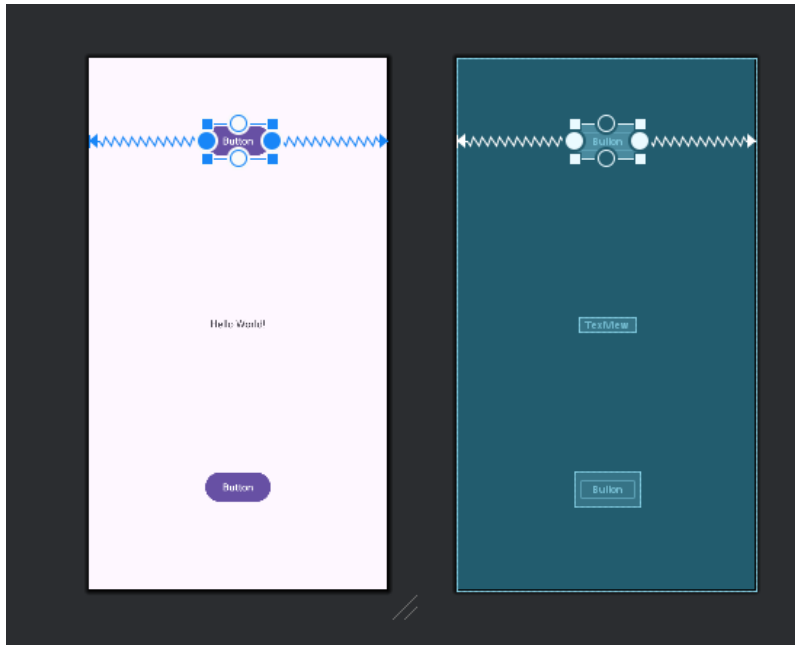
1. **Điều khiển chiều cao:** Điều khiển này xác định thuộc tính `layout_height` và xuất hiện ở hai đoạn trên và dưới của hình vuông. Các góc chỉ ra rằng điều khiển này được đặt thành `wrap_content`, có nghĩa là View sẽ mở rộng theo chiều dọc khi cần để phù hợp với nội dung của nó. "8" chỉ ra một lề tiêu chuẩn được đặt thành 8dp.
2. **Điều khiển chiều rộng:** Điều khiển này xác định thuộc tính `layout_width` và xuất hiện ở hai đoạn bên trái và bên phải của hình vuông. Các góc chỉ ra rằng điều khiển này được đặt thành `wrap_content`. Điều này có nghĩa là View sẽ mở rộng theo chiều ngang khi cần để phù hợp với nội dung của nó, lên đến một lề 8dp.
3. Nút đóng khung Attributes. Nhấp để đóng khung.

Làm theo các bước sau:

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

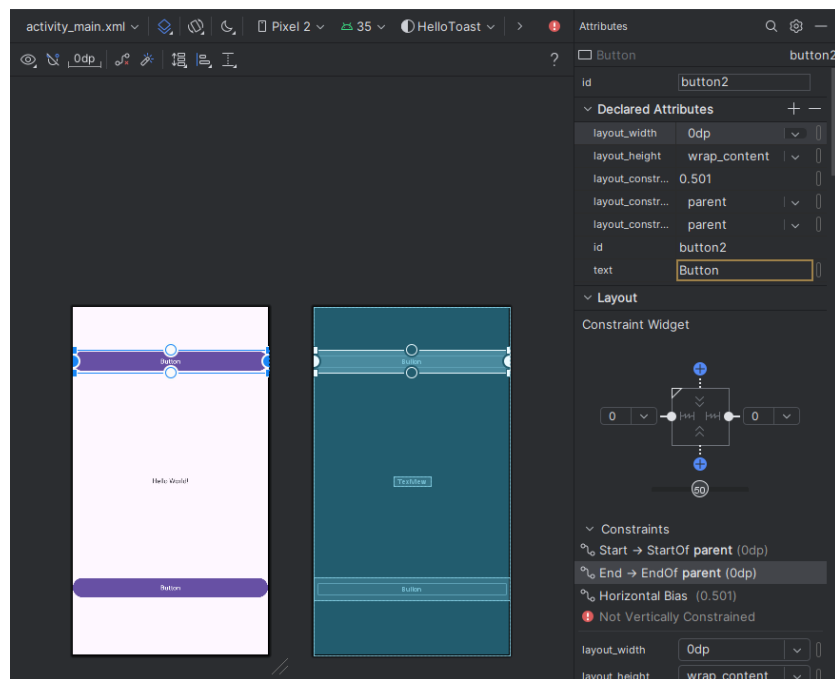


3. Nhấp vào điều khiển chiều rộng hai lần—lần nhấp đầu tiên thay đổi thành Cố định với các đường thẳng, và lần nhấp thứ hai thay đổi thành Khớp với Ràng buộc với các cuộn dây lò xo, như trong hình động dưới đây.



Do kết quả của việc thay đổi điều khiển chiều rộng, thuộc tính `layout_width` trong khung Attributes hiển thị giá trị `match_constraint` và phần tử Button kéo dài theo chiều ngang để lấp đầy không gian giữa bên trái và bên phải của bố cục.

4. Chọn nút thứ hai và thực hiện các thay đổi tương tự với `layout_width` như trong bước trước, như hiển thị trong hình dưới đây.



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

- Cài đặt `match_constraint` mở rộng yếu tố View để lấp đầy bố mẹ của nó theo chiều rộng hoặc chiều cao—tới lề nếu có. Bố mẹ trong trường hợp này là `ConstraintLayout`. Bạn sẽ tìm hiểu thêm về `ConstraintLayout` trong nhiệm vụ tiếp theo.
- Cài đặt `wrap_content` thu hẹp kích thước của yếu tố View sao cho nó vừa đủ để bao bọc nội dung của nó. Nếu không có nội dung, yếu tố View sẽ trở nên vô hình.
- Để chỉ định kích thước cố định điều chỉnh theo kích thước màn hình của thiết bị, sử dụng một số cố định của các điểm ảnh độc lập với mật độ (đơn vị dp). Ví dụ, 16dp có nghĩa là 16 điểm ảnh độc lập với mật độ.

Mẹo: Nếu bạn thay đổi thuộc tính `layout_width` bằng cách sử dụng menu bật lên của nó, thuộc tính `layout_width` sẽ được đặt thành zero vì không có kích thước được đặt. Cài đặt này tương tự như `match_constraint` — view có thể mở rộng hết mức có thể để đáp ứng các ràng buộc và cài đặt lề.

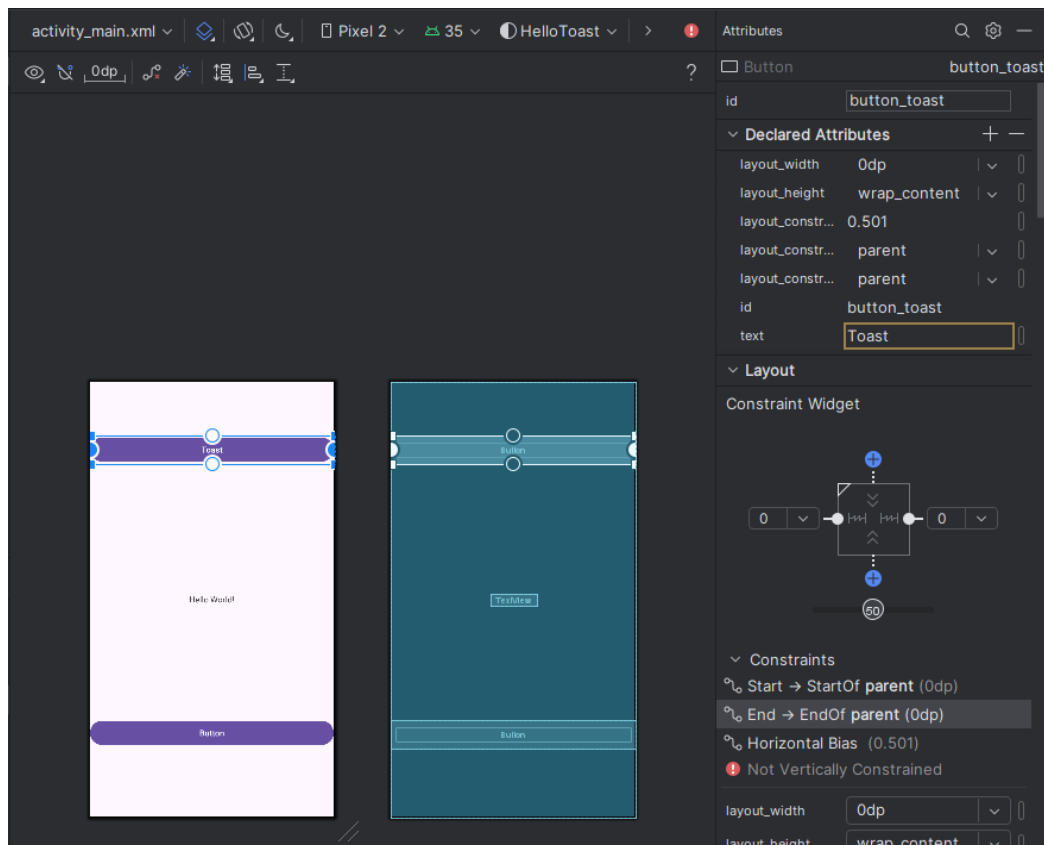
3.2 Thay đổi các thuộc tính của nút

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

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

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

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



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

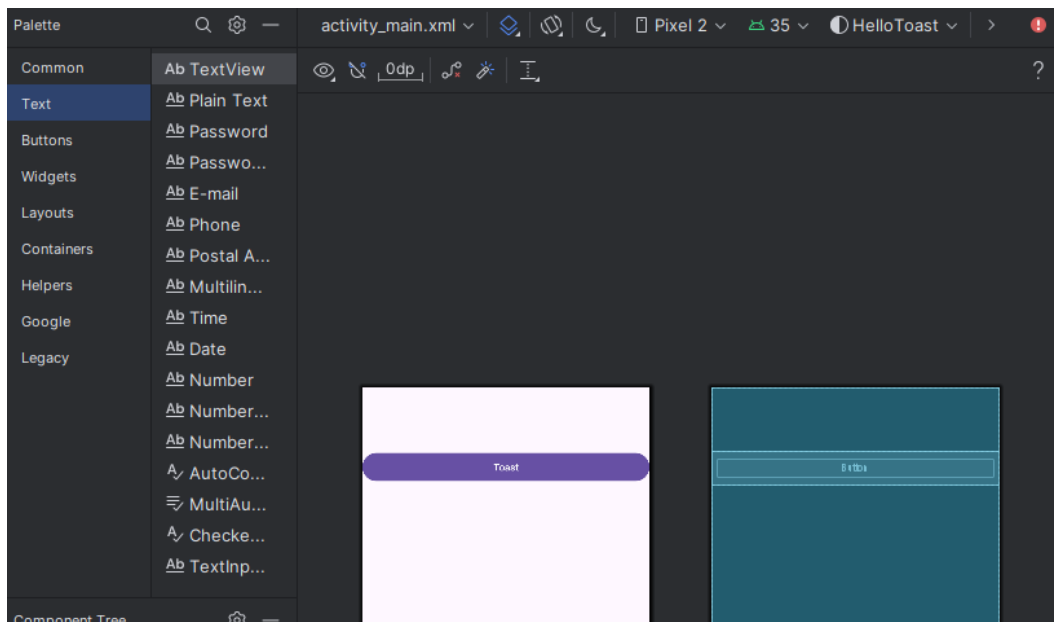
`colorPrimary` là màu chính của chủ đề, một trong những màu cơ bản được xác định trước trong tệp tài nguyên `colors.xml`. Nó được sử dụng cho thanh ứng dụng. Sử dụng các màu cơ bản cho các yếu tố giao diện người dùng khác tạo ra một giao diện người dùng đồng nhất. Bạn sẽ học thêm về chủ đề ứng dụng và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

Nhiệm vụ 4: Thêm `TextEdit` và đặt các thuộc tính

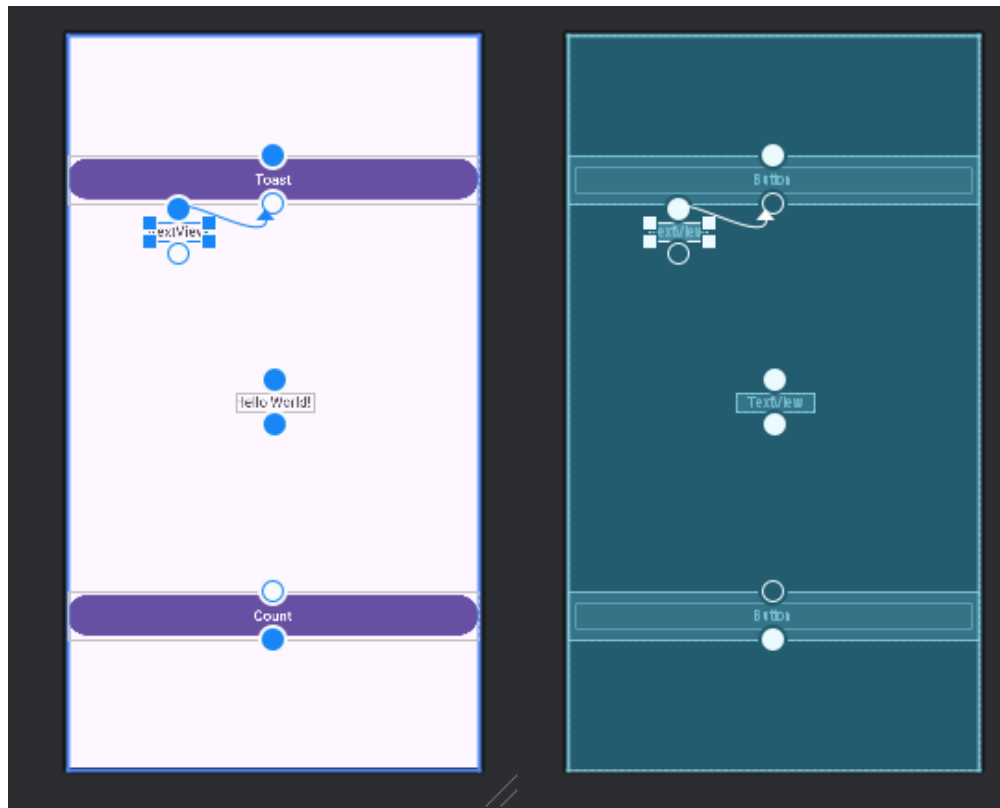
Một trong những lợi ích của `ConstraintLayout` là khả năng căn chỉnh hoặc ràng buộc các yếu tố tương đối với các yếu tố khác. Trong nhiệm vụ này, bạn sẽ thêm một `TextView` ở giữa bố cục, và ràng buộc nó theo chiều ngang vào các lề và theo chiều dọc vào hai yếu tố `Button`. Sau đó, bạn sẽ thay đổi các thuộc tính cho `TextView` trong khung `Attributes`.

4.1 Thêm `TextView` và các ràng buộc

- Như được hiển thị trong hình động dưới đây, kéo một `TextView` từ khung `Palette` lên phần trên của bố cục, và kéo một ràng buộc từ đầu của `TextView` đến tay cầm ở dưới cùng của nút `Toast`. Điều này ràng buộc `TextView` nằm dưới nút.



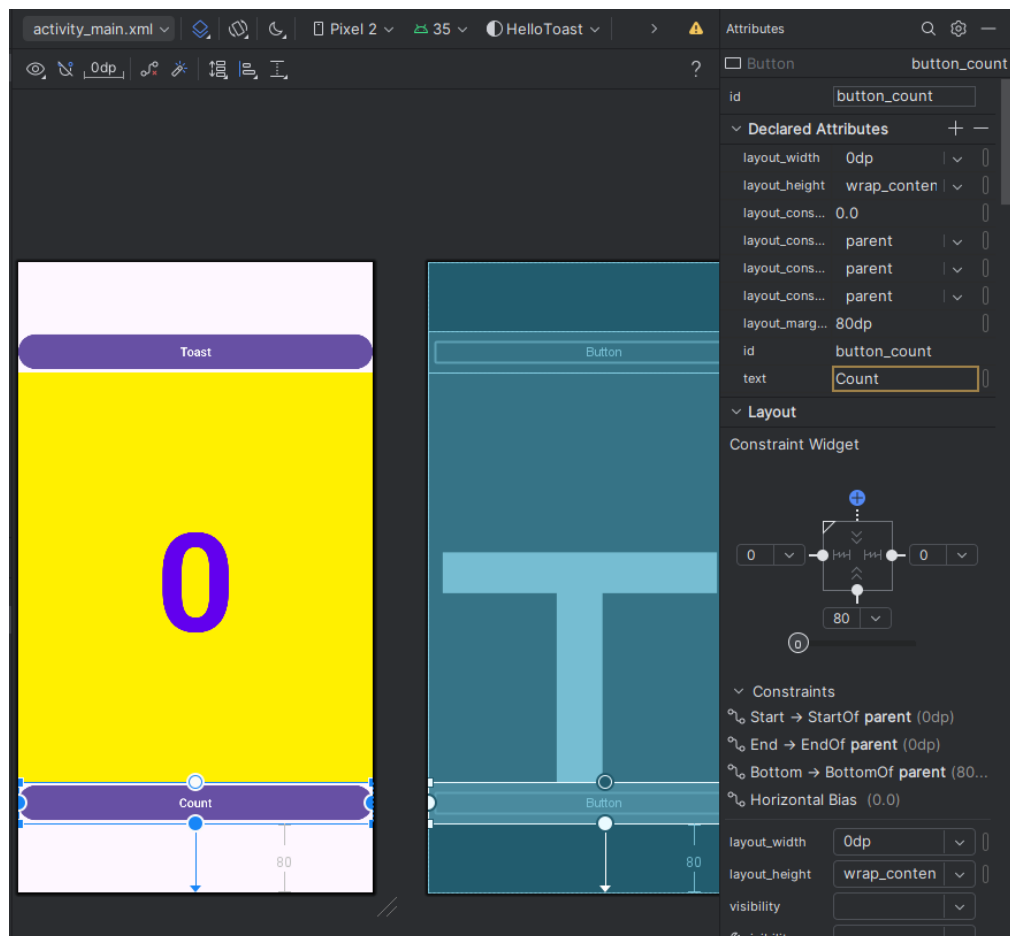
- Như hiển thị trong hình động bên dưới, kéo một ràng buộc từ đáy của TextView đến tay cầm trên đỉnh của Count Button và từ các cạnh của TextView đến các cạnh của bố cục. Điều này ràng buộc TextView ở giữa bố cục giữa hai phần tử Button.



4.2 Đặt các thuộc tính cho TextView


Với TextView được chọn, mở khung Attributes nếu nó chưa được mở. Đặt các thuộc tính cho TextView như hiển thị trong hình động dưới đây. Các thuộc tính bạn chưa gặp phải được giải thích sau hình:

1. Đặt ID thành `show_count`.
2. Đặt văn bản thành 0.
3. Đặt kích thước văn bản thành 160sp.
4. Đặt kiểu văn bản thành B (in đậm) và căn chỉnh văn bản thành ALIGNCENTER (căn giữa đoạn văn).
5. Thay đổi các điều khiển kích thước view ngang và dọc (`layout_width` và `layout_height`) thành `match_constraint`.
6. Đặt màu văn bản thành `@color/colorPrimary`
7. Cuộn xuống khung và nhấp vào Xem tất cả các thuộc tính, cuộn xuống trang thứ hai của các thuộc tính đến background và sau đó nhập #FFF000 cho một sắc màu vàng.
8. Cuộn xuống đến gravity, mở rộng gravity và chọn center_ver (để căn giữa theo chiều dọc).



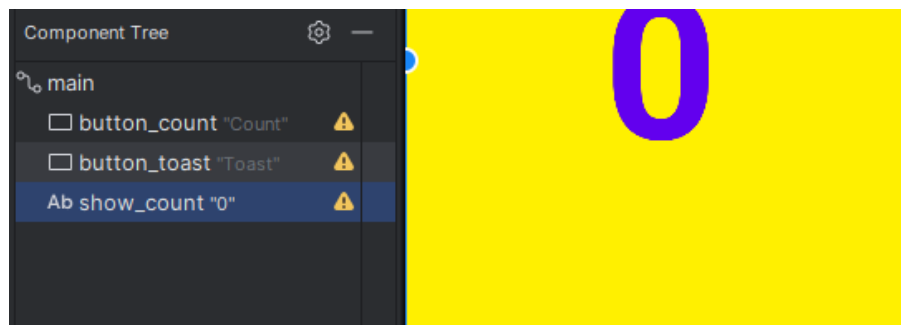
- **textSize**: Kích thước văn bản của TextView. Trong bài học này, kích thước được đặt là 160sp. sp là viết tắt của scale-independent pixel, và giống như dp, là một đơn vị thay đổi theo mật độ màn hình và sở thích kích thước văn bản của người dùng. Sử dụng đơn vị dp khi bạn chỉ định kích thước văn bản để các kích thước được điều chỉnh theo cả mật độ màn hình và sở thích của người dùng.
- **textStyle** và **textAlignment**: Kiểu văn bản được đặt là B (đậm) trong bài học này, và căn chỉnh văn bản được đặt là ALIGNCENTER (căn giữa đoạn văn).
- **gravity**: Thuộc tính gravity xác định cách một View được căn chỉnh trong View hoặc ViewGroup cha của nó. Trong bước này, bạn căn giữa TextView theo chiều dọc trong ConstraintLayout cha.

Bạn có thể nhận thấy rằng thuộc tính background nằm trên trang đầu tiên của khung Attributes cho một Button, nhưng trên trang thứ hai của khung Attributes cho một TextView. Khung Attributes thay đổi cho mỗi loại View: Các thuộc tính phổ biến nhất cho loại View sẽ xuất hiện trên trang đầu tiên và các thuộc tính còn lại sẽ được liệt kê trên trang thứ hai. Để

quay lại trang đầu tiên của khung Attributes, nhấp vào biểu tượng  ở thanh công cụ ở đầu khung.

Nhiệm vụ 5: Chỉnh sửa bố cục trong XML

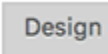
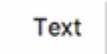
Bố cục ứng dụng Hello Toast gần hoàn thành! Tuy nhiên, một dấu chấm than xuất hiện bên cạnh mỗi yếu tố UI trong Component Tree. Di chuột qua các dấu chấm than này để xem các thông báo cảnh báo, như hiển thị dưới đây. Cùng một cảnh báo xuất hiện cho cả ba yếu tố: các chuỗi cố định nên sử dụng tài nguyên.



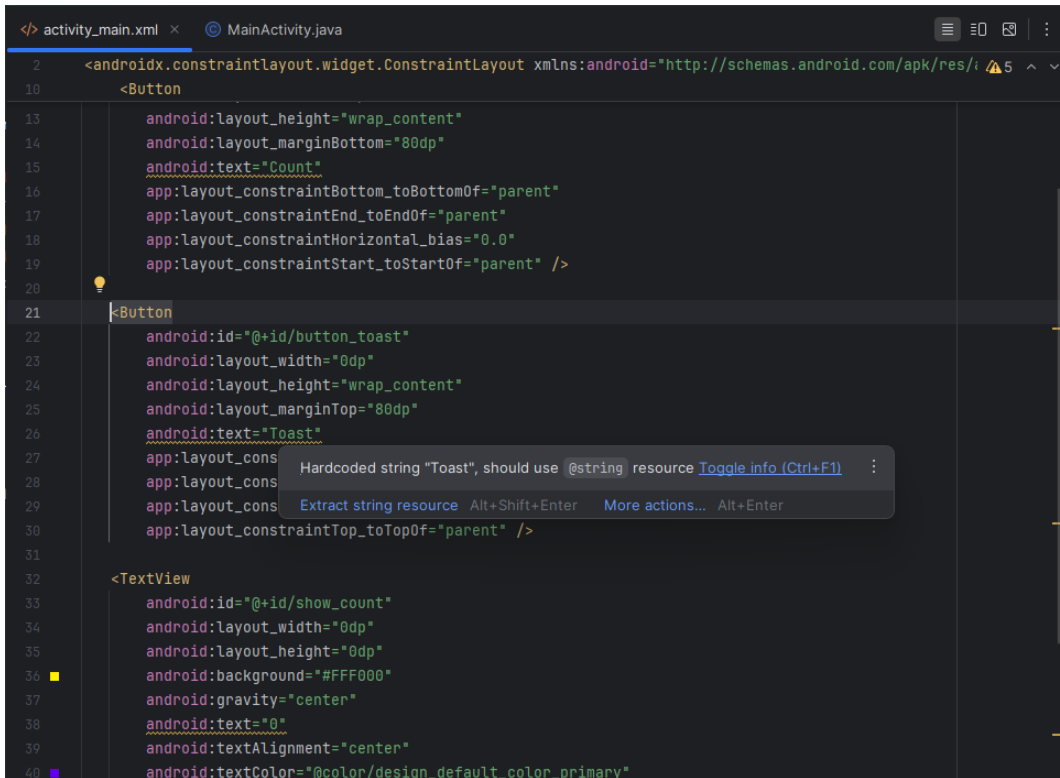
Cách dễ nhất để khắc phục các vấn đề bố cục là chỉnh sửa bố cục trong XML. Mặc dù trình chỉnh sửa bố cục là một công cụ mạnh mẽ, nhưng một số thay đổi dễ thực hiện trực tiếp hơn trong mã nguồn XML.

5.1 Mở mã XML cho bố cục

Đối với nhiệm vụ này, mở tệp `activity_main.xml` nếu nó chưa được mở, và nhấp vào tab Text

  ở dưới cùng của trình chỉnh sửa bố cục.

Trình chỉnh sửa XML xuất hiện, thay thế các khung thiết kế và bản vẽ. Như bạn có thể thấy trong hình dưới đây, hiển thị một phần mã XML cho bố cục, các cảnh báo được đánh dấu - các chuỗi cố định "Toast" và "Count". (Chuỗi cố định "0" cũng được đánh dấu nhưng không được hiển thị trong hình.) Di chuột qua chuỗi cố định "Toast" để xem thông báo cảnh báo.



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/i
10
    <Button
13
        android:layout_height="wrap_content"
14
        android:layout_marginBottom="80dp"
15
        android:text="Count"
16
        app:layout_constraintBottom_toBottomOf="parent"
17
        app:layout_constraintEnd_toEndOf="parent"
18
        app:layout_constraintHorizontal_bias="0.0"
19
        app:layout_constraintStart_toStartOf="parent" />
20
21
    <Button
22
        android:id="@+id/button_toast"
23
        android:layout_width="0dp"
24
        android:layout_height="wrap_content"
25
        android:layout_marginTop="80dp"
26
        android:text="Toast"
27
        app:layout_con
28
        app:layout_con
29
        app:layout_con
30
        app:layout_constraintTop_toTopOf="parent" />
31
32
    <TextView
33
        android:id="@+id/show_count"
34
        android:layout_width="0dp"
35
        android:layout_height="0dp"
36
        android:background="#FFF000"
37
        android:gravity="center"
38
        android:text="0"
39
        android:textAlignment="center"
40
        android:textColor="@color/design_default_color_primary"

```

5.2 Trích xuất tài nguyên chuỗi

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

1. Nhấp một lần vào từ "Toast" (cảnh báo được đánh dấu đầu tiên).
2. Nhấn Alt-Enter trong Windows hoặc Option-Enter trong macOS và chọn Trích xuất tài nguyên chuỗi từ menu bật lên.
3. Nhập button_label_toast cho Tên tài nguyên.
4. Nhấp vào OK. Một tài nguyên chuỗi được tạo trong tệp values/res/string.xml, và chuỗi trong mã của bạn được thay thế bằng một tham chiếu đến tài nguyên: @string/button_label_toast.
5. Trích xuất các chuỗi còn lại: button_label_count cho "Count", và count_initial_value cho "0".
6. Trong khung Dự án > Android, mở rộng values trong res, và sau đó nhấp đúp vào strings.xml để xem các tài nguyên chuỗi của bạn trong tệp strings.xml.

```
<resources>
    <string name="app_name">Hello Toast</string>
    <string name="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 nhiệm vụ tiếp theo hiển thị một 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: Các tài nguyên chuỗi bao gồm tên ứng dụng, 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 các trình xử lý onClick cho các nút

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

6.1. Thêm thuộc tính onClick và trình xử lý cho mỗi nút

Một trình xử lý click là một phương thức được gọi khi người dùng nhấp hoặc nhấn vào một yếu tố UI có thể nhấp. Trong Android Studio, bạn có thể chỉ định tên của phương thức trong trường onClick trong khung Attributes của tab Design. Bạn cũng có thể chỉ định tên của phương thức xử lý trong trình chỉnh sửa XML bằng cách thêm thuộc tính android:onClick vào nút. Bạn sẽ sử dụng phương pháp sau vì bạn chưa tạo các phương thức xử lý và trình chỉnh sửa XML cung cấp một cách tự động để tạo các phương thức đó.

1. chỉnh sửa XML mở (tab Text), tìm nút với android:id được đặt là button_toast.

```
<Button
    android:id="@+id/button_toast"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="80dp"
    android:text="@string/button_label_toast"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    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 số kết thúc `/>`

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

3. Nhấp vào biểu tượng bóng đèn để xuất hiện bên cạnh thuộc tính. Chọn Create click handler, chọn MainActivity, và nhấp vào OK. 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)`' in MainActivity, và nhấp vào OK. Hành động này tạo một phương thức trình giữ chỗ cho phương thức `showToast()` trong MainActivity, như hiển thị ở cuối các bước này.
4. Lặp lại hai bước cuối với nút `button_count`: Thêm thuộc tính `android:onClick` vào cuối, và thêm trình xử lý click:

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

Mã XML cho các yếu tố giao diện người dùng trong `ConstraintLayout` bây giờ trông như thế này:

```
<Button
    android:id="@+id/button_count"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="80dp"
    android:text="@string/button_label_count"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    android:onClick="countUp"/>
```

```
<Button
    android:id="@+id/button_toast"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="80dp"
    android:text="@string/button_label_toast"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
```

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

```
<TextView
    android:id="@+id/show_count"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:background="#FFF000"
    android:gravity="center"
    android:text="@string/count_initial_value"
    android:textAlignment="center"
    android:textColor="@color/design_default_color_primary"
    android:textSize="160sp"
    android:textStyle="bold"
    app:layout_constraintBottom_toTopOf="@+id/button_count"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/button_toast"
    app:layout_constraintVertical_bias="0.5" />
```

5. Nếu MainActivity.java chưa được mở, hãy mở rộng java trong khung Dự án > Chế độ xem Android, mở rộng com.example.android.hellotoast, và sau đó nhấp đúp vào MainActivity. Trình chỉnh sửa mã xuất hiện với mã trong MainActivity:

```
package com.example.hellotoast;
import ...
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
            Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
            v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
            return insets;
        });
    }
    public void showToast(View view) {
    }
    public void countUp(View view) {
    }
}
```

6.2. Chỉnh sửa trình xử lý nút

Toast Bạn sẽ chỉnh sửa phương thức `showToast()` — trình xử lý click nút Toast trong `MainActivity` — để hiển thị một thông báo. Một Toast cung cấp một cách để hiển thị một thông báo đơn giản trong một cửa sổ pop-up nhỏ. Nó chỉ chiếm không gian cần thiết cho thông điệp. Hoạt động hiện tại vẫn hiển thị và tương tác. Một Toast có thể hữu ích để kiểm tra tính tương tác trong ứng dụng của bạn - thêm một thông điệp Toast để hiển thị kết quả của việc nhấn nút hoặc thực hiện một hành động.

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

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

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

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

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

Câu lệnh này chưa hoàn chỉnh cho đến khi bạn hoàn thành tất cả các bước.

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

```
Toast toast = Toast.makeText(this,
```

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

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

5. Cung cấp thời lượng cho việc hiển thị. Ví dụ, `Toast.LENGTH_SHORT` hiển thị toast trong thời gian tương đối ngắn.

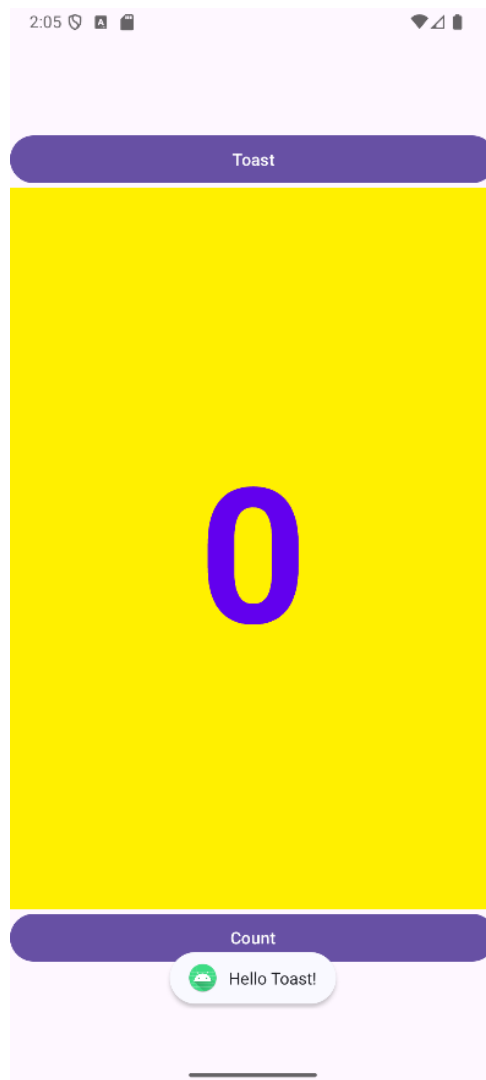
```
Toast toast = Toast.makeText(this, R.string.toast_message, Toast.LENGTH_SHORT);
```

Thời lượng hiển thị của một 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.

6. Hiển thị Toast bằng cách gọi `show()`. Dưới 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 kiểm tra rằng thông báo Toast xuất hiện khi bạn nhấn vào nút Toast.



6.3. Chỉnh sửa trình xử lý nút Count

Bạn sẽ chỉnh sửa phương thức `countUp()` — trình xử lý click nút Count trong MainActivity — để hiển thị số đếm hiện tại sau khi nhấn nút 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 cập nhật đến TextView để hiển thị.

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

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

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

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

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

Nếu biểu tượng bóng đèn đỏ không xuất hiện, chọn biểu thức `mCount++`. Biểu tượng bóng đèn đỏ cuối cùng sẽ xuất hiện.

3. Nhấp vào biểu tượng bóng đèn đỏ và chọn Create field 'mCount' từ menu bật lên. Điều này tạo ra một biến thành viên riêng ở đầu MainActivity, và Android Studio giả định rằng bạn muốn nó 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 thành viên không:

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

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

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

6. Bây giờ bạn đã có mShowCount, bạn có thể lấy một tham chiếu đến TextView bằng cách sử dụng ID mà bạn đã đặt trong tệp bố cục. Để lấy tham chiếu này chỉ một lần, hãy chỉ định nó trong phương thức onCreate(). Như bạn đã học trong một bài học khác, phương thức onCreate() được sử dụng để khởi tạo bố cục, có nghĩa là đặt nội dung của màn hình vào bố cục XML. Bạn cũng có thể sử dụng nó để lấy các tham chiếu đến các yếu tố UI khác trong bố cục, chẳng hạn như TextView. Tìm 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:

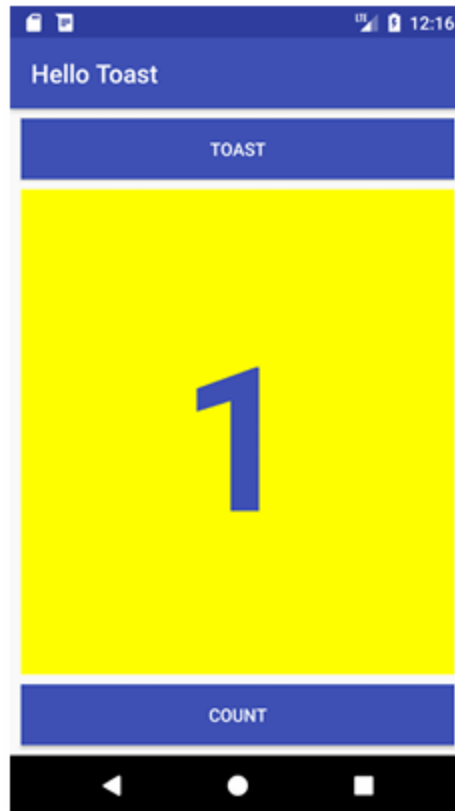
```
@Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_main); mShowCount = (TextView) findViewById(R.id.show_count); }
```

Một View, giống như một chuỗi, là một tài nguyên có thể có ID. Cuộc gọi findViewById lấy ID của một view làm tham số của nó và trả về View. Vì phương thức này trả về một View, bạn phải chuyển đổi kết quả thành kiểu view mà bạn mong đợi, trong trường hợp này là (TextView).

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

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

9. Chạy ứng dụng để kiểm tra rằng số đếm tăng khi bạn nhấn vào nút Count.

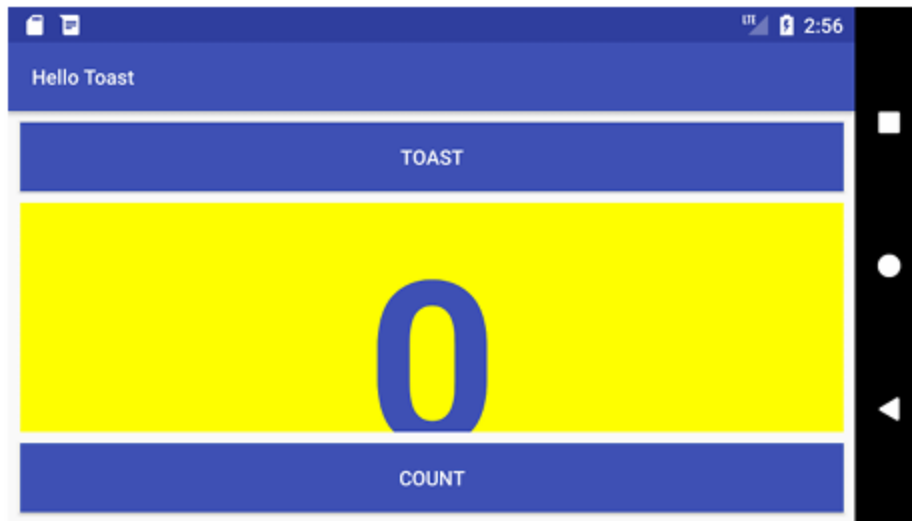


Mẹo: Để có hướng dẫn chuyên sâu về việc sử dụng `ConstraintLayout`, hãy xem Codelab [Sử dụng ConstraintLayout để thiết kế các view của bạn](#).

Thử thách mã hóa

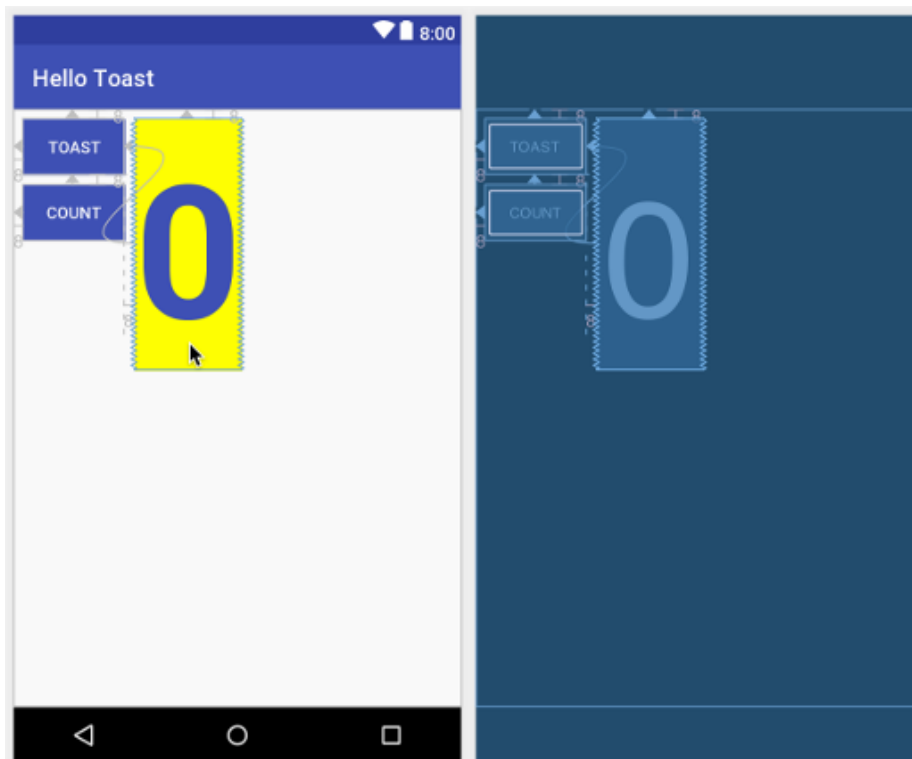
Lưu ý: Tất cả các thử thách mã hóa đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

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



Thử thách: Thay đổi bố cục sao cho trông đẹp ở cả hai 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 cho hướng dẫn sao chép và tái cấu trúc một dự án.)
3. Thay đổi bố cục sao cho nút Toast và nút Count xuất hiện ở bên trái, như trong hình dưới đây. TextView xuất hiện bên cạnh chúng, nhưng chỉ đủ rộng để hiển thị nội dung của nó. (Gợi ý: Sử dụng wrap_content).
4. Chạy ứng dụng trong cả hai hướng ngang và dọc.





Tóm tắt

View, ViewGroup và bố cục:

- Tất cả các yếu tố UI đều là các lớp con của lớp View và do đó thừa kế nhiều thuộc tính của lớp View.
- Các yếu tố View có thể được nhóm lại bên trong một ViewGroup, hoạt động như một hộp chứa. Mỗi quan hệ này là mối quan hệ cha-con, trong đó cha là ViewGroup và con là View hoặc một ViewGroup khác.
- Phương thức onCreate() được sử dụng để khởi tạo bố cục, có nghĩa là đặt nội dung màn hình vào bố cục XML. Bạn cũng có thể sử dụng nó để lấy tham chiếu đến các yếu tố UI khác trong bố cục.
- Một View, giống như một chuỗi, là một tài nguyên có thể có ID. Lệnh gọi findViewById lấy ID của một view làm tham số của nó và trả về View.

Sử dụng trình chỉnh sửa bố cục:

- Nhấp vào tab Design để thao tác các yếu tố và bố cục, và tab Text để chỉnh sửa mã XML cho bố cục.
- Trong tab Design, khung Palettes hiển thị các yếu tố UI mà bạn có thể sử dụng trong bố cục ứng dụng của bạn, và khung Component tree hiển thị cấu trúc phân cấp của các yếu tố UI.
- Các khung thiết kế và bản vẽ của trình chỉnh sửa bố cục hiển thị các yếu tố UI trong bố cục.
- Tab Attributes hiển thị khung Attributes để thiết lập thuộc tính cho một yếu tố UI.

- Tay cầm ràng buộc: Nhấp vào một tay cầm ràng buộc, hiển thị dưới dạng một vòng tròn ở mỗi bên của một yếu tố, sau đó kéo đến một tay cầm ràng buộc khác hoặc đến ranh giới cha để tạo ràng buộc. Ràng buộc được biểu diễn bằng đường zigzag.
- Tay cầm thay đổi kích thước: Bạn có thể kéo tay cầm thay đổi kích thước hình vuông để thay đổi kích thước của yếu tố. Khi kéo, tay cầm thay đổi thành một góc xiên.
- Khi được bật, công cụ Autoconnect tự động tạo hai hoặc nhiều ràng buộc cho một yếu tố UI vào bố cục cha. Sau khi bạn kéo yếu tố vào bố cục, nó tạo các ràng buộc dựa trên vị trí của yếu tố.
- Bạn có thể xóa các ràng buộc từ một yếu tố bằng cách chọn yếu tố đó và di con trỏ qua nó để hiển thị nút Clear Constraints. Nhấp vào nút này để xóa tất cả các ràng buộc trên yếu tố đã chọn. Để xóa một ràng buộc đơn, nhấp vào tay cầm cụ thể đặt ràng buộc đó.
- Khung Attributes cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một yếu tố UI. Nó cũng bao gồm một bảng kích thước hình vuông gọi là view inspector ở trên cùng. Các biểu tượng bên trong hình vuông đại diện cho các cài đặt chiều cao và chiều rộng.

Thiết lập chiều rộng và chiều cao bố cục:

- Các thuộc tính `layout_width` và `layout_height` thay đổi khi bạn thay đổi các điều khiển kích thước chiều cao và chiều rộng trong view inspector. Các thuộc tính này có thể nhận một trong ba giá trị cho một `ConstraintLayout`:
- Cài đặt `match_constraint` mở rộng view để lấp đầy cha của nó bằng chiều rộng hoặc chiều cao – đến một biên nếu được đặt.
- Cài đặt `wrap_content` thu nhỏ kích thước view để view vừa đủ lớn để bao quanh nội dung của nó. Nếu không có nội dung, view sẽ trở nên vô hình.
- Sử dụng số cố định dp (density-independent pixels) để chỉ định kích thước cố định, điều chỉnh cho kích thước màn hình của thiết bị.

Trích xuất tài nguyên chuỗi:

- Thay vì mã hóa chuỗi, tốt nhất là sử dụng tài nguyên chuỗi, là đại diện cho các chuỗi đó. Thực hiện theo các bước sau:
- Nhấp một lần vào chuỗi được mã hóa cứng để trích xuất, nhấn Alt-Enter (Option-Enter trên Mac), và chọn Extract string resources từ menu bật lên.
- Đặt tên tài nguyên.
- Nhấp OK. Điều này tạo một tài nguyên chuỗi trong tệp `values/res/string.xml`, và chuỗi trong mã của bạn được thay thế bằng một tham chiếu đến tài nguyên:
`@string/button_label_toast`

Xử lý các lượt nhấp:

- Một phương thức xử lý click được gọi khi người dùng nhấp hoặc nhấn vào một yếu tố UI.

- Chỉ định một phương thức xử lý click cho một yếu tố UI như Button bằng cách nhập tên của nó trong trường onClick ở khung Attributes của tab Design, hoặc trong trình chỉnh sửa XML bằng cách thêm thuộc tính android:onClick vào một yếu tố UI như Button.
- Tạo phương thức xử lý click trong Activity chính bằng cách sử dụng tham số View. Ví dụ: `public void showToast(View view) {/...}`.
- Bạn có thể tìm 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 thông báo Toast:

- Một Toast cung cấp một cách để hiển thị một thông báo đơn giản trong một cửa sổ pop-up nhỏ. Nó chỉ chiếm không gian cần thiết cho thông điệp.
- Để tạo một thể hiện của Toast, thực hiện theo các bước sau:
- Gọi phương thức `makeText()` trên lớp Toast.
- Cung cấp ngữ cảnh của Activity ứng dụng và thông báo để hiển thị (chẳng hạn như tài nguyên chuỗi).
- Cung cấp thời lượng hiển thị, ví dụ, `Toast.LENGTH_SHORT` cho một khoảng thời gian ngắn. Thời lượng có thể là `Toast.LENGTH_LONG` hoặc `Toast.LENGTH_SHORT`.
- Hiển thị Toast bằng cách gọi `show()`.

1.2 Phần B: Trình chỉnh sửa bố cục

Giới thiệu Như bạn đã học trong Bài 1.2 Phần A: Giao diện người dùng tương tác đầu tiên của bạn, bạn có thể tạo giao diện người dùng (UI) sử dụng `ConstraintLayout` trong trình chỉnh sửa bố cục, đặt các yếu tố UI vào một bố cục bằng cách sử dụng các kết nối ràng buộc đến các yếu tố khác và cạnh của bố cục. `ConstraintLayout` được thiết kế để dễ dàng kéo các yếu tố UI 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 children hoặc child views). Thực hành này cho thấy nhiều tính năng hơn của `ConstraintLayout` và trình chỉnh sửa bố cục. Thực hành này cũng giới thiệu hai lớp con khác của `ViewGroup`:

- **LinearLayout:** Một nhóm căn chỉnh các yếu tố View con bên trong nó theo chiều ngang hoặc dọc.
- **RelativeLayout:** Một nhóm các yếu tố View con trong đó mỗi yếu tố View được đặt và căn chỉnh tương đối với các yếu tố View khác trong `ViewGroup`. Các vị trí của các yếu tố View con được mô tả liên quan đến nhau hoặc đến `ViewGroup` cha.

Những gì bạn nên biết trước:

- Tạo một ứng dụng Hello World với Android Studio.
- Chạy một ứng dụng trên trình giả lập hoặc thiết bị.

- Tạo một bố cục đơn giản cho một ứng dụng với ConstraintLayout.
- Trích xuất và sử dụng tài nguyên chuỗi.

Những gì bạn sẽ học:

- Cách tạo một biến thể bố cục cho hướng ngang (landscape).
- Cách tạo một biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Cách sử dụng ràng buộc cơ sở để căn chỉnh các yếu tố UI với văn bản.
- Cách sử dụng các nút pack và align để căn chỉnh các yếu tố trong bố cục.
- Cách đặt các view trong một LinearLayout.
- Cách đặt các view trong một RelativeLayout.

Những gì bạn sẽ làm:

- Tạo một biến thể bố cục cho hướng hiển thị ngang.
- Tạo một biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Sửa đổi bố cục để thêm các ràng buộc cho các yếu tố UI.
- Sử dụng các ràng buộc cơ sở của ConstraintLayout để căn chỉnh các yếu tố với văn bản.
- Sử dụng các nút pack và align của ConstraintLayout để căn chỉnh các yếu tố.
- Thay đổi bố cục để sử dụng LinearLayout.
- Đặt các yếu tố trong một LinearLayout.
- Thay đổi bố cục để sử dụng RelativeLayout.
- Sắp xếp lại các view trong bố cục chính để tương đối với nhau.

Tổng quan về ứng dụng:

Ứng dụng Hello Toast trong bài học trước sử dụng ConstraintLayout để sắp xếp các yếu tố UI trong bố cục Activity, như được hiển thị trong hình dưới đây.