

Nhà phát triển Android

Cơ bản

(Phiên bản 2) khóa học



Cập nhật lần cuối vào Thứ Ba, ngày 11 tháng 9 năm 2018

Khóa học này được nhóm Đào tạo nhà phát triển của Google tạo ra.

- Để biết thông tin chi tiết về khóa học, bao gồm các liên kết đến tất cả các chương khái niệm, ứng dụng và slide, hãy xem [Cơ bản dành cho nhà phát triển Android \(Phiên bản 2\)](#).

nhaphatnhanh.android.com/courses/adfv2

Ghi chú: Khóa học này sử dụng các thuật ngữ "codelab" và "thực hành" thay thế cho nhau.

Chúng tôi khuyên bạn nên sử dụng [phiên bản trực tuyến](#) của khóa học này thay vì tệp PDF tĩnh này để đảm bảo bạn đang sử dụng nội dung mới nhất.

Nhìn thấy nhaphatnhanh.android.com/courses/adf-v2.

Đơn vị 1: Bắt đầu

Tệp PDF này chứa một bản chụp nhanh một lần về các bài học trong **Phần 1: Bắt đầu**.

Bài học trong đơn vị này

Bài 1: Xây dựng ứng dụng đầu tiên của bạn

1.1: Android Studio và Hello World

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

1.2B: Trình chỉnh sửa bối cảnh

1.3: Văn bản và chế độ xem cuộn

1.4: Tài nguyên có sẵn

Bài 2: Hoạt động

2.1: Hoạt động và mục đích

2.2: Vòng đời và trạng thái hoạt động

2.3: Ý định ngầm định

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

3.1: Trình gỡ lỗi

3.2: Kiểm tra đơn vị

3.3: Thư viện hỗ trợ

Bài 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 điều bạn nên biết

Bạn sẽ có thể:

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

Những gì bạn cần

- Một máy tính chạy Windows hoặc Linux, hoặc một máy Mac chạy macOS. Xem [Trang tải xuống Android Studio](#) để cập nhật các yêu cầu hệ thống.
- Có kết nối Internet hoặc một cách khác để tải phiên bản Android Studio và Java mới nhất vào máy tính của bạn.

Những gì bạn sẽ học được

- 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 mẫu.
- Cách thêm thông báo nhật ký vào ứng dụng của bạn để gỡ lỗi.

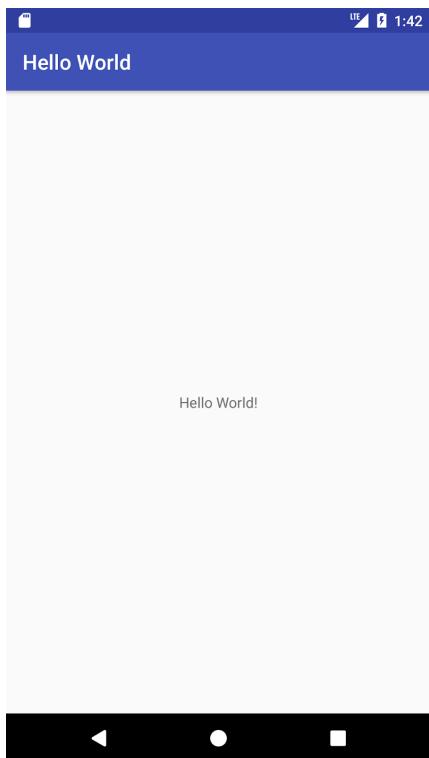
Bạn sẽ làm gì

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

Tổng quan về ứng dụng

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

Ứng dụng hoàn thiện sẽ trông như thế này:



Nhiệm vụ 1: Cài đặt Android Studio

Android Studio cung cấp một môi trường phát triển tích hợp (IDE) hoàn chỉnh bao gồm 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 ứng dụng của mình bằng nhiều trình giả lập được cấu hình sẵn hoặc trên thiết bị di động của riêng bạn, xây dựng ứng dụng sản xuất và xuất bản trên cửa hàng Google Play.

Ghi chú: 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 [Studio Android](#).

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

Để bắt đầu và chạy với Android Studio, trước tiên hãy kiểm tra [yêu cầu hệ thống](#) để đảm bảo hệ thống của bạn đáp ứng được các yêu cầu này. Việc cài đặt tương tự cho tất cả các nền tảng. Mọi sự khác biệt sẽ được ghi chú bên dưới.

- 1.Điều hướng đến [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 đều được chọn để cài đặt.
- 3.Sau khi hoàn tất cài đặt, Trình hướng dẫn thiết lập sẽ tải xuống và cài đặt một số phần bổ sung thành phần 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 vẻ thừa thãi.
- 4.Khi quá trình tải xuống hoàn tất, Android Studio sẽ khởi động và bạn đã sẵn sàng tạo dự án đầu tiên của mình.

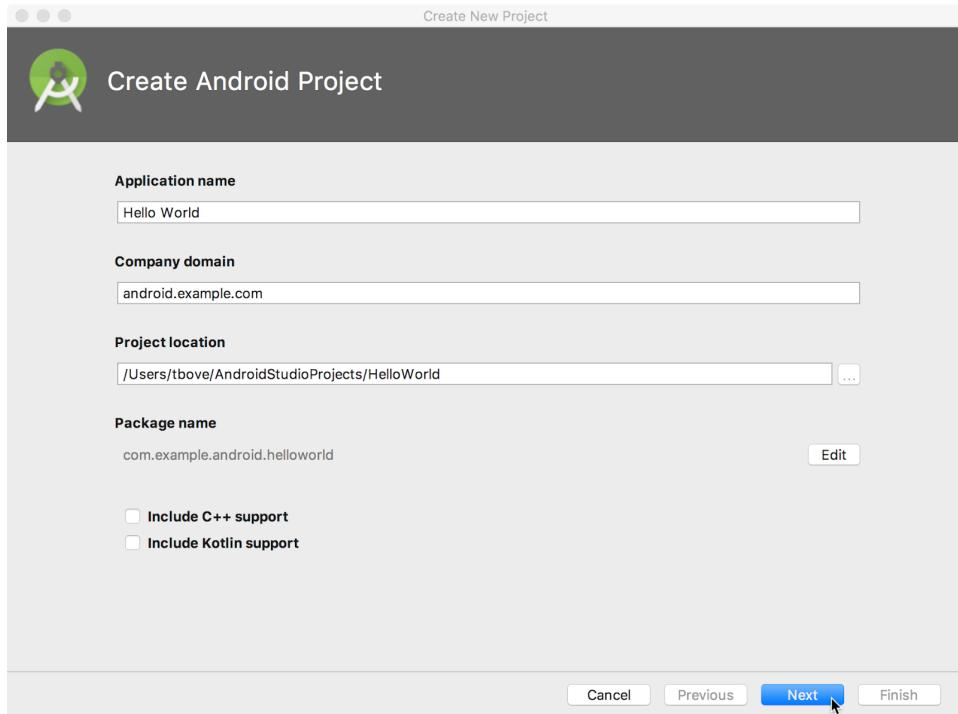
Xử lý sự cố:Nếu bạn gặp vấn đề với cài đặt của mình, hãy kiểm tra [Ghi chú phát hành Android Studio](#) hoặc nhận sự trợ giúp từ người hướng dẫn của bạn.

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 tìm hiểu những điều cơ bản về phát triển bằng Android Studio.

2.1 Tạo dự án ứng dụng

- 1.Mở Android Studio nếu nó chưa được mở.
- 2.Trong chính **Chào mừng đến với Android Studio**cửa sổ, nhấp chuột **Bắt đầu một dự án Android Studio mới**.
- 3.Trong **Tạo dự án Android**cửa sổ, nhập **Xin chào thế giới**cho **Tên ứng dụng**.



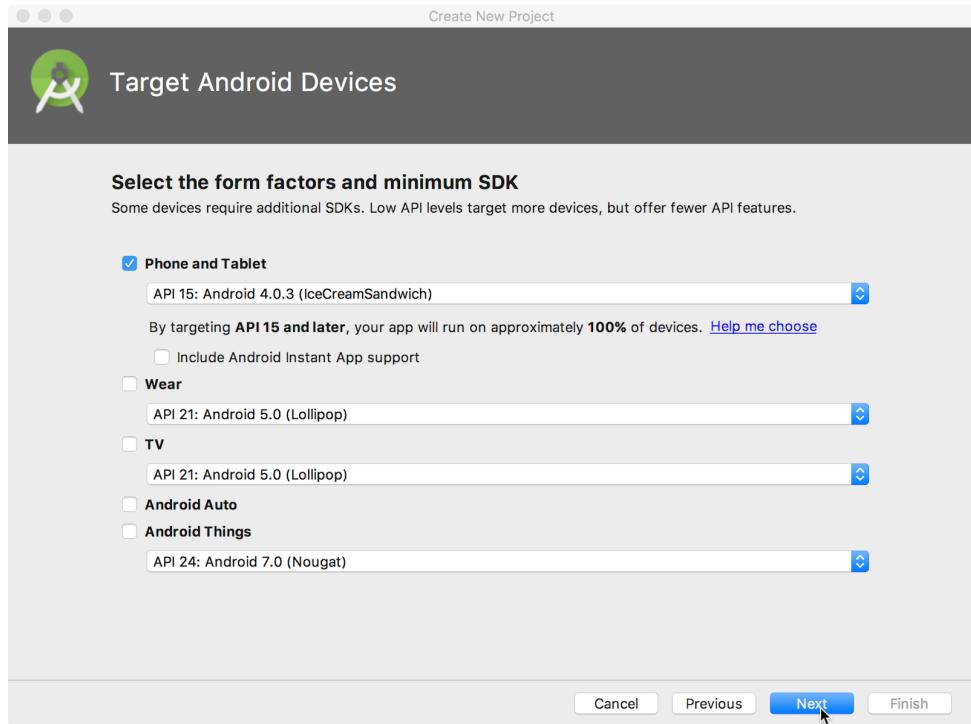
4.Xác minh rằng mặc định **Vị trí dự án** là nơi bạn muốn lưu trữ ứng dụng Hello World và các dự án Android Studio khác hoặc thay đổi nó thành thư mục bạn muốn.

5.Chấp nhận mặc định **android.example.com** vì **Tên miền công ty** hoặc tạo một tên miền công ty độc đáo.

Nếu bạn không có kế hoạch phát hành ứng dụng của mình, bạn có thể chấp nhận mặc định. Lưu ý rằng việc thay đổi tên gói ứng dụng của bạn sau này là công việc bổ sung.

6.Bỏ chọn các tùy chọn để **Bao gồm hỗ trợ C++** và **Bao gồm hỗ trợ Kotlin** và nhấp vào **Kế tiếp**.

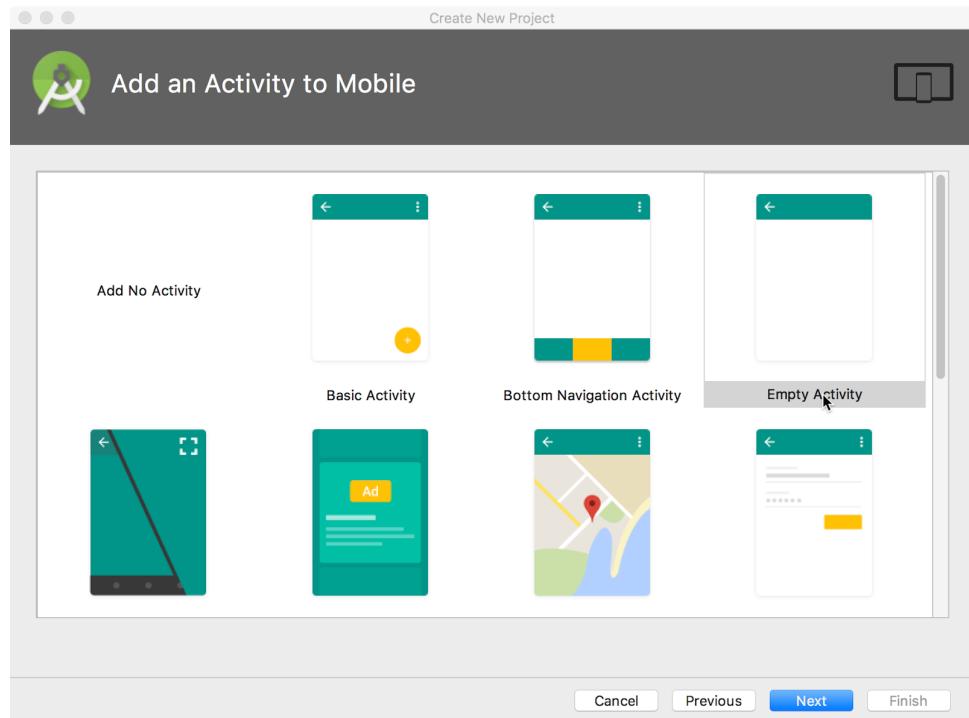
7.Trên **Mục tiêu thiết bị** Android màn hình, **Điện thoại và máy tính bảng** nên được lựa chọn. Đảm bảo rằng **API 15: Android 4.0.3 IceCreamSandwich** được đặt là SDK tối thiểu; nếu không, hãy sử dụng menu bật lên để thiết lập.



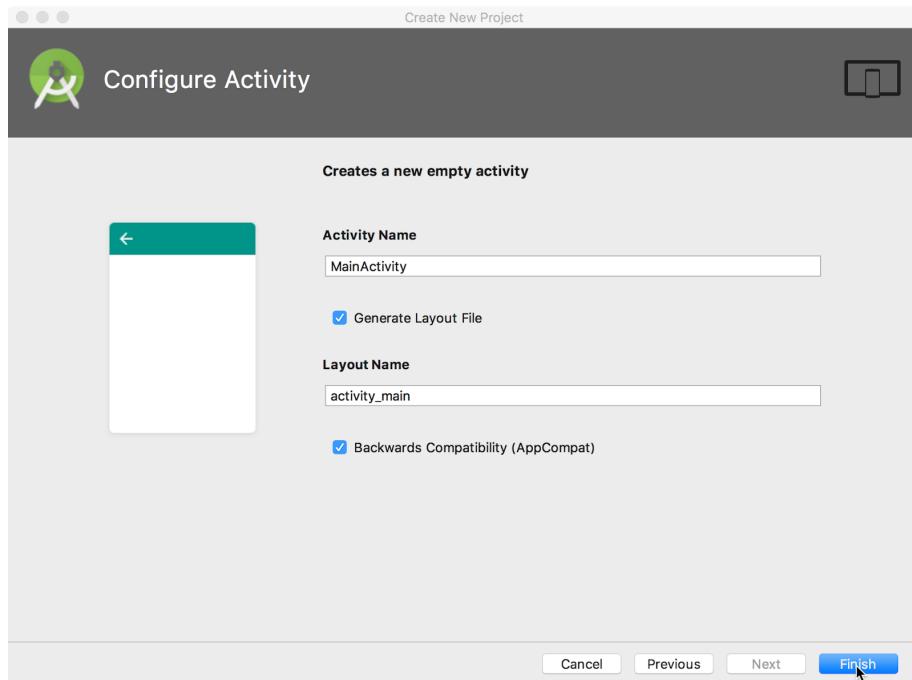
Đây là các thiết lập được sử dụng bởi các ví dụ trong các bài học của khóa học này. Tính đến thời điểm viết bài này, các thiết lập này giúp ứng dụng Hello World của bạn tương thích với 97% thiết bị Android đang hoạt động trên Google Play Store.

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

9.Các**Thêm một hoạt động**cửa sổ xuất hiện. Một**Hoạt động** là một điều duy nhất, tập trung mà người dùng có thể làm. Đây là thành phần quan trọng của bất kỳ ứng dụng Android nào. Một**Hoạt động** thường có một bố cục liên quan đến nó để xác định cách các thành phần UI xuất hiện trên màn hình. Android Studio cung cấp Hoạt động mẫu để giúp bạn bắt đầu. Đối với dự án Hello World, hãy chọn**Hoạt động trống**như hiển thị bên dưới và nhấp vào**Kế tiếp**.



10. Các **Cấu hình hoạt động** màn hình xuất hiện (khác nhau tùy thuộc vào mẫu bạn đã chọn ở bước trước). Theo mặc định, ô trống **Hoạt động** được cung cấp bởi mẫu được đặt tên **Hoạt động chính**. Bạn có thể thay đổi điều này nếu bạn muốn, nhưng bài học này sử dụng **Hoạt động chính**.



11.Hãy chắc chắn rằng **Tạo tệp Bố cục** tùy chọn được chọn. Tên bố cục theo mặc định là `activity_main`.

Bạn có thể thay đổi điều này nếu bạn muốn, nhưng bài học này sử dụng `activity_main`.

12.Hãy chắc chắn rằng **Khả năng tương thích ngược (App Compat)** tùy chọn được chọn. Điều này đảm bảo rằng ứng dụng của bạn sẽ tương thích ngược với các phiên bản Android trước đó.

13.Nhấp chuột **Hoàn thành**.

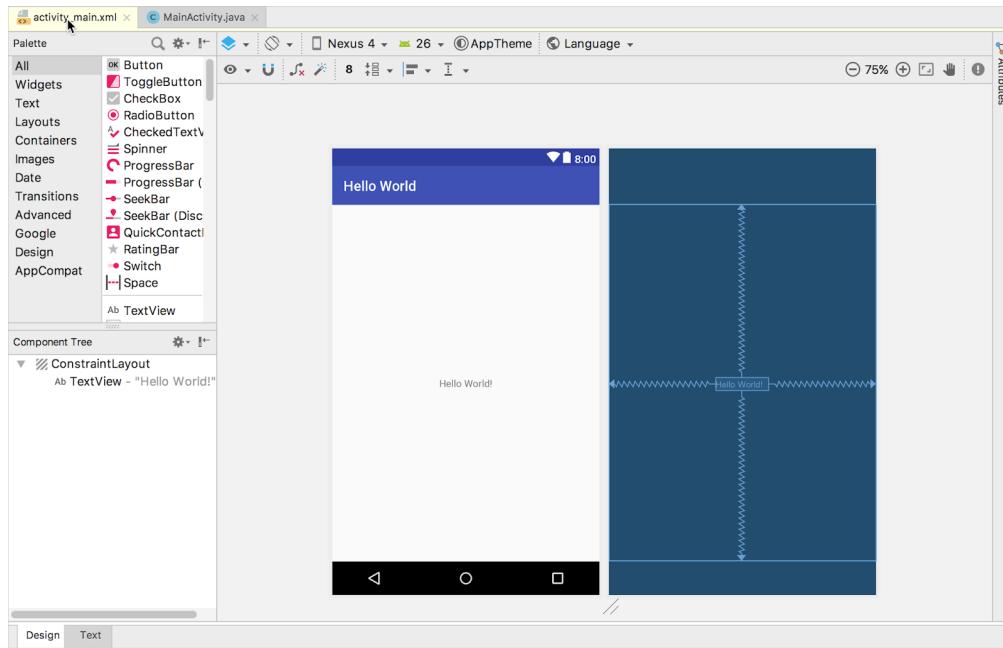
Android Studio tạo một thư mục cho các dự án của bạn và xây dựng dự án với **Tối nghiệp** (quá trình này có thể mất vài phút).

Mẹo: Xem [Cấu hình bản dựng của bạn](#) trang dành cho nhà phát triển để biết thông tin chi tiết.

Bạn cũng có thể thấy thông báo "Mẹo trong ngày" với các phím tắt và các mẹo hữu ích khác. Nhấn vào **Đóng** để đóng tin nhắn.

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

- 1.Nhấp vào **hoạt động_main.xml** tab để xem trình chỉnh sửa bố cục.
- 2.Nhấp vào trình chỉnh sửa bố cục **Thiết kế** tab, nếu chưa được chọn, để hiển thị dạng đồ họa của bố cục như hình dưới đây.



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

Tác phẩm này được cấp phép theo một [Giấy phép Creative Commons Ghi công 4.0 Quốc tế](#). PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

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

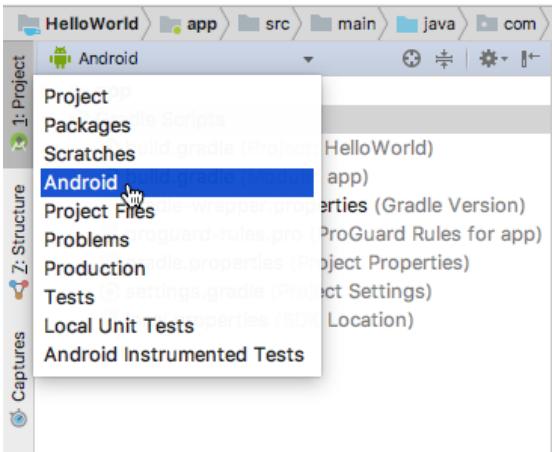
```
1 package com.example.android.helloworld;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6     ...
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_main);
11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

At the bottom of the screen, there are three tabs: 'TODO', 'Event Log' (with a green icon), and 'Gradle Console'.

2.2 Khám phá Dự án > Bảng điều khiển Android

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

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

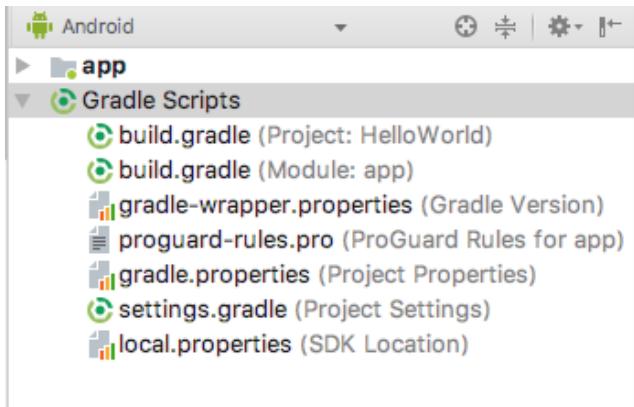


Ghi chú: Chương này và các chương khác đề cập đến ngăn Dự án khi được đặt thành **Android**, như là **Dự án > Android** có.

2.3 Khám phá thư mục Gradle Scripts

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

Khi bạn lần đầu tiên tạo một dự án ứng dụng, **Dự án > Android** ngăn xuất hiện với **Tập lệnh Gradle** thư mục được mở rộng như hiển thị bên dưới.



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

1.Nếu **Tập lệnh Gradle** thư mục không được mở rộng, hãy nhấp vào hình tam giác để mở rộng.

Thư mục này chứa tất cả các tập tin cần thiết cho hệ thống xây dựng.

2.Tìm kiếm **build.gradle**(**Dự án: HelloWorld**) là cái.

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

Theo mặc định, tệp xây dựng cấp cao nhất sử dụng tệp `build.gradle` để xác định kho lưu trữ Gradle và các phụ thuộc chung cho tất cả các mô-đun trong dự án. Khi phụ thuộc của bạn là thứ gì đó khác với thư viện cục bộ hoặc cây tệp, Gradle sẽ tìm kiếm các tệp trong bất kỳ kho lưu trữ trực tuyến nào được chỉ định trong khái kho lưu trữ của tệp này. Theo mặc định, các dự án Android Studio mới khai báo JCenter và Google (bao gồm cả [Kho lưu trữ Google Maven](#)) như các vị trí kho lưu trữ:

```
tất cả các dự án {  
    kho lưu trữ {  
        Google()  
        trung tâm()  
    }  
}
```

3.Tìm kiếm **build.gradle(Mô-đun:Ứng dụng)** là cái.

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

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

Sau đây là **build.gradle(Mô-đun:Ứng dụng)** là cho ứng dụng HelloWord:

```
áp dụng plugin: 'com.android.application'

Android {
    biên dịchSdkPhiên bản 26
    mặc địnhConfig {
        applicationId "com.example.android.helloworld"
        minSdkPhiên bản 15
        targetSdkPhiên bản 26
        Phiên bản Mã 1
        Phiên bản "1.0"
        testInstrumentationRunner
            "android.support.test.runner.AndroidJUnitRunner"
    }
    xây dựng các loại {
        giải phóng {
            minifyEnabled sai
            proguardFiles
                getDefaultProguardFile('proguard-android.txt'),
                'proguard-rules.pro'
        }
    }
}

phụ thuộc {
    fileTree thực hiện(dir: 'libs', include: ['*.jar']) thực hiện
    'com.android.support:appcompat-v7:26.1.0' thực hiện

    'com.android.support.constraint:constraint-layout:1.0.2'
    testImplementation 'junit:junit:4.12'
```

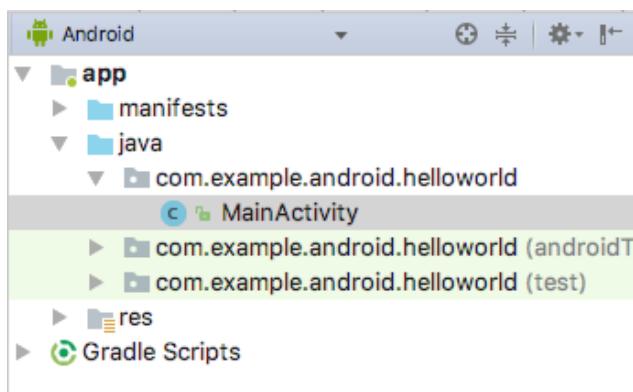
```
    androidTestImplementation 'com.android.support.test:runner:1.0.1'  
    androidTestImplementation  
        'com.android.support.test.espresso:espresso-core:3.0.1'  
}
```

4.Nhấp vào hình tam giác để đóng**Tập lệnh Gradle**.

2.4 Khám phá các thư mục ứng dụng và res

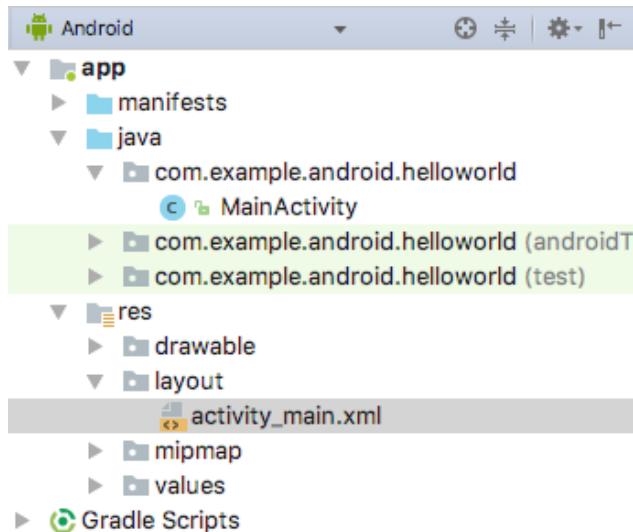
Tất cả mã và tài nguyên cho ứng dụng đều nằm trong ứng dụng và độ phân giải thư mục.

1.Mở rộng **ứng dụng** thư mục, các **java** thư mục và **com.example.android.helloworld** thư mục để xem **Hoạt động chính** tệp java. Nhấp đúp vào tệp để mở tệp đó trong trình soạn thảo mã.



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

2.Mở rộngđộ phân giảithư mục và cách trình bàythư mục và nhấp đúp vào`hoạt động_main.xml`làle để mở nó trong trình soạn thảo bối cục.



Cácđộ phân giảithư mục chứa các tài nguyên, chẳng hạn như bối cục, chuỗi và hình ảnh. MộtHoạt độngthường được liên kết với bối cục của chế độ xem UI được định nghĩa là tệp XML. Tệp này thường được đặt tên theo Hoạt động.

2.5 Khám phá thư mục manifests

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

- 1.Mở rộngbiểu hiệnthư mục.
- 2.Mở`AndroidManifest.xml`làcái.

Các`AndroidManifest.xml`le 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 mỗiHoạt động,phải được khai báo trong tệp XML này. Trong các bài học khác của khóa học, bạn sẽ sửa đổi tệp này để thêm các tính năng và quyền tính năng. Để biết phần giới thiệu, hãy xem[Tổng quan về App Manifest](#).

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 [yêu cầu bổ sung](#) vượt quá các yêu cầu hệ thống cơ bản của Android Studio.

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

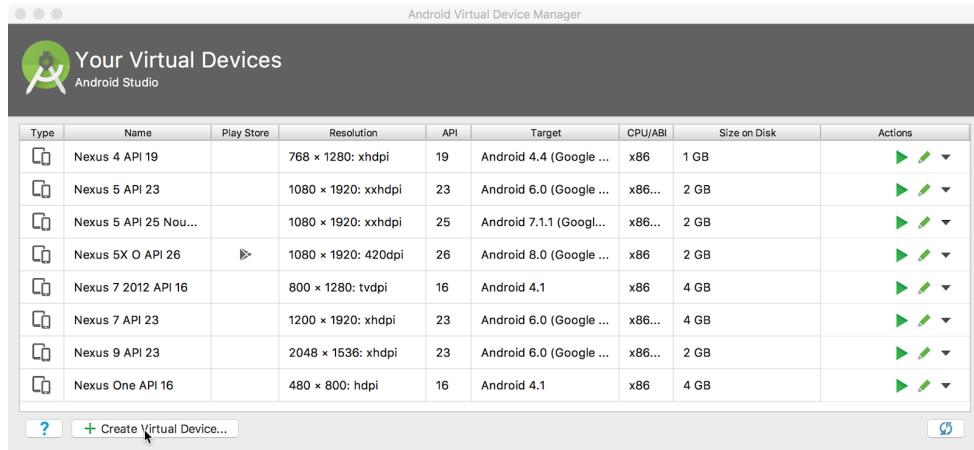
3.1 Tạo thiết bị ảo Android (AVD)

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

1.Trong Android Studio, hãy chọn **Công cụ > Android > Trình quản lý AVD** hoặc nhập vào biểu tượng Trình quản lý AVD

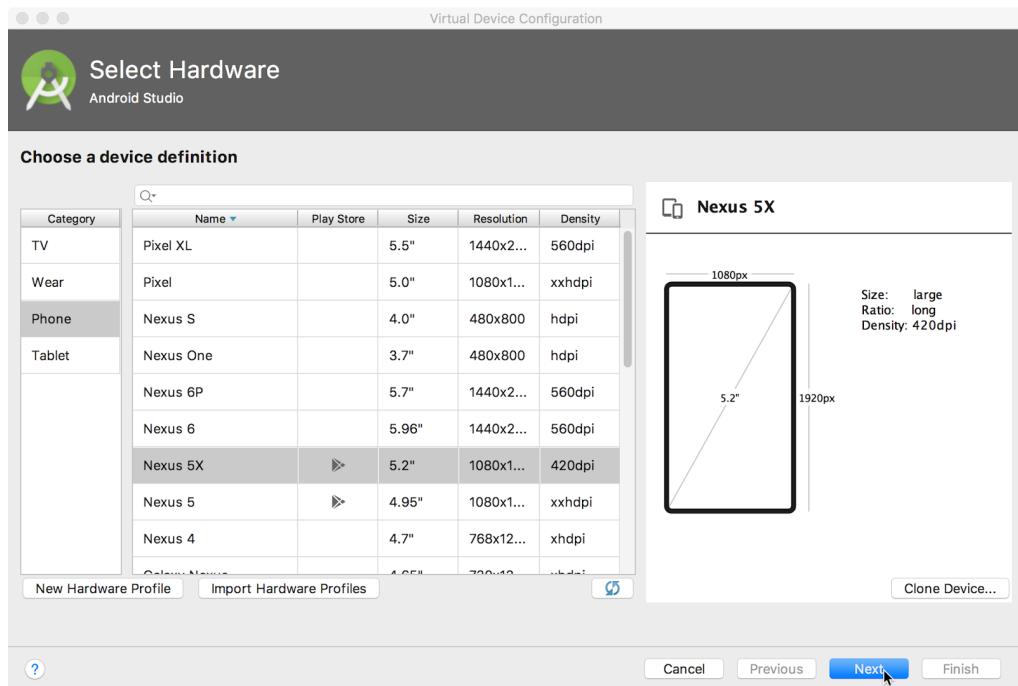


trong thanh công cụ. **Thiết bị ảo của bạn** màn hình xuất hiện. Nếu bạn đã tạo thiết bị ảo, màn hình sẽ hiển thị chúng (như trong hình bên dưới); nếu không, bạn sẽ thấy danh sách trống.



2.Nhấp vào+Tạo thiết bị ảo. Các Chọn phần cứng của sổ xuất hiện hiển thị danh sách

thiết bị phần cứng được cấu hình trước. Đối với mỗi thiết bị, bảng cung cấp một cột cho kích thước hiển thị đường chéo của nó (**Kích cỡ**), độ phân giải màn hình tính bằng pixel (**Nghị quyết**), và mật độ điểm ảnh (**Tỉ trọng**).

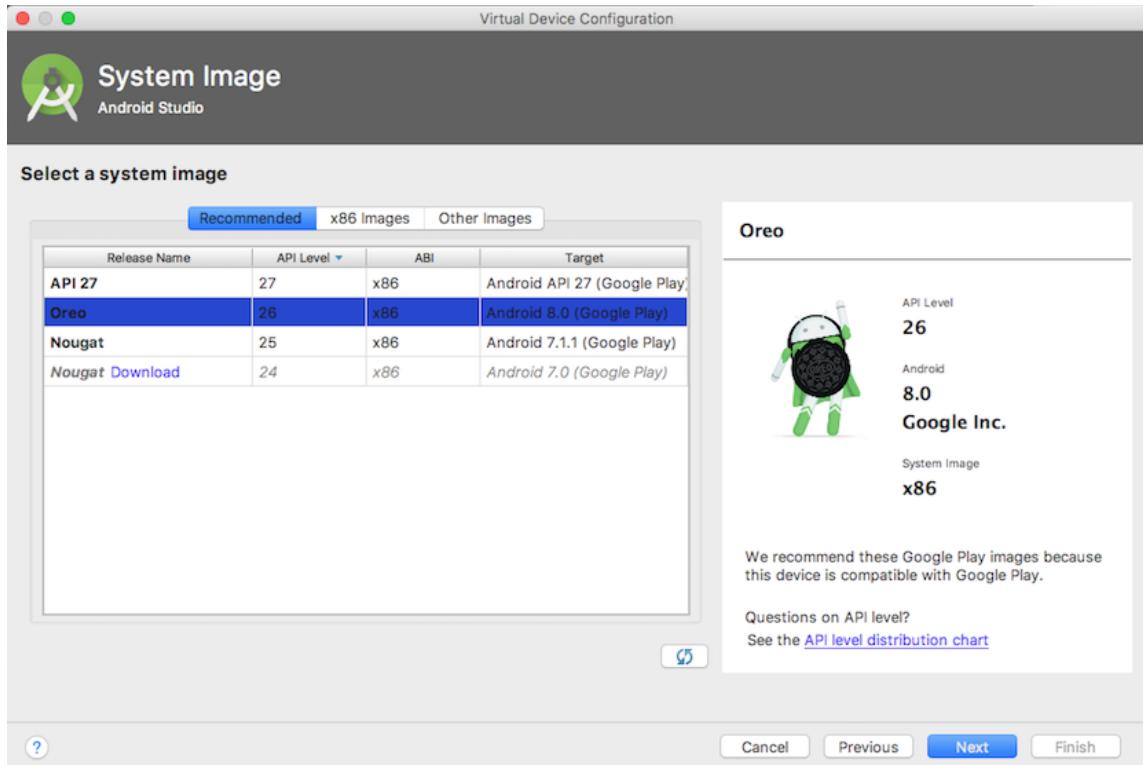


3.Chọn một thiết bị như Điện thoại Nexus 5hoặcĐiểm ảnh XLvà nhấp vàoKế tiếp. Các Hình ảnh hệ thốngmàn hình xuất hiện.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế . PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

4. Nhấp vào **Khuyến khích** tab nếu nó chưa được chọn và chọn phiên bản hệ thống Android nào để chạy trên thiết bị ảo (chẳng hạn như **bánh Oreo**).



Có nhiều phiên bản có sẵn hơn những phiên bản được hiển thị trong **Khuyến khích** tab. Nhìn vào **Hình ảnh x86** và **Hình ảnh khác** tab để xem chúng.

Nếu một **Tài liệu** liên kết có thể nhìn thấy bên cạnh hình ảnh hệ thống bạn muốn sử dụng, nó vẫn chưa được cài đặt. Nhấp vào liên kết để bắt đầu tải xuống và nhấp vào **Hoàn thành** khi nó được thực hiện xong.

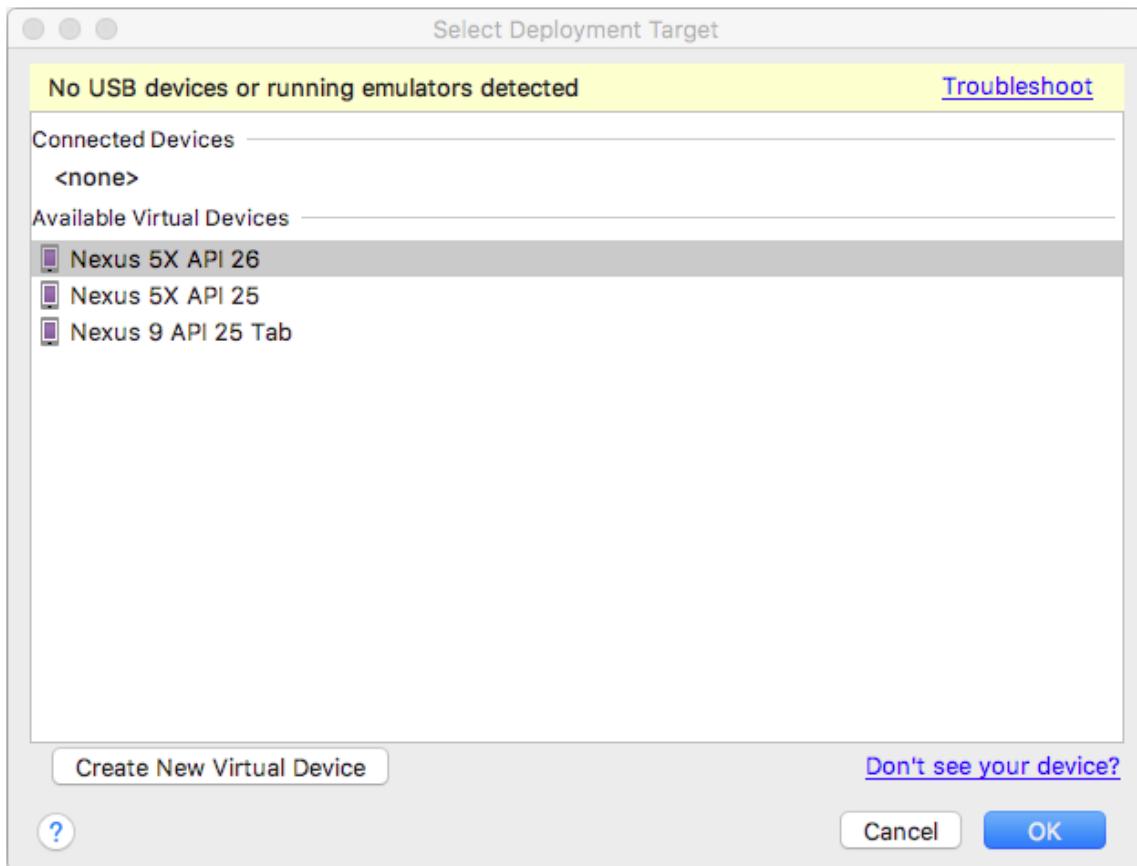
5. Sau khi chọn một hình ảnh hệ thống, hãy nhấp vào **Kế tiếp**. Các **Thiết bị ảo Android (AVD)** cửa sổ xuất hiện. Bạn cũng có thể thay đổi tên của AVD. Kiểm tra cấu hình của bạn và nhấp vào **Hoàn thành**.

3.2 Chạy ứng dụng trên thiết bị ảo

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

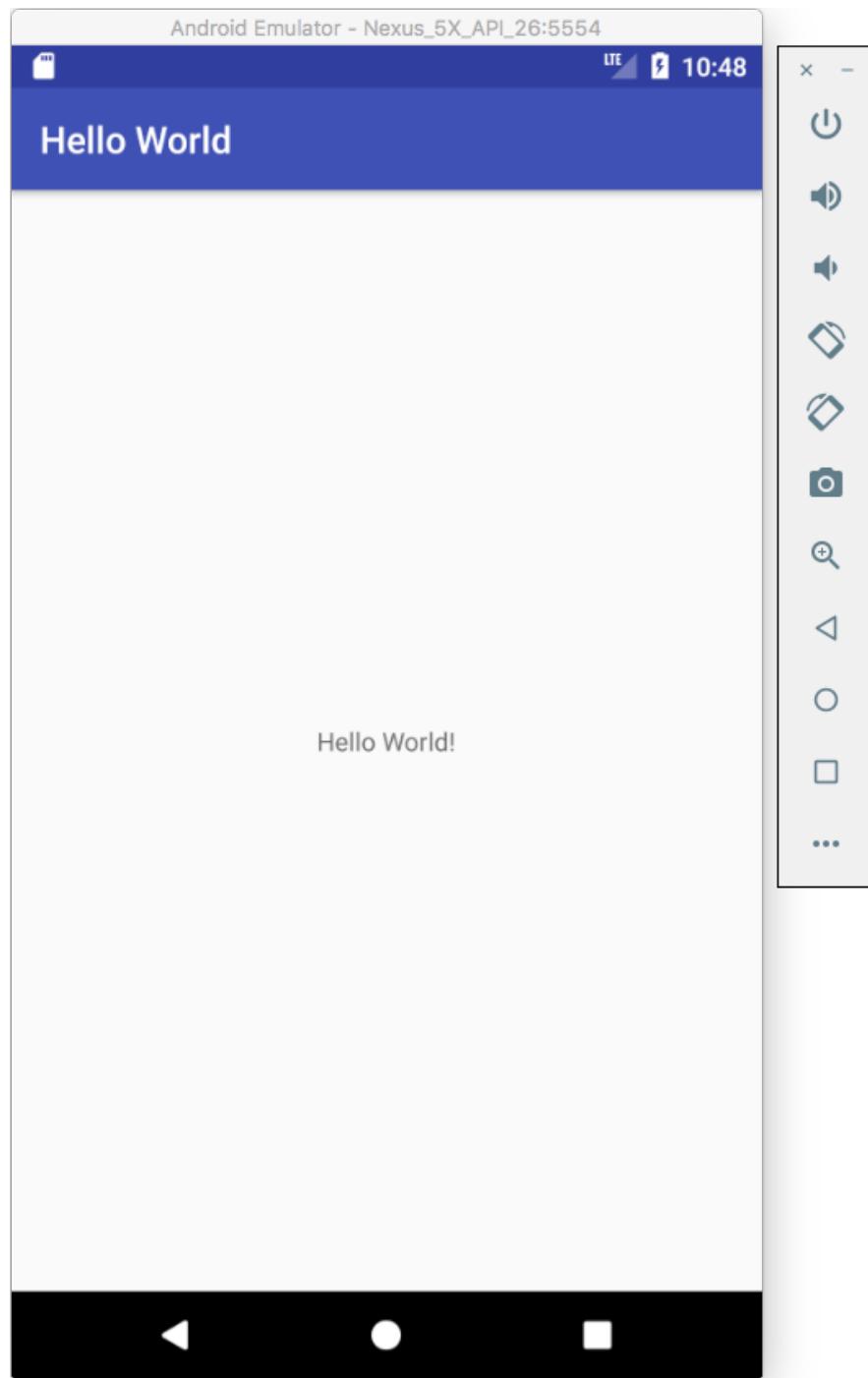
Tác phẩm này được cấp phép theo một [Giấy phép Creative Commons Ghi công 4.0 Quốc tế](#). PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2) để biết thông tin cập nhật mới nhất.

- 1.Trong Android Studio, hãy chọn **Chạy > Chạy ứng dụng** hoặc nhấp vào **Chạy** biểu tượng  trên thanh công cụ.
- 2.Các **Chọn mục tiêu triển khai** cửa sổ, bên dưới **Thiết bị ảo có sẵn**, chọn thiết bị ảo mà bạn vừa tạo và nhấp vào **ĐƯỢC RỒI**.



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

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



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

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

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

Những gì bạn cần:

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

4.1 Bật gỡ lỗi USB

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

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

- 1.Trên thiết bị của bạn, hãy mở **Cài đặt**, tìm kiếm **Về điện thoại**, nhấp vào **Về điện thoại** và chạm **Số bản dựng** bảy lần.
- 2.Trở về màn hình trước đó (**Cài đặt / Hệ thống**). **Tùy chọn nhà phát triển** xuất hiện trong danh sách. Nhấn **Tùy chọn nhà phát triển**.
- 3.Chọn **Gỡ lỗi USB**.

4.2 Chạy ứng dụng của bạn trên thiết bị

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

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

2.Nhấp vào **Chạy** cái nút  trong thanh công cụ. **Chọn mục tiêu triển khai** cửa sổ mở ra với danh sách các trình giả lập và thiết bị được kết nối có sẵn.

3.Chọn thiết bị của bạn và nhấp vào **ĐƯỢC RỒI**.

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

Xử lý sự cố

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

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

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

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

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

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

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

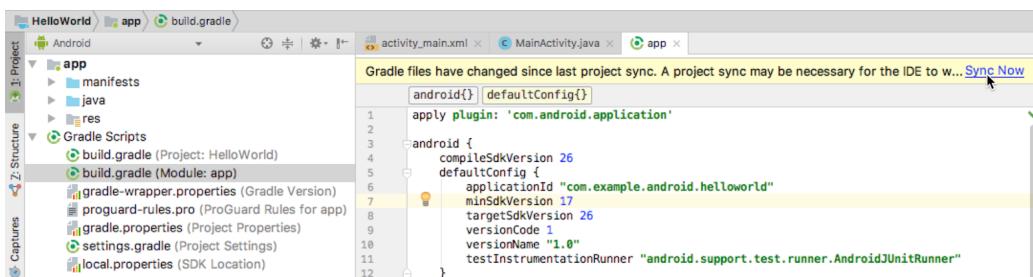
5.1 Thay đổi phiên bản SDK tối thiểu cho ứng dụng

Thực hiện theo các bước sau:

- 1.Mở rộng **Tập lệnh Gradle** thư mục nếu nó chưa được mở và nhấp đúp vào **build.gradle(Mô-đun: ứng dụng)** là cái.

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

- 2.Trong vòng Cấu hình mặc định khôi, thay đổi giá trị của Phiên bản mySdkĐẾN17như được hiển thị bên dưới (ban đầu nó được đặt thành15).



```
Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work... Sync Now
1 android{} defaultConfig{}
2     apply plugin: 'com.android.application'
3
4     android {
5         compileSdkVersion 26
6         defaultConfig {
7             applicationId "com.example.android.helloworld"
8             minSdkVersion 17
9             targetSdkVersion 26
10            versionCode 1
11            versionName "1.0"
12        }
13    }
14
15    testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
```

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

5.2 Đồng bộ cấu hình Gradle mới

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

Để đồng bộ các tập tin dự án, hãy nhấp vào **Đồng bộ ngay** trong thanh thông báo xuất hiện khi thực hiện thay đổi



(như thể hiện trong hình trước đó), hoặc nhấp vào **Đồng bộ dự án với các tập tin Gradle** trong công cụ biểu tượng.

Khi quá trình đồng bộ hóa Gradle hoàn tất, thông báo **Đã hoàn tất việc xây dựng Gradle** xuất hiện ở góc dưới bên trái của cửa sổ Android Studio.

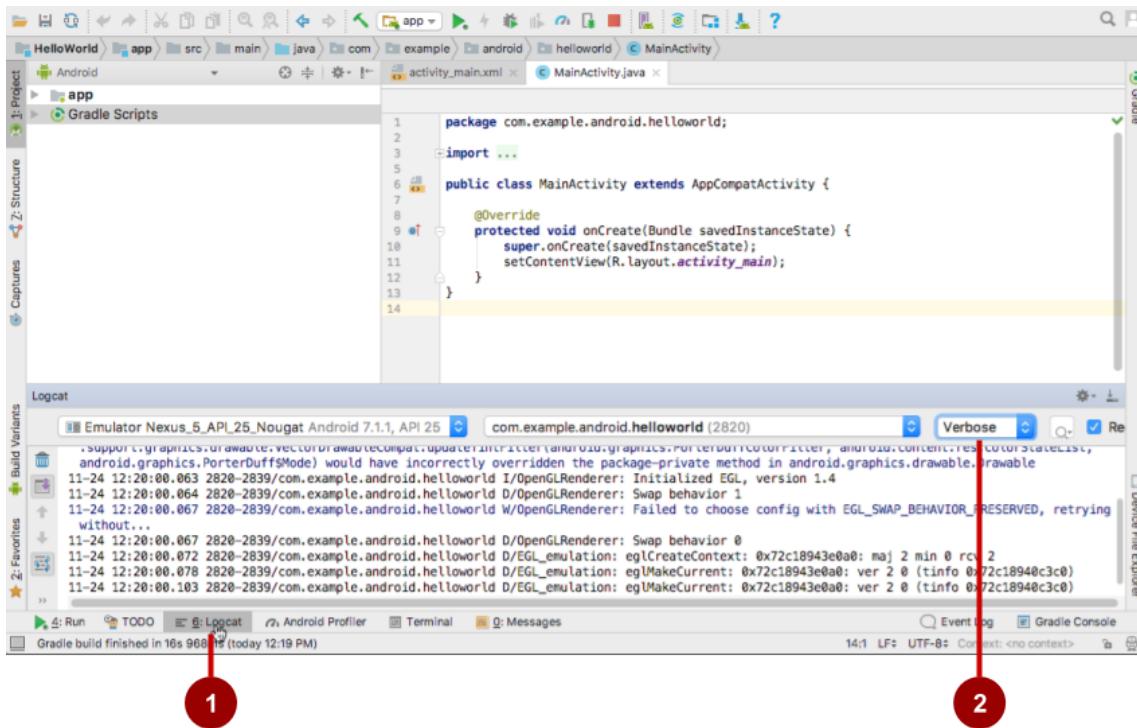
Để có cái nhìn sâu hơn về Gradle, hãy xem [Tổng quan về hệ thống xây dựng](#) và [Cấu hình bản dựng Gradle](#) tài liệu.

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

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

6.1 Xem ngăn Logcat

Để xem **Logcat** ngắn, nhấp vào **Logcat** tab ở cuối cửa sổ Android Studio như thể hiện trong hình bên dưới.



Trong hình trên:

- 1.Các **Logcat** tab để mở và đóng **Logcat** ngắn, hiển thị thông tin về ứng dụng của bạn khi nó đang chạy.
Nếu bạn thêm Nhật ký các câu lệnh cho ứng dụng của bạn, Nhật ký tin nhắn xuất hiện ở đây.
- 2.Các Nhật ký menu cấp độ được thiết lập thành **Dài dòng**(mặc định), hiển thị tất cả Nhật ký tin nhắn. Các thiết lập khác bao gồm **Gỡ lỗi**, **Lỗi**, **Thông tin**, **Và Cảnh báo**.

6.2 Thêm câu lệnh nhật ký vào ứng dụng của bạn

Nhật ký các câu lệnh trong mã ứng dụng của bạn hiển thị thông báo trong ngăn Logcat. Ví dụ:

```
Log.d("MainActivity", "Xin chào thế giới");
```

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

Các phần của tin nhắn bao gồm:

- Nhật ký:Các Nhật ký lớp để gửi tin nhắn nhật ký đến ngăn Logcat.
- ngày:Các Gỡ lỗi Nhật ký thiết lập mức để lọc hiển thị thông báo nhật ký trong ngăn Logcat. Các mức nhật ký khác là Vì lối, Trong và Cảnh báo, Và Tôi và Thông tin.
- "Hoạt động chính":Đối số đầu tiên là một thẻ có thể được sử dụng để lọc tin nhắn trong ngăn Logcat. Đây thường là tên của Hoạt động mà thông điệp bắt nguồn từ đó. Tuy nhiên, bạn có thể biến điều này thành bất cứ điều gì hữu ích cho bạn để gỡ lỗi.

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

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

- "Xin chào thế giới":Lập luận thứ hai chính là thông điệp thực tế.

Thực hiện theo các bước sau:

- 1.Mở ứng dụng Hello World của bạn trong Android studio và mở Hoạt động chính.
- 2.Để tự động thêm các mục nhập rõ ràng vào dự án của bạn (chẳng hạn như `android.util.Log` cần thiết để sử dụng Nhật ký), chọn **Tệp > Cài đặt** trong Windows, hoặc **Android Studio > Tùy chọn** trong macOS.
- 3.Chọn **Biên tập viên > Chung > Tự động nhập**. Chọn tất cả các hộp kiểm và thiết lập **Chèn nhập khẩu vào dán** **ĐẾN** **Tất cả**.
- 4.Nhấp chuột **Áp dụng** và sau đó nhấp vào **ĐƯỢC RỒI**.
- 5.Trong khi tạo() phương pháp của Hoạt động chính, thêm câu lệnh sau:

```
Log.d("MainActivity", "Xin chào thế giới");
```

Cách khi tạo() phương pháp này bây giờ sẽ trông giống như đoạn mã sau:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);
```

```
        Log.d("MainActivity", "Xin chào thế giới");  
    }
```

6.Nếu ngăn Logcat chưa mở, hãy nhấp vào **Logcat** tab ở cuối Android Studio để mở nó.

7.Kiểm tra xem tên mục tiêu và tên gói của ứng dụng đã chính xác chưa.

8.Thay đổi Nhật ký mức độ trong **Logcat** cửa sổ để **Gỡ lỗi** (hoặc để lại như **Dài dòng** vì có rất ít tin nhắn nhật ký).

9.Chạy ứng dụng của bạn.

Thông báo sau sẽ xuất hiện trong ngăn Logcat:

```
11-24 14:06:59.001 4696-4696/? D/Hoạt động chính: Xin chào thế giới
```

Thử thách mã hóa

Ghi chú:Mọi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

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

- 1.Tạo một dự án mới trong Android Studio.
- 2.Đổi lời chào "Hello World" thành "Chúc mừng sinh nhật" và tên của một người nào đó có ngày sinh nhật gần đây.
- 3.((Tùy chọn) Chụp ảnh màn hình ứng dụng đã hoàn thành và gửi email cho người mà bạn quên mất ngày sinh nhật.
- 4.Một cách sử dụng phổ biến của Nhật ký lớp học là để ghi nhật ký Ngoại lệ Java khi chúng xảy ra trong chương trình của bạn. Có một số phương pháp hữu ích, chẳng hạn như Nhật ký.e(), mà bạn có thể sử dụng cho mục đích này. Khám phá

phương pháp bạn có thể sử dụng để bao gồm một ngoại lệ với Nhật ký tin nhắn. Sau đó, viết mã trong ứng dụng của bạn để kích hoạt và ghi lại ngoại lệ.

Bản tóm tắt

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

Các khái niệm liên quan

Tài liệu khái niệm liên quan có trong [1.0: Giới thiệu về Android](#) và [1.1: Ứng dụng Android đầu tiên của bạn](#).

Tìm hiểu thêm

Tài liệu Android Studio:

- [Trang tải xuống Android Studio](#)
- [Ghi chú phát hành Android Studio](#)
- [Làm quen với Android Studio](#)
- [Công cụ dòng lệnh Logcat](#)
- [Trình quản lý thiết bị ảo Android \(AVD\)](#)
- [Tổng quan về App Manifest](#)
- [Cấu hình bản dựng của bạn](#)
- [Nhật ký lớp học](#)
- [Tạo và quản lý thiết bị ảo](#)

Khác:

- [Làm thế nào để cài đặt Java?](#)
- [Cài đặt phần mềm JDK và thiết lập JAVA_HOME](#)
- [Trang web Gradle](#)
- [Cú pháp Apache Groovy](#)
- [Trang Wikipedia Gradle](#)

Bài tập về nhà

Xây dựng và chạy một ứng dụng

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

Trả lời những câu hỏi này

Câu hỏi 1

Tên của tệp bố cục cho hoạt động chính là gì?

- MainActivity.java
- AndroidManifest.xml
- hoạt động_main.xml
- xây dựng.gradle

Câu hỏi 2

Tên của chuỗi tài nguyên chỉ định tên ứng dụng là gì?

- tên_ứng_dụng
- xmlns:ứng dụng
- android:tên
- Ứng dụngId

Câu hỏi 3

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

- Giám sát thiết bị Android
- Quản lý AVD
- Trình quản lý SDK
- Biên tập chủ đề

Câu hỏi 4

Giả sử ứng dụng của bạn bao gồm câu lệnh ghi nhật ký này:

```
Log.i("MainActivity", "Bố cục MainActivity đã hoàn tất");
```

Bạn thấy câu lệnh "Bố cục MainActivity đã hoàn tất" trong **Logcat** nếu menu Cấp độ nhật ký được đặt thành tùy chọn nào sau đây? (Gợi ý: có thể trả lời nhiều câu hỏi.)

- Dài dòng
- Gỡ lỗi
- Thông tin
- Cảnh báo
- Lỗi
- khẳng định

Gửi ứng dụng của bạn để chấm điểm

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

- MỘTHoạt động hiển thị "Hello World" trên màn hình.
- Nhật ký các câu lệnh trong khi tạo() trong hoạt động chính.
- Mức độ ghi nhật ký trong **Logcat** chỉ hiển thị các câu lệnh ghi nhật ký gỡ lỗi hoặc lỗi.

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

Giới thiệu

Giao diện người dùng (UI) xuất hiện trên màn hình của thiết bị Android bao gồm một hệ thống phân cấp các đối tượng được gọi là *lượt xem*—mỗi thành phần của màn hình là một Xem. Các Xem lớp biểu thị khái niệm 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. Thường được sử dụng Xem các lớp con được mô tả trong nhiều bài học bao gồm:

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

- [Xem văn bản](#) để hiển thị văn bản.
- [EditText](#) cho phép người dùng nhập và chỉnh sửa văn bản.
- [Cái nút](#) và các yếu tố có thể nhấp khác (chẳng hạn như[RadioButton](#), [Hộp kiểm](#), Và[Máy quay](#)) để cung cấp hành vi tương tác.
- [CuộnXem](#) Và[RecyclerView](#) để hiển thị các mục có thể cuộn.
- [Hình ảnhXem](#) để hiển thị hình ảnh.
- [Bố cục ràng buộc](#) Và[Bố cục tuyến tính](#) để chứa đựng những thứ khácXemcác yếu tố và định vị chúng.

Mã Java hiển thị và điều khiển UI được chứa trong một lớp mở rộng[Hoạt động](#). MỘT Hoạt động thường được liên kết với bố cục của chế độ xem UI được định nghĩa là tệp XML (Ngôn ngữ đánh dấu mở rộng). Tệp XML này thường được đặt tên theo[Hoạt động](#)và xác định bố cục củaXem các thành phần trên màn hình.

Ví dụ, [Hoạt động chính](#) trong ứng dụng Hello World hiển thị một bố cục được xác định trong [hoat_dong_main.xml](#) tệp tin bố trí, bao gồm một Xem văn bảnvới nội dung "Xin chào thế giới".

Trong các ứng dụng phức tạp hơn, một [Hoạt động](#) 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 tìm hiểu thêm về [Hoạt động](#) lớp học ở bài học khác.

Trong phần 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 ứng dụng bằng mẫu Empty Activity. 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 phần thực hành khác trong khóa học này.

Những điều bạn nên biết

Bạn nên biết về:

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

Những gì bạn sẽ học được

- Cách tạo ứng dụng có tính tương tác.
- Cách sử dụng trình chỉnh sửa bố cục để thiết kế bố cục.
- Cách chỉnh sửa bố cục trong XML.
- Rất nhiều thuật ngữ mới. Kiểm tra [Từ vựng và khái niệm thuật ngữ](#) để có định nghĩa thân thiện.

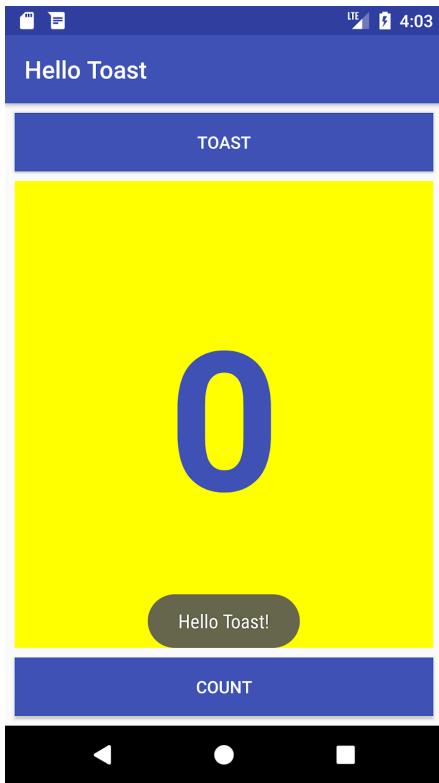
Bạn sẽ làm gì

- Tạo một ứng dụng và thêm hai nút các yếu tố và một Xem văn bản vào bố cục.
- Thao tác từng phần tử trong [Bố cục ràng buộc](#) để giới hạn chúng ở lề và các yếu tố khác.
- Thay đổi thuộc tính của phần tử UI.
- Chỉnh sửa bố cục của ứng dụng trong XML.
- Trích xuất các chuỗi được mã hóa cứng thành các 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 từng phương thức Cái nút.

Tổng quan về ứng dụng

Ứng dụng HelloToast bao gồm hai nút các yếu tố và một Chế độ xem văn bản. Khi người dùng chạm vào đầu tiên Cái nút, nó hiển thị một tin nhắn ngắn (a [Nút](#)) trên màn hình. Chạm vào thứ hai Cái nút tăng bộ đếm "nhấp chuột" được hiển thị trong Chế độ xem văn bản, bắt đầu từ số không.

Sau đây là hình ảnh ứng dụng đã hoàn thành:



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

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

1.1 Tạo dự án Android Studio

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

Thuộc tính	Giá trị

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

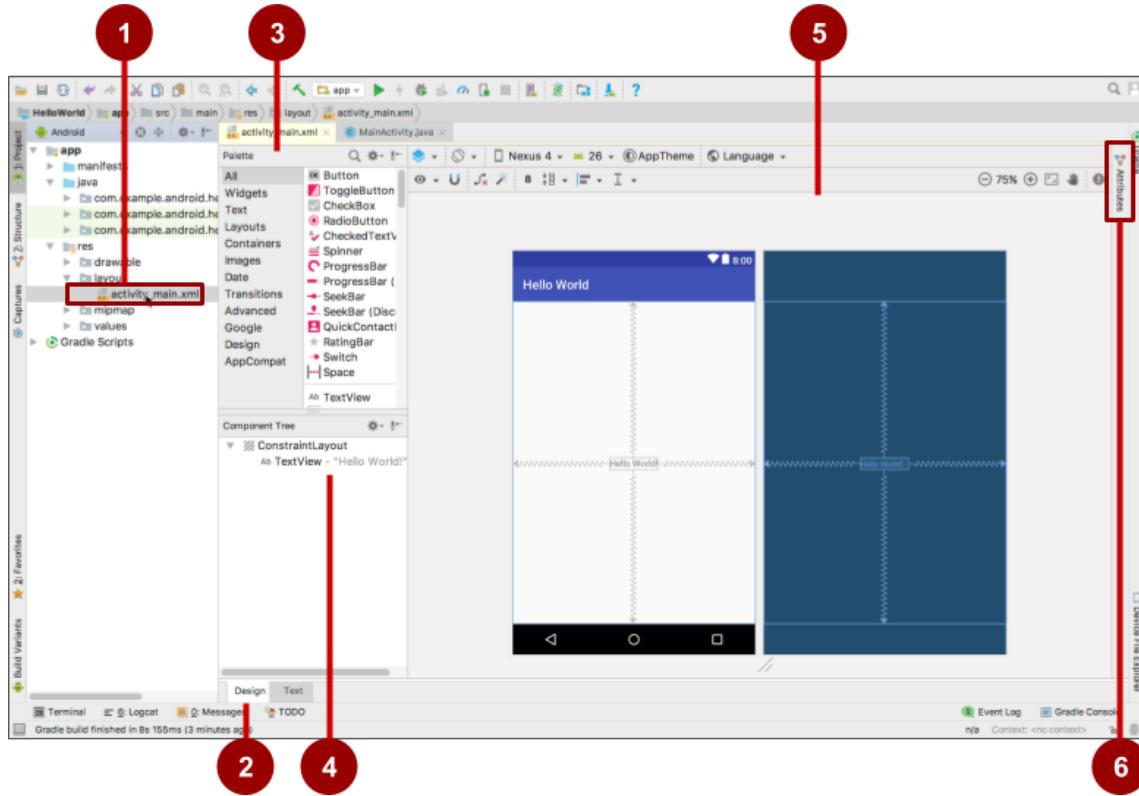
Tên Ứng dụng	Xin chào bánh mì nướng
Tên công ty	com.example.android (hoặc tên miền của riêng bạn)
SDK tối thiểu cho điện thoại và máy tính bảng	API15: Android 4.0.3 IceCreamSandwich
Bản mẫu	Hoạt động trống
Tạo hộp tập tin Bố cục	Đã chọn
Hộp tương thích ngược	Đã chọn

15. Lựa chọn **Chạy > Chạy ứng dụng** hoặc nhấp vào **Biểu tượng chạy**  trên thanh công cụ để xây dựng và thực thi ứng dụng

1.2 Khám phá trình chỉnh sửa bố cục

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

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



- 1.Trong **Ứng dụng > res > bối cục** thư mục trong **Dự án > Android** ngǎn, nhấp đúp vào **activity_main.xml** để mở nó nếu nó chưa được mở.
- 2.Nhấp vào **Thiết kế** tab nếu nó chưa được chọn. Bạn sử dụng **Thiết kế** tab để thao tác các thành phần và bối cục, và **Chữ** tab để chỉnh sửa mã XML cho bối cục.
- 3.Các **Bảng màu** ngǎn hiển thị các thành phần UI mà bạn có thể sử dụng trong bối cục ứng dụng của mình.
- 4.Các **Cây thành phần** ngǎn hiển thị hệ thống phân cấp chế độ xem của các thành phần UI.Xem các yếu tố là được tổ chức thành một hệ thống phân cấp cây gồm cha mẹ và con cái, trong đó con cái thừa hưởng các thuộc tính của cha mẹ. Trong hình trên,Xem văn bản là một đứa con của **Bối cục ràng buộc**. Bạn sẽ tìm hiểu về các yếu tố này ở phần sau của bài học này.
- 5.Các ngăn thiết kế và bảng thiết kế của trình chỉnh sửa bối cục hiển thị các thành phần UI trong bối cục. Trong hình trên, bối cục chỉ hiển thị một thành phần: **TextView** hiển thị "Hello World".
- 6.Các **Thuộc tính** tab hiển thị **Thuộc tính** ngǎn để thiết lập thuộc tính cho một thành phần UI.

Mẹo: Nhìn thấy [Xây dựng giao diện người dùng với Layout Editor](#) để biết chi tiết về cách sử dụng trình chỉnh sửa bối cục và [Làm quen với Android Studio](#) để biết tài liệu đầy đủ về Android Studio.

Tác phẩm này được cấp phép theo một [Giấy phép Creative Commons Ghi công 4.0 Quốc tế](#). PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://www.dev.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

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

Trong tác vụ này, bạn tạo bố cục UI cho ứng dụng HelloToast trong trình chỉnh sửa bố cục bằng cách sử dụng [Bố cục ràng buộc](#) tính năng. Bạn có thể tạo các ràng buộc theo cách thủ công, như được hiển thị sau, hoặc tự động bằng cách sử dụng [Tự động kết nối](#) dung cụ.

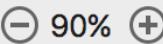
2.1 Kiểm tra các ràng buộc phần tử

Thực hiện theo các bước sau:

- 1.Mở [hoạt động_main.xml](#) từ **Dự án > Android** nếu nó chưa được mở. Nếu **Thiết kế** tab chưa được chọn, hãy nhấp vào tab đó.

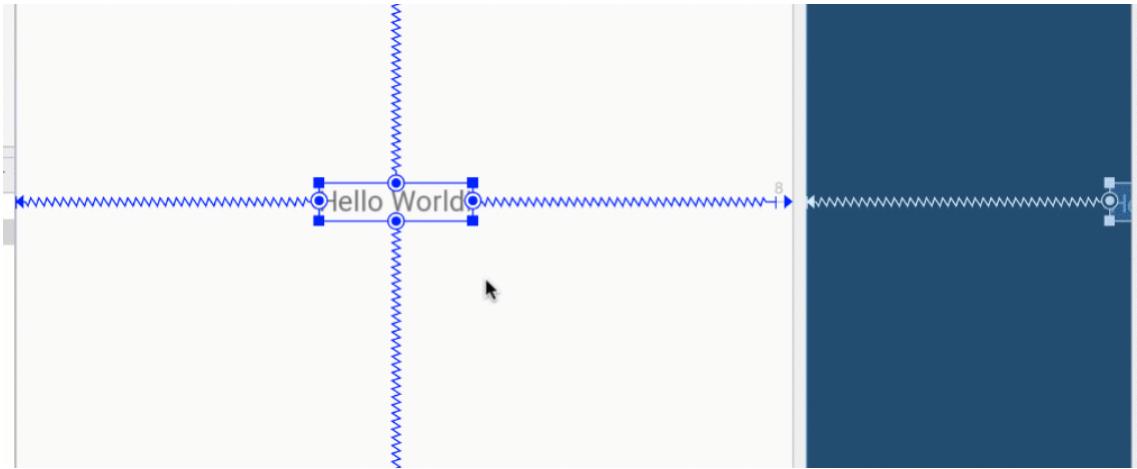
Nếu không có bản thiết kế, hãy nhấp vào **Chọn bề mặt thiết kế** nút  trong thanh công cụ và chọn **Thiết kế + Bản thiết kế**.

- 2.Các **Tự động kết nối** dung cụ  cũng nằm trong thanh công cụ. Nó được bật theo mặc định. Đối với điều này Bước này, đảm bảo rằng công cụ không bị vô hiệu hóa.

- 3.Nhấp vào nút phóng to    nút để phóng to vào các ngăn thiết kế và bản thiết kế cho để xem cận cảnh.

- 4.Lựa chọn **Xem văn bản** trong ngăn Cây thành phần. "Hello World" Xem văn bản được đánh dấu trong khung thiết kế và bản thiết kế và các ràng buộc cho phần tử có thể nhìn thấy được.

- 5.Tham khảo hình ảnh động bên dưới cho bước này. Nhấp vào tay cầm tròn ở bên phải của Xem văn bản để xóa ràng buộc theo chiều ngang liên kết chế độ xem với phía bên phải của bố cục. Xem văn bản này sang bên trái vì nó không còn bị giới hạn ở bên phải nữa. Để thêm lại giới hạn ngang, hãy nhấp vào cùng một tay cầm và kéo một đường sang bên phải của bố cục.



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

- **Xử lý ràng buộc:** Để tạo ràng buộc như trong hình động ở trên, hãy nhấp vào một tay cầm ràng buộc, được hiển thị dưới dạng một vòng tròn ở bên cạnh một phần tử. Sau đó kéo tay cầm đến một tay cầm ràng buộc khác hoặc đến ranh giới cha. Một đường ngoằn ngoèo biểu thị ràng buộc.



- **Thay đổi kích thước tay cầm:** Để thay đổi kích thước phần tử, hãy kéo các tay cầm thay đổi kích thước hình vuông. Tay cầm sẽ thay đổi thành góc nghiêng khi bạn kéo nó.

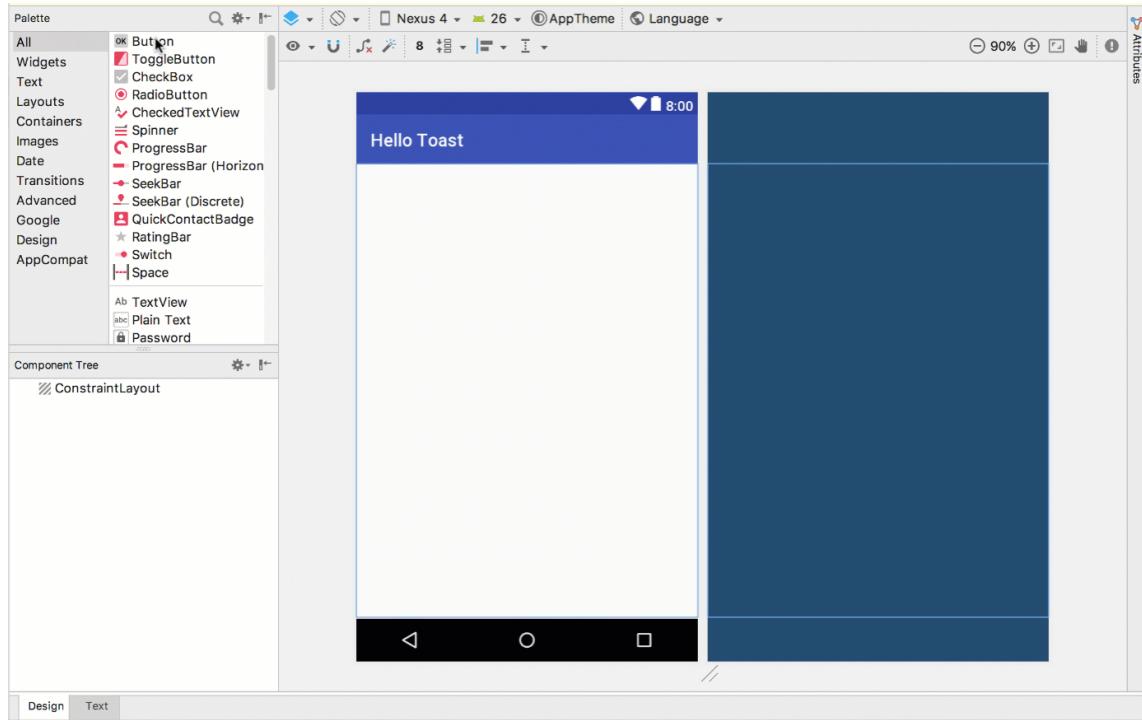


2.2 Thêm nút vào bố cục

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

Thực hiện theo các bước sau để thêm Cái nút:

1. Bắt đầu với một bảng sạch. Xem văn bản phần tử không cần thiết, vì vậy trong khi nó vẫn được chọn, hãy nhấn **Xóa** bỏ chìa khóa hoặc chọn **Sửa > Xóa**. Jetzt bạn có một bố cục hoàn toàn trống.
2. Kéo một **Cái nút** từ **Bảng màu** gần đến bất kỳ vị trí nào trong bố cục. Nếu bạn thả Cái nút ở vùng 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 cùng, bên trái và bên phải của bố cục như minh họa trong hình động bên dưới.



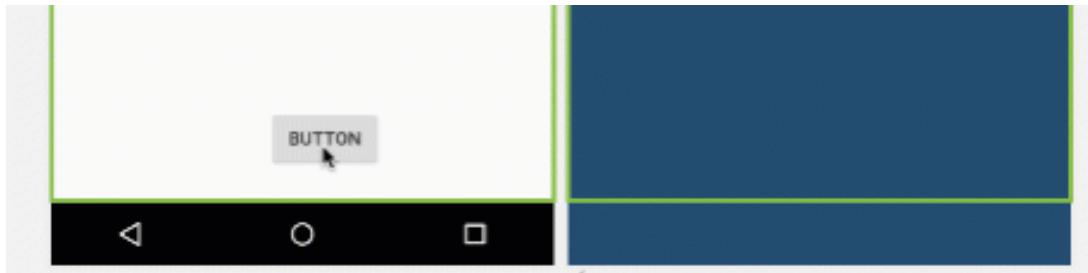
Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

2.3 Thêm Nút thứ hai vào bố cục

1.Kéo một cái khác **Cái nút** từ **Bảng máugắn** vào giữa bố cục như được hiển thị trong hình động bên dưới. Autoconnect có thể cung cấp các ràng buộc theo chiều ngang cho bạn (nếu không, bạn có thể tự kéo chúng).

2.Kéo ràng buộc theo chiều dọc xuống dưới cùng của bố cục (tham khảo hình bên dưới).



Bạn có thể xóa các ràng buộc khỏi một phần tử bằng cách chọn phần tử đó và di con trỏ của bạn

trên đó để hiển thị các ràng buộc rõ ràng nút. Nhấp vào nút này để **xóa tất cả** những hạn chế về phần tử đã chọn. Để xóa một ràng buộc duy nhất, hãy nhấp vào trình xử lý cụ thể đặt ràng buộc đó.

Để xóa tất cả các ràng buộc trong toàn bộ bố cục, hãy nhấp vào **Xóa tất cả ràng buộc** công cụ 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 thuộc tính của phần tử UI

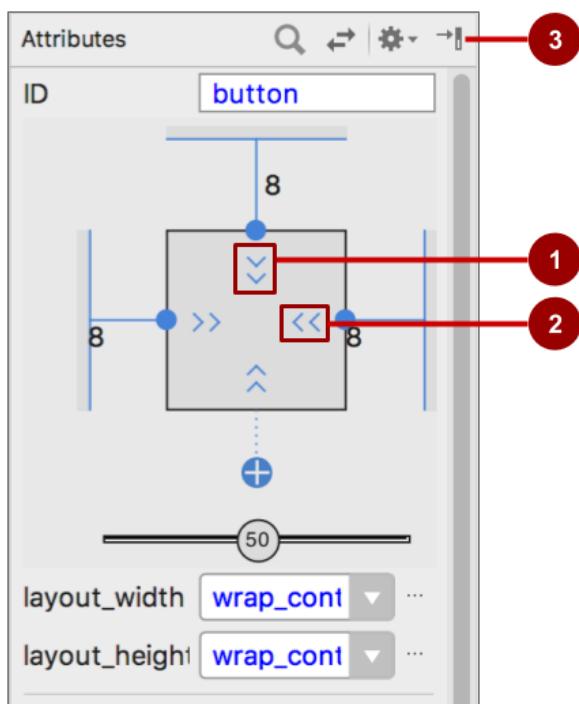
Các **Thuộc tính** cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một phần tử UI. Bạn có thể tìm thấy các thuộc tính (được gọi là *của cả*) chung cho tất cả các quan điểm trong [Xem tài liệu lớp học](#).

Trong nhiệm vụ này, bạn nhập các giá trị mới và thay đổi các giá trị quan trọng Cái nút các thuộc tính, có thể áp dụng cho hầu hết Xem các loại.

3.1 Thay đổi kích thước nút

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

Thay vào đó, hãy sử dụng **Thuộc tính** ngay bên phải của trình chỉnh sửa bối cục để chọn chế độ định cỡ không sử dụng kích thước được mã hóa cứng. **Thuộc tính** bảng điều khiển bao gồm một bảng điều khiển kích thước hình vuông được gọi là *xem thanh tra* ở trên cùng. Các ký hiệu bên trong hình vuông biểu thị các thiết lập chiều cao và chiều rộng như sau:



Trong hình trên:

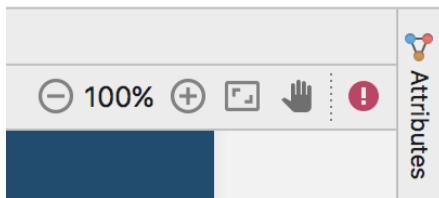
1. **Kiểm soát chiều cao.** Kiểm soát này chỉ định chiều cao bối cục thuộc tính và xuất hiện trong hai các đoạn ở phía trên và phía dưới của hình vuông. Các góc cho biết rằng điều khiển này được đặt thành bối_nội_dung, có nghĩa là Xem sẽ mở rộng theo chiều dọc khi cần để vừa với nội dung của nó. "8" biểu thị lề chuẩn được đặt thành 8dp.
2. **Kiểm soát chiều rộng.** Kiểm soát này chỉ định chiều rộng bối cục và xuất hiện ở hai phân đoạn bên trái và bên phải của hình vuông. Các góc cho biết rằng điều khiển này được đặt thành bối_nội_dung,

có nghĩa là Xemsẽ mở rộng theo chiều ngang khi cần thiết để vừa với nội dung của nó, lên đến biên độ 8dp.

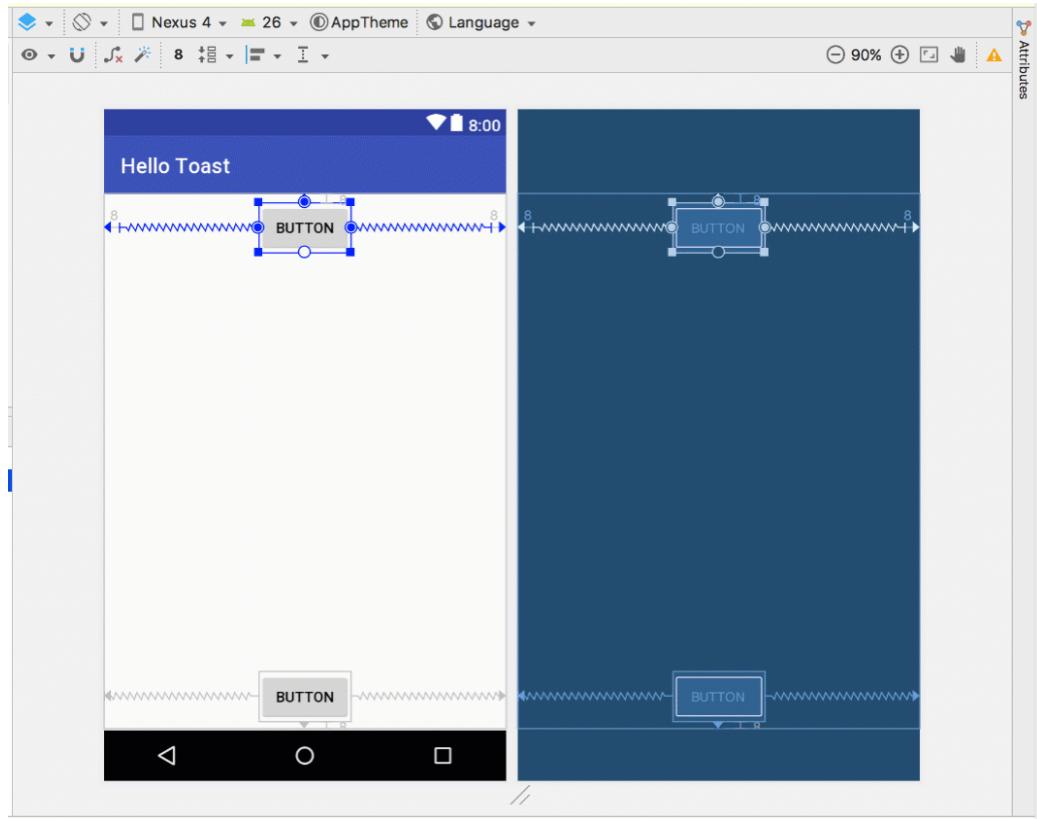
3.Thuộc tínhnút đóng ngắn. Nhấp đê đóng ngắn.

Thực hiện theo các bước sau:

- 1.Chọn đầu trangCái núttrong**Cây thành phần**có.
- 2.Nhấp vào**Thuộc tính**tab ở bên phải cửa sổ trình chỉnh sửa bối cục.

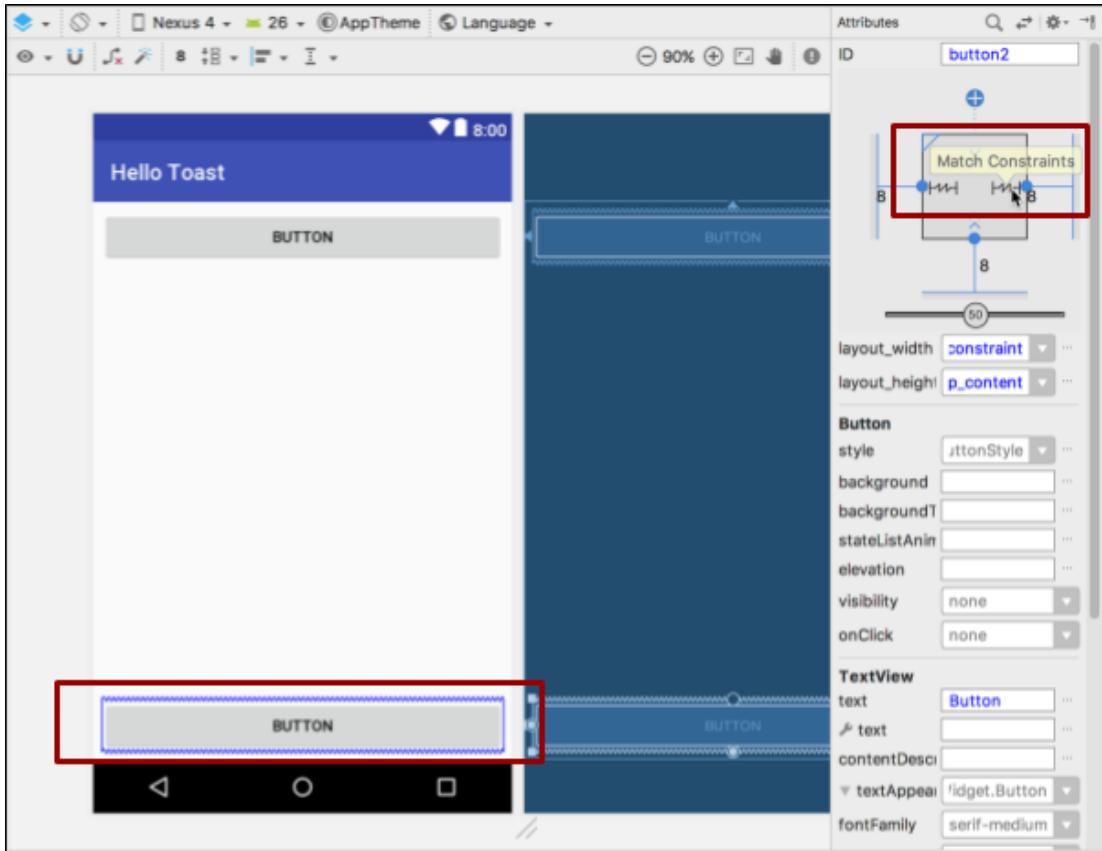


- 3.Nhấp vào điều khiển chiều rộng hai lần—lần nhấp đầu tiên sẽ thay đổi nó thành**Đã sử dụng**với các đường thẳng và nhấp chuột thứ hai thay đổi nó thành**Ràng buộc phù hợp**với các cuộn lò xo, như thể hiện trong hình ảnh động bên dưới.



Kết quả của việc thay đổi điều khiển chiều rộng, chiều rộng bối cảnh thuộc tính trong **Thuộc tính** hiển thị giá trị ràng buộc và phù hợp với nút phím tắt kéo dài theo chiều ngang để lấp đầy khoảng trống giữa bên trái và bên phải của bối cảnh.

- Chọn thứ hai Cái nút, và thực hiện những thay đổi tương tự đối với chiều rộng bối cảnh như trong bước trước, như thể hiện trong hình bên dưới.



Như đã trình bày ở các bước trước, chiều rộng bố cục và chiều cao bố cục các thuộc tính trong **Thuộc tính** thay đổi ngay khi bạn thay đổi các điều khiển chiều cao và chiều rộng trong thanh tra. Các thuộc tính này có thể lấy một trong ba giá trị cho bố cục, đó là **Bố cục ràng buộc**:

- Cá ràng buộc_phù hợp thiết lập mở rộng Xem phần tử để lấp đầy phần tử cha của nó theo chiều rộng hoặc chiều cao—lên đến một lề, nếu có một lề được thiết lập. Cha mẹ trong trường hợp này là **Bố cục ràng buộc**. Bạn tìm hiểu thêm về **Bố cục ràng buộc** trong nhiệm vụ tiếp theo.
- Các nội dung bao gồm thiết lập thu nhỏ Xem kích thước của phần tử sao cho nó vừa đủ lớn để bao gồm nội dung của nó. Nếu không có nội dung, Xem phần tử trở nên vô hình.
- Để chỉ định một kích thước cố định điều chỉnh theo kích thước màn hình của thiết bị, hãy sử dụng một số lượng cố định pixel không phụ thuộc vào mật độ (dpđơn vị). Ví dụ, 16dp có nghĩa là 16 pixel không phụ thuộc vào mật độ.

Mẹo: Nếu bạn thay đổi chiều rộng bố cục thuộc tính sử dụng menu bật lên của nó, chiều rộng bố cục thuộc tính được đặt thành số không vì không có kích thước nào được đặt. Thiết lập này giống như ràng buộc _khớp_hợp—chế độ xem có thể mở rộng tối đa để đáp ứng các ràng buộc và thiết lập lề.

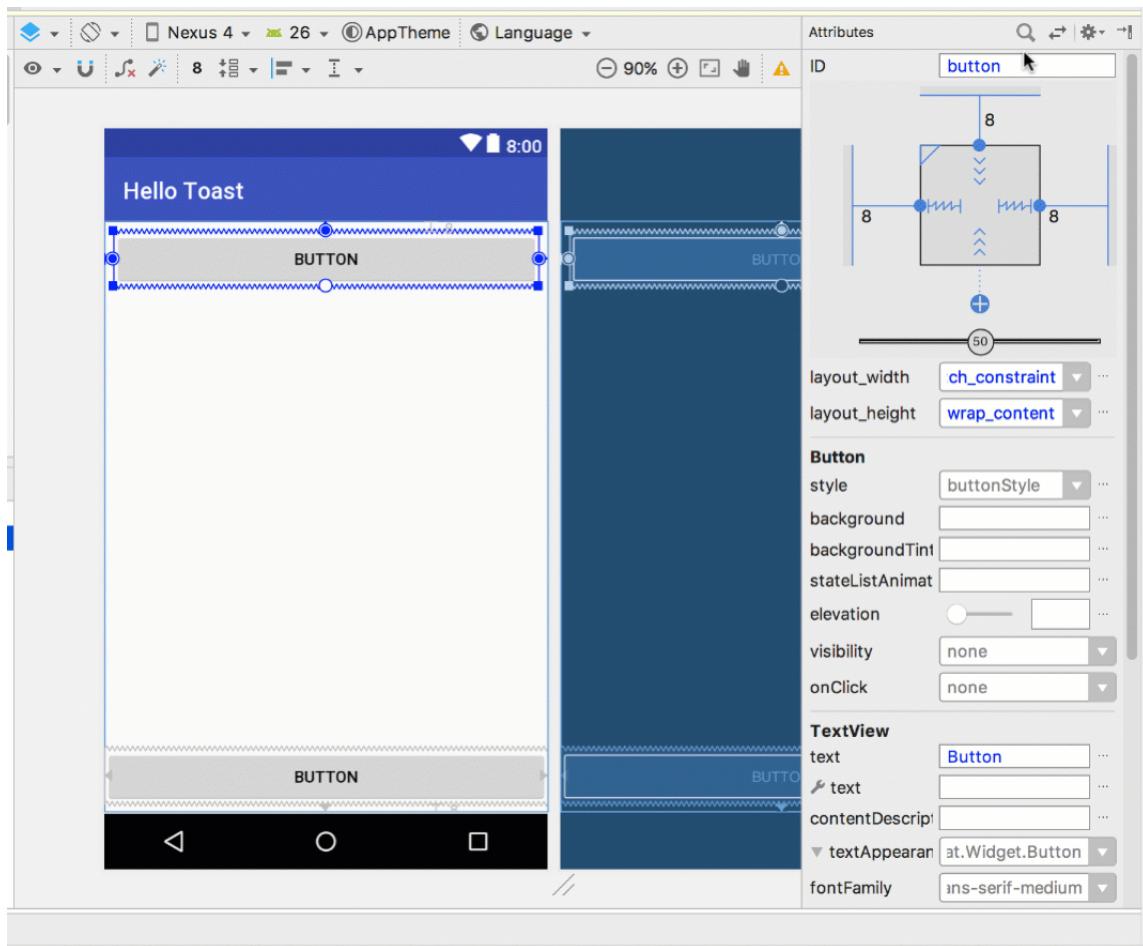
3.2 Thay đổi thuộc tính của Button

Để xác định từng Xem duy nhất trong một Hoạt động bố trí, mỗi Xem hoặc Xem phân lớp (chẳng hạn như Cái nút) cần một ID duy nhất. Và để có thể sử dụng, Cái nút các thành phần cần có văn bản Xem. Các thành phần cũng có thể có nền có thể là màu sắc hoặc hình ảnh.

Các **Thuộc tính** ngang cung cấp quyền truy cập vào tất cả các thuộc tính bạn có thể gán cho Xem phân tử. Bạn có thể nhập giá trị cho mỗi thuộc tính, chẳng hạn như android:id, nền, màu chữ, và chữ thuộc tính.

Hình ảnh hoạt hình sau đây minh họa cách thực hiện các bước này:

- 1.Sau khi chọn đầu tiên Cái nút, chỉnh sửa ID là **cánh đồng** ở phía trên cùng của **Thuộc tính** cửa sổ để **nút_bánh** ... cho `android:id` thuộc tính được sử dụng để xác định phân tử trong bố cục.
- 2.Đặt lý lịch thuộc tính cho **@màu sắc/màu sắc Chính**. (Khi bạn nhập `@c`, các lựa chọn xuất hiện để lựa chọn dễ dàng.)
- 3.Đặt Màu văn bản thuộc tính cho **@android:màu/trắng**.
- 4.Chỉnh sửa chữ thuộc tính cho **Nướng**.



5.Thực hiện các thay đổi thuộc tính tương tự cho lần thứ haiCái nút,sử dụngnút_số_lượngnhư ID, **Đếm**chochữthuộc tính và màu nền và màu văn bản giống như các bước trước.

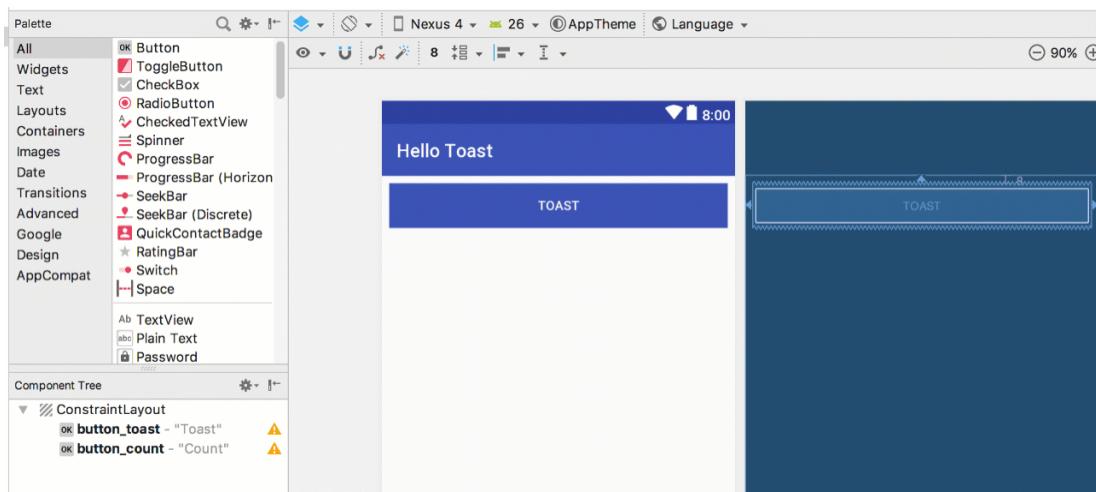
Cácmàuchính là màu chính của chủ đề, một trong những màu cơ bản của chủ đề được xác định trước trongmàu sắc.xmltệp tài nguyên. Nó được sử dụng cho thanh ứng dụng. Sử dụng màu cơ bản cho các thành phần UI khác sẽ tạo ra một UI thống nhất. Bạn sẽ tìm hiểu thêm về chủ đề ứng dụng và Material Design trong bài học khác.

Nhiệm vụ 4: Thêm TextEdit và thiết lập các thuộc tính của nó

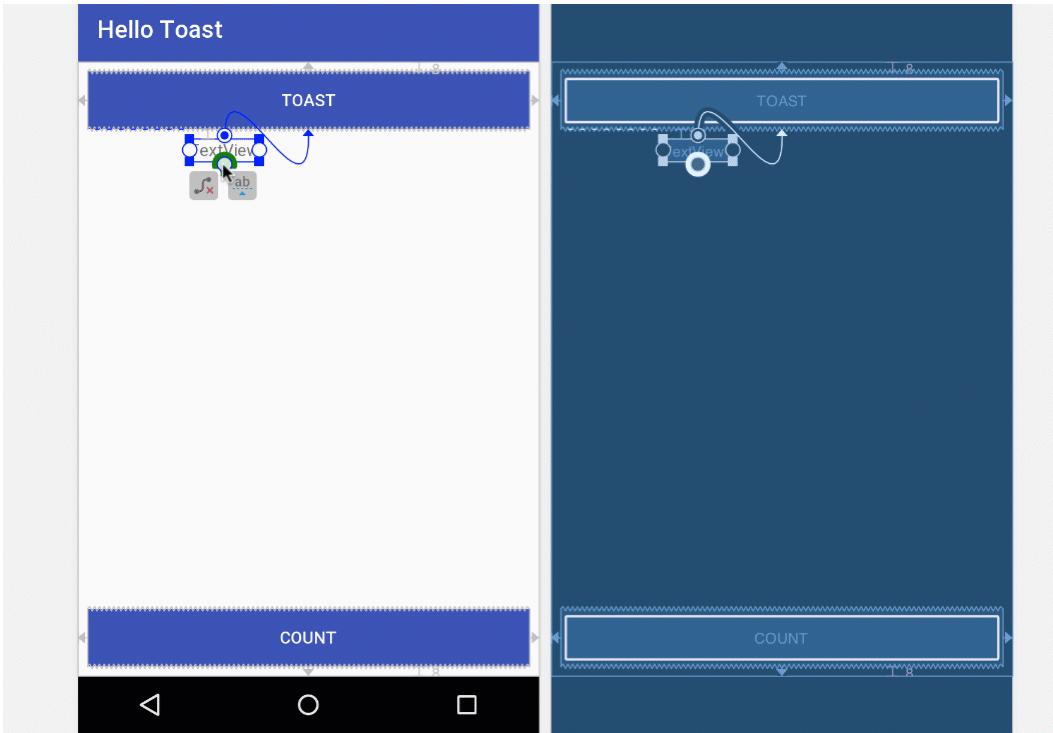
Một trong những lợi ích của **Bố cục ràng buộc** là khả năng căn chỉnh hoặc hạn chế các phần tử liên quan đến các phần tử khác. Trong nhiệm vụ này, bạn sẽ thêm một Xem văn bản ở giữa bố cục và giới hạn nó theo chiều ngang với các lề và theo chiều dọc với hai Cái nút các phần tử. Sau đó, bạn sẽ thay đổi các thuộc tính cho Xem văn bản trong **Thuộc tính** có.

4.1 Thêm TextView và các ràng buộc

1. Như được hiển thị trong hình ảnh động bên dưới, hãy kéo một Xem văn bản từ **Bảng màu** gần đến phần trên của bố cục và kéo một ràng buộc từ đầu Xem văn bản đến tay cầm ở phía dưới **Nút** Cái nút. Điều này hạn chế Xem văn bản ở bên dưới Cái nút.



2. Như được hiển thị trong hình ảnh động bên dưới, hãy kéo một ràng buộc từ dưới cùng của Xem văn bản đến tay cầm ở phía trên **Đếm** Cái nút, và từ các phía của Xem văn bản vào các bên của bố cục. Điều này hạn chế Xem văn bản ở giữa bố cục giữa hai Cái nút các yếu tố.



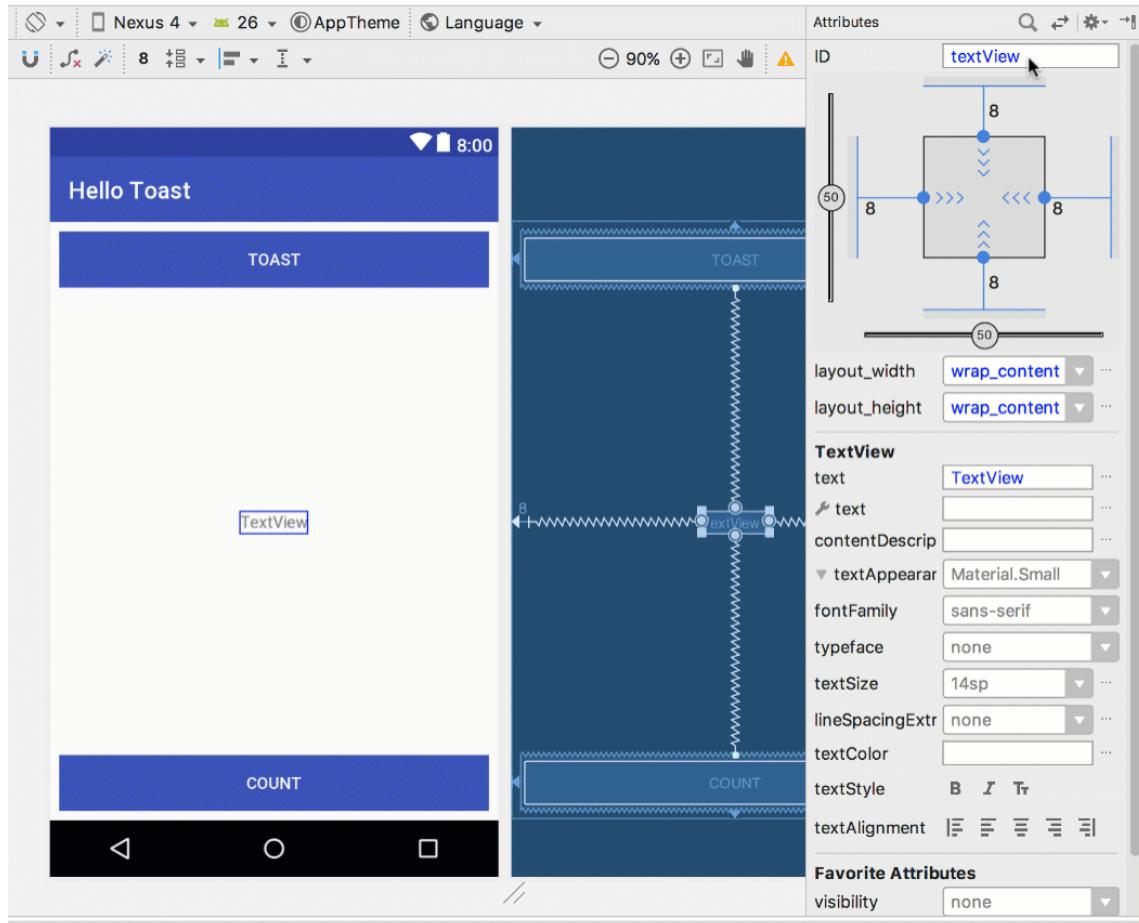
4.2 Thiết lập các thuộc tính TextView

Với Xem văn bản đã chọn, mở **Thuộc tính** ngắn, nếu nó chưa được mở. Đặt thuộc tính cho Xem văn bản như được hiển thị trong hình ảnh động bên dưới. Các thuộc tính bạn chưa gặp phải được giải thích sau hình ảnh:

- 1.Đặt NHẬN DẶNG ĐẾN **hiển thị_số lượng**.
- 2.Đặt **chữ ĐỀN 0**.
- 3.Đặt Kích thước văn bản **ĐỀN 160sp**.
- 4.Đặt Kiểu văn bản **ĐỀN B**(đậm) và **textAlignment** thành **CĂN CHỈNH** (căn giữa đoạn văn).
- 5.Thay đổi các điều khiển kích thước chế độ xem theo chiều ngang và chiều dọc (chiều rộng bố cục và chiều cao bố cục) **ĐỀN ràng buộc_phù hợp**.
- 6.Đặt Màu văn bản **ĐỀN@màu sắc/màu sắc Chính**.

7.Cuộn xuống khung và nhấp vào**Xem tất cả các thuộc tính**, cuộn xuống trang thứ hai của các thuộc tính để lý lịch,và sau đó nhập#**FFF00**để có màu vàng.

8.Cuộn xuống để trọng lực,mở rộng trọng lực,và chọn**trung tâm_xem**((đối với chiều dọc-trung tâm).



- Kích thước văn bản: Kích thước văn bản của Chế độ xem văn bản. Đối với bài học này, kích thước được đặt thành 160sp. Các sp đứng chia pixel không phụ thuộc vào tỷ lệ và thích dp, là đơn vị tỷ lệ với mật độ màn hình và sở thích về kích thước phông chữ của người dùng. Sử dụng đơn vị dp khi bạn chỉ định kích thước phông chữ để kích thước được điều chỉnh cho cả mật độ màn hình và sở thích của người dùng.
- Kiểu văn bản: Và Căn chỉnh văn bản: Kiểu văn bản, được đặt thành B (in đậm) trong bài học này và căn chỉnh văn bản, được đặt thành CĂN CHỈNH (căn giữa đoạn văn).

- trọng lực:Các trọng lực thuộc tính chỉ định cách một Xem được căn chỉnh trong phạm vi của nó cha mẹ Xem hoặc ViewGroup. Trong bước này, bạn căn giữa Xem văn bản được căn giữa theo chiều dọc trong phần cha mẹ Bố cục ràng buộc.

Bạn có thể nhận thấy rằng lý lịch thuộc tính nằm ở trang đầu tiên của **Thuộc tính khung** cho một Cái nút, nhưng ở trang thứ hai của **Thuộc tính khung** cho một Chế độ xem văn bản. Các **Thuộc tính** thay đổi khung cho từng loại Xem: Các thuộc tính phổ biến nhất cho Xem loại xuất hiện trên trang đầu tiên, và phần còn lại được liệt kê trên trang thứ hai. Để trở về trang đầu tiên của **Thuộc tính** có,

nhấp vào  biểu tượng trên thanh công cụ ở đầu ngắn.

Nhiệm vụ 5: Chính sửa bố cục trong XML

Bố cục ứng dụng Hello Toast gần hoàn thiện! Tuy nhiên, một dấu chấm than xuất hiện bên cạnh mỗi phần tử UI trong Component Tree. Di con trỏ qua các dấu chấm than này để xem các thông báo cảnh báo, như hiển thị bên dưới. Cảnh báo tương tự xuất hiện cho cả ba phần tử: chuỗi được mã hóa cứng phải sử dụng tài nguyên.



Cách dễ nhất để khắc phục sự cố 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 trong mã nguồn XML.

5.1 Mở mã XML cho bố cục

Đối với nhiệm vụ này, hãy mở `activity_main.xml` nếu nó chưa được mở và nhấp vào **Chữ tab Design** ở cuối trình chỉnh sửa bố cục.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

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

The screenshot shows the Android Studio code editor with two tabs: `activity_main.xml` and `MainActivity.java`. The `activity_main.xml` tab is active, displaying the following XML code:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.android.hellotoast.MainActivity">

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

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

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

A yellow tooltip appears over the string "Toast" in the first button's `text` attribute, reading: "Hardcoded string 'Toast', should use @string resource more... (⌘F1)".

5.2 Trích xuất tài nguyên chuỗi

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

- 1.Nhấp một lần vào từ "Nướng"(cảnh báo được đánh dấu đầu tiên).
- 2.Nhấn **Alt-Enter** trong Windows hoặc **Tùy chọn-Nhập** trong macOS và chọn **Trích xuất tài nguyên chuỗi** từ menu bật lên.
- 3.Đi vào **button_label_toast** cho **Tên tài nguyên**.
- 4.Nhấp chuột **ĐƯỢC RỒI**. Một tài nguyên chuỗi được tạo ra trong giá trị/res/string.xml file và chuỗi trong mã của bạn được thay thế bằng tham chiếu đến tài nguyên:

```
@string/button_label_toast
```

- 5.Trích xuất các chuỗi còn lại:nút_nhấn_số_lượngvì "Đếm",Vàđếm_giá_trị_ban_đầu vì "0".

- 6.Trong **Dự án > Android** ngắt, mở rộng **giá trị** trong **độ phân giải**, và sau đó nhấp đúp **chuỗi.xml** để xem tài nguyên chuỗi của bạn trong strings.xml ficái:

```
<tài nguyên>
  <string name="app_name">Xin chào Toast</string> <string
  name="button_label_toast">Toast</string> <string
  name="button_label_count">Đếm</string> <string
  name="count_initial_value">0</string> </resources>
```

- 7.Bạn cần một chuỗi khác để sử dụng trong tác vụ tiếp theo hiển thị thông báo. Thêm vào strings.xml file một nguồn tài nguyên chuỗi khác được đặt tên tin nhắn toast cho cụm từ "Xin chào bánh mì nướng!":

```
<tài nguyên>
  <string name="app_name">Xin chào Toast</string> <string
  name="button_label_toast">Toast</string> <string
  name="button_label_count">Đếm</string> <string
  name="count_initial_value">0</string> <string
  name="toast_message">Xin chào Toast!</string>
```

```
</tài nguyên>
```

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

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

Trong nhiệm vụ này, bạn thêm một phương thức Java cho mỗiCái nútTRONGHoạt động chínhthực hiện khi người dùng chạm vàoCái nút.

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

MỘTtrình xử lý nhấp chuộtlà một phương pháp được gọi khi người dùng nhấp hoặc chạm vào một phần tử UI có thể nhấp. Trong Android Studio, bạn có thể chỉ định tên của phương pháp trongtrênClick filinh vực trong**Thiết kế**tab của **Thuộc tính**ngăn. Bạn cũng có thể chỉ định tên của phương thức xử lý trong trình soạn thảo XML bằng cách thêmandroid:onClicktài sản choCái nút.Bạn sẽ sử dụng phương pháp sau vì bạn vẫn chưa tạo các phương thức xử lý và trình soạn thảo XML cung cấp một cách tự động để tạo các phương thức đó.

1.Với trình soạn thảo XML mở (tab Văn bản), hãy tìmCái nútvớiandroid:idđặt thành nút_bật_bật:

```
<Nút
    android:id="@+id/button_toast"
    android:layout_width="0dp"
    ...
    ứng dụng:layout_constraintStart_toStartOf="cha mẹ"
    ứng dụng:layout_constraintTop_toTopOf="cha mẹ" />
```

2.Thêm vàoandroid:onClick thuộc tính đến cuốinút_bánh ...phần tử sau thuộc tính cuối cùng và trước chỉ báo kết thúc />:

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

3.Nhấp vào biểu tượng bóng đèn đỏ xuất hiện bên cạnh thuộc tính. Chọn**Tạo trình xử lý nhấp chuột**, chọn **Hoạt động chính**và nhấp vào**ĐƯỢC RỒI**.

Nếu biểu tượng bóng đèn đỏ không xuất hiện, hãy nhấp vào tên phương pháp ("hiển thị Bánh mì nướng"). Nhấn**Alt-Enter** (Tùy chọn-Nhập trên máy Mac), hãy chọn**Tạo 'showToast(view)' trong MainActiviy**và nhấp vào**ĐƯỢC RỒI**.

Hành động này tạo ra một phương thức giữ chỗ cho hiển thịToast() phương pháp trong Hoạt động chính,như được hiển thị ở cuối các bước này.

4.Lặp lại hai bước cuối cùng vớibutton_count Nút:Thêm vàoandroid:onClick thuộc tính vào cuối và thêm trình xử lý nhấp chuột:

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

Mã XML cho các thành phần UI trongBố cục ràng buộcbây giờ trông như thế này:

```
<Nút
    android:id="@+id/button_toast"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:background="@color/colorPrimary"
    android:text="@string/button_label_toast"
    android:textColor="@android:color/white"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
```

```
Ứng dụng:layout_constraintTop_toTopOf="cha mẹ"
android:onClick="showToast"/>

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

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

5.Nếu như MainActivity.java chưa mở, mở rộng **java** trong chế độ xem Dự án > Android, mở rộng **com.example.android.hellotoast**, và sau đó nhấp đúp **Hoạt động chính**. Trình soạn thảo mã xuất hiện với mã trong Hoạt động chính:

```
gói com.example.android.hellotoast;
```

```
nhập android.support.v7.app.AppCompatActivity; nhập
android.os.Bundle;
nhập android.view.View;

lớp công khai MainActivity mở rộng AppCompatActivity {

    @Ghi đè
    được bảo vệ void onCreate(Gói savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void showToast(Xem chế độ xem) { }

    public void countUp(Xem chế độ xem) { }

}
```

6.2 Chỉnh sửa trình xử lý nút Toast

Bây giờ bạn sẽ chỉnh sửa hiển thịToast() phương pháp—các **Nướng** Cái nút trình xử lý nhấp chuột trong Hoạt động chính —để nó hiển thị một thông điệp. **Nướng** cung cấp một cách để hiển thị một thông điệp đơn giản trong một cửa sổ bật lên nhỏ. Nó chỉ lấp đầy lượng không gian cần thiết cho thông điệp. Hoạt động hiện tại vẫn hiển thị và tương tác. A Nướng có thể hữu ích cho việc kiểm tra tính tương tác trong ứng dụng của bạn—thêm Nướng tin nhắn để hiển thị kết quả của việc chạm vào một Cái nút hoặc thực hiện một hành động.

Thực hiện theo các bước sau để chỉnh sửa **Nướng** Cái nút trình xử lý nhấp chuột:

- Xác định vị trí mới được tạo hiển thịToast() phương pháp.

```
public void showToast(Xem chế độ xem) { }
```

- Để tạo một trường hợp của một **Nướng**, gọi làmVăn bản() phương pháp nhà máy trên **Nướng** lớp học.

```
public void showToast(Xem chế độ xem) {  
    Bánh mì nướng bánh mì nướng = Toast.makeText(  
}
```

Câu lệnh này sẽ không đầy đủ cho đến khi bạn hoàn tất cả các bước.

3.Cung cấp bối cảnh của ứng dụng hoạt động.Bởi vì một Nút hiển thị ở trên cùng của Giao diện người dùng, hệ thống cần thông tin về hiện tại hoạt động.Khi bạn đã ở trong bối cảnh của hoạt động bạn cần ngử cảnh nào, hãy sử dụng cái này như một lối tắt.

```
Toast toast = Toast.makeText(cái này,
```

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

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

5.Cung cấp thời lượng cho màn hình hiển thị. Ví dụ, Toast.LENGTH_SHORT hiển thị thông báo trong một thời gian tương đối ngắn.

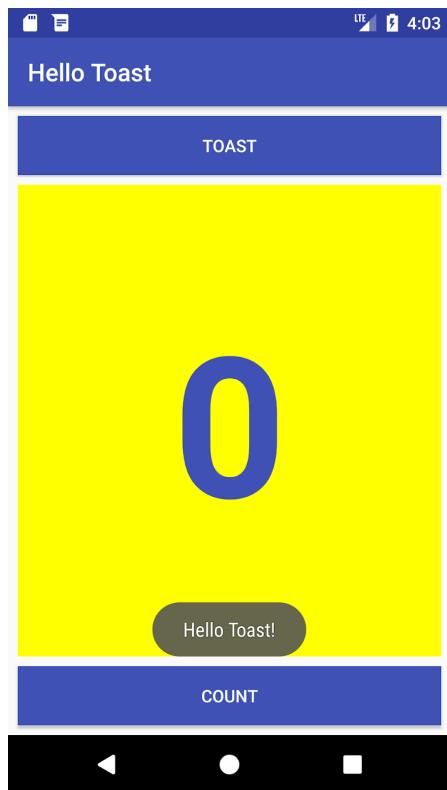
```
Toast toast = Toast.makeText(này, R.string.toast_message,  
    Bánh mì nướng.LENGTH_SHORT);
```

Thời gian của một Nút màn hình có thể là Bánh mì nướng.LENGTH_LONG hoặc Toast.LENGTH_SHORT. Độ dài thực tế là khoảng 3,5 giây cho đoạn dài Nút và 2 giây cho đoạn ngắn Nút.

6.Hiển thị Nút bằng cách gọi trình diễn(). Sau đây là toàn bộ hiển thị Toast() phương pháp:

```
public void showToast(Xem chế độ xem) {  
    Toast toast = Toast.makeText(này, R.string.toast_message,  
        Bánh mì nướng.LENGTH_SHORT);  
    toast.hiển thị();  
}
```

Chạy ứng dụng và xác minh rằng Nút tin nhắn xuất hiện khi **Nhấn** nút được nhấn.



6.3 Chỉnh sửa trình xử lý Nút Đếm

Bây giờ bạn sẽ chỉnh sửa đếm lên() phương pháp—các **Đếm** cái nút trong trình xử lý nhấp chuột trong **Hoạt động chính**—để nó hiển thị số đếm hiện tại sau **Đếm** được chạm vào. Mỗi lần chạm sẽ tăng số lượng lên một.

Mã cho trình xử lý phải:

- Theo dõi số lượng khi nó thay đổi.
- Gửi số lượng cập nhật đến Xem văn bản để hiển thị nó.

Thực hiện theo các bước sau để chỉnh sửa **Đếm** cái nút trong trình xử lý nhấp chuột:

1. Xác định vị trí mới được tạo đếm lên() phương pháp.

```
public void countUp(Xem chế độ xem) { }
```

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

```
public void countUp(Xem chế độ xem) {
    Đếm++;
}
```

Nếu biểu tượng bóng đèn màu đỏ không xuất hiện, hãy chọn **Đếm++** biểu hiện. Cuối cùng bóng đèn màu đỏ xuất hiện.

3. Nhấp vào biểu tượng bóng đèn màu đỏ và chọn **Tạo trường 'mCount'** từ menu bật lên. Điều này tạo một biến thành viên riêng tư ở đầu **Hoạt động chính**, và **Android Studio** cho rằng bạn muốn nó là một số nguyên (số nguyên):

```
lớp công khai MainActivity mở rộng AppCompatActivity {
    int riêng tư mCount;
```

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

```
lớp công khai MainActivity mở rộng AppCompatActivity {  
    int riêng tư mCount = 0;
```

5.Cùng với biến trên, bạn cũng cần một biến thành viên riêng tư để tham chiếu
cái hiển thị số lượng Xem văn bản, mà bạn sẽ thêm vào trình xử lý nhấp chuột. Gọi biến này mSố lần
hiển thị:

```
lớp công khai MainActivity mở rộng AppCompatActivity {  
    int riêng mCount = 0; TextView  
    riêng mShowCount;
```

6.Bây giờ bạn đã có mShowCount, bạn có thể nhận được một tài liệu tham khảo Xem văn bản sử dụng ID bạn thiết lập
trong tệp bố trí. Để chỉ nhận được tham chiếu này một lần, hãy chỉ định nó trong `khi tạo()` phương pháp. Khi
bạn học trong một bài học khác, [khi tạo\(\)](#) phương pháp được sử dụng để *thổi phồng* bố cục, nghĩa là đặt chế độ
xem nội dung của màn hình thành bố cục XML. Bạn cũng có thể sử dụng nó để lấy tham chiếu đến các thành
phần UI khác trong bố cục, chẳng hạn như `Chế độ xem văn bản`. Xác định vị trí khi tạo() phương pháp trong
MainActivity:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
}
```

7.Thêm vào [tìmViewById](#) câu lệnh vào cuối phương thức:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main); mShowCount = (TextView)  
    findViewById(R.id.show_count);  
}
```

MỘT Xem, giống như một chuỗi, là một tài nguyên có thể có một id.tìmViewById cuộc gọi lấy ID của chế độ xem làm tham số của nó và trả về Xem. Bởi vì phương thức trả về một Xem, bạn phải đưa kết quả vào xem loại bạn mong đợi, trong trường hợp này ((Xem văn bản)).

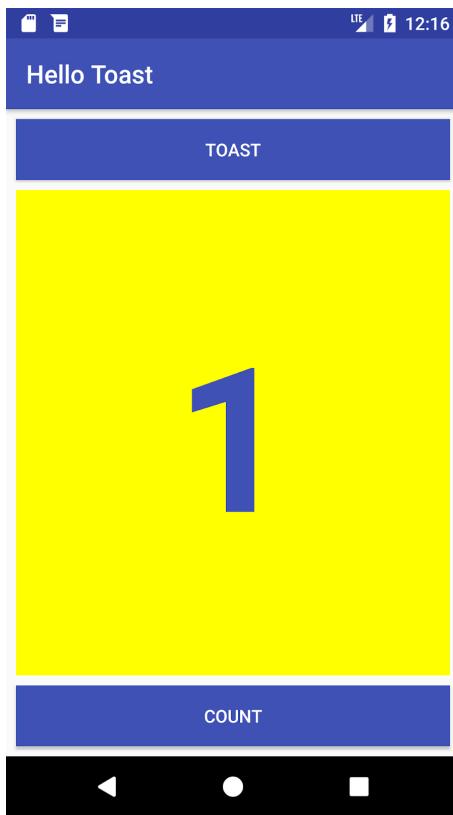
8.Bây giờ bạn đã giao chomHiển thị Số đếm cái Chế độ xem văn bản, bạn có thể sử dụng biến để thiết lập văn bản trong Xem văn bản với giá trị của đếm biến. Thêm phần sau vào đếm Lên() phương pháp:

```
nếu (mShowCount != null)  
    mShowCount.setText(Số nguyên.toString(mCount));
```

Toàn bộ đếm Lên() phương pháp bây giờ trông như thế này:

```
public void countUp(Xem chế độ xem) {  
    ++ Đếm;  
    nếu (mShowCount != null)  
        mShowCount.setText(Số nguyên.toString(mCount));  
}
```

9.Chạy ứng dụng để xác minh số lượng tăng lên khi bạn chạm vào Đếm cái nút.



Mẹo: Để có hướng dẫn chi tiết về cách sử dụng Bố cục ràng buộc, xem Codelab [Sử dụng ConstraintLayout để thiết kế chế độ xem của bạn](#).

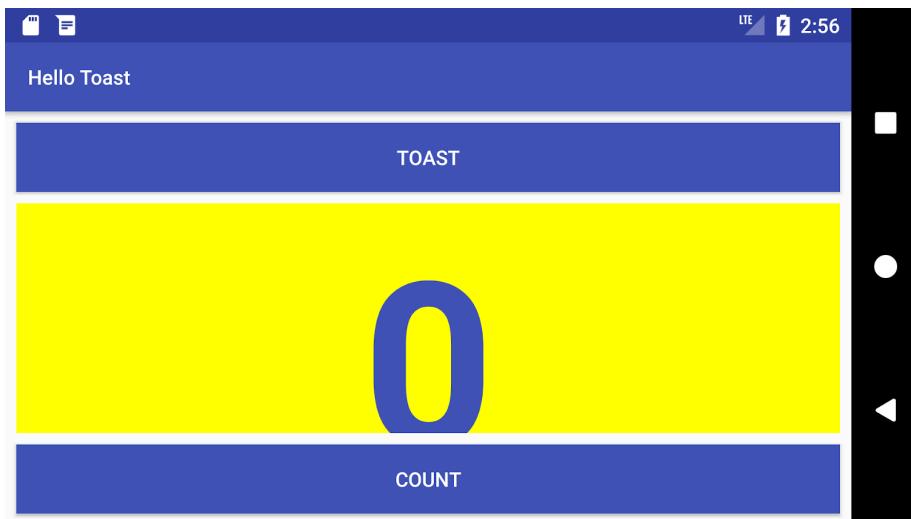
Mã giải pháp

Dự án Android Studio: [Xin chào Toast](#)

Thử thách mã hóa

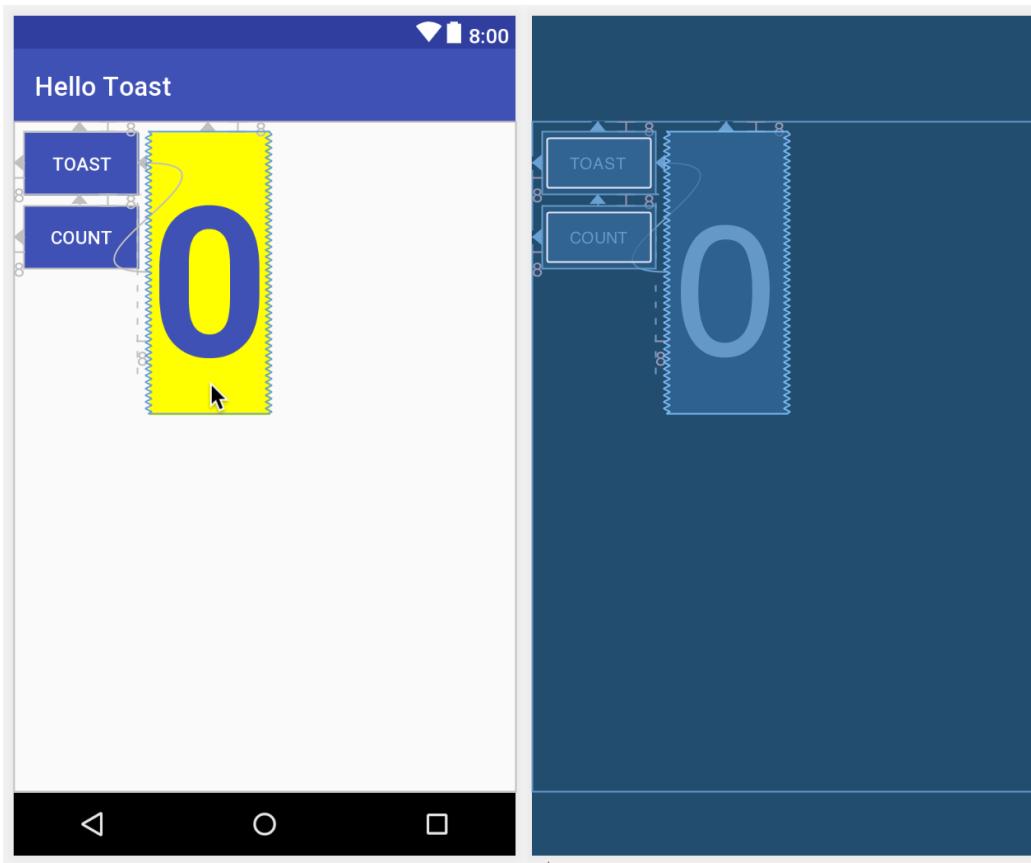
Ghi chú:Mọi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

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 theo chiều ngang, **Đếm**Cái nút có thể chồng chéo lên nhau.Xem văn bản dọc theo phía dưới như thể hiện trong hình bên dưới.



Thử thách: Thay đổi bối cảnh sao cho đẹp mắt theo cả chiều ngang và chiều dọc:

- 1.Trên máy tính của bạn, hãy tạo một bản sao của **Xin chàoToast** thư mục dự án và đổi tên nó thành **Xin chàoToastChallenge**.
- 2.Mở **Xin chàoToastChallenge** trong Android Studio và tái cấu trúc nó. (Xem [Phụ lục: Tiện ích](#) để biết hướng dẫn về cách sao chép và tái cấu trúc một dự án.)
- 3.Thay đổi bối cảnh sao cho **Nướng**Cái nút **Đếm**Cái nút xuất hiện ở phía bên trái, như được hiển thị trong hình bên dưới.Xem văn bản xuất hiện bên cạnh chúng, nhưng chỉ đủ rộng để hiển thị nội dung của nó. (Gợi ý: Sử dụng `boc_nội_dung`.)
- 4.Chạy ứng dụng theo cả chiều ngang và chiều dọc.



Tác phẩm này được cấp phép theo một [Giấy phép Creative Commons Ghi công 4.0 Quốc tế](#). PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.



Mã giải pháp thách thức

Dự án Android Studio:[Xin chàoToastChallenge](#)

Bản tóm tắt

Xem, ViewGroup, và bố cục:

- Tất cả các thành phần UI đều là các lớp con của Xem lớp và do đó thừa hưởng nhiều thuộc tính của Xem siêu lớp.
- Xem các yếu tố có thể được nhóm lại bên trong một ViewGroup, hoạt động như một container. Mối quan hệ là cha-con, trong đó *cha* là một ViewGroup, và *đứa trẻ* là một Xem hoặc một cái khác ViewGroup.

- Các khi tạo() phương pháp được sử dụng để *thổi phồng bố cục*, nghĩa là đặt chế độ xem nội dung của màn hình thành bố cục XML. Bạn cũng có thể sử dụng nó để lấy tham chiếu đến các thành phần UI khác trong bố cục.
- MỘT Xem, giống như một chuỗi, là một tài nguyên có thể có một id. tìmViewById cuộc gọi lấy ID của chế độ xem làm tham số của nó và trả về Xem.

Sử dụng trình chỉnh sửa bố cục:

- Nhấp vào **Thiết kế** tab để thao tác các thành phần và bố cục, và **Chữ** tab để chỉnh sửa mã XML cho bố cục.
- Trong **Thiết kế** tab, các **Bảng màu** ngang hiển thị các thành phần UI mà bạn có thể sử dụng trong bố cục ứng dụng của mình và **Cây thành phần** ngang hiển thị hệ thống phân cấp chế độ xem của các thành phần UI.
- Các ngăn thiết kế và bản thiết kế của trình chỉnh sửa bố cục hiển thị các thành phần UI trong bố cục.
- Các **Thuộc tính** tab hiển thị **Thuộc tính** ngang để thiết lập thuộc tính cho một thành phần UI.
- Tay cầm ràng buộc: Nhấp vào tay cầm ràng buộc, được hiển thị dưới dạng vòng tròn ở mỗi bên của phần tử, sau đó kéo đến tay cầm ràng buộc khác hoặc đến ranh giới cha để tạo ràng buộc.
Ràng buộc được biểu diễn bằng đường ngoằn ngoèo.
- Tay cầm thay đổi kích thước: Bạn có thể kéo tay cầm thay đổi kích thước hình vuông để thay đổi kích thước phần tử. Trong khi kéo, tay cầm sẽ thay đổi thành góc nghiêng.
- Khi được bật, công cụ **Tự động kết nối** sẽ tự động tạo hai hoặc nhiều ràng buộc cho một thành phần UI vào bố cục cha. Sau khi bạn kéo thành phần vào bố cục, công cụ sẽ tạo ra các ràng buộc dựa trên vị trí của thành phần.
- Bạn có thể xóa các ràng buộc khỏi một phần tử bằng cách chọn phần tử và di chuột qua  nút. Nhấp vào nút này để xóa *tất cả* ràng buộc trên phần tử đã chọn. Để xóa một ràng buộc duy nhất, hãy nhấp vào tay cầm cụ thể đặt ràng buộc đó.
- Các **Thuộc tính** pane cung cấp quyền truy cập vào tất cả các thuộc tính XML mà bạn có thể gán cho một phần tử UI. Nó cũng bao gồm một panel kích thước hình vuông được gọi là *xem thanh tra* ở trên cùng. Các ký hiệu bên trong hình vuông biểu thị cài đặt chiều cao và chiều rộng.

Thiết lập chiều rộng và chiều cao của bố cục:

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

- Các ràng buộc phải hợp thiết lập mở rộng chế độ xem để lấp đầy phần cha theo chiều rộng hoặc chiều cao—lên đến một lần nhất định, nếu có.
- Các nội dung bao gồm thiết lập thu nhỏ kích thước chế độ xem để chế độ xem chỉ đủ lớn để bao gồm nội dung của nó. Nếu không có nội dung, chế độ xem sẽ trở nên vô hình.
- Sử dụng một số lượng cố định [pixel không phụ thuộc vào mật độ](#) để chỉ định kích thước cố định, điều chỉnh theo kích thước màn hình của thiết bị.

Trích xuất tài nguyên chuỗi:

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

- 1.Nhấp một lần vào chuỗi được mã hóa cứng để trích xuất, nhấn **Alt-Enter**(Tùy chọn-Nhập trên máy Mac) và chọn **Trích xuất tài nguyên chuỗi** từ menu bật lên.
- 2.Đặt **Tên tài nguyên**.
- 3.Nhấp chuột **ĐƯỢC RỒI**. Điều này tạo ra một tài nguyên chuỗi trong giá trị/res/string.xml file và chuỗi trong mã của bạn được thay thế bằng tham chiếu đến tài nguyên: @chuỗi/nút_nhấn_bánh ...

Xử lý nhấp chuột:

- Một trình xử lý nhấp chuột là phương thức được gọi khi người dùng nhấp hoặc chạm vào phần tử UI.
- Chỉ định trình xử lý nhấp chuột cho một phần tử UI như Cái nút bằng cách nhập tên của nó vào trên Click lĩnh vực trong **Thiết kế** tab của **Thuộc tính** ngăn, hoặc trong trình soạn thảo XML bằng cách thêm android:onClick thuộc tính cho một phần tử UI như Cái nút.
- Tạo trình xử lý nhấp chuột trong chính **Hoạt động** sử dụng Xem tham số. Ví dụ: public void showToast(Xem chế độ xem) {...}.
- Bạn có thể tìm thấy thông tin về tất cả Cái nút các thuộc tính trong [Tài liệu lớp nút](#) và tất cả các Xem văn bản các thuộc tính trong [Tài liệu lớp TextView](#).

Hiển thị tin nhắn Toast:

MỘT [Nướng](#) cung cấp một cách để hiển thị một thông điệp đơn giản trong một cửa sổ bật lên nhỏ. Nó chỉ lấp đầy lượng không gian cần thiết cho thông điệp. Để tạo một phiên bản của [Nướng](#), hãy làm theo các bước sau:

- 1.Gọi cho [làm Văn bản\(\)](#) phương pháp nhà máy trên [Nướng](#) lớp học.
- 2.Cung cấp [bối cảnh](#) của ứng dụng Hoạt động và thông báo hiển thị (chẳng hạn như tài nguyên chuỗi).
- 3.Cung cấp thời lượng hiển thị, ví dụ [Toast.LENGTH_SHORT](#) trong một thời gian ngắn. Thời gian có thể là Bánh mì nướng.[LENGTH_LONG](#) hoặc [Toast.LENGTH_SHORT](#).
- 4.Hiển thị [Nướng](#) bằng cách gọi [trình diễn\(\)](#).

Khái niệm liên quan

Tài liệu khái niệm liên quan có trong [1.2: Bố cục và tài nguyên cho UI](#).

Tìm hiểu thêm

Tài liệu dành cho nhà phát triển Android:

- [Studio Android](#)
- [Xây dựng giao diện người dùng với Layout Editor](#)
- [Xây dựng một giao diện người dùng đáp ứng với ConstraintLayout](#)
- [Bố cục](#)
- [Xem](#)
- [Cái nút](#)
- [Xem văn bản](#)
- [Tài nguyên Android](#)
- [Tài nguyên R.color chuẩn của Android](#)
- [Hỗ trợ các mật độ khác nhau](#)
- [Sự kiện đầu vào Android](#)
- [Bối cảnh](#)

Khác:

- Codelab:[Sử dụng ConstraintLayout để thiết kế chế độ xem của bạn](#)
- [Từ vựng và khái niệm thuật ngữ](#)

Codelab tiếp theo là[Cơ bản về Android 1.2 Phần B: Trình chỉnh sửa bố cục](#).

Bài 1.2 Phần B: Trình chỉnh sửa bố cục

Giới thiệu

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

Bố cục ràng buộc là một[ViewGroup](#), đó là một điều đặc biệt Xem có thể chứa đựng những thứ khác Xem các đối tượng (gọi là *những đứa trẻ* hoặc *lượt xem của trẻ em*). Thực hành này cho thấy nhiều tính năng hơn của Bố cục ràng buộc và trình chỉnh sửa bố cục.

Bài thực hành này cũng giới thiệu hai bài khác [ViewGroup](#) các lớp con:

- [Bố cục tuyến tính](#): Một nhóm liên kết trẻ em Xem các thành phần bên trong theo chiều ngang hoặc chiều dọc.
- [Bố cục tương đối](#): Một nhóm trẻ em Xem các yếu tố trong đó mỗi Xem phần tử được định vị và căn chỉnh tương đối với các phần tử khác Xem phần tử trong [ViewGroup](#). Vị trí của trẻ Xem các yếu tố được mô tả liên quan đến nhau hoặc đến cha mẹ [ViewGroup](#).

Những điều bạn nên biết

Bạn sẽ có thể:

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

Những gì bạn sẽ học được

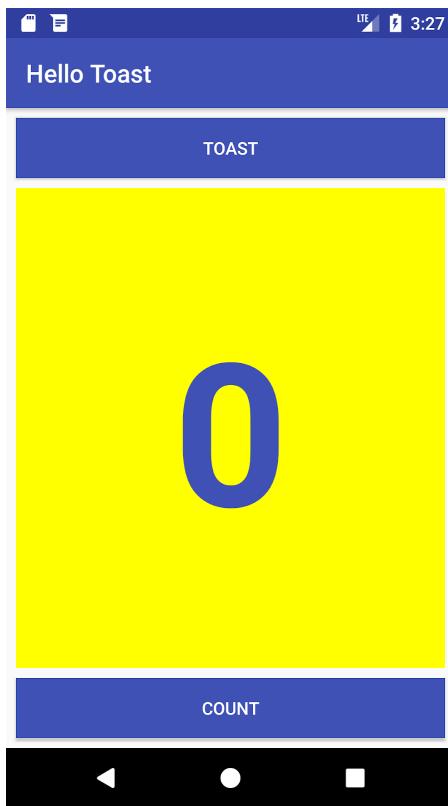
- Cách tạo biến thể bố cục cho hướng ngang (phong cảnh).
- Cách tạo biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Cách sử dụng ràng buộc đường cơ sở để căn chỉnh các thành phần UI với văn bản.
- Cách sử dụng các nút đóng gói và căn chỉnh để căn chỉnh các thành phần trong bố cục.
- Làm thế nào để định vị chế độ xem trong một Bố cục tuyến tính.
- Làm thế nào để định vị chế độ xem trong một Bố cục tương đối.

Bạn sẽ làm gì

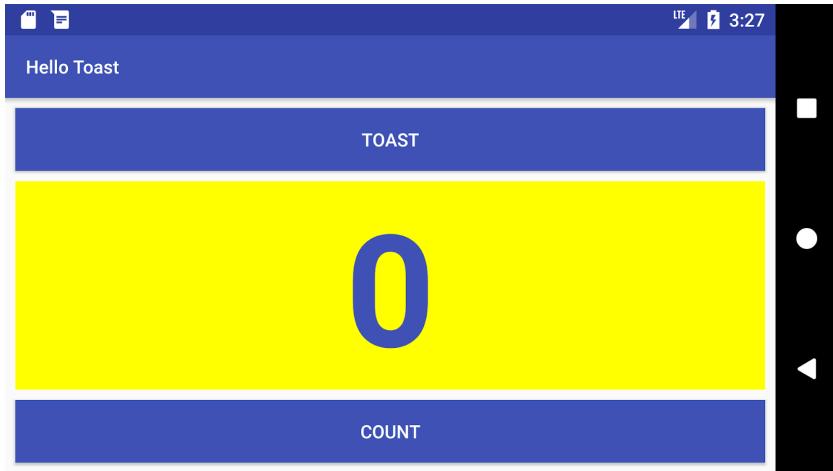
- Tạo một biến thể bố cục cho hướng hiển thị ngang.
- Tạo một biến thể bố cục cho máy tính bảng và màn hình lớn hơn.
- Sửa đổi bố cục để thêm ràng buộc vào các thành phần UI.
- Sử dụng Bố cục ràng buộc ràng buộc cơ sở để căn chỉnh các thành phần với văn bản.
- Sử dụng Bố cục ràng buộc nút đóng gói và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

Tổng quan về ứng dụng

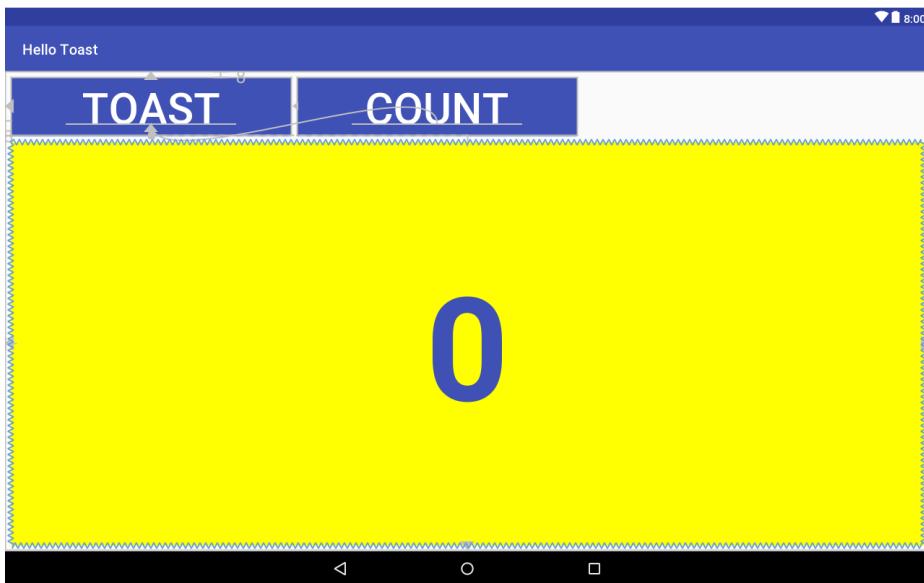
Ứng dụng Hello Toast trong bài học trước sử dụng [Bố cục ràng buộc](#) để sắp xếp các thành phần UI trong Hoạt động Bố cục như thể hiện trong hình bên dưới.



Để có thêm nhiều thực hành với Bối cục ràng buộc, bạn sẽ tạo một biến thể của bối cục này theo hướng ngang như thể hiện trong hình bên dưới.



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

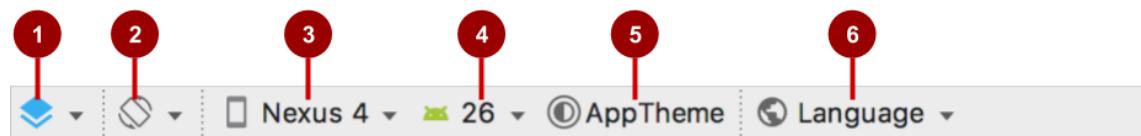


Bạn cũng tìm hiểu về những điều khác `ViewGroup` các lớp con như [Bố cục tuyến tính](#) Và [Bố cục tương đối](#) và thay đổi giao diện ứng dụng Hello Toast để sử dụng chúng.

Nhiệm vụ 1: Tạo các biến thể bố cục

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

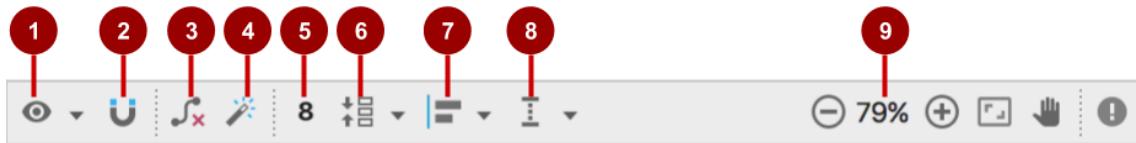
Trong tác vụ này, bạn sẽ sử dụng một số nút trong hai thanh công cụ trên cùng của trình chỉnh sửa bố cục. Thanh công cụ trên cùng cho phép bạn định cấu hình giao diện của bản xem trước bố cục trong trình chỉnh sửa bố cục:



Trong hình trên:

1. **Chọn bề mặt thiết kế:** Lựa chọn **Thiết kế** để hiển thị bản xem trước màu sắc của bố cục của bạn hoặc **Bản thiết kế** để chỉ hiển thị phác thảo cho từng thành phần UI. Để xem cả hai các ô cạnh nhau, chọn **Thiết kế + Bản thiết kế**.
2. **Định hướng trong Trình biên tập:** Lựa chọn **Chân dung** hoặc **Phong cảnh** để hiển thị bản xem trước theo hướng dọc hoặc ngang. Điều này hữu ích để xem trước bố cục mà không cần phải chạy ứng dụng trên trình giả lập hoặc thiết bị. Để tạo bố cục thay thế, hãy chọn **Tạo biến thể cảnh quan** hoặc các biến thể khác.
3. **Thiết bị trong Trình chỉnh sửa:** Chọn loại thiết bị (điện thoại/máy tính bảng, Android TV hoặc Android Wear).
4. **Phiên bản API trong Trình chỉnh sửa:** Chọn phiên bản Android sử dụng để hiển thị bản xem trước.
5. **Chủ đề trong Trình biên tập:** Chọn một chủ đề (chẳng hạn như **Chủ đề Ứng dụng**) để áp dụng cho bản xem trước.
6. **Vị trí trong Trình soạn thảo:** Chọn ngôn ngữ và địa phương để xem trước. Danh sách này chỉ hiển thị ngôn ngữ có sẵn trong tài nguyên chuỗi (xem bài học về bản địa hóa để biết chi tiết về cách thêm ngôn ngữ). Bạn cũng có thể chọn **Xem trước từ phải sang trái** để xem bố cục như thể ngôn ngữ RTL đã được chọn.

Thanh công cụ thứ hai cho phép bạn cấu hình giao diện của các thành phần UI trong một Bố cục ràng buộc, và để phóng to và thu nhỏ bản xem trước:



Trong hình trên:

1. **Trình diễn:** Chọn **Hiển thị ràng buộc** và **Hiển thị lề** để hiển thị chúng trong bản xem trước hoặc dừng hiển thị chúng.
2. **Tự động kết nối:** Bật hoặc tắt Tự động kết nối. Khi Tự động kết nối được bật, bạn có thể kéo bất kỳ phần tử (chẳng hạn như một Cái nút) đến bất kỳ phần nào của bố cục để tạo ra các ràng buộc đối với bố cục gốc.
3. **Xóa tất cả ràng buộc:** Xóa mọi ràng buộc trong toàn bộ bố cục.
4. **Suy ra ràng buộc:** Tạo ràng buộc bằng suy luận.
5. **Lề Mặc Định:** Đặt lề mặc định.
6. **Đóng gói:** Đóng gói hoặc mở rộng các phần tử đã chọn.
7. **Căn chỉnh:** Căn chỉnh các phần tử đã chọn.
8. **Hướng dẫn:** Thêm hướng dẫn theo chiều dọc hoặc chiều ngang.
9. Điều khiển thu phóng/di chuyển: Phóng to hoặc thu nhỏ.

Mẹo: Để tìm hiểu thêm về cách sử dụng trình chỉnh sửa bố cục, hãy xem [Xây dựng giao diện người dùng với Layout Editor](#). Để tìm hiểu thêm về cách xây dựng bố cục với ConstraintLayout, hãy xem [Xây dựng một giao diện người dùng đáp ứng với ConstraintLayout](#).

1.1 Xem trước bố cục theo hướng ngang

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

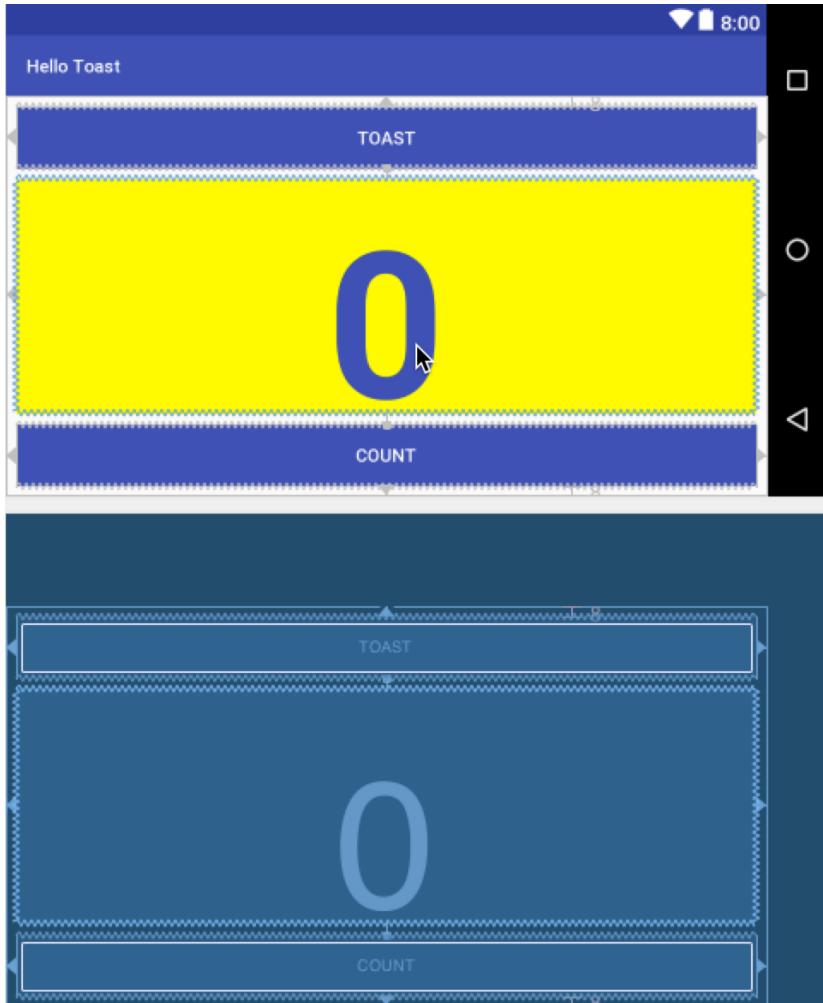
1. Mở ứng dụng Hello Toast từ bài học trước.

Ghi chú: Nếu bạn đã tải xuống [mã giải pháp cho HelloToast](#), bạn cần xóa các bố cục cảnh quan đã hoàn thành và cực lớn mà bạn sẽ tạo trong tác vụ này. Chuyển từ **Dự án > Android** ĐẾN **Dự án > Tệp dự án** trong ngăn Dự án, mở rộng **ứng dụng > ứng dụng > src/main > res**, chọn cả hai **bố trí** đất **thư mục và** **bố trí** **lớn** **thư mục** và chọn **Sửa > Xóa**. Sau đó chuyển ngăn Dự án trở lại **Dự án > Android**.

2. Mở **activity_main.xml** tập tin bố trí. Nhấp vào **Thiết kế** tab nếu nó chưa được chọn.

3. Nhấp vào **Định hướng** trong **Trình biên tập** cái nút ở thanh công cụ phía trên.

4. Lựa chọn **Chuyển sang chế độ phong cảnh** trong menu thả xuống. Bố cục xuất hiện theo hướng ngang như hiển thị bên dưới. Để trở về hướng dọc, hãy chọn **Chuyển sang chế độ Chân dung**.



1.2 Tạo một biến thể bố cục cho hướng ngang

Sự khác biệt trực quan giữa hướng dọc và hướng ngang cho bố cục này là chữ số (0) trong `Hiển thị_số_văn_bản` phần tử quá thấp so với hướng ngang—quá gần

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

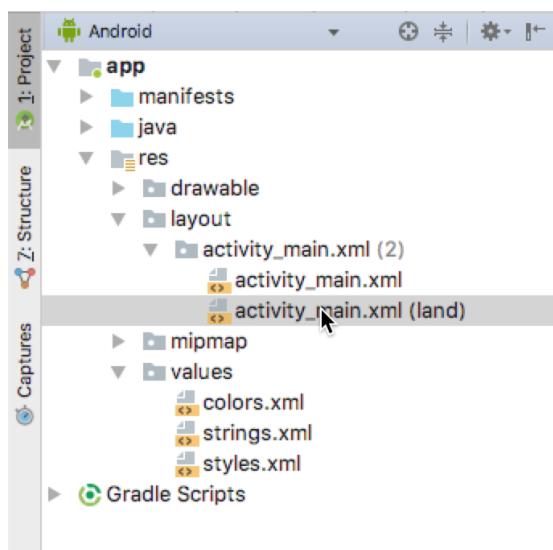
Đếmnút. Tùy thuộc vào thiết bị hoặc trình giả lập bạn sử dụng,Xem văn bảnphần tử có thể xuất hiện quá lớn hoặc không được căn giữa vì kích thước văn bản được cố định ở mức 160sp.

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

- 1.Nhấp vào**Định hướng** trong **Trình biên tập**cái nút  ở thanh công cụ phía trên.
- 2.Chọn**Tạo biến thể cảnh quan**.

Một cửa sổ biên tập mới mở ra với**đất/hoạt động chính.xml**tab hiển thị bố cục cho hướng ngang (ngang). Bạn có thể thay đổi bố cục này, dành riêng cho hướng ngang, mà không thay đổi hướng dọc (dọc) ban đầu.

- 3.Trong**Dự án > Android**cửa sổ, nhìn vào bên trong**res > bối trí**thư mục và bạn sẽ thấy Android Studio tự động tạo biến thể cho bạn, được gọi là**activity_main.xml (đất)**.



1.3 Xem trước bố cục cho các thiết bị khác nhau

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

- 1.Các**đất/hoạt động chính.xml**tab vẫn phải được mở trong trình chỉnh sửa bố cục; nếu không, hãy nhấp đúp vào **activity_main.xml (đất)**là trong**cách trình bày**thư mục.

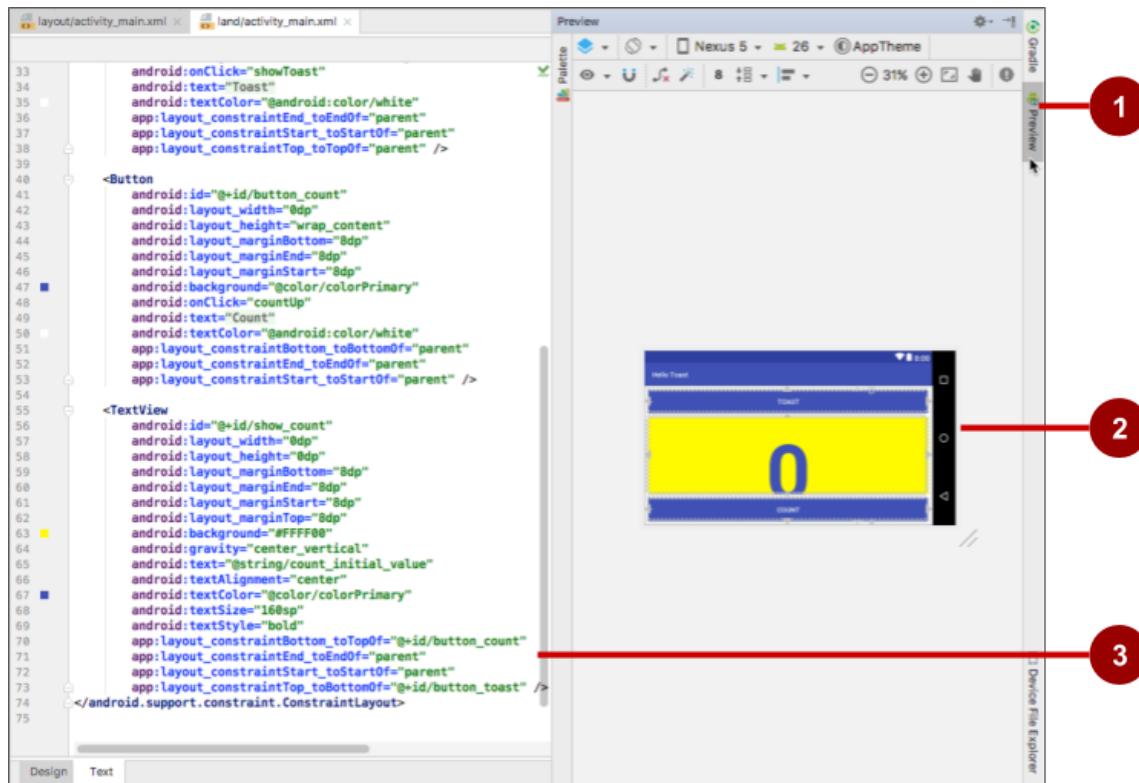
Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế . PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

2.Nhập vào **Thiết bị** trong **Trình chỉnh sửa** cái nút **Nexus 5** ở thanh công cụ phía trên.
 3.Chọn một thiết bị khác trong menu thả xuống. Ví dụ, chọn **Điện thoại Nexus 4**, **Điện thoại Nexus 5**, và sau đó **Điểm ảnh** để xem sự khác biệt trong bản xem trước. Những sự khác biệt này là do kích thước văn bản cố định cho chế độ xem văn bản.

1.4 Thay đổi bố cục theo hướng ngang

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



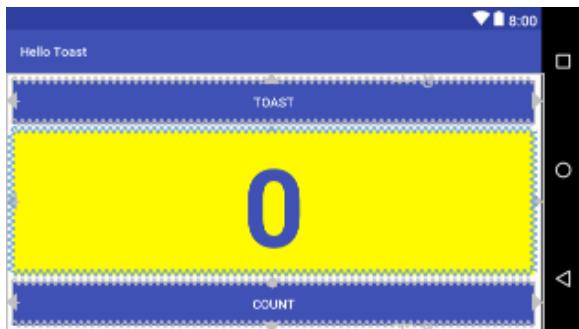
Hình trên cho thấy những điều sau:

1.Các **Xem trước** tab, mà bạn sử dụng để hiển thị khung xem trước

- 2.Khung xem trước
- 3.Mã XML

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

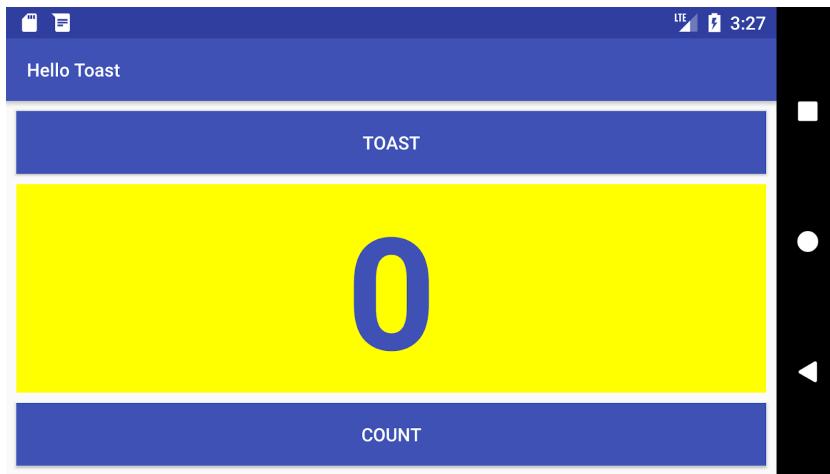
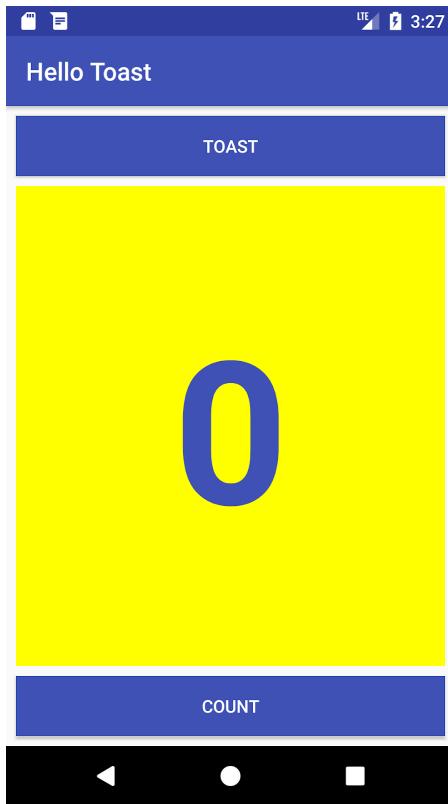
- 1.Các **đất/hoạt động chính.xml** tab vẫn phải được mở trong trình chỉnh sửa bố cục; nếu không, hãy nhấp đúp vào **activity_main.xml (đất)** làle trong cách **trình bày** thư mục.
- 2.Nhấp vào **Chữ** tab và **Xem trước** tab (nếu chưa chọn).
- 3.Tìm thấy **Xem** văn bản phần tử trong mã XML.
- 4.Thay đổi `android:textSize="160sp"` thuộc tính cho `android:textSize="120sp"`. Bản xem trước bố cục hiển thị kết quả:



- 5.Chọn các thiết bị khác nhau trong **Thiết bị** trong **Trình chỉnh sửa** menu thả xuống để xem bố cục trông như thế nào trên các thiết bị khác nhau theo chiều ngang.

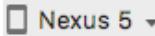
Trong ngăn biên tập, **đất/hoạt động chính.xml** tab hiển thị bố cục cho hướng ngang. **hoạt động_main.xml** tab hiển thị bố cục không thay đổi theo hướng dọc. Bạn có thể chuyển đổi qua lại bằng cách nhấp vào các tab.

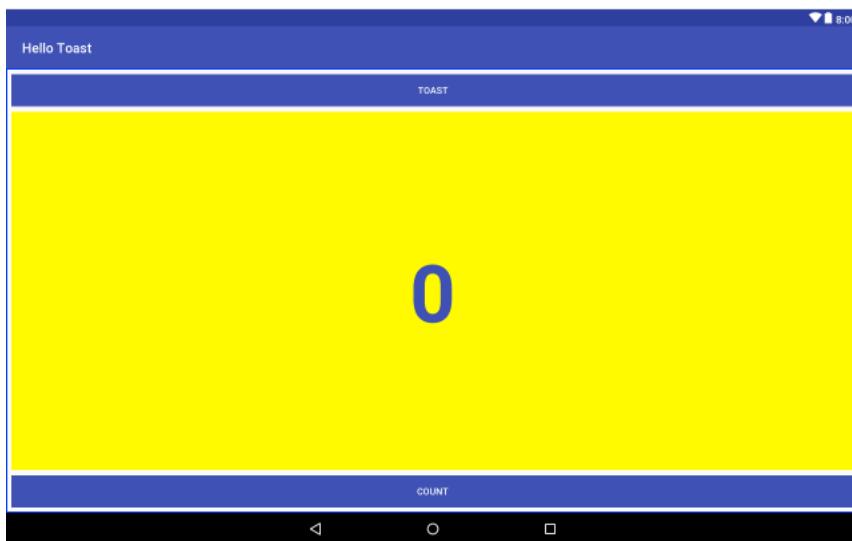
- 6.Chạy ứng dụng trên trình giả lập hoặc thiết bị và chuyển hướng từ dọc sang ngang để xem cả hai bố cục.



1.5 Tạo biến thể bố cục cho máy tính bảng

Như bạn đã học trước đó, bạn có thể xem trước bố cục cho các thiết bị khác nhau bằng cách nhấp vào **Thiết bị trong**

Biên tập viên cá nút  **Nexus 5** trong thanh công cụ trên cùng. Nếu bạn chọn một thiết bị như **Điện thoại Nexus 10** (một máy tính bảng) từ menu, bạn có thể thấy rằng bố cục không lý tưởng cho màn hình máy tính bảng—văn bản của mỗi Cái nút quá nhỏ và sự sắp xếp của Cái nút các thành phần ở trên cùng và dưới cùng không lý tưởng cho máy tính bảng màn hình lớn.



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

1.Nhấp vào **Thiết kế** tab (nếu chưa được chọn) để hiển thị khung thiết kế và bản thiết kế.

2.Nhấp vào **Định hướng** trong Trình biên tập cá nút  ở thanh công cụ phía trên.

3.Chọn **Tạo bố cục x-lớn Biến thể**.

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

1.6 Thay đổi biến thể bố cục cho máy tính bảng

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

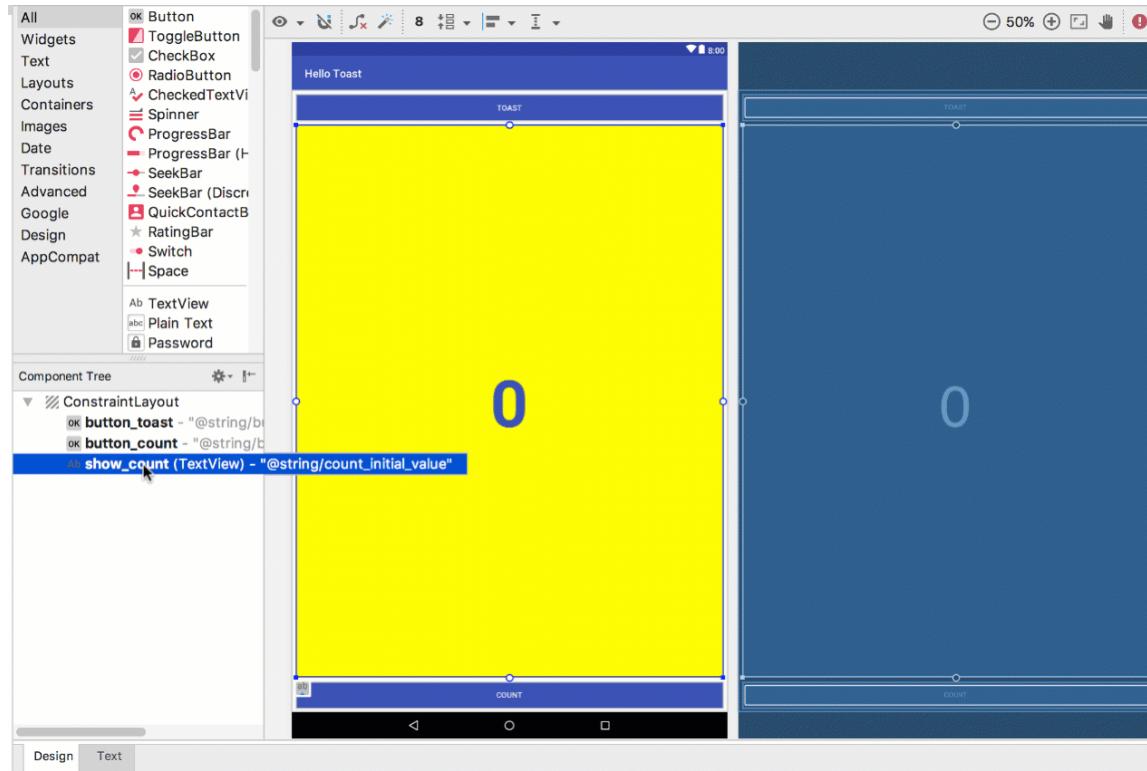
1.Tắt công cụ Autoconnect trên thanh công cụ. Đối với bước này, hãy đảm bảo rằng công cụ đã bị vô hiệu hóa:



2.Xóa tất cả các ràng buộc trong bố cục bằng cách nhập vào **Xóa tất cả ràng buộc** thanh công nút trong cụ.

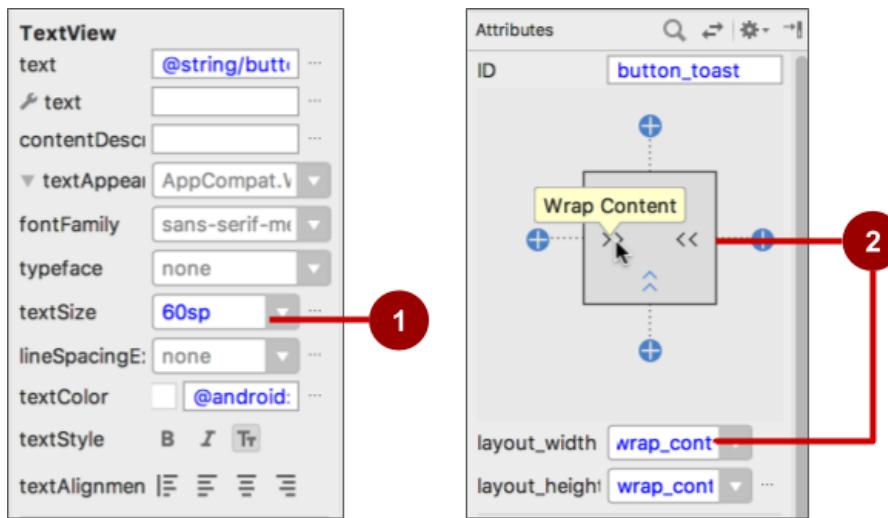
Khi các ràng buộc đã được loại bỏ, bạn có thể di chuyển và thay đổi kích thước các thành phần trên bố cục một cách tự do.

3.Trình chỉnh sửa bố cục cung cấp các nút điều chỉnh kích thước ở cả bốn góc của một phần tử để thay đổi kích thước của nó. Trong **Cây thành phần**, chọn Xem văn bản gọi điện hiển thị số lượng. Để có được Xem văn bản ra khỏi đường để bạn có thể tự do kéo Cái nút các thành phần, kéo một góc của nó để thay đổi kích thước, như thể hiện trong hình động bên dưới.



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

- 4.Chọn button_toast Nút trong **Cây thành phần**, nhấp vào **Thuộc tính** tab để mở **Thuộc tính** nhanh, và thay đổi Kích thước văn bản **ĐẾN 60sp**(#1 trong hình bên dưới) và chiều rộng bối cục **ĐẾN nội dung bọc**(#2 trong hình bên dưới).



Như được hiển thị ở phía bên phải của hình trên (2), bạn có thể nhấp vào điều khiển chiều rộng của thanh tra chế độ xem, xuất hiện ở hai phân đoạn ở phía bên trái và bên phải của hình vuông, cho đến khi nó hiển thị Gói nội dung. Ngoài ra, bạn có thể chọn **nội dung bọc** từ chiều rộng bố cục thực đơn.

Bạn sử dụng nội dung bọc vì vậy nếu Cái nút văn bản được bản địa hóa sang một ngôn ngữ khác, Cái nút sẽ trông rộng hơn hoặc mỏng hơn để phù hợp với từ trong ngôn ngữ khác nhau.

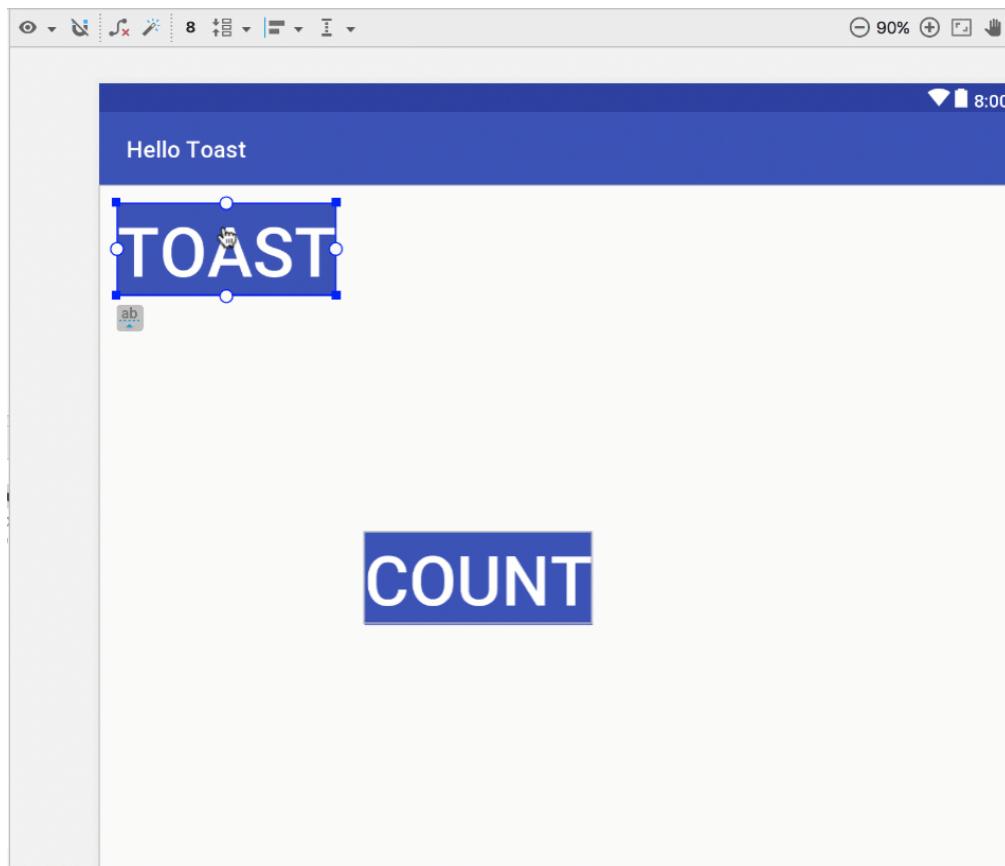
5.Chọn button_count Nút trong **Cây thành phần**, thay đổi Kích thước văn bản ĐẾN **60sp** và chiều rộng bố cục ĐẾN **nội dung bọc** và kéo Cái nút phía trên Xem văn bản vào một khoảng trống trong bố cục.

1.7 Sử dụng ràng buộc cơ sở

Bạn có thể căn chỉnh một phần tử UI có chứa văn bản, chẳng hạn như Xem văn bản hoặc Cái nút, với một thành phần UI khác có chứa văn bản. **Ràng buộc cơ sở** cho phép bạn giới hạn các thành phần sao cho đường cơ sở của văn bản khớp với nhau.

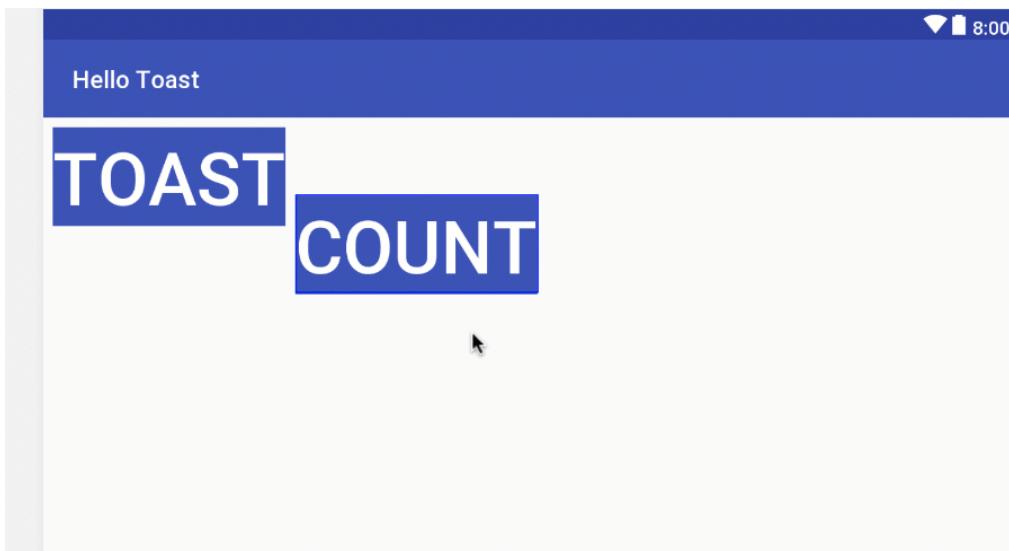
1.Hạn chế button_toast Nút lên cùng và bên trái của bố cục, kéo

button_count Nút đến một không gian gần button_toast Nút, và hạn chế button_count Nút ở phía bên trái của button_toast Nút, như thể hiện trong hình ảnh động:



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

3.Nhấp vào nút ràng buộc đường cơ sở. Tay cầm đường cơ sở xuất hiện, nhấp nháy màu xanh lá cây khi được hiển thị trong hình ảnh động. Nhấp và kéo đường ràng buộc đường cơ sở đến đường cơ sở của nút `bánh` ...yếu tố.

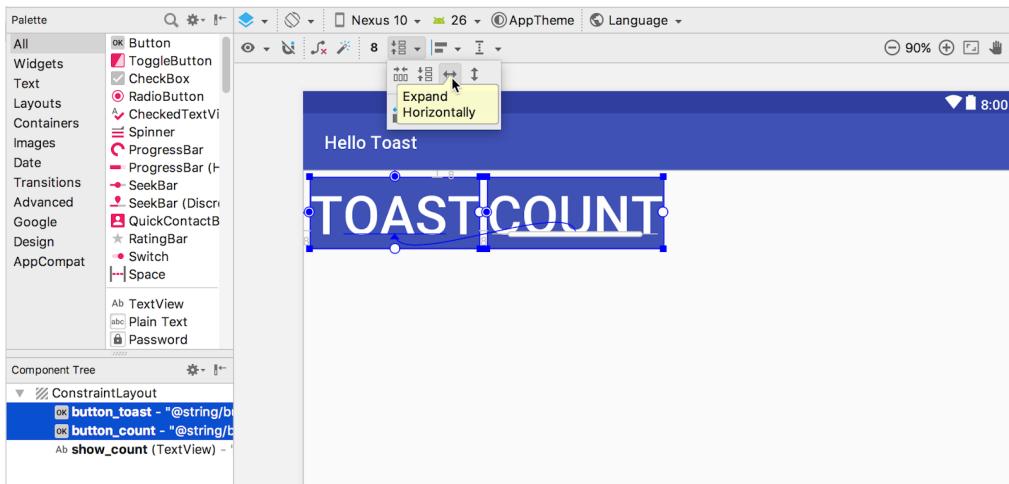


1.8 Mở rộng các nút theo chiều ngang

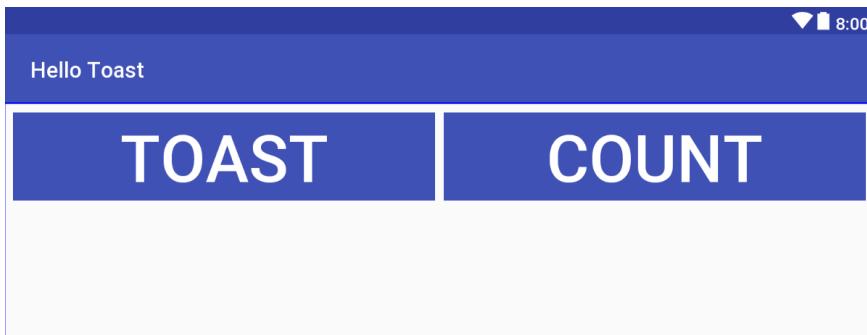
Nút gói  trên thanh công cụ cung cấp các tùy chọn để đóng gói hoặc mở rộng UI đã chọn các yếu tố. Bạn có thể sử dụng nó để sắp xếp đều Cái nút các yếu tố nằm ngang trên bố cục.

1.Chọnbutton_count Nút trong **Cây thành phần** và Shift-chọnbutton_toast Nút để cả hai đều được chọn.

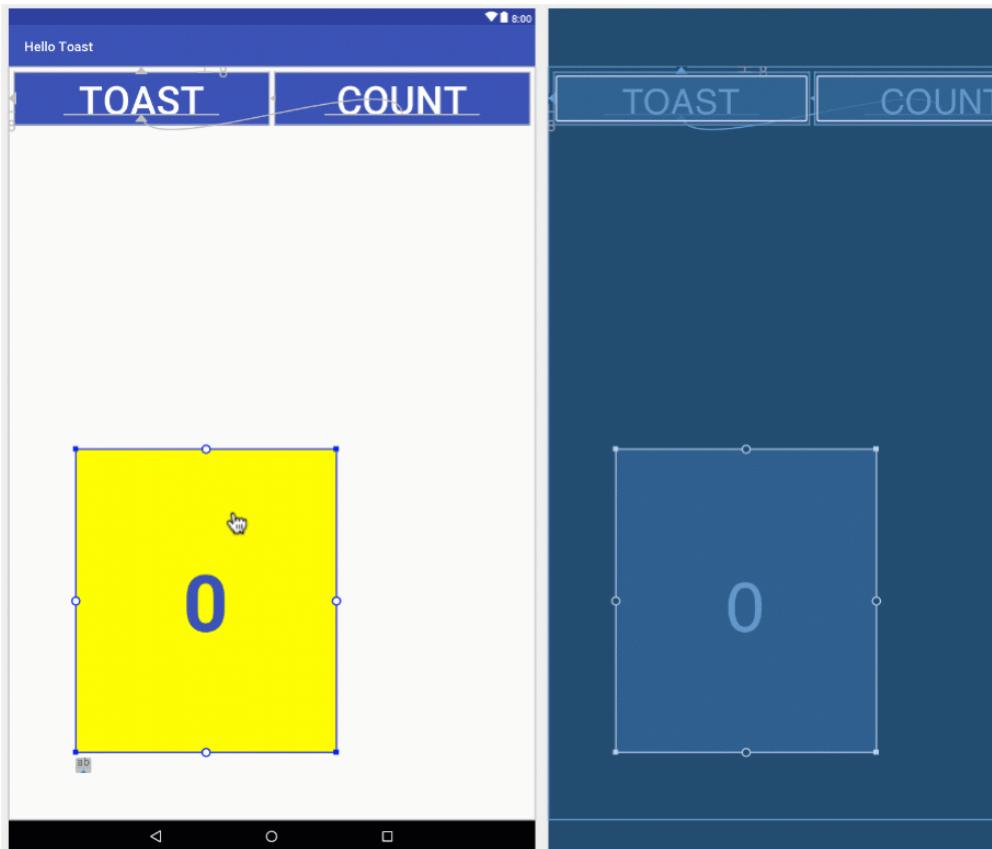
2.Nhấp vào nút gói ở hình  trên thanh công cụ và chọn **Mở rộng theo chiều ngang** như thể hiện trong bên dưới.



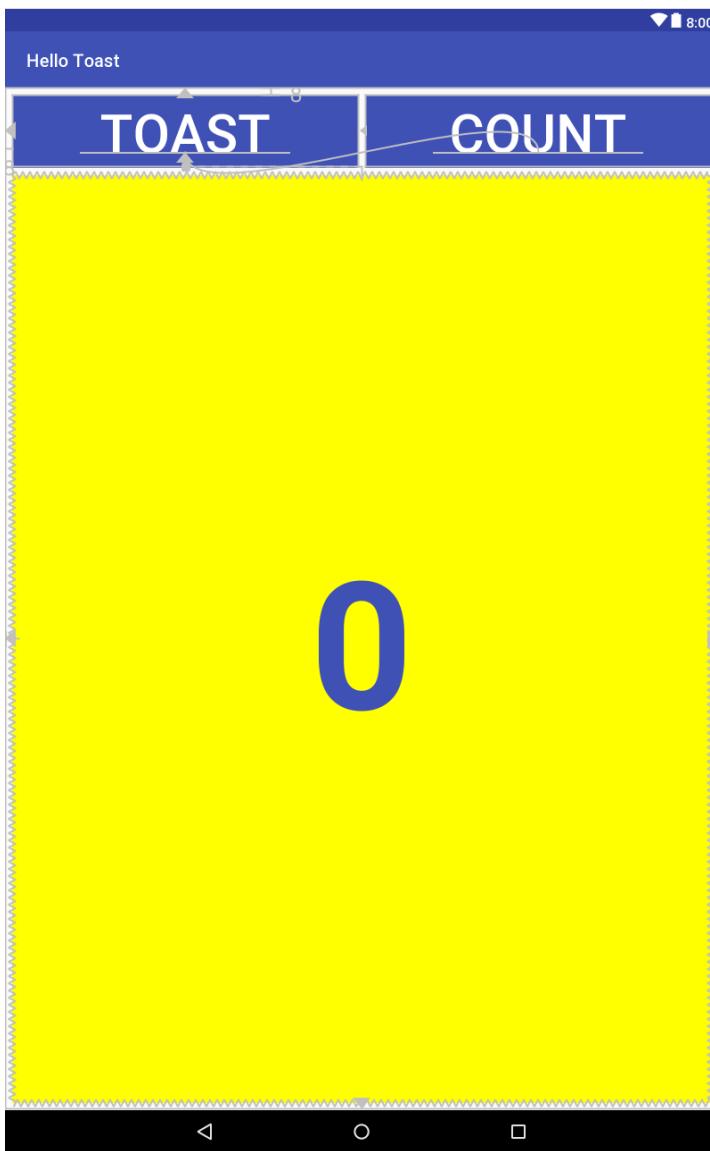
Các nút các thành phần mở rộng theo chiều ngang để lấp đầy bố cục như minh họa bên dưới.



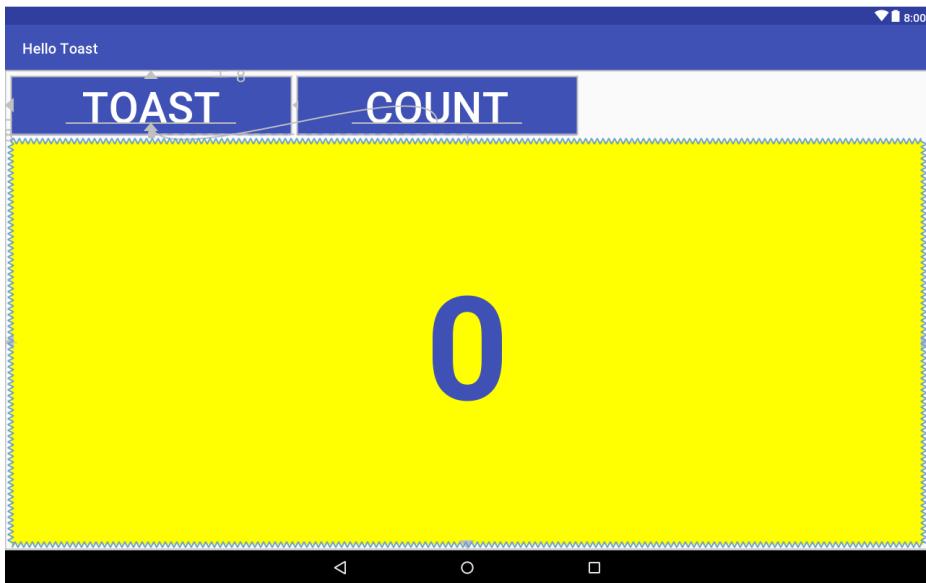
3. Để hoàn thiện bố cục, hãy ràng buộc `Hiển thị số văn bản` đến tận cùng của `button_toast`. Nút và các cạnh và đáy của bố cục, như thể hiện trong hình động bên dưới.



4.Các bước cuối cùng là thay đổi hiển thị số lượng TextView layout_width và chiều cao bối cục ĐẾN **Ràng buộc phù hợp** và Kích thước văn bản ĐẾN **200sp**. Bối cục cuối cùng trông giống như hình bên dưới.



5.Nhấp vào **Định hướng** trong **Trình biên tập** cái nút trong thanh công cụ trên cùng và chọn **Chuyển sang Phong cảnh**. Bố cục máy tính bảng xuất hiện theo hướng ngang như hiển thị bên dưới. (Bạn có thể chọn **Chuyển sang chế độ Chân dung** để trở về hướng dọc.).



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

Mẹo: Để có hướng dẫn chi tiết về cách sử dụngBối cục ràng buộc,nhìn thấySử dụng ConstraintLayout để thiết kế chế độ xem của bạn .

Mã giải bài tập 1

Dự án Android Studio:[Xin chàoToast](#)

Thử thách mã hóa 1

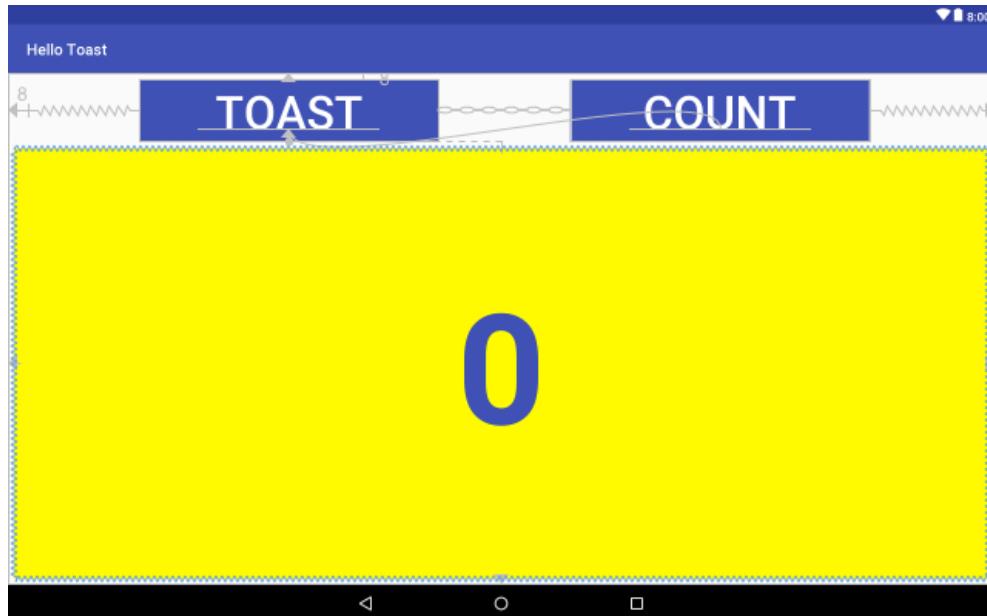
Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế . PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

Ghi chú:Mỗi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách: Để phù hợp với hướng ngang (phong cảnh) cho máy tính bảng, bạn có thể căn giữa các thành phần Nút trong activity_main.xml (cực lớn) để chúng xuất hiện như hình bên dưới.

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



Mã giải pháp thử thách 1

Dự án Android Studio:[Xin chào Toast Challenge 2](#)

Nhiệm vụ 2: Thay đổi bố cục thành LinearLayout

Bố cục tuyến tính là một `ViewGroup` sắp xếp bộ sưu tập các chế độ xem của nó theo hàng ngang hoặc hàng dọc. A Bố cục tuyến tính là một trong những bố cục phổ biến nhất vì nó đơn giản và nhanh chóng. Nó thường được sử dụng trong một nhóm chế độ xem khác để sắp xếp các thành phần UI theo chiều ngang hoặc chiều dọc.

MỘT Bố cục tuyến tính cần phải có những thuộc tính sau:

- chiều rộng bố cục
- chiều cao bố cục
- định hướng

Các chiều rộng bố cục và chiều cao bố cục có thể lấy một trong các giá trị sau:

- khorp_phu_tu: Mở rộng chế độ xem để lấp đầy phần cha mẹ theo chiều rộng hoặc chiều cao. Khi Bố cục tuyến tính là chế độ xem gốc, nó mở rộng theo kích thước của màn hình (chế độ xem cha).
- nội dung bọc: Thu nhỏ kích thước chế độ xem để chế độ xem chỉ đủ lớn để bao gồm nội dung của nó. Nếu không có nội dung, chế độ xem sẽ trở nên vô hình.
- Số lượng cố định của dp ([pixel không phụ thuộc vào mật độ](#)): Chỉ định kích thước cố định, được điều chỉnh theo mật độ màn hình của thiết bị. Ví dụ: 16dp có nghĩa là 16 pixel không phụ thuộc vào mật độ.

Các định hướng có thể là:

- nằm ngang: Các góc nhọn được sắp xếp từ trái sang phải.
- thẳng đứng: Các góc nhọn được sắp xếp từ trên xuống dưới.

Trong nhiệm vụ này bạn sẽ thay đổi Bố cục ràng buộc nhóm xem gốc cho ứng dụng Hello Toast để Bố cục tuyến tính để bạn có thể thực hành sử dụng Bố cục tuyến tính.

2.1 Thay đổi nhóm chế độ xem gốc thành LinearLayout

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở `activity_main.xml` (tập tin bố trí) (nếu nó chưa được mở) và nhập vào **Chữ tab** tại dưới cùng của ngăn chính sửa để xem mã XML. Ở đầu cùng của mã XML là dòng thẻ sau:

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

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

3.Thay đổi <android.support.constraint.ConstraintLayout>năm thẻ vào <**Bố cục tuyến tính**> để mã trông như thế này:

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

4.Đảm bảo thẻ đóng ở cuối mã đã được thay đổi thành </Bố cục tuyến tính> (Android Studio sẽ tự động thay đổi thẻ đóng nếu bạn thay đổi thẻ mở). Nếu thẻ chưa tự động thay đổi, hãy thay đổi thủ công.

5.Dưới <Bố cục tuyến tính>dòng thẻ, thêm thuộc tính sau vào sau android:layout_heightthuộc tính:

```
    android:orientation="dọc"
```

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 Bố cục ràng buộc và không liên quan đến Bố cục tuyến tính.

2.2 Thay đổi thuộc tính phần tử cho LinearLayout

Thực hiện theo các bước sau để thay đổi các thuộc tính của phần tử UI để chúng hoạt động với Bố cục tuyến tính:

- 1.Mở ứng dụng Hello Toast từ tác vụ trước đó.
- 2.Mở **hoạt động_main.xml**tập tin bố trí (nếu nó chưa được mở) và nhấp vào **Chữ tab**.
- 3.Tìm thấy **button_toast** Nút phần tử và thay đổi thuộc tính sau:

Nguyên bản	Thay đổi thành
------------	----------------

android:layout_width="0dp"	android:layout_width="phù hợp với cha mẹ"
----------------------------	---

4.Xóa các thuộc tính sau khỏi nút bánh ...yếu tố:

ứng dụng:layout_constraintEnd_toEndOf="cha mẹ" ứng
dụng:layout_constraintStart_toStartOf="cha mẹ" ứng
dụng:layout_constraintTop_toTopOf="cha mẹ"

5.Tìm thấy button_count Nút phàn tử và thay đổi thuộc tính sau:

Nguyên bản	Thay đổi thành
android:layout_width="0dp"	android:layout_width="phù hợp với cha mẹ"

6.Xóa các thuộc tính sau khỏi nút số lượng yếu tố:

ứng dụng:layout_constraintBottom_toBottomOf="cha mẹ"
ứng dụng:layout_constraintEnd_toEndOf="cha mẹ" ứng
dụng:layout_constraintStart_toStartOf="cha mẹ"

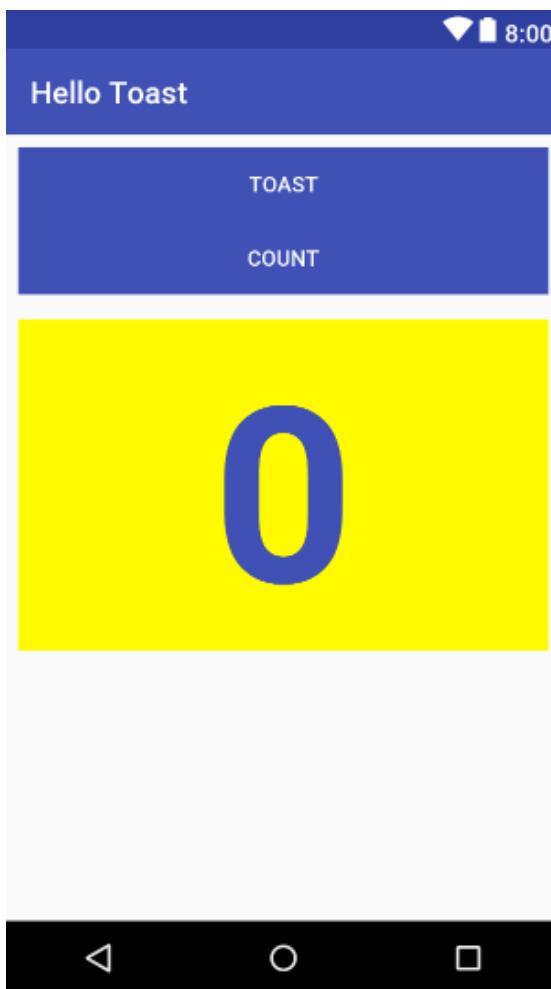
7.Tìm thấy Hiển thị số văn bản phàn tử và thay đổi các thuộc tính sau:

Nguyên bản	Thay đổi thành
android:layout_width="0dp"	android:layout_width="phù hợp với cha mẹ"
android:layout_width="0dp"	android:layout_height="wrap_content"

8.Xóa các thuộc tính sau khỏi hiển thị_số lượng yêu tố:

```
ứng dụng:layout_constraintBottom_toTopOf="@+id/button_count"  
ứng dụng:layout_constraintEnd_toEndOf="parent"  
ứng dụng:layout_constraintStart_toStartOf="cha mẹ" ứng  
dụng:layout_constraintTop_toBottomOf="@+id/button_toast"
```

9.Nhấp vào **Xem trước** tab ở bên phải cửa sổ Android Studio (nếu chưa được chọn) để xem bản xem trước của bố cục cho đến thời điểm này:



2.3 Thay đổi vị trí của các phần tử trong LinearLayout

Bố cục tuyến tính sắp xếp các thành phần của nó theo hàng ngang hoặc hàng dọc. Bạn đã thêm `android:orientation="dọc"` thuộc tính cho `LinearLayout`, do đó các phần tử được xếp chồng lên nhau theo chiều dọc như thể hiện trong hình trước.

Để thay đổi vị trí của họ sao cho **Đếm** nút ở phía dưới, hãy làm theo các bước sau:

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở `activity_main.xml` tệp tin bố trí (nếu nó chưa được mở) và nhấp vào **Chữ tab**.

- 3.Chọnbutton_count Nút và tất cả các thuộc tính của nó, từ <Cái nút> gắn thẻ lên đến và bao gồm thẻ đóng /> và chọn**Chỉnh sửa > Cắt**.
- 4.Nhấp vào sau thẻ đóng /> của Xem văn bản phàn tử nhưng trước khi đóng </LinearLayout> gắn thẻ và chọn**Sửa > Dán**.
5. (Tùy chọn) Để sửa bất kỳ lỗm hoặc vấn đề khoảng cách nào vì mục đích thẩm mỹ, hãy chọn**Mã > Định dạng lại mã** để định dạng lại mã XML với khoảng cách và thụt lề thích hợp.

Mã XML cho các thành phần UI hiện trông như sau:

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

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

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

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

Bằng cách di chuyển button_count Nút bên dưới Chế độ xem bản bối cục bây giờ gần giống với những gì bạn đã có trước đó, với **Đếm** nút ở phía dưới. Bản xem trước của bối cục bây giờ trông như sau:



2.4 Thêm trọng lượng cho phần tử TextView

Chỉ định trọng lực và cân nặng các thuộc tính cung cấp cho bạn quyền kiểm soát bổ sung đối với việc sắp xếp chế độ xem và nội dung trong một bố cục tuyến tính.

Các android:trọng lực thuộc tính chỉ định sự căn chỉnh của nội dung của một Xem trong vòng Xem chính nó. Trong bài học trước bạn đã thiết lập thuộc tính này cho `Hiển thị_số_văn_bản` để căn giữa nội dung (chữ số 0) vào giữa Văn bản Xem:

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

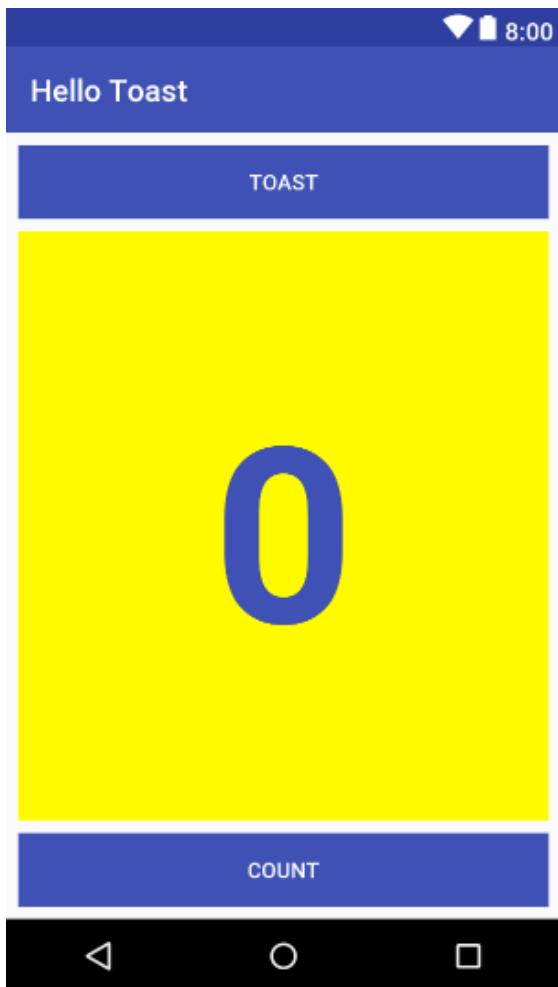
Các android:trọng lượng bổ cục thuộc tính cho biết có bao nhiêu không gian bổ sung trong Bố cục tuyến tính sẽ được phân bổ cho Xem. Nếu chỉ có một Xem có thuộc tính này, nó sẽ có được tất cả không gian màn hình bổ sung. Đối với nhiều Xem các phần tử, không gian được chia theo tỷ lệ. Ví dụ, nếu Cái nút mỗi phần tử có trọng số là 1 và Xem văn bản 2, tổng cộng là 4, Cái nút các phần tử nhận được $\frac{1}{4}$ không gian mỗi phần tử và Xem văn bản một nửa.

Trên các thiết bị khác nhau, bố cục có thể hiển thị `Hiển thị_số_văn_bản` phần tử lấp đầy một phần hoặc hầu hết khoảng trống giữa **Nước** và **Đêm** nút. Để mở rộng Xem văn bản để lấp đầy không gian có sẵn bất kể thiết bị nào được sử dụng, hãy chỉ định android:trọng lực thuộc tính cho Chế độ xem văn bản. Thực hiện theo các bước sau:

1. Mở ứng dụng Hello Toast từ tác vụ trước đó.
2. Mở `activity_main.xml` tập tin bố trí (nếu nó chưa được mở) và nhấp vào **Chữ tab**.
3. Tìm thấy `Hiển thị_số_văn_bản` phần tử và thêm thuộc tính sau:

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

Bản xem trước bây giờ trông giống như hình sau.



Các **Hiển thị số văn bản** phần tử chiếm hết không gian giữa các nút. Bạn có thể xem trước bố cục cho các thiết bị khác nhau, như bạn đã làm trong tác vụ trước đó bằng cách nhấp vào **Thiết bị trong Trình chỉnh sửa** cái nút

Nexus 5 trong thanh công cụ trên cùng của ngăn xem trước và chọn một thiết bị khác. Không quan trọng bạn chọn thiết bị nào để xem trước, **Hiển thị số văn bản** phần tử này phải chiếm hết khoảng trống giữa các nút.

Mã giải bài tập 2

Mã XML trong activity_main.xml:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ứng dụng="http://schemas.android.com/apk/res-auto"
    xmlns:công cụ="http://schemas.android.com/tools" android:bố
    cục_chiều rộng="phù hợp với cha mẹ"
    android:layout_height="match_parent" android:orientation="vertical" công
    cụ:context="com.example.android.hellotoast.MainActivity">

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

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

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

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

Nhiệm vụ 3: Thay đổi bố cục thành RelativeLayout

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

3.1 Đổi LinearLayout thành RelativeLayout

Một cách dễ dàng để thay đổi Bố cục tuyến tính đến một Bố cục tương đối là thêm các thuộc tính XML vào **Chữ tab**.

1. Mở hoạt động_main.xml tập tin bối trí và nhập vào **Chữ tab** ở cuối ngăn chỉnh sửa để xem mã XML.

2. Thay đổi <Bố cục tuyến tính> đầu đến <**Bố cục tương đối**> để câu lệnh trông như thế này:

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

3. Cuộn xuống để đảm bảo rằng thẻ kết thúc </Bố cục tuyến tính> cũng đã thay đổi thành </RelativeLayout>; nếu chưa, hãy thay đổi thủ công.

```

super.onDestroy();
Nhật ký.d(LOG_TAG, "onDestroy");
}

```

Hoạt động thứ hai

Các đoạn mã sau đây hiển thị mã đã thêm vào Hoạt động thứ hai, nhưng không phải toàn bộ lớp.

Ở phía trên cùng của Hoạt động thứ hai lớp học:

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

Các trả về Trả lời() phương pháp:

```

public void returnReply(Xem chế độ xem) {
    Chuỗi trả lời = mReply.getText().toString(); Ý định
    replyIntent = new Intent();
    replyIntent.putExtra(EXTRA_REPLY, trả lời);
    setResult(RESULT_OK, replyIntent); Log.d(LOG_TAG,
    "Kết thúc SecondActivity"); kết thúc();
}

```

Các phương pháp vòng đời khác:

Giống như đối với Hoạt động chính, bên trên.

Nhiệm vụ 2: Lưu và khôi phục trạng thái phiên bản Activity

Tùy thuộc vào tài nguyên hệ thống và hành vi của người dùng, mỗi hoạt động trong ứng dụng của bạn có thể bị phá hủy và tái tạo thường xuyên hơn bạn nghĩ.

Bạn có thể đã nhận thấy hành vi này trong phần cuối cùng khi bạn xoay thiết bị hoặc trình giả lập. Xoay thiết bị là một ví dụ về thiết bị *thay đổi cấu hình*. Mặc dù xoay là cách phổ biến nhất, nhưng tất cả các thay đổi cấu hình đều dẫn đến hiện tại *Hoạt động* bị phá hủy và tái tạo như thế nào mới. Nếu bạn không tính đến hành vi này trong mã của mình, khi thay đổi cấu hình xảy ra, *Hoạt động* Bổ cục có thể trở lại giao diện mặc định và các giá trị ban đầu, và người dùng của bạn có thể mất vị trí, dữ liệu hoặc trạng thái tiến trình của họ trong ứng dụng.

Trạng thái của mỗi hoạt động được lưu trữ dưới dạng một tập hợp các cặp khóa/giá trị trong một *Bộ* đối tượng được gọi là *Hoạt động trạng thái ví dụ*. Hệ thống lưu thông tin trạng thái mặc định vào trạng thái thể hiện *Bóng* gay trước khi *Hoạt động* đã dừng lại và vượt qua *Bộ* đến cái mới *Hoạt động* trường hợp cần khôi phục.

Để tránh mất dữ liệu trong một *Hoạt động* khi nó bị phá hủy và tái tạo bất ngờ, bạn cần phải thực hiện *onSaveInstanceState()* phương pháp. Hệ thống gọi phương pháp này trên *Hoạt động* (giữa tạm dừng() và khi dừng()) khi có khả năng *Hoạt động* có thể bị phá hủy và tái tạo.

Dữ liệu bạn lưu trong trạng thái thể hiện chỉ dành riêng cho thể hiện cụ thể này *Hoạt động* trong phiên ứng dụng hiện tại. Khi bạn dừng và khởi động lại phiên ứng dụng mới, *Hoạt động* trạng thái thể hiện bị mất và *Hoạt động* trở lại giao diện mặc định. Nếu bạn cần lưu dữ liệu người dùng giữa các phiên ứng dụng, hãy sử dụng tùy chọn chia sẻ hoặc cơ sở dữ liệu. Bạn sẽ tìm hiểu về cả hai điều này trong phần thực hành sau.

2.1 Lưu trạng thái thể hiện của Activity bằng *onSaveInstanceState()*

Bạn có thể nhận thấy rằng việc xoay thiết bị không ảnh hưởng đến trạng thái của giây *Hoạt động* hoàn toàn. Điều này là do thứ hai *Hoạt động* bổ cục và trạng thái được tạo ra từ bối cảnh và Ý định đã kích hoạt nó. Ngay cả khi *Hoạt động* được tái tạo, Ý định vẫn còn đó và dữ liệu trong đó Ý định vẫn được sử dụng mỗi lần *khi tạo()* phương pháp thứ hai *Hoạt động* được gọi là.

Ngoài ra, bạn có thể nhận thấy rằng trong mỗi Hoạt động, bất kỳ văn bản nào bạn nhập vào tin nhắn hoặc trả lời Sửa văn bản các thành phần được giữ lại ngay cả khi thiết bị được xoay. Điều này là do thông tin trạng thái của một số Xem các thành phần trong bố cục của bạn được tự động lưu qua các thay đổi cấu hình và giá trị hiện tại của Sửa văn bản là một trong những trường hợp đó.

Vì vậy, duy nhất Hoạt động tiêu bang bạn quan tâm là Xem văn bản các yếu tố cho tiêu đề trả lời và văn bản trả lời trong phần chính Hoạt động. Cả hai Xem văn bản các phần tử mặc định là vô hình; chúng chỉ xuất hiện khi bạn gửi tin nhắn trở lại trang chính Hoạt động từ thứ hai Hoạt động.

Trong nhiệm vụ này, bạn thêm mã để bảo toàn trạng thái hiển thị của hai Xem văn bản các yếu tố sử dụng khi Lưu Trạng Thái Phiên Bản().

1. Mở Hoạt động chính.

2. Thêm triển khai bộ xương này onSaveInstanceState() đến Hoạt động, hoặc sử dụng **Mã số > Ghi đè phương pháp** để chèn ghi đè bộ xương.

```
@Ghi đè  
public void onSaveInstanceState(Gói outState) {  
    super.onSaveInstanceState(outState);  
}
```

3. Kiểm tra xem tiêu đề hiện có hiển thị hay không và nếu có thì hãy đưa trạng thái hiển thị đó vào trạng thái Bó với đặt Boolean() phương pháp và chìa khóa "trả lời hiển thị".

```
nếu (mReplyHeadTextView.getVisibility() == View.VISIBLE) {  
    outState.putBoolean("reply_visible", đúng);  
}
```

Hãy nhớ rằng tiêu đề và văn bản trả lời được đánh dấu là vô hình cho đến khi có phản hồi từ người thứ hai Hoạt động. Nếu tiêu đề hiển thị, thì có dữ liệu trả lời cần được lưu. Lưu ý rằng chúng tôi chỉ quan tâm đến trạng thái hiển thị đó — văn bản thực tế của tiêu đề không cần phải được lưu, vì văn bản đó không bao giờ thay đổi.

4. Bên trong cùng một kiểm tra đó, thêm văn bản trả lời vào Bó.

```
outState.putString("văn bản trả lời",mReplyTextView.getText().toString());
```

Nếu tiêu đề hiển thị, bạn có thể cho rằng bản thân tin nhắn trả lời cũng hiển thị. Bạn không cần phải kiểm tra hoặc lưu trạng thái hiển thị hiện tại của tin nhắn trả lời. Chỉ có văn bản thực tế của tin nhắn mới được đưa vào trạng tháiBówi chìa khóa ""văn_bản_trả_lời".

Bạn chỉ lưu trạng thái của nhữngXemcác yếu tố có thể thay đổi sau khiHoạt độngđược tạo ra. Cái kiaXemcác thành phần trong ứng dụng của bạn (cácChỉnh sửa văn bản,cáiCái nút)có thể được tạo lại từ bối cục mặc định bất cứ lúc nào.

Lưu ý rằng hệ thống sẽ lưu trạng thái của một sốXemcác yếu tố, chẳng hạn như nội dung của Chỉnh sửa văn bản.

2.2 Khôi phục trạng thái thể hiện Activity trong onCreate()

Một khi bạn đã lưuHoạt độngtrạng thái thể hiện, bạn cũng cần khôi phục nó khiHoạt độngđược tạo lại. Bạn có thể làm điều này trongkhi tạo(),hoặc bằng cách thực hiện onRestoreInstanceState()gọi lại, được gọi sau khiBắt đầu()sau khiHoạt độngđược tạo ra.

Hầu hết thời gian là nơi tốt hơn để khôi phục lạiHoạt độngtrạng thái đang ởkhi tạo(),để đảm bảo rằng UI, bao gồm cả trạng thái, có sẵn sớm nhất có thể. Đôi khi thuận tiện để thực hiện điều đó trong onRestoreInstanceState()sau khi tất cả quá trình khởi tạo đã được thực hiện hoặc để cho phép các lớp con quyết định có sử dụng triển khai mặc định của bạn hay không.

- Trongkhi tạo()phương pháp, sau khiXemcác biến được khởi tạo với`findViewById()`,thêm một bài kiểm tra để đảm bảo rằngTrạng thái đã lưukhông phải là null.

```
// Khởi tạo tất cả các biến view. mMessageEditText  
=           tìmViewById(R.id.editText_main);  
mReplyHeadTextXem = findViewById(R.id.text_header_reply);  
mReplyTextView = findViewById(R.id.text_message_reply);  
  
// Khôi phục trạng thái.  
nếu (savedInstanceState != null) { }
```

Khi bạn Hoạt động được tạo ra, hệ thống chuyển trạng thái BỎ ĐỀN khi tạo() như là lý lẽ duy nhất của nó. Lần đầu tiên khi tạo() được gọi và ứng dụng của bạn bắt đầu, BỎ ĐỀN không có trạng thái hiện hữu nào khi ứng dụng của bạn khởi động lần đầu tiên. Các cuộc gọi tiếp theo đến khi tạo() có một bó dữ liệu được lưu trữ trong đó khi Lưu Trạng Thái Phiên Bản().

2. Bên trong kiểm tra đó, lấy khả năng hiển thị hiện tại (đúng hoặc sai) ra khỏi BỎ ĐỀN với chìa khóa "reply_visible".

```
nếu (savedInstanceState != null) {  
    boolean isVisible =  
        savedInstanceState.getBoolean("reply_visible");  
}
```

3. Thêm một bài kiểm tra bên dưới dòng trước đó cho biến isVisible.

```
nếu (isVisible) {  
}
```

Nếu có một trả lời hiển thị chìa khóa trong trạng thái BỎ ĐỀN (Và có thể nhìn thấy do đó là ĐÚNG VẬY), bạn sẽ cần khôi phục lại trạng thái.

4. Bên trong có thể nhìn thấy kiểm tra, làm cho tiêu đề hiển thị.

```
mReplyHeadTextView.setVisibility(View.VISIBLE);
```

5. Nhận tin nhắn trả lời văn bản từ BỎ ĐỀN với chìa khóa "trả lời_văn_bản", và thiết lập trả lời Xem văn bản để hiển thị chuỗi đó.

```
mReplyTextView.setText(savedInstanceState.getString("reply_text"));
```

6.Trả lờiXem văn bản cũngh có thể nhìn thấy:

```
mReplyTextView.setVisibility(View.VISIBLE);
```

7.Chạy ứng dụng. Thử xoay thiết bị hoặc trình giả lập để đảm bảo rằng tin nhắn trả lời (nếu có) vẫn hiển thị trên màn hình sau khi Hoạt độngđược tái tạo.

Mã giải bài tập 2

Các đoạn mã sau đây hiển thị mã giải pháp cho nhiệm vụ này.

Hoạt động chính

Các đoạn mã sau đây hiển thị mã đã thêm vào Hoạt động chính,nhưng không phải toàn bộ lớp.

CáconSaveInstanceState()phương pháp:

```
@Ghi đè
public void onSaveInstanceState(Gói outState) {
    super.onSaveInstanceState(outState);
    // Nếu tiêu đề hiển thị, tin nhắn cần được lưu. // Nếu không, chúng ta vẫn sử
    // dụng bố cục mặc định.
    nếu (mReplyHeadTextView.getVisibility() == View.VISIBLE) {
        outState.putBoolean("reply_visible", đúng);
        outState.putString("reply_text",
                           mReplyTextView.getText().toString());
    }
}
```

```
}
```

Các khi tạo() phương pháp:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    Nhật ký.d(LOG_TAG, "");  
    Nhật ký.d(LOG_TAG, "khi tạo");  
  
    // Khởi tạo tất cả các biến view. mMessageEditText  
    =     tìmViewById(R.id.editText_main);  
    mReplyHeadTextXem     = findViewById(R.id.text_header_reply);  
    mReplyTextView = findViewById(R.id.text_message_reply);  
  
    // Khôi phục trạng thái đã lưu.  
    // Xem onSaveInstanceState() để biết những gì được lưu. if  
(savedInstanceState != null) {  
        boolean isVisible =  
            savedInstanceState.getBoolean("reply_visible");  
        // Hiển thị cả chế độ xem tiêu đề và tin nhắn. Nếu isVisible là // sai hoặc bị thiếu trong  
        // gói, hãy sử dụng bối cục mặc định. if (isVisible) {  
  
            mReplyHeadTextView.setVisibility(View.VISIBLE);  
            mReplyTextView.setText(savedInstanceState  
                .getString("văn bản trả lời"));  
            mReplyTextView.setVisibility(View.VISIBLE);  
        }  
    }  
}
```

Dự án hoàn chỉnh:

Dự án Android Studio:[HaiHoạt ĐộngVòng Đời](#)

Thử thách mã hóa

Ghi chú:Mọi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách:Tạo một ứng dụng danh sách mua sắm đơn giản với hoạt động chính cho danh sách mà người dùng đang xây dựng và hoạt động thứ hai cho danh sách các mặt hàng mua sắm thông thường.

- Hoạt động chính phải bao gồm danh sách cần xây dựng, danh sách này phải bao gồm mười ô trống. Xem văn bản các yếu tố.
- **MỘT** **Thêm mục** nút trên hoạt động chính sẽ khởi chạy hoạt động thứ hai có chứa danh sách các mặt hàng mua sắm thông thường (**Phô mai, Cơm, Táo**, v.v.). Sử dụng Cái nút các thành phần để hiển thị các mục.
- Việc chọn một mục sẽ đưa người dùng trở lại hoạt động chính và cập nhật một mục trốngXem văn bản để bao gồm mục đã chọn.

Sử dụng một **Ý định** để truyền thông tin từ một **Hoạt động** sang thiết bị khác. Đảm bảo trạng thái hiện tại của danh sách mua sắm được lưu khi người dùng xoay thiết bị.

Bản tóm tắt

- Vòng đời hoạt động là một tập hợp các trạng thái Hoạt động di chuyển qua, bắt đầu khi nó được tạo lần đầu tiên và kết thúc khi hệ thống Android lấy lại tài nguyên cho nó Hoạt động.
- Khi người dùng điều hướng từ một Hoạt động đến một cái khác, và bên trong và bên ngoài ứng dụng của bạn, mỗi Hoạt động di chuyển giữa các tiểu bang trong Hoạt động vòng đời.
- Mỗi tiểu bang trong Hoạt động vòng đời có một phương thức gọi lại tương ứng mà bạn có thể ghi đè trong Hoạt động lớp học.
- Các phương pháp vòng đời là `onCreate()`, `onStart()`, `onPause()`, `onRestart()`, `onResume()`, `onStop()`, `onDestroy()`.
- Ghi đè phương thức gọi lại vòng đời cho phép bạn thêm hành vi xảy ra khi Hoạt động chuyển đổi sang trạng thái đó.
- Bạn có thể thêm các phương thức ghi đè bộ xương vào các lớp của mình trong Android Studio bằng **Mã > Ghi đè**.

- Những thay đổi về cấu hình thiết bị như kết quả quay vòng trong Hoạt động bị phá hủy và được tái tạo như mới.
- Một phần của Hoạt động trạng thái được bảo toàn khi thay đổi cấu hình, bao gồm các giá trị hiện tại của Sửa văn bản phần tử. Đối với tất cả dữ liệu khác, bạn phải tự lưu dữ liệu đó một cách rõ ràng.
- Cứu Hoạt động trạng thái hiện trong onSaveInstanceState() phương pháp.
- Dữ liệu trạng thái hiện được lưu trữ dưới dạng cặp khóa/giá trị đơn giản trong Bó. Sử dụng Bó phương pháp đưa dữ liệu vào và lấy dữ liệu ra khỏi Bó.
- Khôi phục trạng thái hiện trong khi tạo(), đó là cách được ưa thích, hoặc khi Khôi phục Trạng thái Phiên bản().

Khái niệm liên quan

Tài liệu khái niệm liên quan có trong [2.2: Vòng đời và trạng thái hoạt động](#).

Tìm hiểu thêm

Tài liệu Android Studio:

- [Làm quen với Android Studio](#)

Tài liệu dành cho nhà phát triển Android:

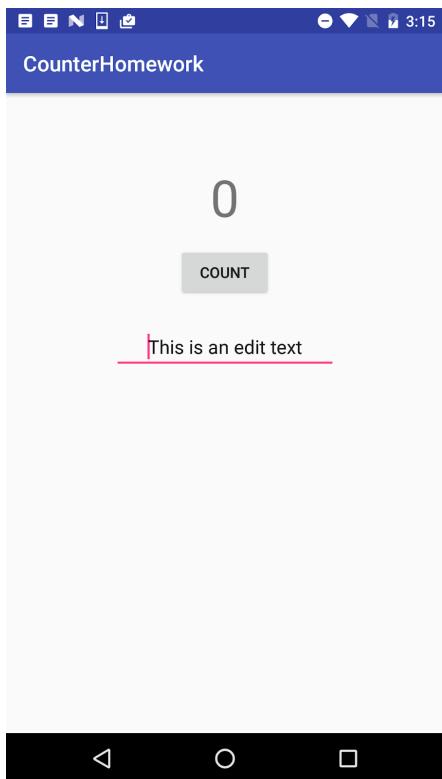
- [Cơ sở ứng dụng](#)
- [Các hoạt động](#)
- [Hiểu về Vòng đời hoạt động](#)
- [Ý định và Bộ lọc ý định](#)
- [Xử lý các thay đổi cấu hình](#)
- [Hoạt động](#)
- [Ý định](#)

Bài tập về nhà

Xây dựng và chạy một ứng dụng

1. Tạo một ứng dụng có bố cục chứa bộ đếm Chế độ xem văn bản, Một nút để tăng bộ đếm và một chỉnh sửa văn bản. Xem ảnh chụp màn hình bên dưới làm ví dụ. Bạn không cần phải sao chép chính xác bố cục.
2. Thêm trình xử lý nhấp chuột cho cái nút điều đó làm tăng bộ đếm.

3. Chạy ứng dụng và tăng bộ đếm. Nhập một số văn bản vào `Chỉnh sửa văn bản`.
4. Xoay thiết bị. Lưu ý rằng bộ đếm được thiết lập lại, nhưng `Sửa văn bản` không phải là.
5. Thực hiện `onSaveInstanceState()` để lưu trạng thái hiện tại của ứng dụng.
6. Cập nhật khi tạo() để khôi phục trạng thái của ứng dụng.
7. Đảm bảo rằng khi bạn xoay thiết bị, trạng thái ứng dụng được giữ nguyên.



Trả lời những câu hỏi này

Câu hỏi 1

Nếu bạn chạy ứng dụng bài tập về nhà trước khi triển khai `onSaveInstanceState()`, điều gì xảy ra nếu bạn xoay thiết bị? Chọn một:

- Các `Sửa văn bản` không còn chứa văn bản bạn đã nhập, nhưng bộ đếm vẫn được giữ nguyên.
- Bộ đếm được thiết lập lại về 0 và `Sửa văn bản` không còn chứa văn bản bạn đã nhập.
- Bộ đếm được thiết lập lại về 0, nhưng nội dung của `Sửa văn bản` được bảo tồn.
- Quầy và nội dung của `Sửa văn bản` được bảo tồn.

Câu hỏi 2

Cái gì Hoạt động phương pháp vòng đời được gọi khi có sự thay đổi cấu hình thiết bị (chẳng hạn như xoay) xảy ra? Chọn một:

- Android ngay lập tức tắt máy của bạn Hoạt động bằng cách gọi `khi dừng()`. Mã của bạn phải khởi động lại Hoạt động.
- Android tắt máy của bạn Hoạt động bằng cách gọi `tạm dừng()`, `dừng()`. Và khi Hủy Bỏ(). Mã của bạn phải khởi động lại Hoạt động.
- Android tắt máy của bạn Hoạt động bằng cách gọi `tạm dừng()`, `dừng()`. Và khi Hủy BỎ(), và sau đó bắt đầu lại, gọi `khi Tạo()`, khi `Bắt đầu()`, Và khi tiếp tục().
- Android ngay lập tức gọi `khi tiếp tục()`.

Câu hỏi 3

Khi ở trong Hoạt động vòng đời là `onSaveInstanceState()` được gọi là? Chọn một:

- `onSaveInstanceState()` được gọi trước `khi Dừng()` phương pháp.
- `onSaveInstanceState()` được gọi trước `onResume()` phương pháp.
- `onSaveInstanceState()` được gọi trước `khi tạo()` phương pháp.
- `onSaveInstanceState()` được gọi trước `khi Hủy BỎ()` phương pháp.

Câu hỏi 4

Cái mà Hoạt động phương pháp vòng đời là tốt nhất để sử dụng để lưu trữ dữ liệu trước khi Hoạt động đã hoàn thành hay bị phá hủy? Chọn một:

- tạm dừng() hoặc `khi Dừng()`
- `onResume()` hoặc `khi tạo()`
- `khi Hủy BỎ()`
- `khi Bắt đầu()` hoặc `khi khởi động lại()`

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Kiểm tra xem ứng dụng có các tính năng sau không:

- Nó hiển thị một bộ đếm, một nút để tăng bộ đếm đó và một chỉnh sửa văn bản.
- Nhấp vào nút tăng bộ đếm lên 1.
- Khi thiết bị được xoay, cả bộ đếm và sửa văn bản các trạng thái được giữ lại.
- Việc thực hiện `MainActivity.java` sử dụng `onSaveInstanceState()` phương pháp lưu trữ giá trị bộ đếm.
- Việc thực hiện khi tạo() kiểm tra sự tồn tại của `Gói outState`. Nếu như thế Bótôn tại, giá trị bộ đếm được khôi phục và lưu vào Chế độ xem văn bản.

Bài 2.3: Ý định ngầm định

Giới thiệu

Trong phần trước, bạn đã tìm hiểu về ý định rõ ràng. Trong ý định rõ ràng, bạn thực hiện một hoạt động trong ứng dụng của mình hoặc trong một ứng dụng khác bằng cách gửi ý định với tên lớp đủ điều kiện của hoạt động. Trong phần này, bạn tìm hiểu thêm về *ngầm định* mục đích và cách sử dụng chúng để thực hiện các hoạt động.

Với ý định ngầm, bạn khởi tạo một hoạt động mà không biết ứng dụng hoặc hoạt động nào sẽ xử lý tác vụ. Ví dụ: nếu bạn muốn ứng dụng của mình chụp ảnh, gửi email hoặc hiển thị vị trí trên bản đồ, bạn thường không quan tâm ứng dụng hoặc hoạt động nào thực hiện tác vụ.

Ngược lại, hoạt động của bạn có thể khai báo một hoặc nhiều bộ lọc ý định trong `AndroidManifest.xml` để quảng cáo rằng hoạt động có thể chấp nhận các ý định ngầm định và để xác định các loại ý định mà hoạt động sẽ chấp nhận.

Để khớp yêu cầu của bạn với ứng dụng được cài đặt trên thiết bị, hệ thống Android sẽ khớp ý định ngầm của bạn với một hoạt động có bộ lọc ý định cho biết chúng có thể thực hiện hành động. Nếu nhiều ứng dụng khớp, người dùng sẽ được cung cấp trình chọn ứng dụng cho phép họ chọn ứng dụng họ muốn sử dụng để xử lý ý định.

Trong phần thực hành này, bạn sẽ xây dựng một ứng dụng gửi ý định ngầm định để thực hiện từng tác vụ sau:

- Mở URL trong trình duyệt web.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

- Mở một vị trí trên bản đồ.
- Chia sẻ văn bản.

Chia sẻ—gửi một thông tin cho người khác qua email hoặc phương tiện truyền thông xã hội—là một tính năng phổ biến trong nhiều ứng dụng. Đối với hành động chia sẻ, bạn sử dụng `IntentBuilder` lớp, giúp dễ dàng xây dựng ý định ngầm định để chia sẻ dữ liệu.

Cuối cùng, bạn tạo một trình tiếp nhận ý định đơn giản chấp nhận ý định ngầm định cho một hành động cụ thể.

Những điều bạn nên biết

Bạn sẽ có thể:

- Sử dụng trình chỉnh sửa bố cục để sửa đổi bố cục.
- Chính sửa mã XML của bố cục.
- Thêm một Cái nút và trình xử lý nhấp chuột.
- Tạo và sử dụng một `Hoạt động`.
- Tạo và gửi một Ý định giữa một `Hoạt động` và một cái nữa.

Những gì bạn sẽ học được

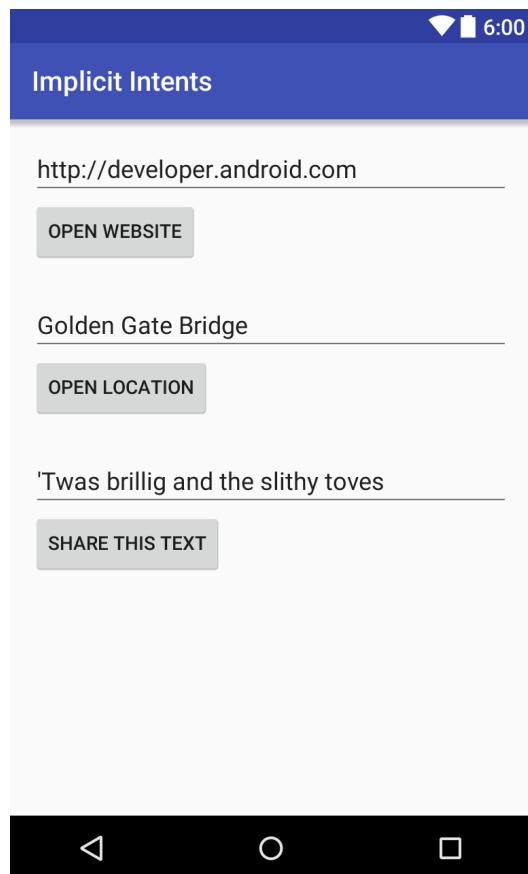
- Làm thế nào để tạo ra một Ý định, và sử dụng các hành động và danh mục của nó.
- Làm thế nào để sử dụng `IntentBuilder` lớp trợ giúp để tạo ra một Ý định để chia sẻ dữ liệu.
- Làm thế nào để quảng cáo rằng ứng dụng của bạn có thể chấp nhận một cách ngầm định Ý định bằng cách tuyên bố Ý định file bộ lọc trong `AndroidManifest.xml` là.

Bạn sẽ làm gì

- Tạo một ứng dụng mới để thử nghiệm với Ý định.
- Thực hiện một ngầm định Ý định mở một trang web và một trang khác mở một vị trí trên bản đồ.
- Thực hiện hành động để chia sẻ một đoạn văn bản.
- Tạo một ứng dụng mới có thể chấp nhận ngầm định Ý định để mở một trang web.

Tổng quan về ứng dụng

Trong phần này bạn tạo một ứng dụng mới với một Hoạt động và ba tùy chọn cho hành động: mở một trang web, mở một vị trí trên bản đồ và chia sẻ một đoạn văn bản. Tất cả các trường văn bản đều có thể chỉnh sửa (Sửa đổi văn bản), nhưng chứa các giá trị mặc định.



Nhiệm vụ 1: Tạo dự án và bối cảnh

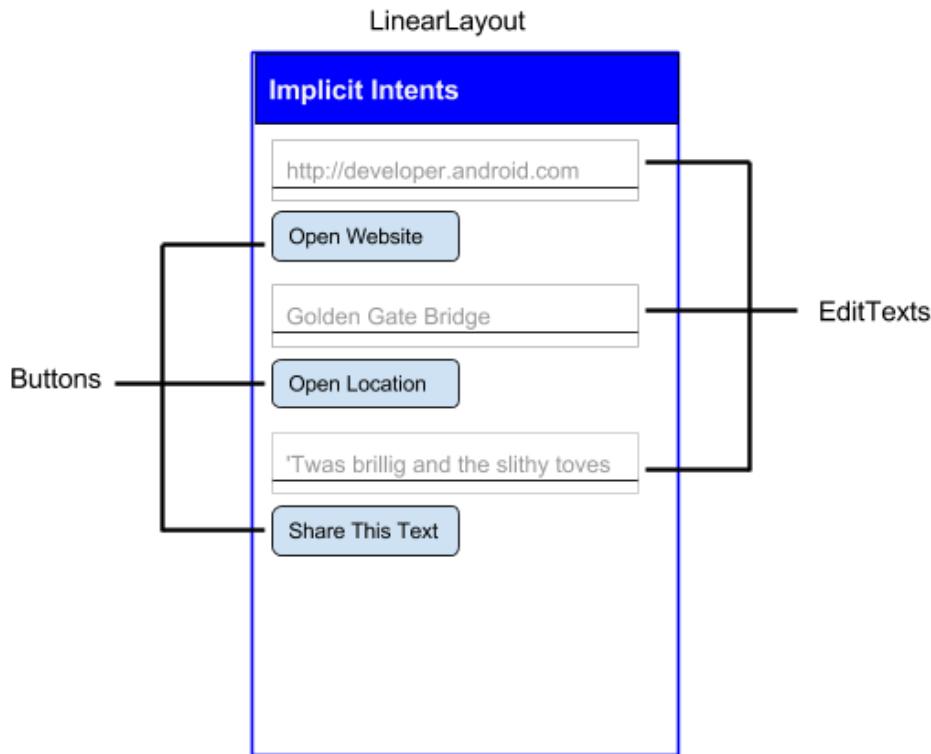
Đối với bài tập này, bạn tạo một dự án và ứng dụng mới có tên là Implicit Intents với bối cảnh mới.

1.1 Tạo dự án

- 1.Khởi động Android Studio và tạo một dự án Android Studio mới. Đặt tên cho ứng dụng của bạn **Ý định ngầm**.
- 2.Chọn **Hoạt động trống** cho mẫu dự án. Nhấp vào **Kế tiếp**.
- 3.Chấp nhận mặc định **Hoạt động** tên (Hoạt động chính). Hãy chắc chắn rằng **Tạo tệp Bối cảnh** hộp được chọn. Nhấp vào **Hoàn thành**.

1.2 Tạo bối cảnh

Trong nhiệm vụ này, hãy tạo bối cảnh cho ứng dụng. Sử dụng **Bối cảnh** tuyến tính, ba Cái nút các yếu tố, và ba Sửa văn bản các yếu tố như thế này:



1. Mở ứng dụng > res > giá trị > strings.xml trong Dự án > Android và thêm các tài nguyên chuỗi sau:

```
<chuỗi    name="edittext_uri">http://developer.android.com</string>
<chuỗi    name="button_uri">Mở trang web</string>

<string name="edittext_loc">Cầu Cổng Vàng</string> <string
name="button_loc">Vị trí mở</string>

<string name="edittext_share">'Twas brillig và slithy toves</string> <string
name="button_share">Chia sẻ văn bản này</string>
```

- 2.Mởres > bối cục > activity_main.xmltrongDự án > Androidngăn. Nhấp vàoChữtab để chuyển sang mã XML.
- 3.Thay đổi android.support.constraint.ConstraintLayoutĐẾNBối cục tuyến tính,như bạn đã học trong bài thực hành trước.
- 4.Thêm vàoandroid: định hướng thuộc tính có giá trị "thẳng đứng".Thêm vào android:đệm thuộc tính có giá trị "16dp".

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:ứng dụng="http://schemas.android.com/apk/resauto"  
    xmlns:công cụ="http://schemas.android.com/tools" android:bối  
    cục_chiều rộng="phù hợp với cha mẹ"  
    android:layout_height="match_parent"  
    android:orientation="vertical"  
    android:padding="16dp"  
    công cụ:context="com.example.android.implicitintents.MainActivity">
```

- 5.Loại bỏXem văn bản hiển thị "Xin chào thế giới".
- 6.Thêm một tập hợp các thành phần UI vào bối cục choMở trang webnút. Bạn cần mộtSửa văn bản phần tử và mộtCái nútphần tử. Sử dụng các giá trị thuộc tính này:

Thuộc tính EditText	Giá trị
android:id	"@+id/trang web_edittext"
android:bối cục_chiều_rộng	"phù hợp với cha mẹ"
android: layout_height	"bọc_nội_dung"
android:văn bản	"@string/edittext_uri"
Thuộc tính nút	Giá trị
android:id	"@+id/nút mở trang web"
android:bối cục_chiều_rộng	"bọc_nội_dung"
android: layout_height	"bọc_nội_dung"
android:layout_marginBottom	"24dp"
android:văn bản	"@string/button_uri"

android:onClick	"mở trang web"
-----------------	----------------

Giá trị cho `android:onClick` thuộc tính sẽ vẫn được gạch chân màu đỏ cho đến khi bạn xác định phương thức gọi lại trong tác vụ tiếp theo.

7.Thêm một tập hợp các thành phần UI (Sửa văn bản Và Cái nút) để bố trí cho **Mở Vị trí** cái nút.

Sử dụng các thuộc tính giống như trong bước trước, nhưng sửa đổi chúng như được hiển thị bên dưới. (Bạn có thể sao chép các giá trị từ **Mở trang web** và sửa đổi chúng.)

Thuộc tính EditText	Giá trị
android:id	"@+id/vị trí_sửa văn bản"
android:văn bản	"@string/edittext_loc"
Thuộc tính nút	Giá trị
android:id	"@+id/nút_mở_vị_trí"
android:văn bản	"@string/button_loc"
android:onClick	"mởVị trí"

Giá trị cho `android:onClick` thuộc tính sẽ vẫn được gạch chân màu đỏ cho đến khi bạn xác định phương thức gọi lại trong tác vụ tiếp theo.

8.Thêm một tập hợp các thành phần UI (Sửa văn bản Và Cái nút) để bố trí cho **Chia sẻ** cái nút. Sử dụng các thuộc tính được hiển thị bên dưới. (Bạn có thể sao chép các giá trị từ **Mở trang web** và sửa đổi chúng.)

Thuộc tính EditText	Giá trị
android:id	"@+id/share_edittext"
android:văn bản	"@string/edittext_share"
Thuộc tính nút	Giá trị
android:id	"@+id/nút chia sẻ văn bản"
android:văn bản	"@string/button_share"
android:onClick	"chia sẻ văn bản"

Tùy thuộc vào phiên bản Android Studio của bạn, hoạt động_main.xml mà nên trông như thế nào tương tự như sau. Các giá trị cho android:onClick các thuộc tính sẽ vẫn được gạch chân màu đỏ cho đến khi bạn xác định phương thức gọi lại trong tác vụ tiếp theo.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ứng dụng="http://schemas.android.com/apk/resauto"
    xmlns:công cụ="http://schemas.android.com/tools" android:bố
    cục_chiều rộng="phù hợp với cha mẹ"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp"
    công cụ:context="com.example.android.implicitintents.MainActivity">

    <Sửa đổi văn bản
        android:id="@+id/website_edittext"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/edittext_uri"/>

    <Nút
        android:id="@+id/open_website_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="24dp"
        android:text="@string/button_uri"
        android:onClick="openWebsite"/>

    <Sửa đổi văn bản
        android:id="@+id/location_edittext"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/edittext_uri"/>

    <Nút
        android:id="@+id/nút_mở_vị_trí"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_marginBottom="24dp"
        android:text="@string/button_loc"
        android:onClick="openLocation"/>

    <Sửa đổi văn bản
        android:id="@+id/share_edittext"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/edittext_share"/>

    <Nút
        android:id="@+id/share_text_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="24dp"
        android:text="@string/button_share"
        android:onClick="shareText"/>

</LinearLayout>
```

Nhiệm vụ 2: Triển khai nút Mở trang web

Trong tác vụ này, bạn triển khai phương thức xử lý khi nhấp cho nút đầu tiên trong bố cục, **Mở trang web**. Hành động này sử dụng một hàm `Intent` để gửi URI đã cho đến một `Activity` có thể xử lý được điều đó ngầm hiểu (`Intent` chẳng hạn như trình duyệt web).

2.1 Định nghĩa openWebsite()

- 1.Nhấp vào "mởTrang web" trong `activity_main.xml` XML.
- 2.Nhấn Alt+Enter (Tùy chọn+Enter trên máy Mac) và chọn **Tạo 'openWebsite(View)' trong 'MainActivity'**.

Các Hoạt động chính file mở ra và Android Studio tạo ra một phương thức khung cho mở Trang web() người xử lý.

```
public void openWebsite(Xem chế độ xem) { }
```

3. TRONG Hoạt động chính, thêm một biến riêng tư ở đầu lớp để giữ Sửa văn bản đổi tượng cho URI của trang web.

```
riêng tư EditText mWebsiteEditText;
```

4. Trong khi tạo() phương pháp cho Hoạt động chính, sử dụng tìmViewById() để có được một tham chiếu đến Sửa văn bản và gán nó vào biến riêng đó:

```
mWebsiteEditText = findViewById(R.id.website_edittext);
```

2.2 Thêm mã vào openWebsite()

1. Thêm một tuyên bố vào mới mở Trang web() phương pháp lấy giá trị chuỗi của Chính sửa văn bản:

```
Chuỗi url = mWebsiteEditText.getText().toString();
```

2. Mã hóa và phân tích chuỗi đó thành một đối tượng Uri:

```
Trang web Uri = Uri.parse(url);
```

3. Tạo một cái mới Ý định với Ý định ACTION_VIEW là hành động và URI là dữ liệu:

```
Ý định intent = new Intent(Intent.ACTION_VIEW, trang web);
```

Cái này Ý định constructor khác với constructor bạn dùng để tạo một constructor rõ ràng Ý định. Trong hàm tạo trước đó, bạn đã chỉ định ngữ cảnh hiện tại và một thành phần cụ thể (Hoạt động lặp) để gửi Ý định. Trong trình xây dựng này, bạn chỉ định một hành động và dữ liệu cho hành động đó. Các hành động được xác định bởi Ý định lớp và có thể bao gồm XEM_HÀNH_ĐỘNG (để xem dữ liệu đã cho), HÀNH_ĐỘNG_CHỈNH_SỬA (để chỉnh sửa dữ liệu đã cho), hoặc HÀNH_ĐỘNG_QUAY_SỐ (để quay số điện thoại). Trong trường hợp này, hành động là HÀNH_ĐỘNG_XEM bởi vì bạn muốn hiển thị trang web được chỉ định bởi URI trong trang web biến đổi.

4. Sử dụng giải quyết Hoạt động() phương pháp và trình quản lý gói Android để tìm một Hoạt động có thể xử lý được sự ngầm hiểu của bạn Ý định. Đảm bảo rằng yêu cầu được giải quyết thành công.

```
nếu (intent.resolveActivity(getApplicationContext()) != null) {}
```

Yêu cầu này phù hợp với bạn Ý định hành động và dữ liệu với Ý định filter bộ lọc cho các ứng dụng đã cài đặt trên thiết bị. Bạn sử dụng nó để đảm bảo có ít nhất một Hoạt động có thể xử lý được yêu cầu của bạn.

5. Bên trong nếu như tuyên bố, gọi bắt đầu Hoạt động() để gửi Ý định.

```
startActivity(y Ý định);
```

6.Thêm một khái khái để in một Nhật ký tin nhắn nếu Ý định không thể giải quyết được.

```
} khác {
    Log.d("ImplicitIntents", "Không xử lý được!");
}
```

Các mở Trang web() phương pháp bây giờ sẽ trông như sau. (Đã thêm bình luận để rõ ràng hơn.)

```
public void openWebsite(Xem chế độ xem) {
    // Lấy văn bản URL.
    Chuỗi url = mWebsiteEditText.getText().toString();

    // Phân tích URI và tạo mục đích. Uri webpage =
    Uri.parse(url);
    Ý định intent = new Intent(Intent.ACTION_VIEW, trang web);

    // Tìm một hoạt động để chuyển giao ý định và bắt đầu hoạt động đó. if
    (intent.resolveActivity(getApplicationContext()) != null) {
        startActivity(Ý định);
    } khác {
        Log.d("ImplicitIntents", "Không thể xử lý ý định này!");
    }
}
```

Nhiệm vụ 3: Triển khai nút Mở vị trí

Trong nhiệm vụ này, bạn triển khai phương thức xử lý khi nhấp vào nút thứ hai trong UI, **Mở Vị trí**. Phương pháp này gần giống với mở Trang web() phương pháp. Sự khác biệt là sử dụng geo URI để chỉ ra vị trí bắn đỗ. Bạn có thể sử dụng geo URI với vĩ độ và kinh độ hoặc sử dụng chuỗi truy vấn cho vị trí chung. Trong ví dụ này, chúng tôi đã sử dụng phương pháp sau.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

3.1 Định nghĩa openLocation()

- 1.Nhập vào "mởVị trí" trong hoat_dong_main.xml Mã XML.
- 2.Nhấn Alt+Enter (Tùy chọn+Enter trên máy Mac) và chọn **Tạo 'openLocation(View)' trong MainActivity.**

Android Studio tạo ra một phương thức khung xương trong Hoạt động chính chomởVị trí() người xử lý.

```
public void openLocation(Xem chế độ xem) {}
```

- 3.Thêm một biến riêng tư ở đầu Hoạt động chính để giữ Sửa văn bản đổi tương cho URI vị trí.

```
riêng tư EditText mLocationEditText;
```

- 4.Trong khi tạo() phương pháp, sử dụng `tìmViewByID()` để có được một tham chiếu đến Sửa văn bản và gán nó vào biến riêng đó:

```
mLocationEditText = findViewById(R.id.location_edittext);
```

3.2 Thêm mã vào openLocation()

- Trong cái mới mở Vị trí() phương pháp, thêm một câu lệnh để lấy giá trị chuỗi của mLocationEditText.Chỉnh sửa văn bản.

```
Chuỗi loc = mLocationEditText.getText().toString();
```

- Phân tích chuỗi đó thành đối tượng Uri bằng truy vấn tìm kiếm theo vị trí địa lý:

```
Địa chỉ Uri Uri = Uri.parse("geo:0,0?q=" + loc);
```

- Tạo một cái mới Ý định với Ý định.ACTION_VIEW là hành động và loc là dữ liệu.

```
Ý định intent = new Intent(Intent.ACTION_VIEW, addressUri);
```

- Giải quyết Ý định và kiểm tra để đảm bảo rằng Ý định đã giải quyết thành công. Nếu vậy, bắt đầu Hoạt động(), nếu không sẽ ghi lại thông báo lỗi.

```
nếu (intent.resolveActivity(getApplicationContext()) != null) {  
    startActivityForResult(ý định);  
} khác {  
    Log.d("ImplicitIntents", "Không thể xử lý ý định này!");  
}
```

CácmởVị trí()phương pháp này bây giờ sẽ trông như sau (thêm bình luận để rõ ràng hơn):

```
public void openLocation(Xem chế độ xem) {  
    // Lấy chuỗi chỉ ra vị trí. Đầu vào không được xác thực; nó được // truyền nguyên vẹn đến trình  
    // xử lý vị trí.  
    Chuỗi loc = mLocationEditText.getText().toString();  
  
    // Phân tích vị trí và tạo mục đích. Uri addressUri =  
    Uri.parse("geo:0,0?q=" + loc);  
    Ý định intent = new Intent(Intent.ACTION_VIEW, addressUri);  
  
    // Tìm một hoạt động để xử lý ý định và bắt đầu hoạt động đó. if  
    (intent.resolveActivity(getApplicationContext()) != null) {  
        startActivity(Ý định);  
    } khác {  
        Log.d("ImplicitIntents", "Không thể xử lý ý định này!");  
    }  
}
```

Nhiệm vụ 4: Triển khai nút Chia sẻ văn bản này

Hành động chia sẻ là cách dễ dàng để người dùng chia sẻ các mục trong ứng dụng của bạn với các mạng xã hội và các ứng dụng khác. Mặc dù bạn có thể xây dựng hành động chia sẻ trong ứng dụng của riêng mình bằng cách sử dụng mộtÝ định,Android cung cấp[Chia sẻCompat.IntentBuilder](#) lớp trợ giúp để thực hiện chia sẻ dễ dàng. Bạn có thể sử dụngChia sẻCompat.IntentBuilderđể xây dựng mộtÝ địnhvà khởi chạy trình chọn để người dùng có thể chọn ứng dụng đích để chia sẻ.

Trong nhiệm vụ này, bạn thực hiện chia sẻ một đoạn văn bản trong bản chỉnh sửa văn bản bằng cách sử dụngChia sẻCompat.IntentBuilder lớp học.

4.1 Định nghĩa shareText()

- 1.Nhấp vào "chia sẻ Văn bản" trong hoạt động_main.xml Mã XML.
- 2.Nhấn Alt+Nhập (Tùy chọn + Nhập trên máy Mac) và chọn **Tạo 'shareText(View)' trong MainActivity.**

Android Studio tạo ra một phương thức khung xương trong Hoạt động chính cho chia sẻ Văn bản() người xử lý.

```
public void shareText(Xem chế độ xem) { }
```

- 3.Thêm một biến riêng tư ở đầu Hoạt động chính để giữ Chính sửa văn bản.

```
riêng tư EditText mShareTextEdit;
```

- 4.TRONG khi tạo(), sử dụng tìm `findViewById()` để có được một tham chiếu đến Sửa văn bản và gán nó vào biến riêng đó:

```
mShareTextEdit = findViewById(R.id.share_edittext);
```

4.2 Thêm mã vào shareText()

- 1.Trong cái mới chia sẻ Văn bản() phương pháp, thêm một câu lệnh để lấy giá trị chuỗi của mShareTextEdit.Chỉnh sửa văn bản.

```
Chuỗi txt = mShareTextEdit.getText().toString();
```

2.Xác định loại MIME của văn bản cần chia sẻ:

```
Chuỗi mimeType = "text/plain";
```

3.GọiChia sẻCompat.IntentBuildervới những phương pháp sau:

```
Chia sẻCompat.IntentBuilder
    . từ(cái này)
    . thiết lập Kiểu(MimeKiểu)
    . setChooserTitle("Chia sẻ văn bản này với: ")
    . thiết lập Văn bản(txt)
    .startChooser();
```

4.Trích xuất giá trị của .thiết lậpChọnTiêu đềđến một tài nguyên chuỗi.

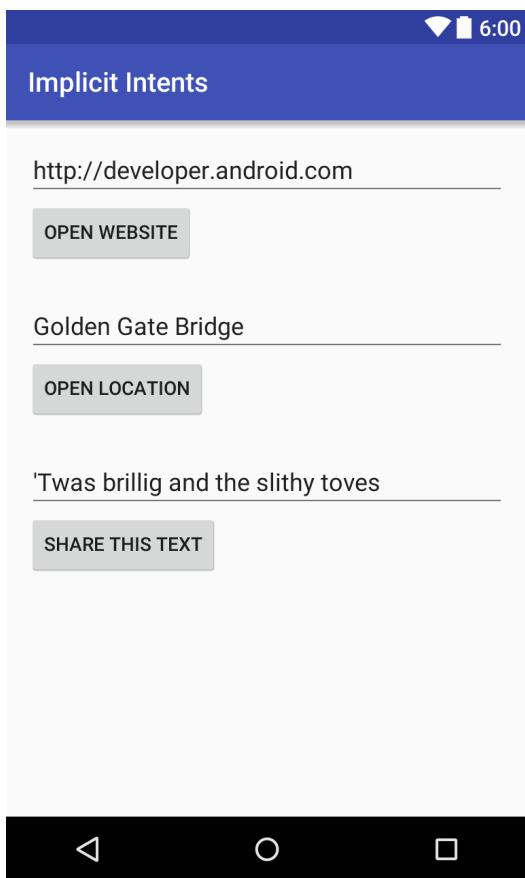
Cuộc gọi đếnChia sẻCompat.IntentBuildersử dụng các phương pháp sau:

Phương pháp	Sự miêu tả
tù()	Các Hoạt động ra mắt chia sẻ này Ý định (điều này).
đặtKiểu()	Loại MIME của mục được chia sẻ.
đặtChooserTitle()	Tiêu đề xuất hiện trên trình chọn ứng dụng hệ thống.
đặtVăn bản()	Văn bản thực tế sẽ được chia sẻ
startChooser()	Hiển thị trình chọn ứng dụng hệ thống và gửi Ý định.

Định dạng này, với tất cả các phương thức thiết lập của trình xây dựng được xâu chuỗi lại với nhau trong một câu lệnh, là một cách viết tắt dễ dàng để tạo và khởi chạy định. Bạn có thể thêm bất kỳ phương pháp bổ sung nào vào danh sách này.

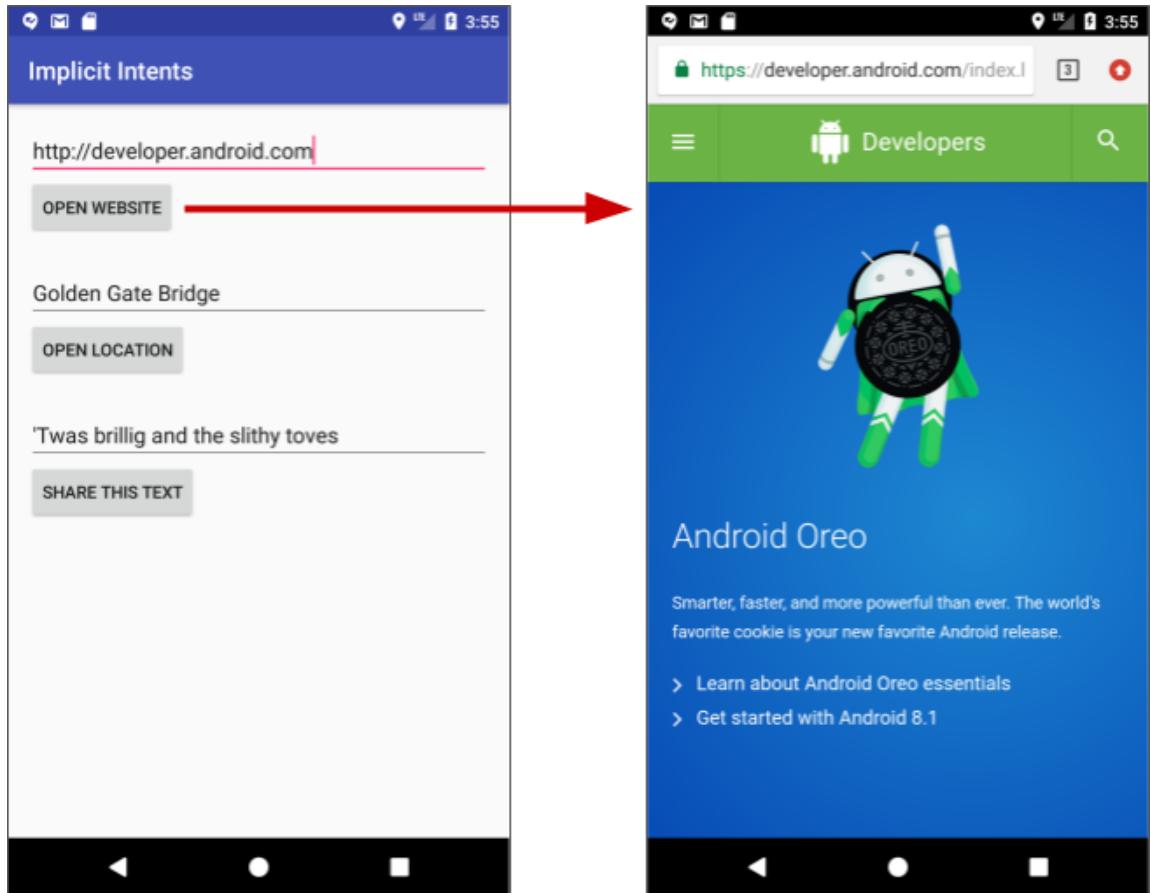
Các chia sẻ Văn bản() phương pháp bây giờ sẽ trông như sau:

```
public void shareText(Xem chế độ xem) {  
    Chuỗi txt = mShareTextEdit.getText().toString(); Chuỗi mimeType =  
    "text/plain";  
    Chia sẻCompat.IntentBuilder  
        .tù(cái này)  
        .thiết lập Kiểu(MimeKiểu)  
        .setChooserTitle(R.string.share_text_with)  
        .thiết lập Văn bản(txt)  
        .startChooser();  
}
```

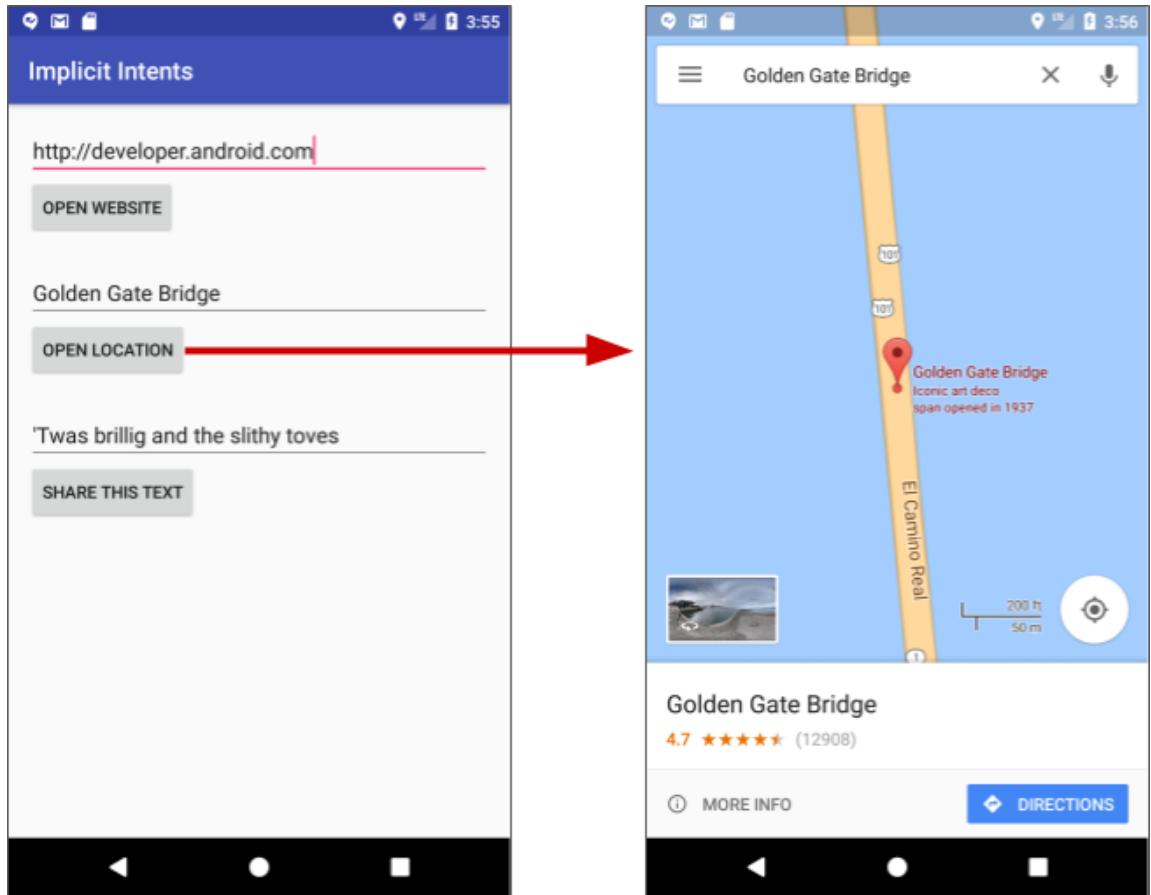


4.3 Chạy ứng dụng

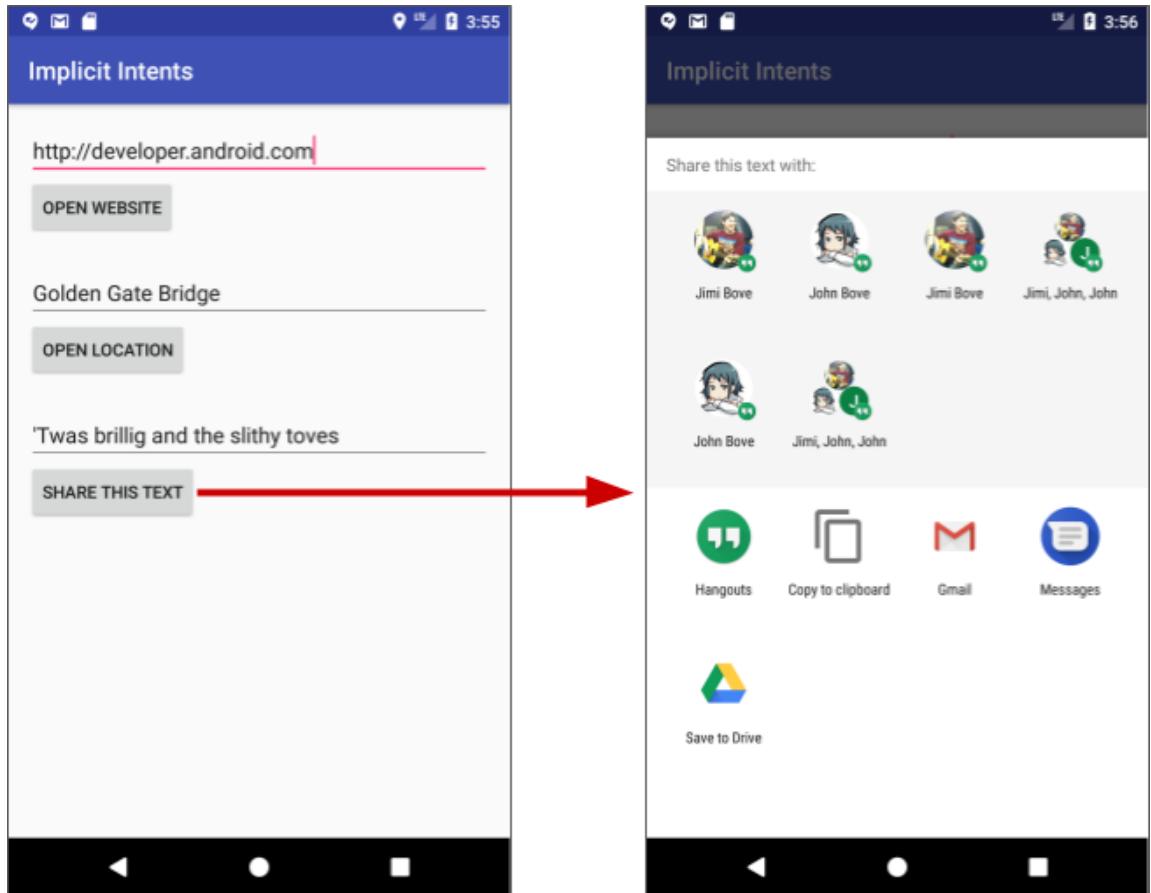
- 1.Chạy ứng dụng.
- 2.Nhấp vào **Mở trang web**nút để khởi chạy trình duyệt có URL trang web trong Sửa văn bản phía trên Cái nút.Trình duyệt và trang web sẽ hiển thị như hình dưới đây.



3.Nhấp vào **Mở Vị trí** nút để khởi chạy bản đồ với vị trí trong **Sửa văn bản** phía trên **Cái nút**. Bản đồ có vị trí sẽ hiển thị như hình bên dưới.



4.Nhấp vào**Chia sẻ văn bản này**để khởi chạy hộp thoại với các lựa chọn để chia sẻ văn bản. Hộp thoại với các lựa chọn sẽ xuất hiện như hình dưới đây.



Mã giải bài tập 4

Dự án Android Studio: [Ý định ngầm định](#)

Nhiệm vụ 5: Nhận một Intent ngầm định

Cho đến nay, bạn đã tạo ra một ứng dụng sử dụng hàm `Intent` để khởi chạy một số ứng dụng khác. Trong nhiệm vụ này, bạn nhìn nhận vấn đề theo hướng ngược lại: cho phép một ứng dụng của bạn để phản hồi một hàm `Intent` được gửi từ một ứng dụng khác.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

MỘTHoạt độngtrong ứng dụng của bạn luôn có thể được kích hoạt từ bên trong hoặc bên ngoài ứng dụng của bạn với một lệnh rõ ràng Ý định. Để cho phép một Hoạt độngđể nhận được một ngầm địnhÝ định,bạn định nghĩa mộtÝ định lọc trong ứng dụng của bạn AndroidManifest.xml để chỉ ra loại ẩn dụ nàoÝ địnhcủa bạnHoạt độngquan tâm đến việc xử lý.

Để khớp yêu cầu của bạn với một ứng dụng cụ thể được cài đặt trên thiết bị, hệ thống Android sẽ khớp với yêu cầu ngầm định của bạn.Ý địnhvới mộtHoạt độngcủa aiÝ định fibộ lọc cho biết họ có thể thực hiện hành động đó. Nếu có nhiều ứng dụng được cài đặt phù hợp, người dùng sẽ được cung cấp trình chọn ứng dụng cho phép họ chọn ứng dụng nào họ muốn sử dụng để xử lý ứng dụng đóÝ định.

Khi một ứng dụng trên thiết bị gửi một lệnh ngầmÝ định,hệ thống Android khớp với hành động và dữ liệu của nóÝ địnhvới bất kỳ có sẵnHoạt độngbao gồm cả quyềnÝ định fibộ lọc. Khi Ý định fibộ lọc cho mộtHoạt độngphù hợp vớiMục đích:

- Nếu chỉ có một kết quả phù hợpHoạt động,Android cho phépHoạt độngxử lýÝ địnhchính nó.
- Nếu có nhiều kết quả trùng khớp, Android sẽ hiển thị trình chọn ứng dụng để cho phép người dùng chọn ứng dụng mà họ muốn thực hiện hành động đó.

Trong nhiệm vụ này, bạn tạo một ứng dụng rất đơn giản nhận được một lệnh ngầm địnhÝ địnhđể mở URI cho một trang web. Khi được kích hoạt bởi một hàm ẩnÝ định, ứng dụng đó hiển thị URI được yêu cầu dưới dạng một chuỗi trong Chế độ xem văn bản.

5.1 Tạo dự án và bố cục

- 1.Tạo một dự án Android Studio mới với tên ứng dụng**Người nhận ý định ngầm định**và chọn **Hoạt động trống**cho mẫu dự án.
- 2.Chấp nhận mặc địnhHoạt độngtên (Hoạt động chính).Nhấp chuột**Kế tiếp**.
- 3.Hãy chắc chắn rằng**Tạo tệp Bố cục**được chọn. Nhấp vào**Hoàn thành**.
- 4.Mở**hoạt động_main.xml**.
- 5.Trong hiện tại ("Hello World")Chế độ xem văn bản,xóa bỏ android:văn bản thuộc tính. Không có văn bản trong nàyXem văn bản theo mặc định, nhưng bạn sẽ thêm URI từÝ địnhTRONG khi Tạo().
- 6.Rời khỏi rằng buộc bố cục chỉ các thuộc tính, nhưng thêm các thuộc tính sau:

Thuộc tính	Giá trị
android:id	"@+id/text_uri_message"
android:kích thước văn bản	"18sp"
android:kiểu chữ	"in đậm"

5.2 Sửa đổi AndroidManifest.xml để thêm bộ lọc Intent

1.MởAndroidManifest.xml là.

2.Lưu ý rằng Hoạt động chính đã có cái này rồi Ý định filoc:

```
<bộ lọc ý định>
<hành động android:tên="android.intent.action.MAIN" /> <thể loại
    android:tên="android.intent.category.LAUNCHER" </intentfilter>           />
```

Cái này Ý định filter, là một phần của bản kê khai dự án mặc định, chỉ ra rằng điều này Hoạt động là điểm vào chính cho ứng dụng của bạn (nó có một Ý định hành động của "android.intent.action.MAIN"), và điều này Hoạt động sẽ xuất hiện như một mục cấp cao nhất trong trình khởi chạy (thể loại của nó là "android.intent.category.LAUNCHER").

3.Thêm một < thứ hai bộ lọc ý định> thẻ bên trong <hoạt động>, và bao gồm các yếu tố sau:

```
<hành động android:tên="android.intent.action.VIEW" <thể loại      />
    android:name="android.intent.category.MẶC ĐỊNH" />
<thể loại      android:name="android.intent.category.CÓ THẺ DUYỆT"          />
<dữ liệu android:scheme="http" android:host="developer.android.com" />
```

Những dòng này xác định một Ý định filoc cho Hoạt động, nghĩa là loại Ý định rằng Hoạt động có thể xử lý. Điều này Ý định filter khai báo các phần tử sau:

Lọc kiểu	Giá trị	Trận đấu
-------------	---------	----------

hoạt động	"android.intent.action.VIEW"	Bất kỳ định với các hành động xem.
loại	"android.intent.category.MÃC ĐỊNH"	Bất kỳ nguy ý nào định. Thể loại này phải được bao gồm cho bạn. Hoạt động để nhận được bất kỳ ngầm hiểu định.
loại	"android.intent.category.BROWSABLE"	Yêu cầu có thể duyệt liên kết từ các trang web, email hoặc các nguồn khác.
dữ liệu	android:scheme="http" android:host="developer.android.com"	URI chứa một lược đồ http và tên máy chủ của nhà phát triển.android.com.

Lưu ý rằng dữ liệu bộ lọc có hạn chế về cả loại liên kết mà nó sẽ chấp nhận và tên máy chủ cho các URI đó. Nếu bạn muốn máy thu của mình có thể chấp nhận bất kỳ liên kết nào, bạn có thể bỏ qua <dữ liệu> yếu tố.

Các ứng dụng phần của AndroidManifest.xml bây giờ sẽ trông như sau:

```
<ứng dụng
    android:allowBackup="true" android:icon="@mipmap/
    ic_launcher" android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"

    android:theme="@style/AppTheme"> <hoạt động
    android:name=".MainActivity">
        <bộ lọc ý định>
            <hành động android:tên="android.intent.action.MAIN" <thể loại      />
            android:tên="android.intent.category.LAUNCHER" </intentfilter>           />

        <bộ lọc ý định>
            <hành động android:tên="android.intent.action.VIEW" <thể loại      />
            android:tên="android.intent.category.MÃC ĐỊNH"                         />
            <thể loại      android:tên="android.intent.category.CÓ THỂ DUYỆT"       />
            <dữ liệu android:scheme="http"                                              />
                            android:host="developer.android.com"                         />
        </bộ lọc ý định>
    </hoạt động>
</ứng dụng>
```

```
</hoạt động>  
</ứng dụng>
```

5.3 Xử lý ý định

Trong khi tạo() phương pháp cho bạn Hoạt động, xử lý các thông tin đến Ý định cho bất kỳ dữ liệu hoặc phần bổ sung nào mà nó bao gồm. Trong trường hợp này, dữ liệu ngầm định đến Ý định có URI được lưu trữ trong Ý định dữ liệu.

1. Mở Hoạt động chính.

2. Trong khi tạo() phương pháp, nhận được thông tin đến Ý định được sử dụng để kích hoạt Hoạt động:

```
Ý định intent = getIntent();
```

3. Nhận được Ý định dữ liệu. Ý định dữ liệu luôn là một đối tượng URI:

```
Uri uri = intent.getData();
```

4. Kiểm tra để đảm bảo rằng uri biến không phải là vô giá trị. Nếu kiểm tra đó thành công, hãy tạo một chuỗi từ đối tượng URI đó:

```
nhếu (uri != null) {  
    Chuỗi uri_string = "URI: " + uri.toString();  
}
```

5.Trích xuất "URI: "một phần của phần trên thành một chuỗi tài nguyên (nhãn uri).

6.Bên trong cùng một nút như chặng, lấy Xem văn bản cho tin nhắn:

```
TextView textView = findViewById(R.id.text_uri_message);
```

7.Cũng bên trong nút như khôi, thiết lập văn bản đó Xem văn bản đến URI:

```
textView.setText(uri_string);
```

Cách khi tạo() phương pháp cho Hoạt động chính bấy giờ sẽ trông như sau:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    Ý định ý định = getIntent(); Uri uri = ý  
    định.getData(); nếu (uri != null) {  
  
        Chuỗi uri_string = getString(R.string.uri_label)  
            + uri.toString();  
        Chế độ xem văn bản textView = findViewById(R.id.text_uri_message); chế độ  
        xem văn bản.setText(uri_string);  
    }  
}
```

5.4 Chạy cả hai ứng dụng

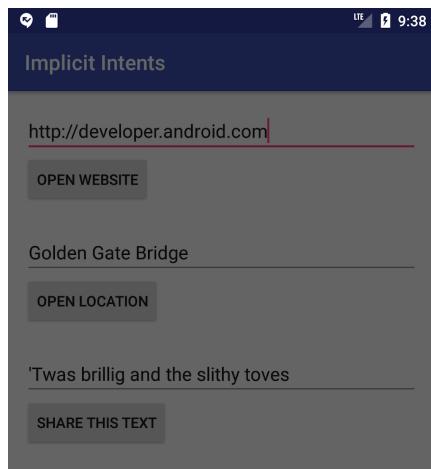
Để hiển thị kết quả của việc nhận được một hàm ẩn Ý định, bạn sẽ chạy cả ứng dụng Implicit Intents Receiver và Implicit Intents trên trình giả lập hoặc thiết bị của bạn.

1. Chạy ứng dụng Implicit Intents Receiver.

Chạy ứng dụng một mình sẽ hiển thị một khoảng trống Hoạt động không có văn bản. Điều này là do Hoạt động đã được kích hoạt từ trình khởi chạy hệ thống, và không phải với Ý định từ một ứng dụng khác.

2. Chạy ứng dụng Implicit Intents và nhấp vào Mở trang web với URI mặc định.

Trình chọn ứng dụng sẽ xuất hiện hỏi bạn có muốn sử dụng trình duyệt mặc định (Chrome trong hình bên dưới) hay ứng dụng Implicit Intents Receiver không. Chọn **Người nhận ý định ngầm định** và nhấp vào **Chỉ một lần**. Ứng dụng Implicit Intents Receiver sẽ khởi chạy và thông báo hiển thị URI từ yêu cầu ban đầu.



Open with

Implicit Intents Receiver

Chrome

JUST ONCE ALWAYS



3. Nhấn nút Quay lại và nhập một URI khác. Nhấp vào Mở trang web.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

Ứng dụng máy thu có một hạn chế rất lớn Ý định fib bộ lọc chỉ khớp với giao thức URI chính xác (<http://>) và máy chủ (developer.android.com). Bất kỳ URI nào khác đều mở trong trình duyệt web mặc định.

Mã giải bài tập 5

Dự án Android Studio:[Người nhận ý định ngầm định](#)

Thử thách mã hóa

Ghi chú: Mọi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách: Trong thử thách thực tế trước đó, bạn đã tạo ra một trình xây dựng ứng dụng danh sách mua sắm với **Hoạt động** để hiển thị danh sách và một danh sách khác **Hoạt động** để chọn một mục. Thêm một **Sửa văn bản** và một **Cái nút** vào danh sách mua sắm **Hoạt động** để xác định vị trí một cửa hàng cụ thể trên bản đồ.

Bản tóm tắt

- Một ẩn ý Ý định cho phép bạn kích hoạt một **Hoạt động** nếu bạn biết hành động, nhưng không biết ứng dụng cụ thể hoặc **Hoạt động** sẽ xử lý hành động đó.
- MỘT **Hoạt động** có thể nhận được một ngầm Ý định **fib** bộ lọc trong **AndroidManifest.xml** phù hợp với một hoặc nhiều Ý định **hành động** và danh mục.
- Hệ thống Android khớp với nội dung của một hàm ẩn Ý định và Ý định **fib** bộ lọc của bất kỳ có sẵn **Hoạt động** để xác định cái nào **Hoạt động** để kích hoạt. Nếu có nhiều hơn một **Hoạt động**, hệ thống cung cấp chức năng chọn để người dùng có thể chọn một.
- Các **Chia sẻ Compat.IntentBuilder** lớp giúp dễ dàng xây dựng một lớp ngầm Ý định để chia sẻ dữ liệu lên mạng xã hội hoặc email.

Khái niệm liên quan

Tài liệu khái niệm liên quan có trong [2.3: Ý định ngầm định](#).

Tìm hiểu thêm

Tài liệu dành cho nhà phát triển Android:

- [Cơ sở ứng dụng](#)
- [Các hoạt động](#)
- [Hiểu về Vòng đời hoạt động](#)
- [Ý định và Bộ lọc ý định](#)
- [Cho phép các ứng dụng khác bắt đầu hoạt động của bạn](#)
- [Ý định của Google Maps dành cho Android](#)
- [Hoạt động](#)
- [Ý định](#)
- [<bộ lọc ý định>](#)
- [<hoạt động>](#)
- [Uri](#)
- [Chia sẻCompat.IntentBuilder](#)

Bài tập về nhà

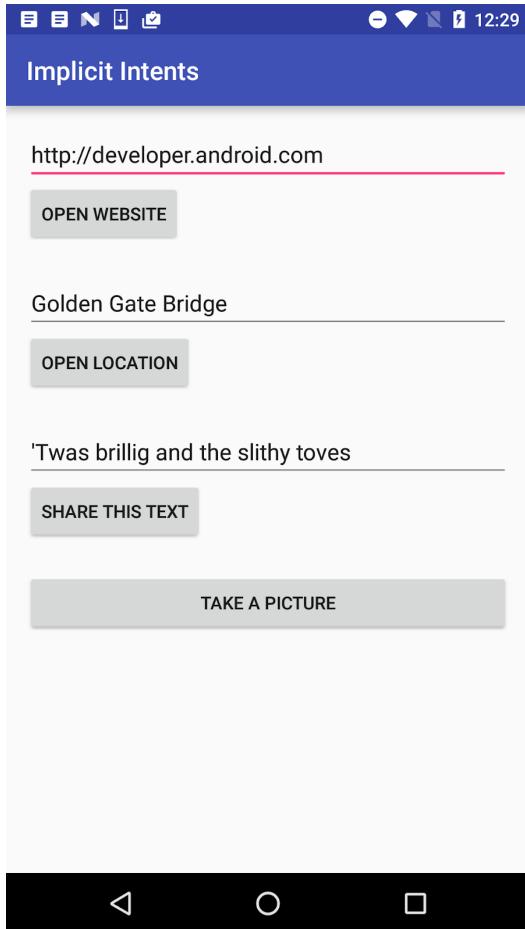
Xây dựng và chạy một ứng dụng

Mở [Ý định ngầm định](#) ứng dụng bạn đã tạo.

1. Thêm một nút khác ở cuối màn hình.
2. Khi Cái nút được nhấp, hãy khởi chạy ứng dụng máy ảnh để chụp ảnh. (Bạn không cần phải trả lại ảnh về ứng dụng gốc.)

Ghi chú:

Nếu bạn sử dụng trình giả lập Android để kiểm tra camera, hãy mở cấu hình trình giả lập trong trình quản lý AVD Android, chọn **Cài đặt nâng cao**, và sau đó chọn **Mô phỏng** cho cả camera trước và sau. Khởi động lại trình giả lập nếu cần.



Trả lời những câu hỏi này

Câu hỏi 1

Bạn sử dụng phương thức xây dựng nào để tạo một hàm ẩn? Ý định để khởi chạy ứng dụng máy ảnh?

- Ý định mới()
- new Intent(Context context, Class<?> class)

- new Intent(String action, Uri uri)
- new Intent(Chuỗi hành động)

Câu hỏi 2

Khi bạn tạo ra một ẩn Ý định vật, câu nào sau đây là đúng?

- Không chỉ định cụ thể Hoạt động hoặc thành phần khác để khởi chạy.
- Thêm một Ý định hành động hoặc Ý định danh mục (hoặc cả hai).
- Giải quyết Ý định với hệ thống trước khi gọi bắt đầu Hoạt động() hoặc bắt đầu Hoạt động cho Kết quả().
- Tất cả những điều trên.

Câu hỏi 3

Cái mà Ý định Bạn dùng thao tác nào để chụp ảnh bằng ứng dụng camera?

- Ý định takePicture = new Intent(Intent.ACTION_VIEW);
- Ý định takePicture = new Intent(Intent.ACTION_MAIN);
- Ý định takePicture = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
- Ý định takePicture = new Intent(Intent.ACTION_GET_CONTENT);

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Kiểm tra xem ứng dụng có các tính năng sau không:

- Nó hiển thị một **Chụp một bức ảnh** nút ở cuối ứng dụng.
- Khi nhấp vào, nút này sẽ khởi chạy ứng dụng camera trên thiết bị.

- Trước khi gửi ý định, khi nhấp vào() phương pháp cho **Chụp một bức ảnh** Cái nút đảm bảo rằng một ứng dụng có sẵn trên thiết bị, bằng cách sử dụng **giải quyết** Hoạt động(). Và lấy **PackageManager()** phương pháp.

Bài 3.1: Trình gỡ lỗi

Giới thiệu

Trong các bài thực hành trước bạn đã sử dụng **Nhật ký** lớp để in thông tin vào nhật ký hệ thống, xuất hiện trong **Logcat** trong Android Studio khi ứng dụng của bạn chạy. Thêm các câu lệnh ghi nhật ký vào ứng dụng của bạn là một cách để tìm lỗi và cải thiện hoạt động của ứng dụng. Một cách khác là sử dụng trình gỡ lỗi được tích hợp trong Android Studio.

Trong phần thực hành này, bạn sẽ học cách gỡ lỗi ứng dụng trong trình giả lập và trên thiết bị, thiết lập và xem điểm dừng, kiểm tra mã và kiểm tra các biến.

Những điều bạn nên biết

Bạn sẽ có thể:

- Tạo một dự án Android Studio.
- Sử dụng trình chỉnh sửa bối cảnh để làm việc với Sửa văn bản và Cài nút các yếu tố.
- Xây dựng và chạy ứng dụng của bạn trong Android Studio, trên cả trình giả lập và trên thiết bị.
- Đọc và phân tích dấu vết ngắn xếp, bao gồm lệnh bắt đầu cùng và lệnh tắt đầu tiên.
- Thêm các câu lệnh nhật ký và xem nhật ký hệ thống (**Logcat pane**) trong Android Studio.

Những gì bạn sẽ học được

- Cách chạy ứng dụng ở chế độ gỡ lỗi trên trình giả lập hoặc trên thiết bị.

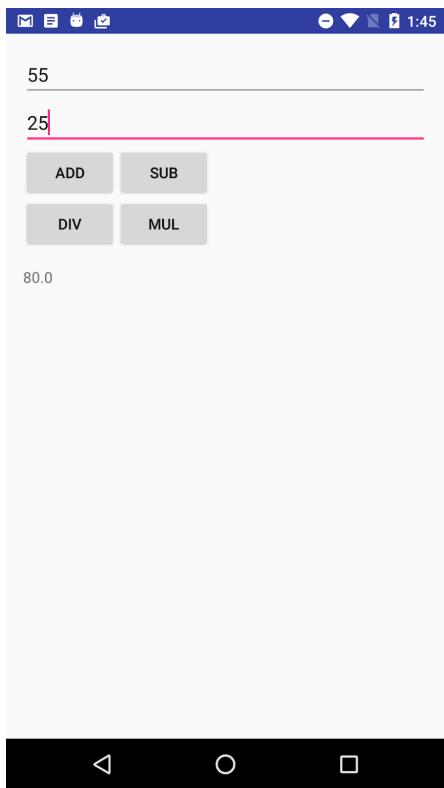
- Cách thực hiện từng bước trong quá trình triển khai ứng dụng của bạn.
- Cách thiết lập và sắp xếp điểm dừng.
- Cách kiểm tra và sửa đổi các biến trong trình gỡ lỗi.

Bạn sẽ làm gì

- Xây dựng ứng dụng SimpleCalc.
- Đặt và xem điểm dừng trong mã cho SimpleCalc.
- Duyệt qua mã của bạn khi nó chạy.
- Kiểm tra các biến và đánh giá biểu thức.
- Xác định và khắc phục sự cố trong ứng dụng mẫu.

Tổng quan về ứng dụng

Ứng dụng SimpleCalc có hai Sửa văn bản các yếu tố và bốn Cái nút các yếu tố. Khi bạn nhập hai số và nhấn vào Cái nút, ứng dụng thực hiện tính toán cho điều đó Cái nút và hiển thị kết quả.



Nhiệm vụ 1: Khám phá dự án và ứng dụng SimpleCalc

Đối với bài thực hành này, bạn sẽ không tự xây dựng ứng dụng SimpleCalc. Toàn bộ dự án có sẵn tại [Tính toán đơn giản](#). Trong nhiệm vụ này, bạn sẽ mở dự án SimpleCalc trong Android Studio và khám phá một số tính năng chính của ứng dụng.

1.1 Tải xuống và mở Dự án SimpleCalc

1.Tải về [Tính toán đơn giản](#) và giải nén tập tin.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

2.Khởi động Android Studio và chọn **Tệp > Mở**.

3.Điều hướng đến thư mục SimpleCalc, chọn tệp thư mục đó và nhấp vào **ĐƯỢC RỒI**. Dự án SimpleCalc được xây dựng.

4.Mở **Dự án > Android** nếu nó chưa được mở.

Cảnh báo: Ứng dụng này chứa các lỗi mà bạn sẽ tìm thấy và sửa. Nếu bạn chạy ứng dụng trên thiết bị hoặc trình giả lập, bạn có thể gặp phải hành vi không mong muốn, bao gồm cả sự cố trong ứng dụng.

1.2 Khám phá Bố cục

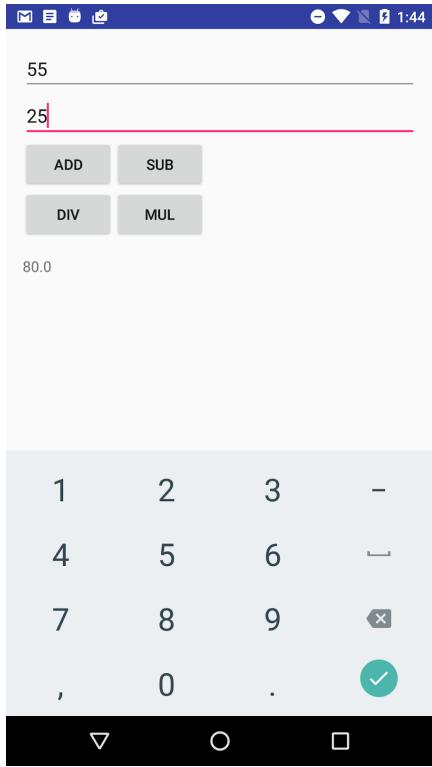
1.Mở **hoạt động_main.xml**.

2.Nhấp vào **Chữ tab** để xem mã XML.

3.Nhấp vào **Xem trước** tab để xem trước bố cục.

Kiểm tra mã XML và thiết kế bố cục và lưu ý những điều sau:

- Bố cục chứa hai Sửa văn bản các yếu tố cho đầu vào, bốn Cái nút các yếu tố cho các phép tính, và một Xem văn bản để hiển thị kết quả.
- Mỗi phép tính Cái nút có riêng của nó android:onClick trình xử lý nhấp chuột (onAdd, OnSub, và vân vân.)
- Các Xem văn bản vì mặc định kết quả sẽ không có văn bản nào.
- Hai Sửa văn bản các yếu tố có android:kiểu đầu vào thuộc tính và giá trị "số thập phân". Thuộc tính này chỉ ra rằng Sửa văn bản chỉ chấp nhận số làm đầu vào. Bàn phím xuất hiện trên màn hình sẽ chỉ chứa số. Bạn sẽ tìm hiểu thêm về các loại đầu vào cho Sửa văn bản các yếu tố trong phần thực hành sau.



1.3 Khám phá mã ứng dụng

1. Mở rộng ứng dụng>javathư mục trong **Dự án > Android**ngăn. Ngoài ra còn có **Hoạt động chính**lớp học, dự án này cũng bao gồm một tiện ích **Máy tính**lớp học.
2. Mở **Máy tính**và kiểm tra mã. Lưu ý rằng các hoạt động mà máy tính có thể thực hiện được xác định bởi **Toán tử enum**, và tất cả các phương pháp hoạt động đều công cộng.
3. Mở **Hoạt động chính**và kiểm tra mã và bình luận.

Lưu ý những điều sau:

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

- Tất cả các định nghĩa android:onClick trong xử lý nhấp chuột gọi là riêng tư tính toán() phương pháp, với tên hoạt động là một trong các giá trị từ Máy tính.Tính toán tử sự liệt kê.
- Cáctính toán()phương pháp gọi riêng tư phương pháp lây Toán tử() (mà lần lượt gọi lấy Toán tử và Văn bản()) để lấy lại các giá trị số từ Sửa văn bản các yếu tố.
- Cáctính toán()phương pháp sau đó sử dụng một công tắc trên tên toán hạng để gọi phương thức thích hợp trong Máy tính ví dụ (Máy tính).
- Các phương pháp tính toán trong Máy tính Lớp thực hiện phép tính số học thực tế và trả về một giá trị.
- Phần cuối cùng của tính toán() phương pháp cập nhật Xem văn bản với kết quả của phép tính.

1.4 Chạy ứng dụng

Chạy ứng dụng và làm theo các bước sau:

- 1.Nhập cả giá trị số nguyên và số dấu phẩy động để tính toán.
- 2.Nhập các giá trị dấu phẩy động với phân số thập phân lớn (ví dụ:**1.6753456**)
- 3.Chia một số cho số không.
- 4.Để lại một hoặc cả hai Sửa văn bản các phần tử rỗng và thử bất kỳ phép tính nào.
- 5.Nhấp vào **Logcat** tab ở cuối cửa sổ Android Studio để mở **Logcat** ngăn (nếu chưa mở). Kiểm tra dấu vết ngăn xếp tại điểm ứng dụng báo cáo lỗi.

Nếu một hoặc cả hai Sửa văn bản các phần tử trong SimpleCalc trống, ứng dụng báo cáo ngoại lệ, như thể hiện trong hình bên dưới và nhật ký hệ thống hiển thị trạng thái của ngăn xếp thực thi tại thời điểm ứng dụng tạo ra lỗi. Dấu vết ngăn xếp thường cung cấp thông tin quan trọng về lý do xảy ra lỗi.

```

02-22 17:10:16.649 2869-2869/com.example.android.SimpleCalc E/CalculatorActivity: NumberFormatException: empty String
    at sun.misc.FloatingDecimal.readJavaFormatString(FloatingDecimal.java:1842)
    at sun.misc.FloatingDecimal.parseDouble(FloatingDecimal.java:110)
    at java.lang.Double.parseDouble(Double.java:539)
    at com.example.android.SimpleCalc.MainActivity.getOperand(MainActivity.java:136)
    at com.example.android.SimpleCalc.MainActivity.compute(MainActivity.java:182)
    at com.example.android.SimpleCalc.MainActivity.onSub(MainActivity.java:94) <1 internal calls>
    at android.view.View$DeclaredOnClickListener.onClick(View.java:5331)
    at android.view.View.performClick(View.java:24697)
    at android.os.Handler.handleCallback(Handler.java:789)
    at android.os.Handler.dispatchMessage(Handler.java:98)
    at android.os.Looper.loop(Looper.java:164)
    at android.app.ActivityThread.main(ActivityThread.java:6541) <1 internal calls>
    at java.lang.reflect.Method.invoke(Native Method)
    at com.android.internal.os.Zygote$MethodAndArgsCaller.run(Zygote.java:240)
    at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:767)

```

Nhiệm vụ 2: Chạy SimpleCalc trong trình gỡ lỗi

Trong nhiệm vụ này, bạn sẽ được giới thiệu về trình gỡ lỗi trong Android Studio và tìm hiểu cách đặt điểm dừng và chạy ứng dụng ở chế độ gỡ lỗi.

2.1 Khởi động và chạy ứng dụng của bạn ở chế độ gỡ lỗi

1.Trong Android Studio, hãy chọn **Chạy > Gỡ lỗi Ứng dụng** hoặc nhấp vào **Biểu tượng gỡ lỗi** trên thanh công cụ.

2.Nếu ứng dụng của bạn đã chạy, bạn sẽ được hỏi xem bạn có muốn khởi động lại ứng dụng ở chế độ gỡ lỗi không. Nhập vào **Khởi động lại ứng dụng**.

Android Studio xây dựng và chạy ứng dụng của bạn trên trình giả lập hoặc trên thiết bị. Gỡ lỗi là như nhau trong cả hai trường hợp. Trong khi Android Studio đang khởi tạo trình gỡ lỗi, bạn có thể thấy thông báo "Đang chờ trình gỡ lỗi" trên thiết bị trước khi bạn có thể sử dụng ứng dụng của mình.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

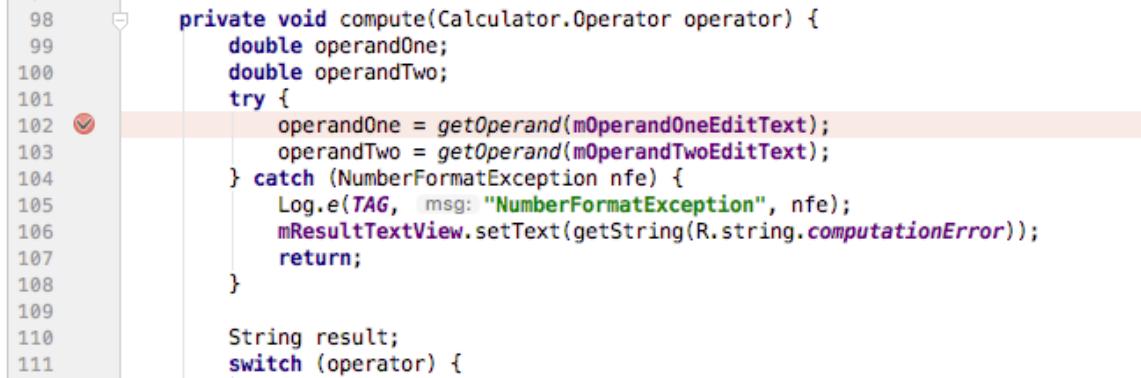
để biết thông tin cập nhật mới nhất.

- 3.Nhấp vào **Gỡ lỗi** tab ở cuối cửa sổ Android Studio để hiển thị **Gỡ lỗi** khung cửa sổ (hoặc chọn **Xem > Công cụ Windows > Gỡ lỗi**). Các **Trình gỡ lỗi** tab trong ngăn phải được chọn, hiển thị **Trình gỡ lỗi** khung cửa sổ.

2.2 Đặt điểm dừng

Điểm dừng là một vị trí trong mã của bạn mà bạn muốn tạm dừng việc thực thi bình thường của ứng dụng để thực hiện các hành động khác như kiểm tra biến hoặc đánh giá biểu thức hoặc thực thi từng dòng mã của bạn để xác định nguyên nhân gây ra lỗi thời gian chạy. Bạn có thể đặt điểm dừng trên bất kỳ dòng mã thực thi nào.

- 1.Mở **Hoạt động chính** và nhấp vào dòng thứ tư của tính toán() phương pháp (dòng ngay sau **hỗn**).
- 2.Nhấp vào rãnh bên trái của khung soạn thảo tại dòng đó, bên cạnh số dòng. Một chấm đỏ xuất hiện tại dòng đó, biểu thị điểm dừng. Dấu chấm đỏ bao gồm dấu kiểm nếu ứng dụng đã chạy ở chế độ gỡ lỗi.
Ngoài ra, bạn có thể chọn **Chạy > Chuyển đổi điểm ngắt dòng** hoặc nhấn **Control-F8** (**Lệnh-F8** trên máy Mac) để đặt hoặc xóa điểm dừng tại một dòng.



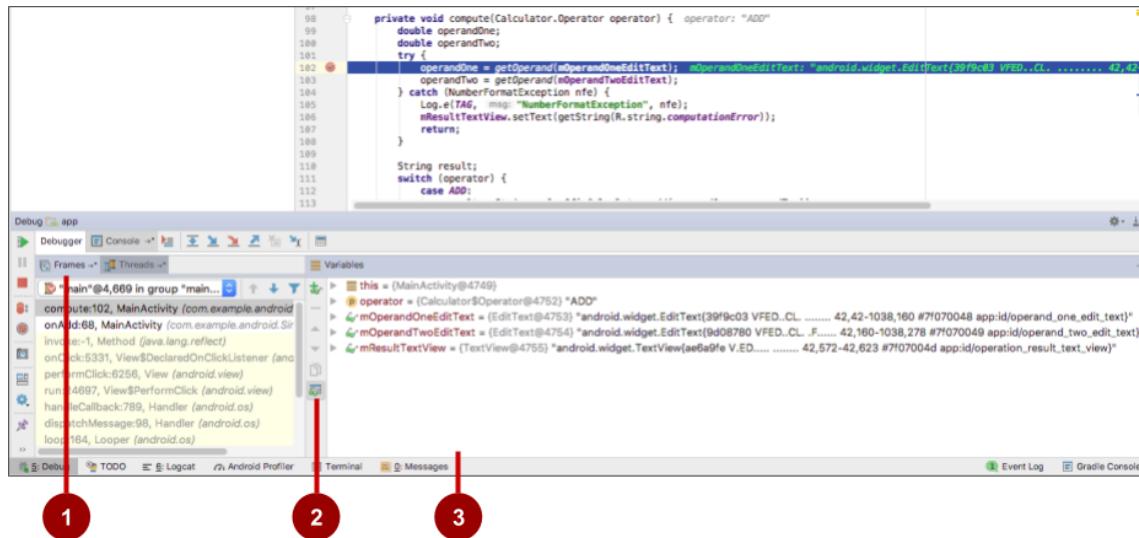
```
98
99
100
101
102
103
104
105
106
107
108
109
110
111
```

```
private void compute(Calculator.Operator operator) {
    double operandOne;
    double operandTwo;
    try {
        operandOne = getOperand(mOperandOneEditText);
        operandTwo = getOperand(mOperandTwoEditText);
    } catch (NumberFormatException nfe) {
        Log.e(TAG, msg: "NumberFormatException", nfe);
        mResultTextView.setText(getString(R.string.computationError));
        return;
    }
    String result;
    switch (operator) {
```

Nếu bạn vô tình nhấp vào điểm dừng, bạn có thể hoàn tác bằng cách nhấp vào điểm dừng. Nếu bạn nhấp vào một dòng mã không thể thực thi, dấu chấm đỏ sẽ bao gồm một "x" và xuất hiện cảnh báo rằng dòng mã này không thể thực thi được.

- 3.Trong ứng dụng SimpleCalc, nhập số vào Sửa văn bản các yếu tố và nhấp vào một trong các tính toán Cái nút các yếu tố.

Việc thực thi ứng dụng của bạn sẽ dừng lại khi đạt đến điểm dừng mà bạn đặt và trình gỡ lỗi sẽ hiển thị trạng thái hiện tại của ứng dụng tại điểm dừng đó như thể hiện trong hình bên dưới.



Hình trên cho thấy **Gỡ lỗi** cửa sổ với **Trình gỡ lỗi** và **Bảng điều khiển** tab. Các **Trình gỡ lỗi** tab được chọn, hiển thị **Trình gỡ lỗi** bảng điều khiển có các tính năng sau:

1. Khung tab: Nhập để hiển thị **Khung** ngắt với các khung ngắt xếp thực thi hiện tại cho một luồng đã cho. Ngắt xếp thực thi hiển thị từng lớp và phương thức đã được gọi trong ứng dụng của bạn và trong thời gian chạy Android, với phương thức gần đây nhất ở trên cùng.

Nhấp vào **Chủ đề** tab để thay thế **Khung** cửa sổ với **Chủ đề** ngắt. Ứng dụng của bạn hiện đang chạy trong luồng chính và ứng dụng đang thực thi tính toán() phương pháp trong Hoạt động chính.

2. Đồng hồ cái nút: Nhập để hiển thị **Đồng hồ** khung bên trong **Biến số khung**, hiển thị giá trị cho bất kỳ biến nào bạn đã thiết lập sẽ theo dõi. Theo dõi cho phép bạn theo dõi một biến cụ thể trong chương trình của mình và xem biến đó thay đổi như thế nào khi chương trình của bạn chạy.

3. Biến số khung cửa sổ: Hiển thị các biến trong phạm vi hiện tại và giá trị của chúng. Ở giai đoạn thực thi ứng dụng này, các biến khả dụng là: cái này (cho Hoạt động), toán tử (tên nhà điều hành từ Máy tính). Toán tử mà phương thức được gọi từ, cũng như các biến toàn cục cho Sửa văn bản các yếu tố và Chế độ xem văn bản. Mỗi biến trong ngăn này có một biểu tượng mở rộng để mở rộng danh sách các thuộc tính đối tượng cho biến đó. Hãy thử mở rộng một biến để khám phá các thuộc tính của nó.

2.3 Tiếp tục thực thi ứng dụng của bạn

Tiếp tục thực thi ứng dụng của bạn bằng cách chọn **Chạy > Tiếp tục chương trình**, hoặc nhấp vào **Bản tóm tắt** ở phía bên trái của cửa sổ gỡ lỗi.



biểu tượng

Ứng dụng SimpleCalc vẫn tiếp tục chạy và bạn có thể tương tác với ứng dụng cho đến khi mã thực thi đến điểm dừng.

2.4 Gỡ lỗi ứng dụng đang chạy

Nếu ứng dụng của bạn đang chạy trên thiết bị hoặc trình giả lập và bạn quyết định muốn gỡ lỗi ứng dụng đó, bạn có thể chuyển ứng dụng đang chạy sang chế độ gỡ lỗi.



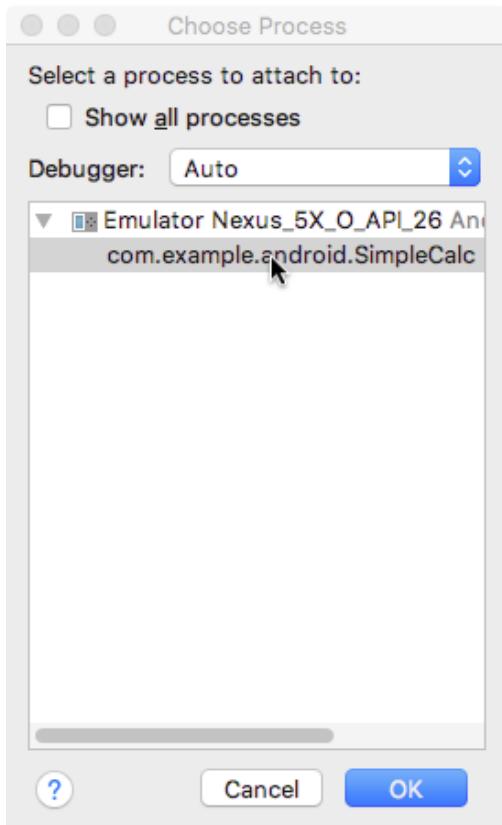
1.Chạy ứng dụng SimpleCalc bình thường, với **Chạy**



biểu tượng trong

2.Lựa chọn **Chạy > Đính kèm trình gỡ lỗi vào quy trình Android** hoặc nhấp vào **Gắn** thanh công cụ.

3.Chọn quy trình của ứng dụng của bạn từ hộp thoại xuất hiện (hiển thị bên dưới). Nhấp vào **ĐƯỢC RỒI**.



Các **Gỡ lỗi** xuất hiện với **Trình gỡ lỗi** mở cửa sổ và bây giờ bạn có thể gỡ lỗi ứng dụng của mình như thể bạn đã khởi động nó ở chế độ gỡ lỗi.

Ghi chú: Nếu **Gỡ lỗi** không tự động xuất hiện, hãy nhấp vào **Gỡ lỗi** tab ở cuối màn hình. Nếu nó chưa được chọn, hãy nhấp vào **Trình gỡ lỗi** tab trong **Gỡ lỗi** để hiển thị **Trình gỡ lỗi** khung cửa sổ.

Nhiệm vụ 3: Khám phá các tính năng gỡ lỗi

Trong nhiệm vụ này, chúng ta sẽ khám phá nhiều tính năng khác nhau trong trình gỡ lỗi Android Studio, bao gồm thực thi từng dòng ứng dụng, làm việc với điểm dừng và kiểm tra các biến.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

3.1 Thực hiện từng bước ứng dụng của bạn

Sau điểm dừng, bạn có thể sử dụng trình gỡ lỗi để thực thi từng dòng mã trong ứng dụng của mình và kiểm tra trạng thái của các biến khi ứng dụng chạy.

- 1.Gỡ lỗi ứng dụng của bạn trong Android Studio bằng điểm dừng mà bạn đặt ở tác vụ cuối cùng.
- 2.Trong ứng dụng, nhập số vào cả hai Sửa văn bản các yếu tố và nhấp vào **Thêm** và **cái** nút.

Việc thực thi ứng dụng của bạn dừng lại ở điểm dừng mà bạn đã đặt trước đó và **Trình gỡ lỗi** hiển thị trạng thái hiện tại của ứng dụng. Dòng hiện tại được tô sáng trong mã của bạn.

- 3.Nhấp vào **Bước qua**  nút ở đầu cửa sổ gỡ lỗi.

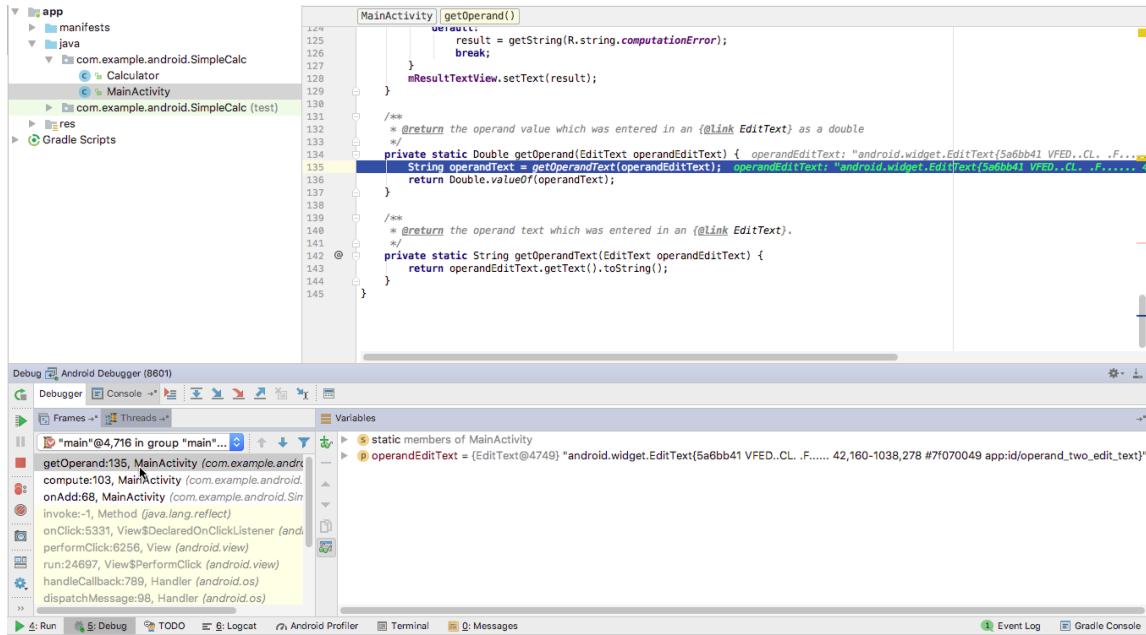
Trình gỡ lỗi thực thi dòng hiện tại trong tính toán() phương pháp (nơi điểm dừng là, việc chỉ định chuoán hạngMột), và điểm nối bật di chuyển đến dòng tiếp theo trong mã (nhiệm vụ chuoán hạngHai). Các **Biến** sẽ cập nhật để phản ánh trạng thái thực thi mới và các giá trị hiện tại của biến cũng xuất hiện sau mỗi dòng mã nguồn của bạn bằng chữ in nghiêng.

Bạn cũng có thể sử dụng **Chạy > Bước qua**, hoặc nhấn **F8**, để bước qua mã của bạn.

- 4.Ở dòng tiếp theo (nhiệm vụ chuoán hạngHai), nhấp vào **Bước vào** 

Bước vào hay vào việc thực hiện một lệnh gọi phương thức trong dòng hiện tại (so với việc chỉ thực hiện phương thức đó và vẫn ở trên cùng một dòng). Trong trường hợp này, vì lệnh gán đó bao gồm một lệnh gọi đến **Lấy Toán hạng()**, trình gỡ lỗi cuộn **Hoạt động chính** mã cho định nghĩa phương thức đó.

Khi bạn bước vào một phương pháp, **Khung** cập nhật ngắn để chỉ ra khung mới trong ngắn xếp cuộc gọi (ở đây, **Lấy Toán hạng()**), và **Biến** sẽ hiển thị các biến có sẵn trong phạm vi phương pháp mới. Bạn có thể nhấp vào bất kỳ dòng nào trong **Khung** để xem điểm trong khung ngắn xếp trước đó nơi phương thức được gọi.



Bạn cũng có thể sử dụng **Chạy > Bước vào**, hoặc F7, để bước vào một phương pháp.

1.Nhấp chuột **Bước qua** để chạy từng dòng tronglấy Toán tử().Lưu ý rằng khi phương pháp hoàn tất trình gỡ lỗi sẽ đưa bạn trở lại điểm đầu tiên bạn bước vào phương pháp và tất cả các bảng điều khiển sẽ cập nhật để hiển thị thông tin mới.

2.Nhấp chuột **Bước qua** hai lần để di chuyển điểm thực hiện đến dòng đầu tiên bên trongtrường hợptuyên bố cho **THÊM VÀO**.

3.Nhấp chuột **Bước vào** .

Trình gỡ lỗi thực hiện phương pháp thích hợp được xác định trongMáy tínhlớp học, mở Máy tính.java file và cuộn đến điểm thực thi trong lớp đó. Một lần nữa, các ngăn khác nhau cập nhật để phản ánh trạng thái mới.

4.Sử dụng **Bước ra ngoài** biểu tượng để thực hiện phần còn lại của phương pháp tính toán đó và bật lên quay trở lại tính toán()phương pháp trongHoạt động chính.Sau đó bạn có thể tiếp tục gỡ lỗi tính toán()phương pháp từ nơi bạn dừng lại.

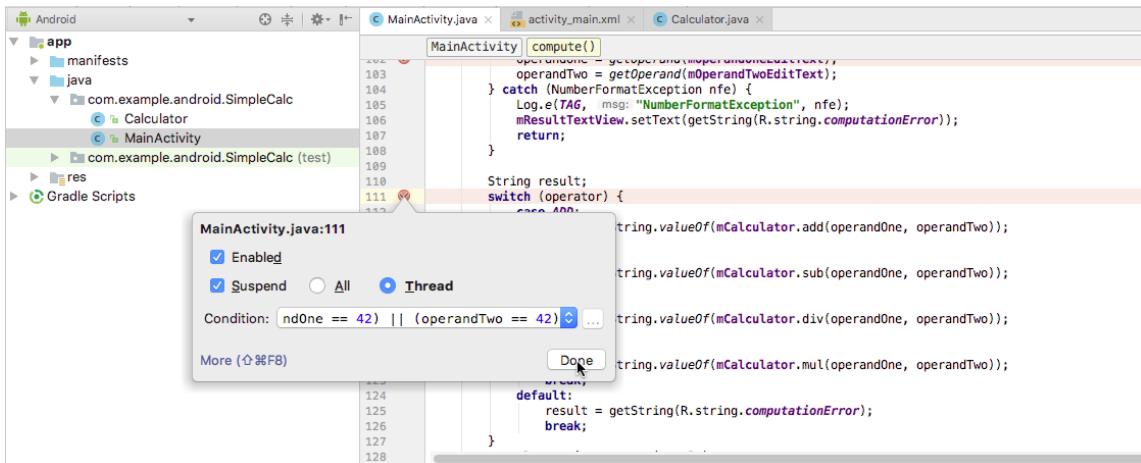
Bạn cũng có thể sử dụng **Chạy > Bước ra** hoặc nhấn **Shift-F8**để thoát khỏi việc thực hiện phương thức.

3.2 Làm việc với Điểm dừng

Sử dụng điểm dừng để chỉ ra vị trí trong mã mà bạn muốn ngắt quá trình thực thi ứng dụng để gỡ lỗi phần đó của ứng dụng.

- 1.Tìm điểm dừng bạn đặt trong tác vụ cuối cùng—khi bắt đầu tính toán() phương pháp trong Hoạt động chính.
- 2.Thêm một điểm dừng vào đầu công tắc duyên bố.
- 3.Nhấp chuột phải vào điểm dừng mới đó để nhập điều kiện, như thể hiện trong hình bên dưới, và nhập thử nghiệm sau vào **Tình trạng** có lĩnh vực:

(operandOne == 42) | | (operandTwo == 42)



- 4.Nhấp chuột **Xong**.

Điểm dừng thứ hai này là một **có điều kiện** breakpoint. Việc thực thi ứng dụng của bạn sẽ chỉ dừng lại tại breakpoint này nếu phép thử trong điều kiện là đúng. Trong trường hợp này, biểu thức chỉ đúng nếu một hoặc các toán hạng khác mà bạn nhập là **42**. Bạn có thể nhập bất kỳ biểu thức Java nào làm điều kiện miễn là nó trả về một boolean.

- 5.Chạy ứng dụng của bạn ở chế độ gỡ lỗi (**Chạy > Gỡ lỗi**), hoặc nhấp vào **Bản tóm tắt**  nếu nó đã chạy. Trong ứng dụng, nhập hai số khác 42 và nhấp vào **Thêm và** nút. Thực hiện dừng lại ở điểm dừng đầu tiên trong tính toán() phương pháp.

- 6.Nhấp chuột**Bản tóm tắt**để tiếp tục gỡ lỗi ứng dụng. Lưu ý rằng quá trình thực thi không dừng lại ở điểm dừng thứ hai của bạn vì điều kiện không được đáp ứng.
- 7.Trong ứng dụng, nhập**42**trong lần đầu tiênSửa văn bảnvà nhấp vào bất kỳCái nút.Nhấp chuột**Bản tóm tắt**để tiếp tục thực hiện sau điểm dừng đầu tiên. Lưu ý rằng điểm dừng thứ hai tại công tắc tuyênbối—các có điều kiệnđiểm dừng—dừng thực thi vì điều kiện đã được đáp ứng.
- 8.Nhấp chuột phái(hoặcControl-nhấp chuột)điểm dừng đầu tiên trong tính toán(và bỏ chọnĐã bật. Nhấp chuột **Xong**. Lưu ý rằng biểu tượngđiểm dừnghiện có một chấm màu xanh lá cây với đường viền màu đỏ.

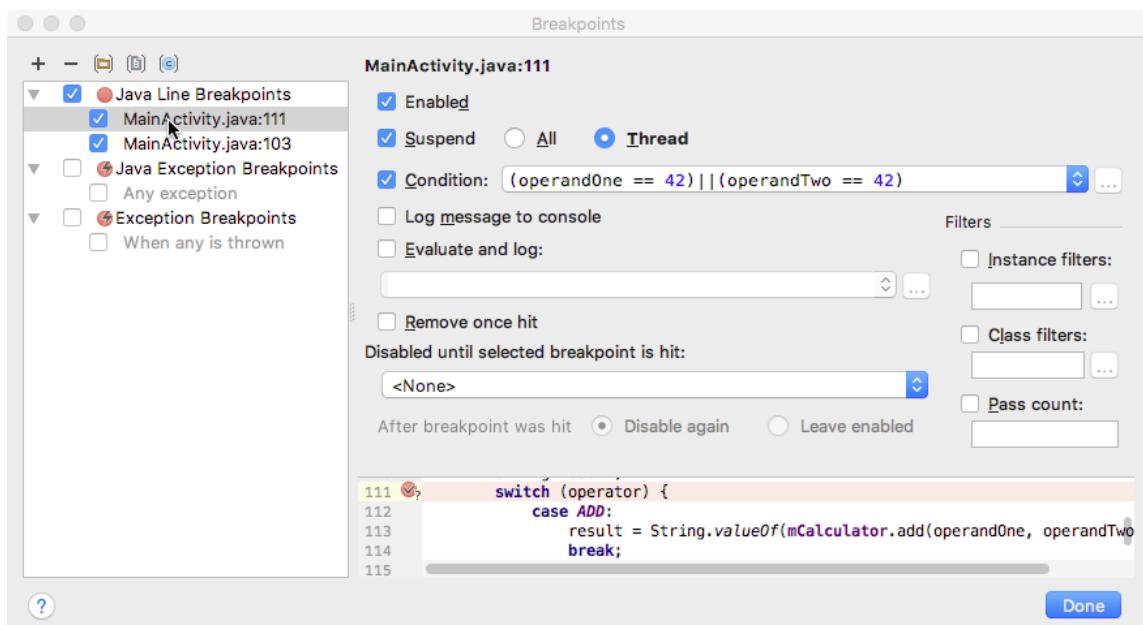
Vô hiệu hóa một breakpoint cho phép bạn tạm thời "tắt tiếng" breakpoint đó mà không thực sự xóa nó khỏi mã của bạn. Nếu bạn xóa hoàn toàn một breakpoint, bạn cũng sẽ mất mọi điều kiện bạn đã tạo cho breakpoint đó, vì vậy vô hiệu hóa nó thường là lựa chọn tốt hơn.

Bạn cũng có thể tắt tiếng tất cả các điểm dừng trong ứng dụng của mình cùng một lúc bằng**Tắt tiếng điểm ngắt** biểu tượng.



- 9.Nhấp chuột**Xem Điểm ngắt** ở cạnh trái của cửa sổ gỡ lỗi.**Điểm ngắt** cửa sổ xuất hiện.

Các**Điểm ngắt**cửa sổ cho phép bạn xem tất cả các điểm dừng trong ứng dụng của mình, bật hoặc tắt từng điểm dừng và thêm các tính năng bổ sung của điểm dừng bao gồm điều kiện, sự phụ thuộc vào các điểm dừng khác và ghi nhật ký.



Để đóng**Điểm ngắt**cửa sổ, nhấp chuột**Xong**.

3.3 Kiểm tra và sửa đổi các biến

Trình gỡ lỗi Android Studio cho phép bạn kiểm tra trạng thái của các biến trong ứng dụng khi ứng dụng đó chạy.

1.Chạy ứng dụng SimpleCalc ở chế độ gỡ lỗi nếu ứng dụng chưa chạy.

2.Trong ứng dụng, nhập hai số, một trong số đó **42** và nhấp vào **Thêm** và **cái** nút.

Điểm dừng đầu tiên trong tính toán (vẫn bị tắt tiếng). Thực hiện dừng lại ở điểm dừng thứ hai (điểm dừng có điều kiện tại công tắc câu lệnh) và trình gỡ lỗi sẽ xuất hiện.

3.Quan sát trong **Biến** khung mà toán hạng Một Vào toán hạng Hai biến có các giá trị bạn đã nhập vào ứng dụng.

The screenshot shows the Android Studio debugger interface. The top half displays the Java code for a switch statement:

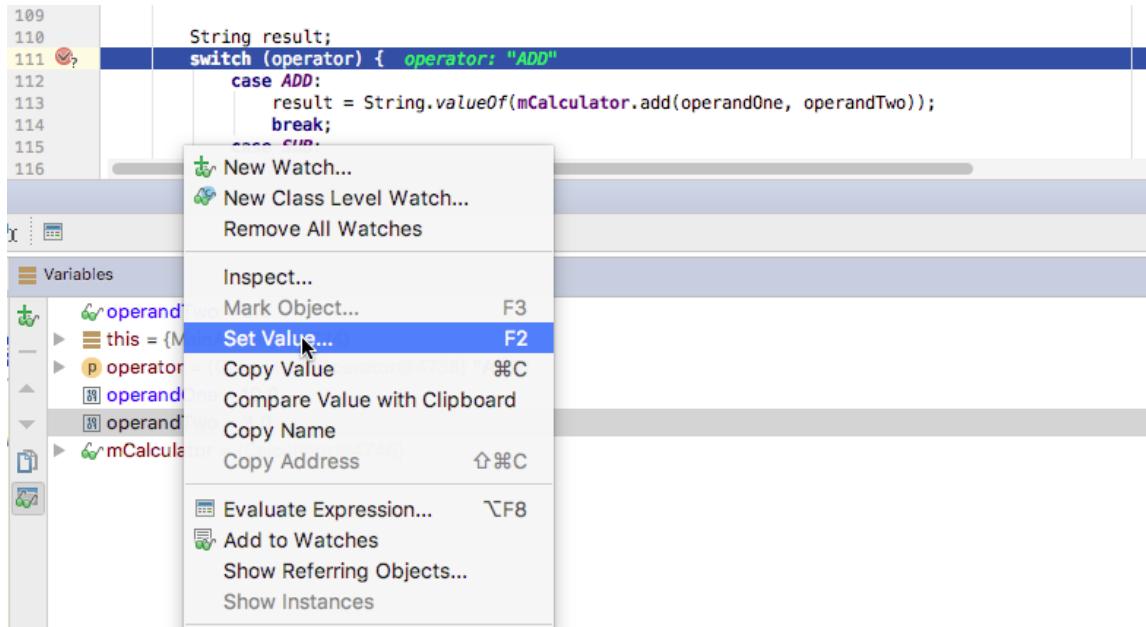
```
109
110
111     String result;
112     switch (operator) { operator: "ADD"
113         case ADD:
114             result = String.valueOf(mCalculator.add(operandOne, operandTwo));
115             break;
116         case SUB:
```

The line `case ADD:` is highlighted in blue, indicating it is the current point of execution. The bottom half shows the **Variables** tool window with the following variable list:

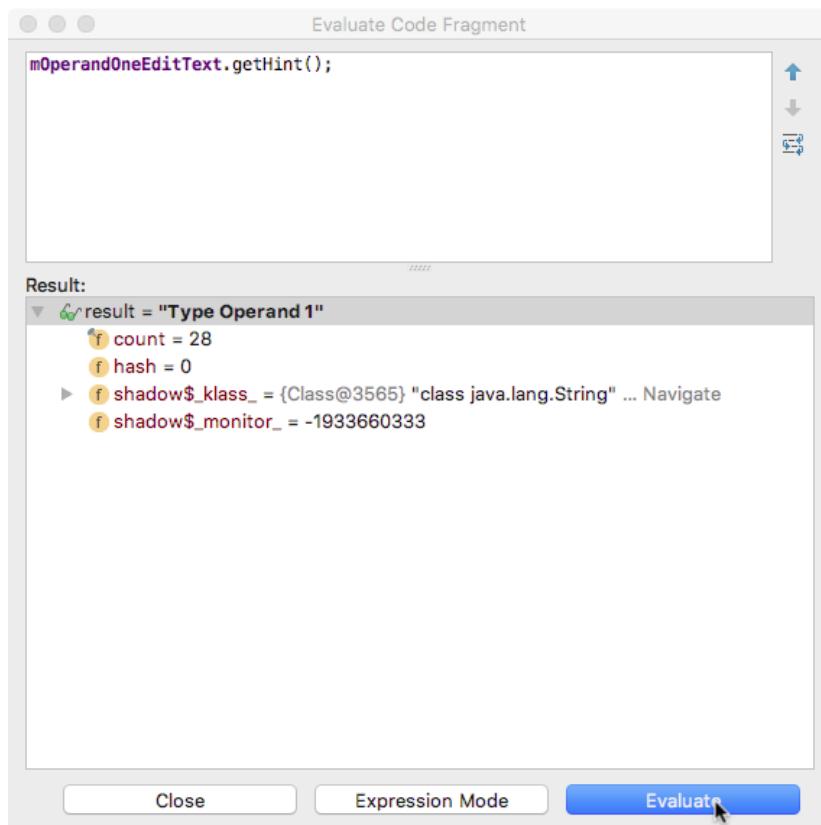
variable	value
<code>operandTwo</code>	3.0
<code>this</code>	{MainActivity@4734}
<code>operator</code>	{Calculator\$Operator@4738} "ADD"
<code>operandOne</code>	42.0
<code>operandTwo</code>	3.0
<code>mCalculator</code>	{Calculator@4745}

4.Các biến này là một **Hoạt động chính đối tượng**. Nhấp vào biểu tượng mở rộng để xem danh sách các biến thành viên của đối tượng đó. Nhấp vào biểu tượng mở rộng một lần nữa để đóng danh sách.

5.Nhấp chuột phải (hoặc **Control-nhấp chuột**) vào **toán hạng Một** biến trong **Biến** sổ găt và chọn **Đặt giá trị**.



- 6.Thay đổi giá trị của toán hạng Môn ĐỀN 10 và nhấn **Trở lại**.
- 7.Thay đổi giá trị của toán hạng Hai ĐỀN 10 theo cùng một cách và nhấn **Trở lại**.
- 8.Lưu ý rằng kết quả trong ứng dụng hiện dựa trên các giá trị biến bạn đã thay đổi trong trình gõ lỗi; ví dụ, vì bạn đã nhấp vào **Thêm vào** Cái nút ở Bước 2, kết quả trong ứng dụng bây giờ là 20.
- 9.Nhấp vào **Bản tóm tắt** biểu tượng để tiếp tục chạy ứng dụng của bạn.
- 10.Trong ứng dụng, các mục nhập gốc (bao gồm **42**) được bảo quản trong **Sửa văn bản** các yếu tố. (Của họ giá trị chỉ được thay đổi trong trình gõ lỗi.) Nhấp vào **Thêm vào** Nút. Quá trình thực thi lại dừng tại điểm dừng.
- 11.Nhấp vào **Đánh giá biểu thức** biểu tượng, hoặc chọn **Chạy > Đánh giá biểu thức**. Bạn cũng có thể **nhấp chuột phải** (hoặc **Control-nhấp chuột**) bất kỳ biến nào và chọn **Đánh giá biểu thức**.
Các **Đánh giá đoạn mã** cửa sổ xuất hiện. Sử dụng nó để khám phá trạng thái của các biến và đối tượng trong ứng dụng của bạn, bao gồm cả việc gọi các phương thức trên các đối tượng đó. Bạn có thể nhập bất kỳ mã nào vào cửa sổ này.



- 12.Nhập câu lệnh **mOperandOneEditText.getHint();** vào trường trên cùng của **Đánh giá đoạn mã** cửa sổ (như thể hiện trong hình trên) và nhấp vào **Đánh giá**.
13.Trường Result hiển thị kết quả của biểu thức đó. Gợi ý cho điều này Sửa văn bản là chuỗi "Kiểu toán hạng 1", như đã được định nghĩa ban đầu trong XML cho điều đó Chính sửa văn bản.

Kết quả bạn nhận được từ việc đánh giá một biểu thức dựa trên trạng thái hiện tại của ứng dụng. Tùy thuộc vào giá trị của các biến trong ứng dụng tại thời điểm bạn đánh giá biểu thức, bạn có thể nhận được các kết quả khác nhau.

Lưu ý rằng nếu bạn sử dụng **Đánh giá biểu thức** để thay đổi giá trị của biến hoặc thuộc tính đối tượng, bạn thay đổi trạng thái đang chạy của ứng dụng.

- 14.Nhấp chuột **Đóng** để đóng **Đánh giá đoạn mã** cửa sổ.

Thử thách mã hóa

Ghi chú:Mọi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách:Vào cuối Nhiệm vụ 1, bạn đã thử chạy ứng dụng SimpleCalc mà không có giá trị nào trong một trong các Sửa văn bảncác phần tử, dẫn đến lỗi. Sử dụng trình gỡ lỗi để thực hiện từng bước mã và xác định chính xác lý do tại sao lỗi này xảy ra. Sửa lỗi gây ra lỗi này.

Bản tóm tắt

- Xem thông tin ghi nhật ký trong Android Studio bằng cách nhấp vào**Logcat** tab.
- Chạy ứng dụng của bạn ở chế độ gỡ lỗi bằng cách nhấp vào biểu tượng Gỡ lỗi hoặc chọn**Chạy > Gỡ lỗi ứng dụng**.
- Nhấp vào**Gỡ lỗi** tab để hiển thị**Gỡ lỗi** ngăn. Nhấp vào**Trình gỡ lỗi** tab trong**Gỡ lỗi** ngăn để hiển thị **Trình gỡ lỗi** ngăn (nếu chưa được chọn).
- Các**Trình gỡ lỗi** ngăn hiển thị (ngăn xếp)**Khung**,**Biến số** trong một khung cụ thể, và**Đồng hồ** (theo dõi chủ động một biến trong khi chương trình đang chạy).
- Điểm dừng là một vị trí trong mã của bạn mà bạn muốn tạm dừng thực thi bình thường của ứng dụng để thực hiện các hành động khác. Đặt hoặc xóa điểm dừng gỡ lỗi bằng cách nhấp vào rãnh bên trái của cửa sổ trình soạn thảo ngay bên cạnh dòng mục tiêu

Khái niệm liên quan

Tài liệu khái niệm liên quan có trong[3.1: Trình gỡ lỗi Android Studio](#) .

Tìm hiểu thêm

Tài liệu Android Studio:

- [Hướng dẫn sử dụng Android Studio](#)
- [Gỡ lỗi ứng dụng của bạn](#)
- [Viết và Xem Nhật ký](#)
- [Phân tích một Stack Trace](#)
- [Cầu qữ lỗi Android](#)
- [Trình tạo hồ sơ Android](#)
- [Trình tạo hồ sơ mạng](#)
- [Bộ định hình CPU](#)
- [Theo dõi](#)

Khác:

- Băng hình:[Gỡ lỗi và kiểm tra trong Android Studio](#)

Bài tập về nhà

Xây dựng và chạy một ứng dụng

Mở [Tính toán đơn giản](#) ứng dụng.

1. Trong Hoạt động chính, đặt một điểm dừng ở dòng đầu tiên của `trên` `Thêm()` phương pháp.
2. Chạy ứng dụng trong trình gỡ lỗi. Thực hiện thao tác thêm trong ứng dụng. Quá trình thực thi dừng tại điểm dừng.
3. Nhấp vào **Bước vào** để theo dõi quá trình thực hiện ứng dụng từng bước. Lưu ý rằng **Bước vào** mở và thực thi các tệp từ nền tảng Android, cho phép bạn xem cách Android hoạt động trên mã của bạn.
4. Kiểm tra cách thức **Gỡ lỗi** ngăn thay đổi khi bạn thực hiện từng bước trong mã cho khung ngăn xếp hiện tại và các biến cục bộ.
5. Kiểm tra cách chú thích mã trong khung soạn thảo khi mỗi dòng được thực thi.
6. Nhấp vào **Bước ra** ngoài để quay lại ứng dụng của bạn nếu ngăn xếp thực thi quá sâu để hiểu.

Trả lời những câu hỏi này

Câu hỏi 1

Chạy ứng dụng SimpleCalc mà không có trình gỡ lỗi. Đề lại một hoặc cả hai Sửa văn bản các phần tử rỗng và thử bất kỳ phép tính nào. Tại sao lỗi xảy ra?

- java.lang.NumberFormatException: Chuỗi rỗng
- W/OpenGLRenderer: Không thể chọn cấu hình với EGL_SWAP_BEHAVIOR_PRESERVED
- Ứng dụng có thể đang làm quá nhiều việc trên luồng chính.
- Dung lượng bộ nhớ đệm mã được tăng lên tới 128KB.

Câu hỏi 2

Bạn thực hiện chức năng nào trong ngăn Gỡ lỗi để thực thi dòng hiện tại tại điểm dừng, sau đó dừng ở dòng tiếp theo trong mã? Chọn một trong các chức năng sau:

- **Bước vào**
- **Bước qua**
- **Bước ra ngoài**
- **Bản tóm tắt**

Câu hỏi 3

Bạn thực hiện chức năng nào trong ngăn Gỡ lỗi để chuyển đến phần thực thi lệnh gọi phương thức từ dòng hiện tại nơi có điểm dừng? Chọn một trong các chức năng sau:

- **Bước vào**
- **Bước qua**
- **Bước ra ngoài**
- **Bản tóm tắt**

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Không có ứng dụng nào để nộp bài tập về nhà này.

Bài 3.2: Kiểm tra đơn vị

Giới thiệu

Kiểm thử mã của bạn có thể giúp bạn phát hiện lỗi sớm trong quá trình phát triển, khi lỗi ít tốn kém nhất để giải quyết.

Khi ứng dụng của bạn lớn hơn và phức tạp hơn, việc kiểm thử sẽ cải thiện độ mạnh mẽ của mã. Với các bài kiểm thử trong mã, bạn có thể thực hiện các phần nhỏ của ứng dụng một cách riêng biệt và bạn có thể kiểm thử theo cách có thể tự động hóa và lặp lại.

Android Studio và Android Testing Support Library hỗ trợ nhiều loại thử nghiệm và khuôn khổ thử nghiệm khác nhau. Trong bài thực hành này, bạn sẽ khám phá chức năng thử nghiệm tích hợp của Android Studio và học cách viết và chạy các thử nghiệm đơn vị cụb bộ.

Kiểm tra đơn vị *cực bộ* là các bài kiểm tra được biên dịch và chạy hoàn toàn trên máy cục bộ của bạn với Java Virtual Machine (JVM). Bạn sử dụng các bài kiểm tra đơn vị cục bộ để kiểm tra các phần của ứng dụng không cần truy cập vào khuôn khổ Android hoặc thiết bị hoặc trình giả lập chạy bằng Android, ví dụ như logic bên trong. Bạn cũng sử dụng các bài kiểm tra đơn vị cục bộ để kiểm tra các phần của ứng dụng mà bạn có thể tạo các đối tượng giả ("mock" hoặc stub) giả vờ hoạt động giống như các đối tượng tương đương của khuôn khổ.

Các bài kiểm tra đơn vị được viết bằng JUnit, một khuôn khổ kiểm tra đơn vị phổ biến cho Java.

Những điều bạn nên biết

Bạn sẽ có thể:

- Tạo một dự án Android Studio.
- Xây dựng và chạy ứng dụng của bạn trong Android Studio, trên cả trình giả lập và trên thiết bị.
- Điều hướng **Dự án > Android** trong Android Studio.

- Tìm các thành phần chính của một dự án Android Studio, bao gồm `AndroidManifest.xml`, tài nguyên, tệp Java và tệp Gradle.

Những gì bạn sẽ học được

- Cách tổ chức và chạy thử nghiệm trong Android Studio.
- Hiểu bài kiểm tra đơn vị là gì.
- Viết các bài kiểm tra đơn vị cho mã của bạn.

Bạn sẽ làm gì

- Chạy thử nghiệm ban đầu trong ứng dụng SimpleCalc.
- Thêm nhiều bài kiểm tra hơn vào ứng dụng SimpleCalc.
- Chạy thử nghiệm đơn vị để xem kết quả.

Tổng quan về ứng dụng

Thực tế này sử dụng [Tính toán đơn giản](#) ứng dụng từ codelab thực hành trước đó ([Cơ bản về Android 3.1: Trình gỡ lỗi](#)). Bạn có thể sửa đổi ứng dụng đó tại chỗ hoặc tạo một bản sao thư mục dự án trước khi tiếp tục.

Nhiệm vụ 1: Khám phá và chạy CalculatorTest

Bạn viết và chạy các bài kiểm tra của mình (cả bài kiểm tra đơn vị và bài kiểm tra có công cụ) bên trong Android Studio, cùng với mã cho ứng dụng của bạn. Mỗi dự án Android mới đều bao gồm các lớp mẫu cơ bản để kiểm tra mà bạn có thể mở rộng hoặc thay thế cho mục đích sử dụng của riêng mình.

Trong nhiệm vụ này, bạn sẽ quay lại ứng dụng SimpleCalc, bao gồm lớp kiểm thử đơn vị cơ bản.

1.1 Khám phá các tập nguồn và CalculatorTest

Bộ nguồn là các tập hợp mã trong dự án của bạn dành cho các mục tiêu xây dựng khác nhau hoặc các "hương vị" khác của ứng dụng của bạn. Khi Android Studio tạo dự án của bạn, nó sẽ tạo ra ba bộ nguồn:

- Các *chủ yếu* bộ nguồn dành cho mã và tài nguyên của ứng dụng.
- (Bài kiểm tra)bộ nguồn, cho các bài kiểm tra đơn vị cục bộ của ứng dụng của bạn. Bộ nguồn hiển thị (Bài kiểm tra)sau tên gói.
- (((Kiểm tra android)bộ nguồn, cho các thử nghiệm được thiết lập trên Android. Bộ nguồn hiển thị (Kiểm tra android)sau tên gói.

Trong nhiệm vụ này, bạn sẽ khám phá cách hiển thị bộ nguồn trong Android Studio, kiểm tra cấu hình Gradle để thử nghiệm và chạy các bài kiểm tra đơn vị cho ứng dụng SimpleCalc.

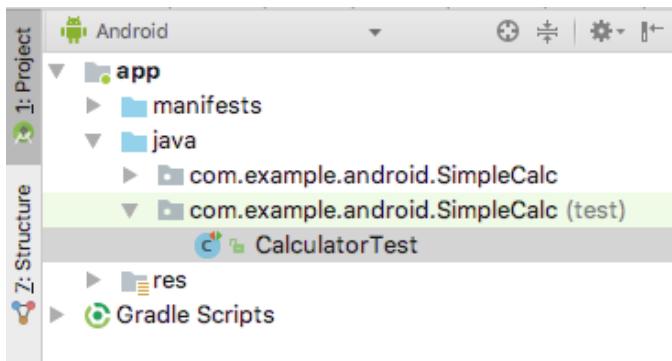
Ghi chú:(((Kiểm tra android)bộ nguồn đã được xóa khỏi ví dụ này để đơn giản hóa. Nó được giải thích chi tiết hơn trong bài học khác.

1.Mở [Tính toán đơn giản](#) dự án trong Android Studio, nếu bạn chưa thực hiện.

1.Mở **Dự án > Android** ngan và mở rộng **Ứng dụng**Và javathư mục.

Cácjavathư mục trong chế độ xem Android liệt kê tất cả các bộ nguồn trong ứng dụng theo tên gói.

Trong trường hợp này (như được hiển thị bên dưới), mã ứng dụng nằm trong com.android.example.SimpleCalc bộ nguồn. Mã kiểm tra nằm trong bộ nguồn với Bài kiểm tra xuất hiện trong dấu ngoặc đơn sau tên gói:com.android.example.SimpleCalc (kiểm tra).



2. Mở rộng com.android.example.SimpleCalc (kiểm tra) thư mục.

Thư mục này là nơi bạn đặt các bài kiểm tra đơn vị cục bộ của ứng dụng. Android Studio tạo một lớp kiểm tra mẫu cho bạn trong thư mục này cho các dự án mới, nhưng đối với SimpleCalc, lớp kiểm tra được gọi là CalculatorTest.

1. Mở Máy tính thử nghiệm.

Kiểm tra mã và lưu ý những điều sau:

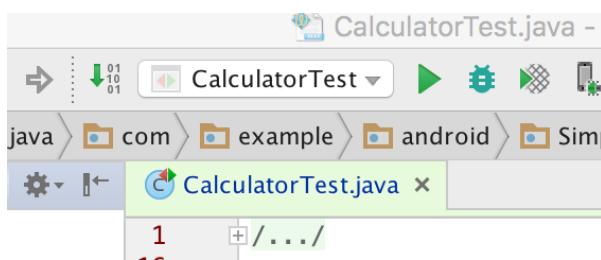
- Các mặt hàng nhập khẩu duy nhất là `org.junit`, `org.hamcrest`, và `android.test`. Không có sự phụ thuộc nào vào các khung Android.
- @Chạy với(`JUnit4.class`) chú thích chỉ ra trình chạy sẽ được sử dụng để chạy các bài kiểm tra trong lớp này. Trình chạy kiểm tra là một thư viện hoặc bộ công cụ cho phép thực hiện kiểm tra và in kết quả vào nhật ký. Đối với các bài kiểm tra có yêu cầu về thiết lập hoặc cơ sở hạ tầng phức tạp hơn (như Espresso), bạn sẽ sử dụng các trình chạy kiểm tra khác nhau. Đối với ví dụ này, chúng tôi đang sử dụng trình chạy kiểm tra JUnit4 cơ bản.
- @Kiểm tra nhỏ chú thích cho biết rằng tất cả các bài kiểm tra trong lớp này đều là các bài kiểm tra đơn vị không có sự phụ thuộc và chạy trong mili giây. @Kiểm tra nhỏ, @Kiểm tra trung bình, và @Kiểm tra lớn chú thích là những quy ước giúp dễ dàng hơn trong việc gom các nhóm bài kiểm tra thành các bộ có chức năng tương tự.
- Các `cài đặt()` phương pháp được sử dụng để thiết lập môi trường trước khi thử nghiệm và bao gồm `@Trước` chú thích. Trong trường hợp này, thiết lập tạo ra một phiên bản mới của Máy tính lớp và gán nó cho Máy tính biến thành viên.
- Các `thêm Hai Số()` phương pháp là một thử nghiệm thực tế và được chú thích bằng `@Bài kiểm tra`. Chỉ các phương thức trong lớp kiểm tra có `@Bài kiểm tra` chú thích được coi là các bài kiểm tra đối với người chạy thử nghiệm. Lưu ý rằng theo quy ước, các phương pháp kiểm tra không bao gồm từ "kiểm tra".
- Dòng đầu tiên của `thêm Hai Số()` gọi là `thêm vào()` phương pháp từ Máy tính lớp. Bạn chỉ có thể kiểm tra các phương pháp công cộng hoặc được bảo vệ bằng gói. Trong trường hợp này, Máy tính là một lớp công cộng học với công cộng phương pháp, nên mọi việc đều ổn.
- Dòng thứ hai là khẳng định cho bài kiểm tra. Khẳng định là những biểu thức phải đánh giá và đưa ra kết quả ĐÚNG VẬY để bài kiểm tra vượt qua. Trong trường hợp này, khẳng định là kết quả bạn nhận được từ phương pháp cộng ($1 + 1$) khớp với số đã cho là 2. Bạn sẽ tìm hiểu thêm về cách tạo khẳng định sau trong phần thực hành này.

1.2 Chạy thử nghiệm trong Android Studio

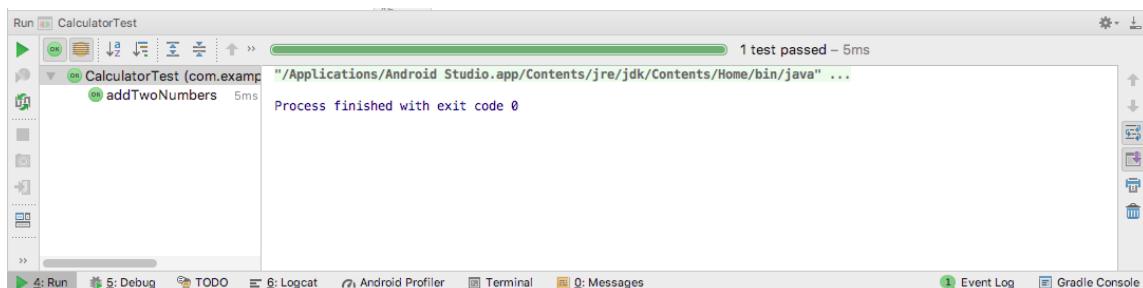
Trong nhiệm vụ này, bạn sẽ chạy các bài kiểm tra đơn vị trong thư mục kiểm tra và xem kết quả của cả các bài kiểm tra thành công và không thành công.

1.Trong **Dự án > Android**cửa sổ,**nhấp chuột phải**(hoặc**Control-nhấp chuột**)**Máy tính thử nghiệm**và chọn**Chạy 'CalculatorTest'**.

Dự án xây dựng, nếu cần thiết, và**Máy tính thử nghiệm**ngăn xuất hiện ở dưới cùng của màn hình. Ở đầu ngăn, danh sách thả xuống cho các cấu hình thực thi có sẵn cũng thay đổi thành**Máy tính thử nghiệm**.



Tất cả các bài kiểm tra trong lớp CalculatorTest đều chạy và nếu các bài kiểm tra đó thành công, thanh tiến trình ở đầu chế độ xem sẽ chuyển sang màu xanh lá cây. (Trong trường hợp này, hiện tại chỉ có một bài kiểm tra.) Một thông báo trạng thái ở chân trang cũng báo cáo "Bài kiểm tra đã vượt qua".



1.Mở**Máy tính thử nghiệm**nếu nó chưa được mở và thay đổi khẳng định trong **HaiSố()** ĐẾN:

```
assertThat(resultAdd, là(bằng(3d)));
```

2.Trong menu thả xuống cấu hình chạy ở đầu màn hình, hãy chọn **Máy tính thử nghiệm**(nếu như nó chưa được chọn) và nhấp vào **Chạy** .

Kiểm tra lại chạy như trước, nhưng lần này khẳng định không thành công (3 không bằng 1 + 1). Thanh tiến trình trong chế độ xem chạy chuyển sang màu đỏ và nhật ký kiểm tra cho biết kiểm tra (khẳng định) không thành công ở đâu và tại sao.

3.Thay đổi khẳng định trong `HaiSo()` quay lại bài kiểm tra chính xác và chạy lại bài kiểm tra để đảm bảo chúng đạt yêu cầu.

4.Trong danh sách thả xuống cấu hình chạy, hãy chọn **Ứng dụng** để chạy ứng dụng của bạn một cách bình thường.

Nhiệm vụ 2: Thêm nhiều bài kiểm tra đơn vị vào CalculatorTest

Với thử nghiệm đơn vị, bạn lấy một đoạn mã nhỏ trong ứng dụng của mình như phương thức hoặc lớp và tách nó khỏi phần còn lại của ứng dụng, để các thử nghiệm bạn viết đảm bảo rằng một đoạn mã nhỏ hoạt động theo cách bạn mong đợi. Thông thường, một thử nghiệm đơn vị gọi một phương thức với nhiều điều vào khác nhau và xác minh rằng phương thức thực hiện những gì bạn mong đợi và trả về những gì bạn mong đợi.

Trong nhiệm vụ này, bạn sẽ tìm hiểu thêm về cách xây dựng các bài kiểm tra đơn vị. Bạn sẽ viết các bài kiểm tra đơn vị bổ sung cho Máy tính các phương pháp tiện ích trong ứng dụng SimpleCalc và chạy các thử nghiệm đó để đảm bảo chúng tạo ra kết quả như bạn mong đợi.

Ghi chú: Kiểm thử đơn vị, phát triển theo hướng kiểm thử và API JUnit 4 đều là những chủ đề lớn và phức tạp, nằm ngoài phạm vi của khóa học này.

2.1 Thêm nhiều bài kiểm tra hơn cho phương thức add()

Mặc dù không thể kiểm tra mọi giá trị có thể có thêm vào() phương pháp có thể thấy, tốt nhất là nên kiểm tra đầu vào có thể bất thường. Ví dụ, hãy xem xét điều gì xảy ra nếu thêm vào() phương pháp nhận được các đối số:

- Với toán hạng âm
- Với số dấu phẩy động

- Với số lượng lớn đặc biệt
- Với các toán hạng có kiểu khác nhau (ví dụ như float và double)
- Với toán hạng bằng không
- Với một toán hạng là vô cực

Trong nhiệm vụ này, chúng ta sẽ thêm nhiều bài kiểm tra đơn vị hơn cho thêm vào() phương pháp kiểm tra các loại đầu vào khác nhau.

1.Thêm một phương pháp mới vào Máy tính thử nghiệm gọi điệntêmTwoNumbersNegative(). Sử dụng bộ xương này:

```
@Bài kiểm tra  
công khai void addTwoNumbersNegative() { }
```

Phương pháp thử nghiệm này có cấu trúc tương tự như thêmHaiSố(): đó là một công cộng phương pháp, không có tham số, trả về vô hiệu. Nó được chú thích bằng @Bài kiểm tra, điều này cho thấy đây là một bài kiểm tra đơn vị.

Tại sao không chỉ thêm nhiều khẳng định hơn vào thêmHaiSố()? Việc nhóm nhiều hơn một khẳng định vào một phương pháp duy nhất có thể khiến các bài kiểm tra của bạn khó gỡ lỗi hơn nếu chỉ có một khẳng định không thành công và làm lu mờ các bài kiểm tra thành công. Quy tắc chung cho các bài kiểm tra đơn vị là cung cấp một phương pháp kiểm tra cho từng khẳng định riêng lẻ.

2.Chạy tất cả các bài kiểm tra trong Máy tính thử nghiệm, như trước đây.

Trong cửa sổ thử nghiệm cả hai thêmHaiSố() và thêmHaiSốÂm() đều được liệt kê là các bài kiểm tra có sẵn (và đạt yêu cầu) trong bảng bên trái. thêmHaiSốÂm() là bài kiểm tra vẫn vượt qua ngay cả khi nó không chứa bất kỳ mã nào—một bài kiểm tra không làm gì vẫn được coi là một bài kiểm tra thành công.

3.Thêm một dòng vào thêmTwoNumbersNegative() để triệu tập thêm vào() phương pháp trong Máy tính lớp có toán hạng âm.

```
double resultAdd = mCalculator.add(1d, 2d);
```

Các ngày kí hiệu sau mỗi toán hạng chỉ ra rằng đây là những số có kiểu gấp đôi. Bởi vì thêm vào() phương pháp được định nghĩa với các tham số kép, một trôi nổi hoặc số nguyên cũng sẽ hoạt động. Chỉ định loại rõ ràng cho phép bạn kiểm tra các loại khác riêng biệt, nếu bạn cần.

4.Thêm một khẳng định với `khẳng địnhThat()`.

```
assertThat(resultAdd, là(bằng(1d)));
```

Cách `khẳng địnhĐó()` phương pháp là một khẳng định JUnit4 tuyên bố rằng biểu thức trong đối số đầu tiên bằng với biểu thức trong đối số thứ hai. Các phiên bản cũ hơn của JUnit sử dụng các phương pháp `khẳng định cụ thể hơn` (`khẳng định bằng()`, `khẳng địnhNull()`, `hoặc khẳng định Đúng()`). Nhưng `khẳng địnhĐó()` là định dạng linh hoạt hơn, dễ gõ lỗi hơn và thường dễ đọc hơn.

Cách `khẳng địnhĐó()` phương pháp được sử dụng với *người so khớp*. Các bộ so khớp là các lệnh gọi phương thức được nối tiếp trong toán hạng thứ hai của khẳng định này, `là(bằng())`. Khung Hamcrest định nghĩa các bộ so khớp có sẵn mà bạn có thể sử dụng để xây dựng một khẳng định. ("Hamcrest" là một từ đảo chữ của "matchers.") Hamcrest cung cấp nhiều bộ so khớp cơ bản cho hầu hết các khẳng định cơ bản. Bạn cũng có thể định nghĩa các bộ so khớp tùy chỉnh của riêng mình cho các khẳng định phức tạp hơn.

Trong trường hợp này, khẳng định là kết quả của `thêm vào()` phép toán $(-1 + 2)$ bằng 1.

5.Thêm một bài kiểm tra đơn vị mới vào Máy tính thử nghiệm đối với số dấu phẩy động:

```
@Bài kiểm tra
công khai void addTwoNumbersFloats() {
    kết quả képThêm = mCalculator.thêm(1.111f, 1.111d); khẳng
    địnhĐó(kết quảThêm, là(bằng(2.222d)));
}
```

Một lần nữa, một bài kiểm tra rất giống với phương pháp kiểm tra trước đó, nhưng có một đối số `thêm vào()` đó là loại rõ ràng trôi nổi còn hơn là gấp đôi. Các `thêm vào()` phương pháp được định nghĩa với các tham số kiểu gấp đôi, vì vậy bạn có thể gọi nó bằng một trôi nổi loại, và số đó được tăng cấp lên gấp đôi.

6.Nhấp chuột  để chạy lại tất cả các bài kiểm tra.

Lần này thử nghiệm không thành công và thanh tiến trình có màu đỏ. Đây là phần quan trọng của thông báo lỗi:

```
java.lang.AssertionError: Dự kiến:  
    là <2.222>  
Nhưng:   đã từng là <2.2219999418258665>
```

Phép tính với số dấu phẩy động không chính xác và việc quảng bá dẫn đến tác dụng phụ là độ chính xác bổ sung. Khẳng định trong bài kiểm tra về mặt kỹ thuật là sai: giá trị mong đợi không bằng giá trị thực tế.

Câu hỏi đặt ra ở đây là: Khi bạn gặp vấn đề về độ chính xác khi quảng bá rồi đổi số, đó có phải là vấn đề với mã của bạn hay là vấn đề với bài kiểm tra của bạn không? Trong trường hợp cụ thể này, cả hai đổi số đều vào chothêm vào()phương pháp từ ứng dụng SimpleCalc sẽ luôn là loại gấp đôi, vì vậy đây là một bài kiểm tra tùy ý và không thực tế. Tuy nhiên, nếu ứng dụng của bạn được viết sao cho đầu vào chothêm vào()phương pháp có thể là gấp đôi hoặc trái ngược, và bạn chỉ quan tâm đến *một số* độ chính xác, bạn cần phải cung cấp một số khoảng trống cho thử nghiệm để "đủ gần" được coi là thành công.

7.Thay đổi khẳng định Đó() phương pháp sử dụng đóngTo() người so khớp:

```
khẳng định Đó(kết quả Thêm, là(gần(2,222, 0,01)));
```

Bạn cần phải đưa ra lựa chọn cho bộ so khớp. Nhấp vào **gần** hai lần (cho đến khi toàn bộ biểu thức được gạch chân) và nhấn Alt+Enter (Tùy chọn+Trả về trên máy Mac). Chọn **isCloseTo.closeTo (org.hamcrest.number)**.

8.Nhấp chuột **Chạy**  để chạy lại tất cả các bài kiểm tra.

Lần này bài kiểm tra đã thành công.

Với đóngTo() bộ so khớp, thay vì kiểm tra sự bình đẳng chính xác, bạn có thể kiểm tra sự bình đẳng trong một delta cụ thể. Trong trường hợp này, đóngTo() Phương thức matcher lấy hai đối số: giá trị mong đợi và lượng delta. Trong ví dụ trên, delta đó chỉ là độ chính xác hai chữ số thập phân.

2.2 Thêm các bài kiểm tra đơn vị cho các phương pháp tính toán khác

Sử dụng những gì bạn đã học trong nhiệm vụ trước để điền vào các bài kiểm tra đơn vị cho Máy tính lớp học.

- 1.Thêm một bài kiểm tra đơn vị được gọi làsubHaiSố()mà kiểm tra phụ()phương pháp.
- 2.Thêm một bài kiểm tra đơn vị được gọi làsubWorksWithNegativeResults()mà kiểm tra phụ()phương pháp trong đó phép tính đưa ra cho kết quả là một số âm.
- 3.Thêm một bài kiểm tra đơn vị được gọi làmulTwoNumbers()mà kiểm tra phu()phương pháp.
- 4.Thêm một bài kiểm tra đơn vị được gọi làmulTwoNumbersZero()mà kiểm tra phu()phương pháp có ít nhất một đối số là số không.
- 5.Thêm một bài kiểm tra đơn vị được gọi làdivTwoNumbers()mà kiểm tra phu()phương pháp có hai đối số khác không.
- 6.Thêm một bài kiểm tra đơn vị được gọi làdivTwoNumbersZero()mà kiểm tra phu()phương pháp với một gãy đôi số bị chia và số 0 là số chia.

Tất cả các bài kiểm tra này đều phải vượt qua, ngoại trừ divTwoNumbersZero()gây ra ngoại lệ đối số bất hợp pháp khi chia cho số không. Nếu bạn chạy ứng dụng, hãy nhập số không làm Toán hạng 2 và nhấp vào **Phân chia** để phép chia cho kết quả sai.

Mã giải bài tập 2

Dự án Android Studio:[Kiểm tra đơn giản](#)

Đoạn mã sau đây hiển thị các bài kiểm tra cho nhiệm vụ này:

```
@Bài kiểm tra
```

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

```
công khai void addTwoNumbers() {
    double resultAdd = mCalculator.add(1d, 1d);
    assertThat(resultAdd, is(equalTo(2d)));
}

@Bài kiểm tra
công khai void addTwoNumbersNegative() {
    double resultAdd = mCalculator.add(1d, 2d);
    assertThat(resultAdd, is(equalTo(1d)));
}

@Bài kiểm tra
công khai void addTwoNumbersFloats() {
    kết quả képThêm = mCalculator.thêm(1.111f, 1.111d); khẳng
địnhĐó(kết quảThêm, là(gần(2.222, 0.01))); }

@Bài kiểm tra
công khai void subTwoNumbers() {
    double resultSub = mCalculator.sub(1d, 1d);
    assertThat(resultSub, is(equalTo(0d)));
}

@Bài kiểm tra
công khai void subWorksWithNegativeResult() {
    kết quả képSub = mCalculator.sub(1ngày, 17ngày); khẳng
địnhĐó(resultSub, là(bằng(16ngày)));
}

@Bài kiểm tra
công khai void mulTwoNumbers() {
    kết quả képMul = mCalculator.mul(32d, 2d);
    assertThat(resultMul, là(bằng(64d)));
}

@Bài kiểm tra
công khai void divTwoNumbers() {
    kết quả képDiv = mCalculator.div(32d, 2d); khẳng
địnhĐó(resultDiv, là(bằng(16d)));
}

@Bài kiểm tra
công khai void divTwoNumbersZero() {
    kết quả képDiv = mCalculator.div(32d, 0); khẳng định rằng(kết quảDiv,
là(bằng(Double.POSITIVE_INFINITY)));
}
```

Thách thức mã hóa

Ghi chú:Mỗi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách 1: Chia cho số không luôn đáng để thử nghiệm, vì đây là trường hợp đặc biệt trong số học. Bạn có thể thay đổi ứng dụng như thế nào để xử lý chia cho số không một cách duyên dáng hơn? Để hoàn thành thử thách này, hãy bắt đầu bằng một bài kiểm tra cho thấy hành vi đúng đắn nên là gì.

Loại bỏ divTwoNumbersZero() phương pháp từ Máy tính thử nghiệm, và thêm một bài kiểm tra đơn vị mới được gọi là divByZeroThrows() mà kiểm tra div() phương pháp với đối số thứ hai bằng không, với kết quả mong đợi là Lớp ngoại lệ bất hợp pháp. Bài kiểm tra này sẽ thành công và kết quả là nó sẽ chứng minh rằng bất kỳ phép chia nào cho số không cũng sẽ dẫn đến ngoại lệ này.

Sau khi bạn học cách viết mã cho một [Ngoại lệ](#) trình xử lý, ứng dụng của bạn có thể xử lý ngoại lệ này một cách khéo léo bằng cách, ví dụ, hiển thị một [Nướng](#) thông báo cho người dùng để thay đổi Toán hạng 2 từ số không sang một số khác.

Thử thách 2:Đôi khi rất khó để cô lập một đơn vị mã khỏi tất cả các phụ thuộc bên ngoài của nó. Thay vì sắp xếp mã của bạn theo những cách phức tạp chỉ để bạn có thể kiểm tra nó dễ dàng hơn, bạn có thể sử dụng một khuôn khổ giả để tạo các đối tượng giả ("giả") giả vờ là các phụ thuộc. Nghiên cứu [Mockito](#) khung và tìm hiểu cách thiết lập nó trong Android Studio. Viết một lớp thử nghiệm cho Nút tính toán() phương pháp trong SimpleCalc và sử dụng Mockito để mô phỏng bối cảnh Android trong đó các bài kiểm tra của bạn sẽ chạy.

Bản tóm tắt

Android Studio có các tính năng tích hợp để chạy thử nghiệm đơn vị cục bộ:

- Kiểm thử đơn vị cục bộ sử dụng JVM của máy cục bộ của bạn. Chúng không sử dụng khuôn khổ Android.
- Các bài kiểm tra đơn vị được viết bằng JUnit, một khuôn khổ kiểm tra đơn vị phổ biến cho Java.
- Các bài kiểm tra JUnit nằm ở (Bài kiểm tra) thư mục trong Android Studio **Dự án > Android** khung cửa sổ.
- Các bài kiểm tra đơn vị cục bộ chỉ cần những gói sau: org.junit, org.hamcrest, và android.test.
- @Chạy với(JUnit4.class) chú thích yêu cầu trình chạy thử nghiệm chạy thử nghiệm trong lớp này.
- @Kiểm tra nhỏ, @Kiểm tra trung bình, và @Kiểm tra lớn chú thích là những quy ước giúp dễ dàng hơn trong việc nhóm các nhóm bài kiểm tra tương tự
- @Kiểm tra nhỏ chú thích cho biết tất cả các bài kiểm tra trong một lớp đều là các bài kiểm tra đơn vị không có sự phụ thuộc và chạy trong mili giây.
- Kiểm thử có công cụ là kiểm thử chạy trên thiết bị hoặc trình giả lập chạy Android.
Kiểm thử có công cụ có quyền truy cập vào khuôn khổ Android.
- Trình chạy thử nghiệm là một thư viện hoặc bộ công cụ cho phép thực hiện thử nghiệm và in kết quả vào nhật ký.

Khái niệm liên quan

Tài liệu khái niệm liên quan có trong [3.2: Kiểm tra ứng dụng](#).

Tìm hiểu thêm

Tài liệu Android Studio:

- [Hướng dẫn sử dụng Android Studio](#)
- [Viết và Xem Nhật ký](#)

Tài liệu dành cho nhà phát triển Android:

- [Thực hành tốt nhất để thử nghiệm](#)
- [Bắt đầu với thử nghiệm](#)
- [Xây dựng các bài kiểm tra đơn vị cục bộ](#)

Khác:

- [Trang chủ JUnit 4](#)
- [Tài liệu tham khảo API JUnit 4](#)
- [java.lang.Toán](#)
- [Java Hamcrest](#)
- [Trang chủ Mockito](#)
- [Bảng hình:Hỗ trợ kiểm tra Android - Mẫu kiểm tra](#)
- [Codelab kiểm thử Android](#)
- [Mẹo chuyên nghiệp về công cụ Android: Chú thích kích thước thử nghiệm](#)
- [Lợi ích của việc sử dụng assertThat so với các phương thức Assert khác trong các bài kiểm tra đơn vị](#)

Bài tập về nhà

Xây dựng và chạy một ứng dụng

Mở [Tính toán đơn giản](#) ứng dụng từ thực tế về việc sử dụng trình gõ lỗi. Bạn sẽ thêm một **BẤN** ... nút để bố trí. Nút tính toán hạng đầu tiên được nâng lên lũy thừa của toán hạng thứ hai. Ví dụ, cho các toán hạng là 5 và 4, ứng dụng sẽ tính toán 5^{4} được nâng lên lũy thừa của 4 hoặc 625.

Trước bạn viết phần triển khai nút nguồn của mình, hãy cân nhắc loại thử nghiệm bạn có thể muốn thực hiện bằng cách sử dụng phép tính này. Những giá trị bất thường nào có thể xảy ra trong phép tính này?

1. Cập nhật [Máy tính](#) lớp trong ứng dụng để bao gồm một [sức mạnh\(\)](#) phương pháp. Gợi ý: Tham khảo tài liệu cho [java.lang.Toán](#) lớp học.
2. Cập nhật [Hoạt động](#) chính lớp để kết nối **BẤN** ... Cái nút để tính toán.

Bây giờ hãy viết từng bài kiểm tra sau cho [bạn](#) [sức mạnh\(\)](#) phương pháp. Chạy bộ kiểm thử của bạn mỗi khi bạn viết một bài kiểm thử và sửa phép tính ban đầu trong ứng dụng của bạn nếu cần:

- Một phép thử với toán hạng số nguyên dương.
- Một phép thử với số nguyên âm là toán hạng đầu tiên.
- Một phép thử với số nguyên âm làm toán hạng thứ hai.
- Một phép thử với 0 là toán hạng đầu tiên và một số nguyên dương là toán hạng thứ hai.

- Một phép thử với 0 là toán hạng thứ hai.
- Một bài kiểm tra với 0 là toán hạng đầu tiên và -1 là toán hạng thứ hai. (Gợi ý: tham khảo tài liệu choĐôi.POSITIVE_INFINITY.)
- Một phép thử với -0 là toán hạng đầu tiên và bất kỳ số âm nào là toán hạng thứ hai.

Trả lời những câu hỏi này

Câu hỏi 1

Câu nào mô tả tốt nhất về bài kiểm tra đơn vị cục bộ? Chọn một câu:

- Các bài kiểm tra chạy trên thiết bị hoặc trình giả lập chạy Android và có quyền truy cập vào nền tảng Android.
- Các bài kiểm tra cho phép bạn viết các phương pháp kiểm tra UI tự động.
- Các bài kiểm tra được biên dịch và chạy hoàn toàn trên máy cục bộ của bạn bằng Máy ảo Java (JVM).

Câu hỏi 2

Bộ nguồn là tập hợp các mã liên quan. Bạn có khả năng tìm thấy các bài kiểm tra đơn vị trong bộ nguồn nào? Chọn một trong các mục sau:

- ứng dụng/res
- com.example.android.SimpleCalcTest
- com.example.android.SimpleCalcTest (kiểm tra)
- com.example.android.SimpleCalcTest (androidTest)

Câu hỏi 3

Chú thích nào được sử dụng để đánh dấu một phương pháp là một bài kiểm tra thực tế? Chọn một trong các chú thích sau:

- @Chạy với(JUnit4.class)
- @Kiểm tra nhỏ
- @Trước
- @Bài kiểm tra

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Kiểm tra xem ứng dụng có các tính năng sau không:

- Nó hiển thị một **BẮN** ... Cái nút cung cấp phép tính theo cấp số nhân ("lũy thừa").
- Việc thực hiện **Hoạt động chính** bao gồm một trình xử lý nhấp chuột cho **BẮN** ... Cái nút.
- Việc thực hiện **Máy tính** bao gồm một **sức mạnh()** phương pháp thực hiện phép tính.
- Các **Máy tính thử nghiệm()** phương pháp bao gồm các phương pháp thử nghiệm riêng biệt cho **sức mạnh()** phương pháp trong **Máy tính**. Lớp thực hiện các thử nghiệm cho các toán hạng âm và 0, và cho trường hợp 0 và -1 là toán hạng.

Bài 3.3: Thư viện hỗ trợ

Giới thiệu

Android SDK bao gồm Android Support Library, là một tập hợp gồm nhiều thư viện. Các thư viện này cung cấp các tính năng không được tích hợp vào khuôn khổ Android, bao gồm:

- Các phiên bản tương thích ngược của các thành phần khung, để các ứng dụng chạy trên các phiên bản cũ hơn các phiên bản của nền tảng Android có thể hỗ trợ các tính năng có sẵn trong các phiên bản mới hơn của nền tảng
- Bổ cục bổ sung và các thành phần giao diện người dùng
- Hỗ trợ cho các dạng thiết bị khác nhau, chẳng hạn như thiết bị TV hoặc thiết bị đeo
- Các thành phần hỗ trợ các thành phần Material Design
- Các tính năng khác, bao gồm hỗ trợ bảng màu, chú thích, kích thước bố cục dựa trên phần trăm và tùy chọn

Những điều bạn nên biết

Bạn sẽ có thể:

- Tạo một dự án Android Studio.
- Sử dụng trình chỉnh sửa bố cục để làm việc với Sửa văn bản và Cái nút các yếu tố.
- Xây dựng và chạy ứng dụng của bạn trong Android Studio, trên cả trình giả lập và trên thiết bị.
- Điều hướng **Dự án > Android** trong Android Studio.
- Tìm các thành phần chính của một dự án Android Studio, bao gồm `AndroidManifest.xml`, tài nguyên, tệp Java và tệp Gradle.

Những gì bạn sẽ học được

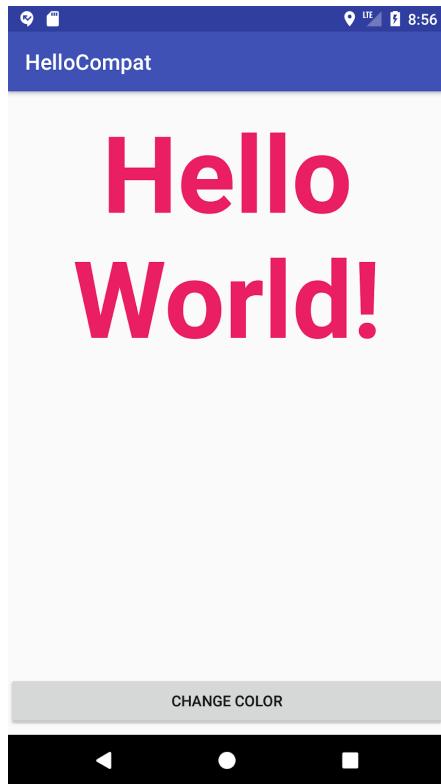
- Cách xác minh Thư viện hỗ trợ Android có sẵn trong cài đặt Android Studio của bạn hay không.
- Cách chỉ định các lớp thư viện hỗ trợ trong ứng dụng của bạn.
- Làm thế nào để biết sự khác biệt giữa các giá trị `compileSdkVersion`, `targetSdkVersion`, và `Phiên bản SDK` tối thiểu.
- Cách nhận biết các API đã lỗi thời hoặc không khả dụng trong mã của bạn.
- Tìm hiểu thêm về thư viện hỗ trợ Android.

Bạn sẽ làm gì

- Tạo một ứng dụng mới với một Xem văn bản và một Cái nút.
- Xác minh rằng Kho lưu trữ hỗ trợ Android (chứa Thư viện hỗ trợ Android) có sẵn trong cài đặt Android Studio của bạn.
- Khám phá xây dựng `gradle` file dự án ứng dụng của bạn.
- Quản lý các lệnh gọi lớp hoặc phương thức không khả dụng cho phiên bản Android mà ứng dụng của bạn hỗ trợ.
- Sử dụng lớp tương thích từ thư viện hỗ trợ để cung cấp khả năng tương thích ngược cho ứng dụng của bạn.

Tổng quan về ứng dụng

Trong bài thực hành này, bạn sẽ tạo một ứng dụng có tên là HelloCompat bằng một Xem văn bản hiển thị "Hello World" trên màn hình và một Cái nút thay đổi màu sắc của văn bản. Có 20 màu có thể có, được định nghĩa là tài nguyên trong color.xml và mỗi lần nhấp vào nút sẽ chọn ngẫu nhiên một trong những màu đó.



Các phương pháp để lấy giá trị màu từ tài nguyên của ứng dụng đã thay đổi với các phiên bản khác nhau cho khuôn khổ Android. Ví dụ này sử dụng `Tương thích ngữ cảnh` lớp trong Thư viện hỗ trợ Android, cho phép bạn sử dụng phương thức hoạt động với mọi phiên bản.

Nhiệm vụ 1: Thiết lập dự án của bạn để sử dụng các thư viện hỗ trợ

Đối với nhiệm vụ này, bạn sẽ thiết lập một dự án mới cho ứng dụng HelloCompat và triển khai bộ cục và hành vi cơ bản.

1.1 Xác minh rằng Kho lưu trữ hỗ trợ Android có sẵn

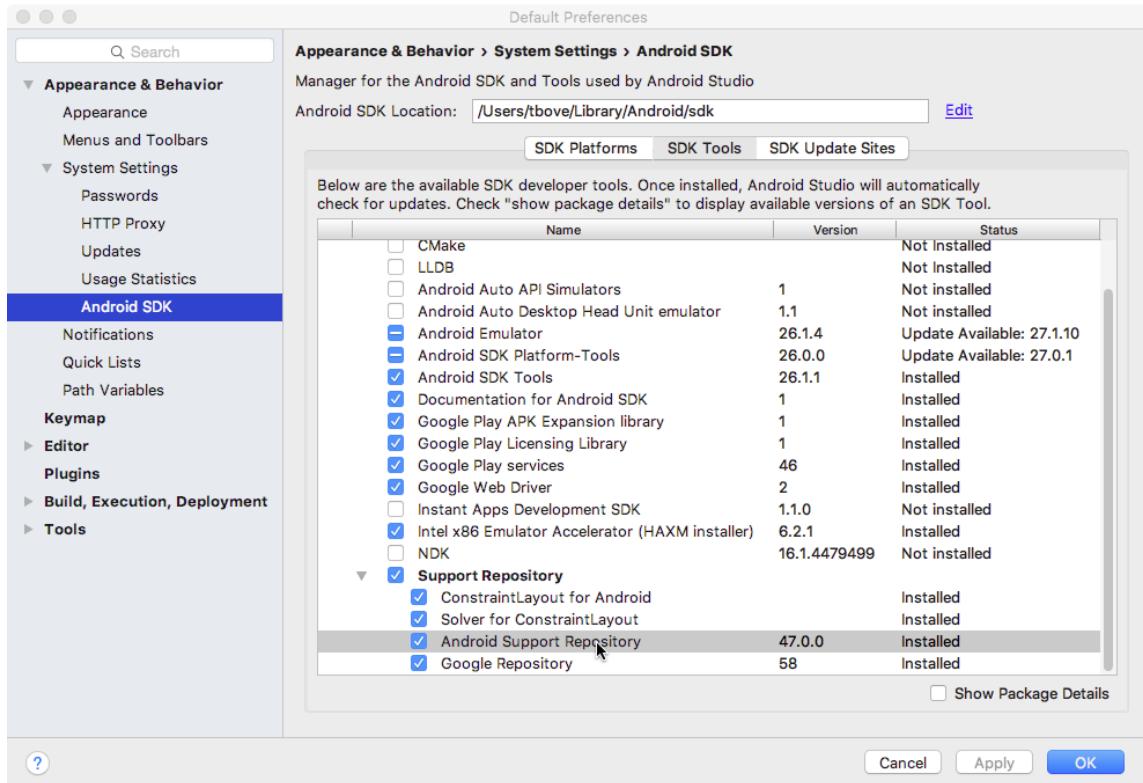
Thư viện hỗ trợ Android được tải xuống như một phần của Android SDK và có sẵn trong trình quản lý Android SDK. Trong Android Studio, bạn sẽ sử dụng Android Support Repository—kho lưu trữ cục bộ cho các thư viện hỗ trợ—để truy cập vào các thư viện từ bên trong các tệp dựng Gradle của bạn. Trong tác vụ này, bạn sẽ xác minh rằng Android Support Repository đã được tải xuống và có sẵn cho các dự án của bạn.

15.Trong Android Studio, hãy chọn **Công cụ > Android > Trình quản lý SDK** hoặc nhấp vào biểu tượng Trình quản lý SDK.



Bộ công cụ phát triển phần mềm Android **Tùy chọn mặc định** gần xuất hiện.

16.Nhấp vào **Công cụ SDK** tab và mở rộng **Kho lưu trữ hỗ trợ**, như thể hiện trong hình bên dưới.



17.Tìm kiếm **Kho lưu trữ hỗ trợ Android** trong danh sách.

Nếu như **Đã cài đặt** xuất hiện trong cột Trạng thái, bạn đã hoàn tất. Nhấp vào **Hủy bỏ**.

Nếu như **Chưa cài đặt** hoặc **Cập nhật có sẵn** xuất hiện, nhấp vào hộp kiểm bên cạnh **Kho lưu trữ hỗ trợ Android**. Một biểu tượng tải xuống sẽ xuất hiện bên cạnh hộp kiểm. Nhấp vào **ĐƯỢC RỒI**.

18.Nhấp chuột **ĐƯỢC RỒI** một lần nữa, và sau đó **Hoàn thành** khi kho lưu trữ hỗ trợ đã được cài đặt.

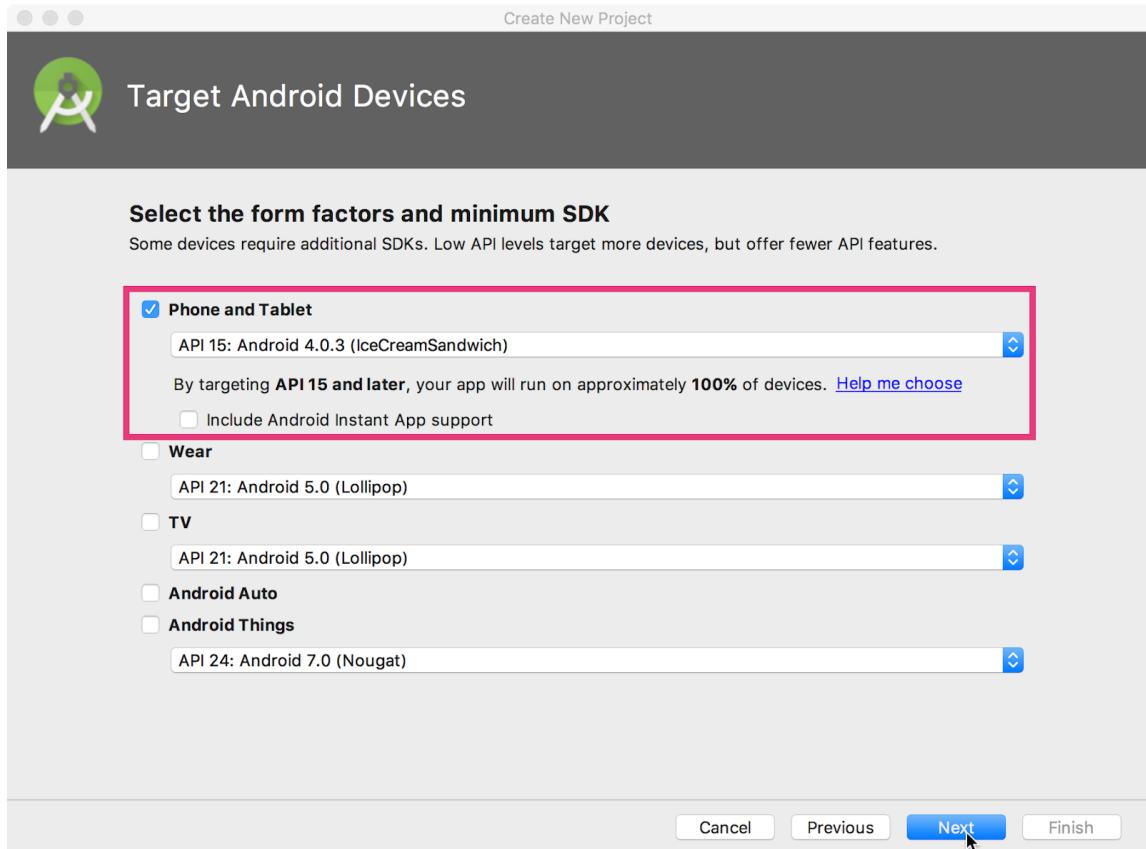
1.2 Thiết lập dự án và kiểm tra build.gradle

1.Tạo một dự án mới có tên là **Xin chào Compat**.

Trên trang Thiết bị Android mục tiêu, **API 15: Android 4.0.3 (IceCreamSandwich)** được chọn cho SDK tối thiểu. Như bạn đã học trong các bài học trước, đây là phiên bản cũ nhất của nền tảng Android mà ứng dụng của bạn sẽ hỗ trợ.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.



- 1.Nhấp chuộtKế tiếpvà chọn**Hoạt động** trốngbản mẫu.
- 2.Nhấp chuộtKế tiếpvà đảm bảo rằng**Tạo tệp Bố cục**Và**Khả năng tương thích ngược (App Compat)**tùy chọn được chọn. Tùy chọn sau đảm bảo rằng ứng dụng của bạn sẽ tương thích ngược với các phiên bản Android trước đó.
- 3.Nhấp chuột**Hoàn thành**.

Khám phá build.gradle (Module:app)

- 1.Trong Android Studio, hãy đảm bảo**Dự án > Android**cửa sổ đang mở.
- 2.Mở rộng**Tập lệnh Gradle**và mở**build.gradle (Mô-đun: Ứng dụng)**có là.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế . PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

Lưu ý rằng `xây dựng.gradle` cho toàn bộ dự án (`build.gradle` (Dự án: `helpcompat`)) là một tập tin khác với `xây dựng.gradle` mô-đun ứng dụng. Trong `build.gradle` (Mô-đun: Ứng dụng) file.

3. Xác định vị trí biên dịch `SdkVersion` dòng gần đầu tệp. Ví dụ:

`biên dịchSdkPhiên bản 26`

Các `biên soạn` là phiên bản khung Android mà ứng dụng của bạn được biên dịch trong Android Studio. Đối với các dự án mới, phiên bản biên dịch là bộ API khung mới nhất mà bạn đã cài đặt. Giá trị này ảnh hưởng đến *chỉ một* bản thân Android Studio và các cảnh báo hoặc lỗi bạn nhận được trong Android Studio nếu bạn sử dụng API cũ hơn hoặc mới hơn.

4. Xác định vị trí `minSdkPhiên bản` dòng trong Cấu hình mặc định đoạn văn cách đó vài dòng.

`minSdkPhiên bản 15`

Các `tối thiểu` là phiên bản API Android cũ nhất mà ứng dụng của bạn chạy dưới. Đây là số giống với số bạn đã chọn ở Bước 1 khi tạo dự án. Cửa hàng Google Play sử dụng số này để đảm bảo ứng dụng của bạn có thể chạy trên thiết bị của người dùng nhất định. Android Studio cũng sử dụng số này để cảnh báo bạn về việc sử dụng API đã lỗi thời.

5. Xác định vị trí Phiên bản `targetSdk` dòng trong Cấu hình mặc định phần. Ví dụ:

`targetSdkPhiên bản 26`

Các `mục tiêu` cho biết phiên bản API mà ứng dụng của bạn được thiết kế và thử nghiệm. Nếu API của nền tảng Android cao hơn số này (tức là ứng dụng của bạn đang chạy trên thiết bị mới hơn), nền tảng có thể kích hoạt các hành vi tương thích để đảm bảo rằng ứng dụng của bạn tiếp tục hoạt động theo cách mà nó được thiết kế. Ví dụ: Android 6.0 (API 23) cung cấp mô hình cấp phép thời gian chạy mới. Nếu ứng dụng của bạn nhắm mục tiêu đến cấp API thấp hơn, nền tảng sẽ quay lại mô hình cấp phép thời gian cài đặt cũ hơn.

Mặc dù SDK mục tiêu có thể có cùng số với SDK biên dịch, nhưng thường thì số đó sẽ thấp hơn, biểu thị phiên bản API mới nhất mà bạn đã thử nghiệm ứng dụng của mình.

6.Xác định vị trí sự phụ thuộc phần build.gradle, gần cuối tệp. Ví dụ:

```
phụ thuộc {  
    fileTree thực hiện(dir: 'libs', include: ['*.jar']) thực hiện  
        'com.android.support:appcompatv7:26.1.0'  
    thực hiện  
        'com.android.support.constraint:constraintlayout:1.0.2' thử nghiệm Thực  
hiện 'junit:junit:4.12'  
    androidTestImplementation 'com.android.support.test:runner:1.0.1'  
    androidTestImplementation  
        'com.android.support.test.espresso:espresso:3.0.1'  
}
```

Các sự phụ thuộc phần dành cho dự án mới bao gồm một số phụ thuộc để cho phép thử nghiệm với Espresso và JUnit, cũng như thư viện hỗ trợ appcompat v7. Số phiên bản cho các thư viện này trong dự án của bạn có thể khác với số được hiển thị ở đây.

Thư viện hỗ trợ appcompat v7 cung cấp khả năng tương thích ngược cho các phiên bản Android cũ hơn, kể từ API 9. Thư viện này cũng bao gồm thư viện tương thích v4, do đó bạn không cần phải thêm cả hai vào làm thành phần phụ thuộc.

7.Cập nhật số phiên bản nếu cần thiết.

Nếu số phiên bản hiện tại của thư viện thấp hơn số phiên bản thư viện hiện có, Android Studio sẽ đánh dấu dòng đó và cảnh báo bạn rằng đã có phiên bản mới ("Đã có phiên bản mới hơn của com.android.support:appcompat-v7"). Chính sửa số phiên bản thành phiên bản đã cập nhật.

Mẹo: Bạn cũng có thể nhấp vào bất kỳ đâu trong dòng tô sáng và nhấn Alt+Enter (Tùy chọn+Trả về trên máy Mac). Chọn **Thay đổi thành xx.xx.x** từ menu, nơi xx.xx.x là phiên bản mới nhất hiện có.

8.Cập nhật biến `dichSdkVersion`, nếu cần thiết.

Số phiên bản chính của thư viện hỗ trợ (số đầu tiên) phải khớp với biến `dichSdkVersion`. Khi bạn cập nhật phiên bản thư viện hỗ trợ, bạn cũng có thể cần cập nhật biến `dichSdkVersion` để phù hợp.

9.Nhấp chuột **Đồng bộ ngay** để đồng bộ hóa các tệp Gradle đã cập nhật của bạn với dự án, nếu được nhắc.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

10.Cài đặt các tệp nền tảng SDK bị thiếu nếu cần.

Nếu bạn cập nhật biến `dichSdkVersion`, bạn có thể cần phải cài đặt các thành phần nền tảng SDK để phù hợp.
Nhấn vào **Cài đặt nền tảng bị thiếu và đồng bộ hóa dự án** để bắt đầu quá trình này.

Nhiệm vụ 2: Triển khai bố cục và MainActivity

Đối với nhiệm vụ này, bạn sẽ triển khai bố cục và hành vi cơ bản cho **Hoạt động chính** lớp học.

2.1 Thay đổi bố cục và màu sắc

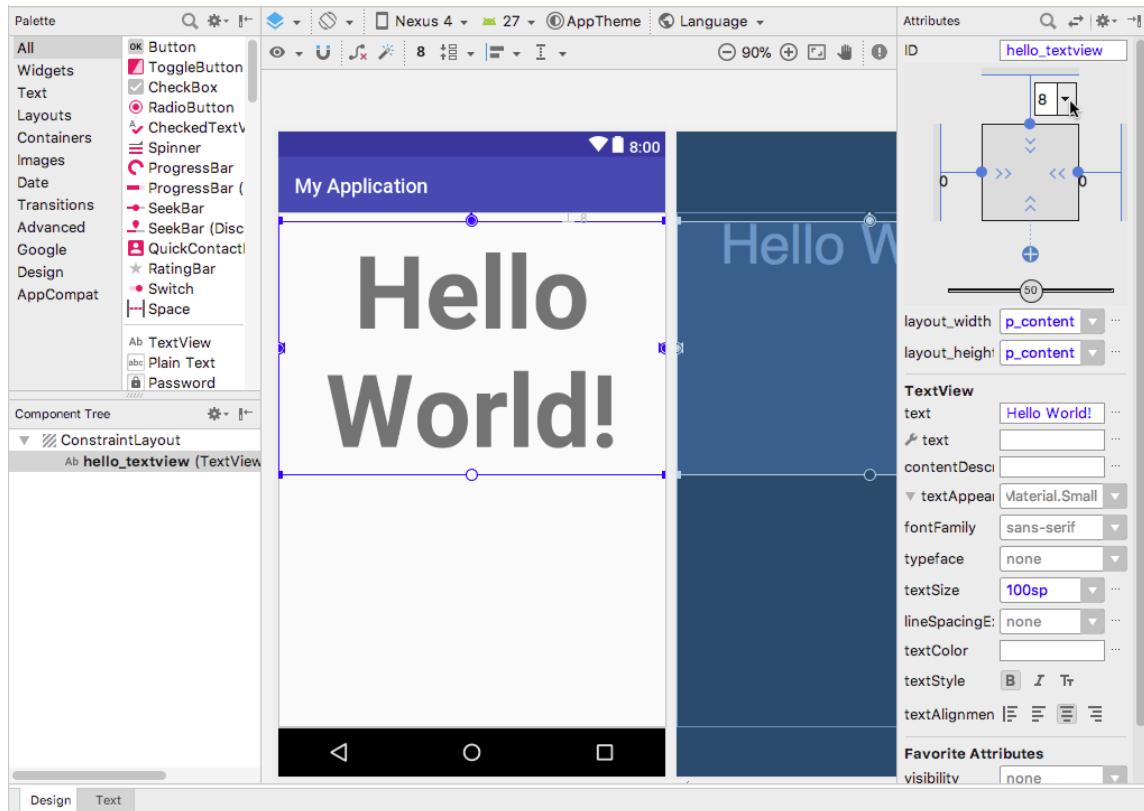
Trong nhiệm vụ này bạn sẽ sửa đổi `activity_main.xml` để bổ sung cho ứng dụng.

- 1.Mở `activity_main.xml` trong **Dự án > Bảng điều khiển Android**.
- 2.Nhấn vào **Thiết kế** tab (nếu chưa được chọn) để hiển thị trình chỉnh sửa bố cục.
- 3.Chọn "Xin chào thế giới" Xem văn bản trong bố cục và mở **Thuộc tính** khung cửa sổ.
- 4.Thay đổi Xem văn bản các thuộc tính như sau:

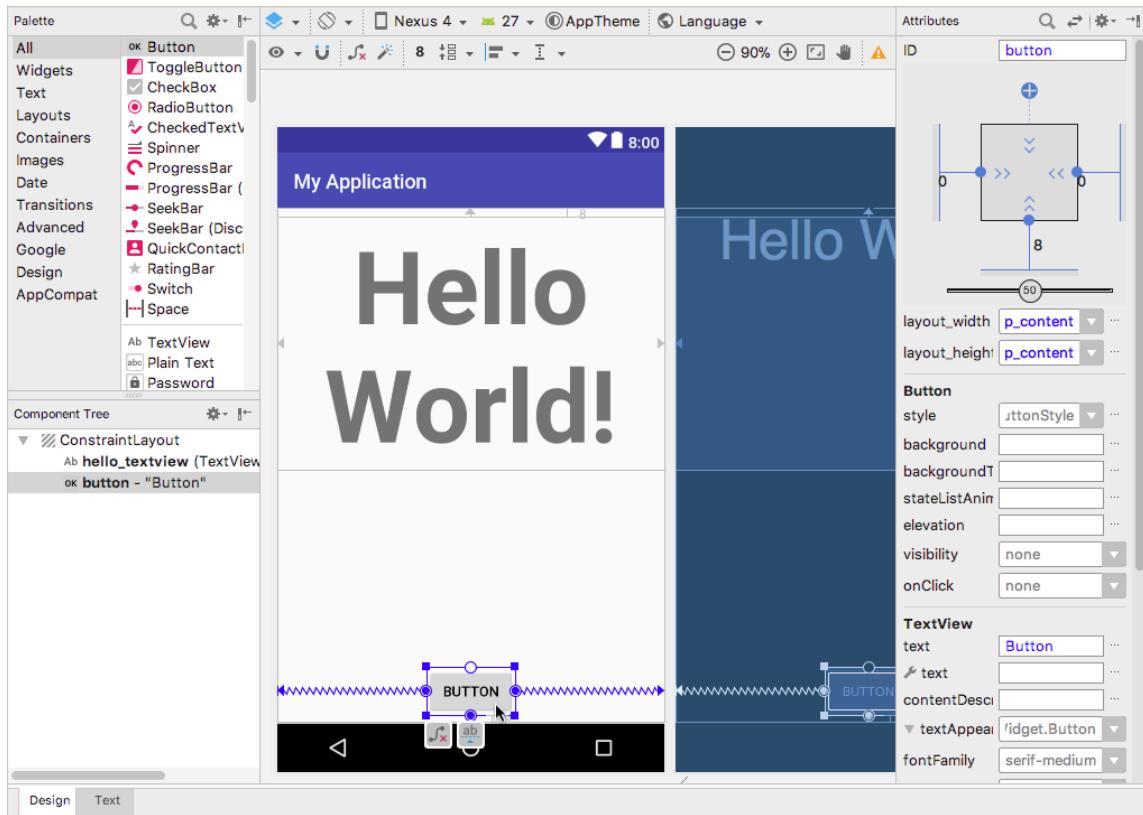
Trường thuộc tính	Nhập thông tin sau:
NHÂN DẠNG	xin chào_văn_bản
Kiểu văn bản	B (in đậm)
Căn chỉnh văn bản	Biểu tượng căn giữa đoạn văn
Kích thước văn bản	100sp

Điều này thêm vào `android:id="@+id/txtHelloWorld"` cho Xem văn bản và định dạng đặt thành `xin_chào_xem_văn_bản`, thay đổi căn chỉnh văn bản, làm cho văn bản đậm và đặt kích thước văn bản lớn hơn `100sp`.

5.Xóa ràng buộc kéo dài từ dưới cùng của hello_textview Xem văn bản vào cuối bố cục, để Xem văn bản bám vào đầu bố cục và chọn 8(8dp) cho lề trên như minh họa bên dưới.



6.Kéo một Cái nútvào cuối bố cục và thêm các ràng buộc vào bên trái, bên phải và phía dưới của bố cục, như thể hiện trong hình bên dưới.



7.Thay đổi **chiều rộng** **bố cục** thuộc tính trong **Thuộc tính khung** cho Cái nút **ĐEN ràng buộc_phù hợp.**

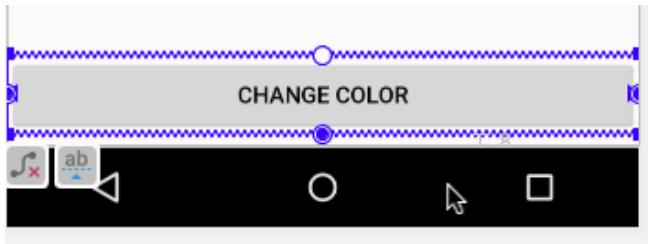
8.Thay đổi các thuộc tính khác trong **Thuộc tính khung** cho Cái nút như sau:

Trường thuộc tính	Nhập thông tin sau:
NHÂN DẠNG	nút_màu
chữ	"Đổi màu"

Các Cái nút bây giờ sẽ xuất hiện trong bố cục như hiển thị bên dưới:

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.



9.Trong bài học trước, bạn đã học cách trích xuất một tài nguyên chuỗi từ một chuỗi văn bản theo nghĩa đen. Nhấp vào **Chữ** tab để chuyển sang mã XML và trích xuất "Xin chào Text!" và "Thay đổi màu sắc" chuỗi trong Xem văn bản và Cái nút, và nhập tên tài nguyên chuỗi cho chúng.

10.Thêm những điều sau đây android:onClick thuộc tính cho Cái nút:

```
    android:onClick="thay đổi màu"
```

11. Để thêm màu sắc, hãy mở rộng **độ phân giải** và **giá trị** trong **Dự án > Android** cửa sổ và mở **màu sắc.xml**.

12.Thêm các nguồn màu sau vào tệp:

```
<màu sắc tên="đỏ">#F44336</color>
<màu sắc tên="hồng">#E91E63</color>
<màu sắc tên="tím">#9C27B0</color>
<màu sắc tên="tím_sâu">#673AB7</color>
<màu sắc tên="chàm">#3F51B5</color>
<màu sắc tên="xanh">#2196F3</color>
<màu sắc tên="xanh_xanh_sáng">#03A9F4</color>
<màu sắc tên="xanh_lục">#00BCD4</color>
<màu sắc tên="xanh_tím">#009688</color>
<màu sắc tên="xanh_xanh">#4CAF50</color>
<màu sắc tên="xanh_sáng">#8BC34A</color>
<màu sắc tên="vàng">#CDDC39</color>
<màu sắc tên="vàng">#FFEB3B</color> tên="hổ
<màu sắc phách">#FFC107</color>
<màu sắc tên="cam">#FF9800</color>
<màu sắc tên="cam_sâu">#FF5722</color>
<màu sắc tên="nâu">#795548</color>
<màu sắc tên="xám">#9E9E9E</color>
<màu sắc tên="xanh_xám">#607D8B</color>
<màu sắc tên="đen">#000000</color>
```

Các giá trị và tên màu này xuất phát từ bảng màu được đề xuất cho các ứng dụng Android được xác định tại [Thiết kế vật liệu - Phong cách - Màu sắc](#). Các mã này biểu thị giá trị màu RGB theo hệ thập lục phân.

2.2 Thêm hành vi vào MainActivity

Trong nhiệm vụ này, bạn sẽ hoàn tất việc thiết lập dự án bằng cách thêm riêng tư biến và thực hiện khi tạo() và khi Lưu Trạng Thái Phiên Bản().

1. Mở **Hoạt động chính**.

2. Thêm một riêng tư biến ở đầu lớp để giữ Xem văn bản sự vật.

TextView riêng tư mHelloTextView;

3. Thêm mảng màu sau ngay sau biến riêng tư:

```
private String[] mColorArray = {"đỏ", "hồng", "tím", "tím_sâu",
    "chàm", "xanh lam", "xanh lam nhạt", "xanh lơ", "xanh lục", "xanh lục nhạt", "chanh",
    "vàng", "hổ phách", "cam", "cam đậm", "nâu", "xám", "xanh lam xám", "đen"};
```

Mỗi tên màu tương ứng với tên của một nguồn tài nguyên màu trong màu sắc.xml.

4. Trong khi tạo() phương pháp, sử dụng `findViewById()` để có được một tham chiếu đến Xem văn bản và gán nó vào biến riêng đó:

```
mHelloTextView = findViewById(R.id.hello_textview);
```

1.Cũng trong khi tạo(), khôi phục trạng thái phiên bản đã lưu, nếu có:

```
// khôi phục trạng thái phiên bản đã lưu (màu văn bản) if  
(savedInstanceState != null) {  
    mHelloTextView.setTextColor(savedInstanceState.getInt("color"));  
}
```

2.Thêm vào onSaveInstanceState() phương pháp để hoạt động chính để lưu màu văn bản:

```
@Ghi đè  
public void onSaveInstanceState(Gói outState) {  
    super.onSaveInstanceState(outState); // lưu màu  
    văn bản hiện tại  
    outState.putInt("color", mHelloTextView.getCurrentTextColor());  
}
```

Mã giải bài tập 2

Sau đây là mã giải pháp cho bộ cục XML và một đoạn mã trong lớp hoạt động chính cho ứng dụng HelloCompat cho đến nay.

Bối cảnh XML

Bối cảnh XML cho activity_main.xml file được hiển thị bên dưới. thay đổi Màn hình xử lý nhấp chuột cho android:onClick thuộc tính cho Cái nút được gạch chân màu đỏ vì nó vẫn chưa được định nghĩa. Bạn định nghĩa nó trong nhiệm vụ tiếp theo.

```
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ứng dụng="http://schemas.android.com/apk/res-auto"
    xmlns:công cụ="http://schemas.android.com/tools" android:bối
    cục_chiều rộng="phù hợp với cha mẹ"
    android:layout_height="match_parent" công
    cụ:context="com.example.android.myapplication">

    <Xem văn bản
        android:id="@+id/hello_textview"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:text="@string/hello_world"
        android:textAlignment="center"
        android:textSize="100sp"
        android:textStyle="bold"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Nút
        android:id="@+id/color_button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="8dp"
        android:text="@string/change_color"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        android:onClick="changeColor"/>

</android.support.constraint.ConstraintLayout>
```

Hoạt động chính

Các **Hoạt động chính** lớp học bao gồm những điều sau đây riêng tư các biến ở đầu lớp:

```
// Xem văn bản cho Hello World. private  
TextView mHelloTextView;  
// mảng tên màu, chúng khớp với các tài nguyên màu trong color.xml private String[]  
mColorArray = {"red", "pink", "purple", "deep_purple",  
    "chàm", "xanh lam", "xanh lam nhạt", "xanh lơ", "xanh lục", "xanh lục nhạt",  
    "chanh", "vàng", "hổ phách", "cam", "cam đậm", "nâu", "xám", "xanh lam xám",  
    "đen" };
```

Các Hoạt động chính lớp học bao gồm những điều sau đây khi tạo() và onSaveInstanceState() phương pháp:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main); mHelloTextView =  
    findViewById(R.id.hello_textview); // khôi phục trạng thái phiên  
    bản đã lưu (màu văn bản) if (savedInstanceState != null) {  
  
        mHelloTextView.setTextColor(savedInstanceState.getInt("color"));  
    }  
}  
  
@Ghi đè  
public void onSaveInstanceState(Gói outState) {  
    super.onSaveInstanceState(outState); // lưu  
    màu văn bản hiện tại  
    outState.putInt("color", mHelloTextView.getCurrentTextColor());  
}
```

Nhiệm vụ 3: Triển khai hành vi của Button

Các **Thay đổi màu sắc** nút trong ứng dụng HelloCompat chọn một trong 20 màu từ màu sắc.xml tệp tài nguyên ngẫu nhiên và đặt màu của văn bản thành màu đó. Trong nhiệm vụ này, bạn sẽ triển khai hành vi cho Cái nút trình xử lý nhập chuột.

2.1 Thêm trình xử lý nhấp chuột changeButton()

- 1.Mở **hoạt động_main.xml**, nếu nó chưa được mở. Nhấp vào **Chữ tab** để hiển thị mã XML.
- 2.Nhấp vào "đổi màu" trong **android:onClick** thuộc tính bên trong Cái nút yếu tố.
- 3.Nhấn Alt+Enter (Tùy chọn+Enter trên máy Mac) và chọn **Tạo trình xử lý sự kiện onClick**.
- 4.Chọn **Hoạt động chính** và nhấp vào **ĐƯỢC RỒI**.

Điều này tạo ra một phương thức giữ chỗ cho **đổi màu()** phương pháp trong **Hoạt động chính**:

```
public void changeColor(Xem chế độ xem) { }
```

2.2 Triển khai hành động Button

- 1.Chuyển sang **Hoạt động chính**.
- 2.Trong **đổi màu()** phương pháp, tạo một đối tượng số ngẫu nhiên bằng cách sử dụng **Ngẫu nhiên lớp** (một lớp Java) để tạo ra các số ngẫu nhiên đơn giản.

```
Ngẫu nhiên ngẫu nhiên = new Random();
```

- 3.Sử dụng **ngẫu nhiên** trường hợp để chọn một màu ngẫu nhiên từ **mColorMảng**:

```
Chuỗi colorName = mColorArray[random.nextInt(20)];
```

Cáctiếp theo **nextInt()** phương pháp với đối số 20 sẽ nhận được một số nguyên ngẫu nhiên khác giữa 0 và 19. Bạn sử dụng số nguyên đó làm chỉ mục của mảng để lấy tên màu.

4.Lấy mã định danh tài nguyên (số nguyên) cho tên màu từ các tài nguyên:

```
int colorResourceName = getResources().getIdentifier(colorName,  
        "màu sắc", getApplicationContext().getPackageName());
```

Khi ứng dụng của bạn được biên dịch, hệ thống Android sẽ chuyển đổi các định nghĩa trong tệp XML của bạn thành các tài nguyên có ID số nguyên nội bộ. Có các ID riêng cho cả tên và giá trị. Dòng này khớp với các chuỗi màu từ Tên màu màu mang với các ID tên màu tương ứng trong tệp tài nguyên XML. `lấy Tài nguyên()` phương pháp này lấy tất cả các tài nguyên cho ứng dụng của bạn. `lấy định danh()` phương pháp tra cứu tên màu (chuỗi) trong tài nguyên màu ("màu sắc") cho tên gói hiện tại.

5.Lấy ID số nguyên cho màu thực tế từ các nguồn và gán nó cho một `màuRes` biến và sử dụng `lấyTheme()` phương pháp để lấy chủ đề cho bối cảnh ứng dụng hiện tại.

```
int colorRes =  
    getResources().getColor(colorResourceName, this.getTheme());
```

Cáclấy Tài nguyên() phương pháp này lấy tập hợp các tài nguyên cho ứng dụng của bạn và `lấyMàu()` phương pháp này lấy một màu cụ thể từ các tài nguyên đó theo ID của tên màu. Tuy nhiên, `lấyMàu()` có phần gạch chân màu đỏ nổi bật.

Nếu bạn chỉ vào `lấy màu()`, Android Studio báo cáo: "Cuộc gọi yêu cầu API 23 (tối thiểu hiện tại là 15)". Bởi vì `minSdkVersion` là 15, bạn sẽ nhận được thông báo này nếu bạn thử sử dụng bất kỳ API nào được giới thiệu sau API 15. Bạn vẫn có thể biên dịch ứng dụng của mình, nhưng vì phiên bản này `lấyMàu()` không khả dụng trên các thiết bị trước API 23, ứng dụng của bạn sẽ bị sập khi người dùng chạm vào **Thay đổi màu sắc** cái nút.

Ở giai đoạn này, bạn có thể kiểm tra phiên bản nền tảng và sử dụng phiên bản phù hợp `lấyMàu()` tùy thuộc vào nơi ứng dụng đang chạy. Một cách tốt hơn để hỗ trợ cả API Android cũ và mới mà không có cảnh báo là sử dụng một lớp tương thích trong thư viện hỗ trợ.

6.Thay đổi `màuRes` dòng chỉ định để sử dụng Tương thích ngữ cảnh lớp học:

```
int colorRes = ContextCompat.getColor(này, colorResourceName);
```

Tương thích ngữ cảnh cung cấp nhiều phương pháp tương thích để giải quyết các khác biệt về API trong bối cảnh ứng dụng và tài nguyên ứng dụng. `lấyMàu()` phương pháp trong `Tương thích ngữ cảnh` lấy hai đối số: bối cảnh hiện tại (ở đây, `Hoạt động` ví dụ, cái này), và tên của màu sắc.

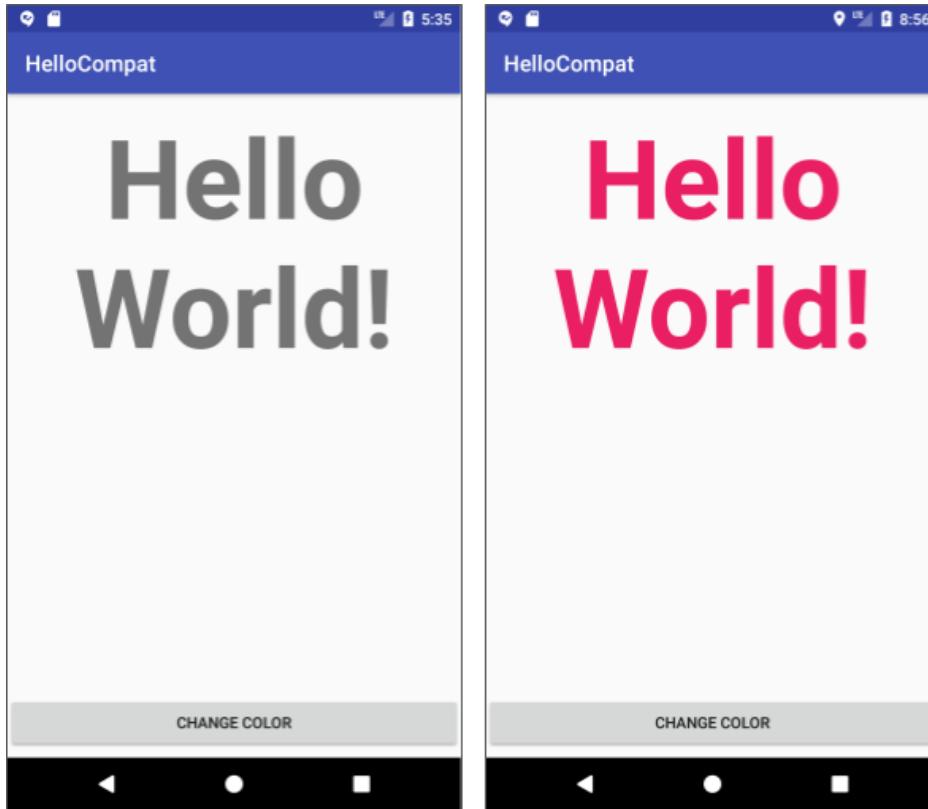
Việc triển khai phương pháp này trong thư viện hỗ trợ sẽ ẩn đi sự khác biệt trong việc triển khai ở các phiên bản khác nhau của API. Bạn có thể gọi phương pháp này bất kể phiên bản SDK biên dịch hay phiên bản SDK tối thiểu của bạn mà không có cảnh báo, lỗi hoặc sự cố nào.

1. Đặt màu của Xem văn bản đến ID tài nguyên màu:

```
mHelloTextView.setTextColor(colorRes);
```

2. Chạy ứng dụng trên thiết bị hoặc trình giả lập và nhấp vào **Thay đổi màu sắc** Cái nút.

Các **Thay đổi màu sắc** Cái nút bây giờ sẽ thay đổi màu của văn bản trong ứng dụng như hiển thị bên dưới.



Mã giải pháp

Giải pháp MainActivity

Sau đây là đổi màu() trình xử lý nhấp chuột trong Hoạt động chính:

```
/**  
 * Phương pháp này xử lý việc nhấp vào nút Thay đổi màu bằng cách  
 * chọn màu ngẫu nhiên từ một mảng màu.  
 *  
 * @param view Chế độ xem đã được nhấp
```

```
* /  
public void changeColor(Xem chế độ xem) {  
    // Lấy tên màu ngẫu nhiên từ mảng màu (20 màu). Random random = new  
    Random();  
    Chuỗi colorName = mColorArray[random.nextInt(20)];  
  
    // Lấy mã định danh màu khớp với tên màu. int colorResourceName =  
    getResources().getIdentifier(colorName,  
        "màu sắc", getApplicationContext().getPackageName());  
  
    // Lấy ID màu từ các tài nguyên.  
    int colorRes = ContextCompat.getColor(này, colorResourceName);  
  
    // Đặt màu văn bản.  
    mHelloTextView.setTextColor(colorRes);  
}
```

Dự án Android Studio

Dự án Android Studio:[Xin chào Compat](#)

Thử thách mã hóa

Ghi chú:Mọi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách:Thay vì sử dụngTương thích ngữ cảnhđể có được nguồn màu, hãy sử dụng phép thử các giá trị trong[Xây dựng](#) lớp để thực hiện một hoạt động khác nếu ứng dụng đang chạy trên thiết bị hỗ trợ phiên bản Android cũ hơn API 23.

Bản tóm tắt

Cài đặt Thư viện hỗ trợ Android:

- Sử dụng Trình quản lý SDK để cài đặt Kho lưu trữ hỗ trợ Android. Chọn**Công cụ > Android > Trình quản lý SDK**, nhấp vào**Công cụ SDK**tab và mở rộng**Kho lưu trữ hỗ trợ**.

Tác phẩm này được cấp phép theo một*Giấy phép Creative Commons Ghi công 4.0 Quốc tế*. PDF này là bản chụp nhanh một lần. Xem[dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

- Nếu như **Đã cài đặt** xuất hiện trong cột Trạng thái cho **Kho lưu trữ hỗ trợ Android**, nhấp chuột **Hủy bỏ**; nếu như **Chưa cài đặt** hoặc **Cập nhật có sẵn** xuất hiện, nhấp vào hộp kiểm. Một biểu tượng tải xuống sẽ xuất hiện bên cạnh hộp kiểm. Nhấp vào **ĐƯỢC RỒI**.

Android sử dụng ba chỉ thị để chỉ ra cách ứng dụng của bạn hoạt động đối với các phiên bản API khác nhau:

- Phiên bản minSdk: phiên bản API tối thiểu mà ứng dụng của bạn hỗ trợ.
- biên dịchSdk: phiên bản API mà ứng dụng của bạn sẽ được biên dịch.
- Phiên bản targetSdk: phiên bản API mà ứng dụng của bạn được thiết kế.

Để quản lý các phụ thuộc trong dự án của bạn:

- Mở rộng **Tập lệnh Gradle** trong **Dự án > Android** ngang, và mở **build.gradle (Mô-đun: Ứng dụng)** có là.
- Bạn có thể thêm các phụ thuộc vào **sự phụ thuộc** phần.

Các **Tương thích ngữ cảnh** Lớp này cung cấp các phương thức tương thích với ngữ cảnh và các phương thức liên quan đến tài nguyên cho cả cấp độ API cũ và mới.

Khái niệm liên quan

Tài liệu khái niệm liên quan có trong [3.3: Thư viện hỗ trợ Android](#).

Tìm hiểu thêm

Tài liệu Android Studio:

- [Hướng dẫn sử dụng Android Studio](#)

Tài liệu dành cho nhà phát triển Android:

Tác phẩm này được cấp phép theo một [Giấy phép Creative Commons Ghi công 4.0 Quốc tế](#). PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

- [Thư viện hỗ trợ Android](#) (giới thiệu)
- [Thiết lập thư viện hỗ trợ](#)
- [Hỗ trợ tính năng thư viện](#)
- [Hỗ trợ các phiên bản nền tảng khác nhau](#)
- [Chỉ mục gói](#) (tất cả các gói API bắt đầu bằng android.support)

Khác:

- [Chọn compileSdkVersion, minSdkVersion và targetSdkVersion của bạn](#)
- [Hiểu về Thư viện hỗ trợ Android](#)
- [Tất cả mọi thứ tương thích](#)

Bài tập về nhà

Chạy một ứng dụng

Mở [Xin chào Compat](#) ứng dụng bạn đã tạo trong phần thực hành sử dụng thư viện hỗ trợ.

1. Đặt điểm dừng gỡ lỗi trên dòng trong đổi màu() phương pháp thực sự thay đổi màu sắc:

```
int colorRes = ContextCompat.getColor(này, colorResourceName);
```

2. Chạy ứng dụng ở chế độ gỡ lỗi trên thiết bị hoặc trình giả lập đang chạy API phiên bản 23 trở lên. Nhấp vào **Bước vào** để bước vào lấy màu() phương pháp và theo dõi các cuộc gọi phương pháp sâu hơn vào ngăn xếp. Kiểm tra cách Tương thích ngữ cảnh lớp xác định cách lấy màu từ các tài nguyên và sử dụng các lớp khung nào khác.

Một số lớp có thể đưa ra cảnh báo rằng "mã nguồn không khớp với mã byte". Nhấp vào **Bước ra ngoài** để quay lại tệp nguồn đã biết hoặc tiếp tục nhấp **Bước vào** cho đến khi trình gỡ lỗi tự động trả về.

3. Lặp lại bước trước đó cho thiết bị hoặc trình giả lập chạy phiên bản API cũ hơn 23. Lưu ý các đường dẫn khác nhau mà khuôn khổ thực hiện để có được màu sắc.

Trả lời những câu hỏi này

Câu hỏi 1

Lớp nào xuất hiện khi bạn có `đầu tiên` **Bước vào** `cái ContextCompat.getColor()` phương pháp? Chọn một:

- Hoạt động chính
- Tương thích ngữ cảnh
- Hoạt động tương thích của ứng dụng
- Bối cảnh

Câu hỏi 2

Trong lớp xuất hiện, câu lệnh nào được thực thi nếu phiên bản dựng là API phiên bản 23 hoặc mới hơn?
Chọn một:

- trả về context.getColor(id);
- trả về context.getResources().getColor(id);
- throw new IllegalArgumentException("quyền là null");
- trả về mResources == null ? super.getResources() : mResources;

Câu hỏi 3

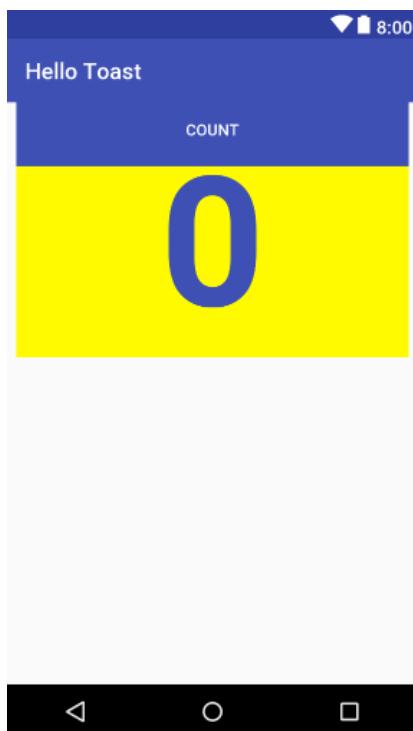
Nếu bạn thay đổi `ContextCompat.getColor()` phương pháp quay lìa lấy màu () phương pháp, điều gì sẽ xảy ra khi bạn chạy ứng dụng? Chọn một:

- Nếu bạn minSdkVersion là 15, từ lấy màu được gạch chân màu đỏ trong trình soạn thảo mã. Di con trỏ qua đó và Android Studio sẽ báo cáo "Cuộc gọi yêu cầu API 23 (min hiện tại là 15)".
- Ứng dụng sẽ chạy mà không có lỗi trên trình giả lập và thiết bị sử dụng API 23 hoặc mới hơn.
- Ứng dụng sẽ bị sập khi người dùng chạm vào **Thay đổi màu sắc** nếu trình giả lập hoặc thiết bị đang sử dụng API 17.
- Tất cả những điều trên.

3.2 Sắp xếp lại chế độ xem trong RelativeLayout

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

- Nhấp vào **Xem trước** tab ở bên cạnh trình soạn thảo (nếu chưa được chọn) để xem bản xem trước bối cục, bây giờ trông giống như hình bên dưới.



Với sự thay đổi Bối cục tương đối, trình chỉnh sửa bối cục cũng thay đổi một số thuộc tính của chế độ xem. Ví dụ:

- Các **Đếm** Nút (`button_count`) chồng lên nhau **Nướng** Cái nút, đó là lý do tại sao bạn không thể nhìn thấy **Nướng** Nút (`button_toast`).
- Phần trên cùng của Xem văn bản (`hiển thị_số lượng`) chồng lên nhau Cái nút các yếu tố.

- Thêm vào `android:bố_cục_bên_dưới` thuộc tính cho `button_count` Nút để định vị Cái nút ngay bên dưới `Hiển thị_số_lượt` xem văn bản. Thuộc tính này là một trong số nhiều thuộc tính để định vị chế độ xem trong một Bối cục tương đối—you đặt chế độ xem liên quan đến các chế độ xem khác.

```
    android:layout_below="@+id/hiển thị_số_lượng"
```

3.Thêm vào android:layout_centerHorizontal thuộc tính cho cùng một Cái nút để căn giữa chế độ xem theo chiều ngang trong phần cha của nó, trong trường hợp này là Bố cục tương đối xem nhóm.

```
    android:layout_centerHorizontal="đúng"
```

Mã XML đầy đủ chobutton_count Nút là như sau:

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

4.Thêm các thuộc tính sau vào Hiển thị_số_lượt xem văn bản:

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

Các android:layout_align Cha mẹ bên trái căn chỉnh chế độ xem sang phía bên trái của Bố cục tương đối nhóm chế độ xem cha mẹ. Trong khi thuộc tính này tự nó đủ để căn chỉnh chế độ xem

ở phía bên trái, bạn có thể muốn chế độ xem căn chỉnh sang phía bên phải nếu ứng dụng đang chạy trên một thiết bị sử dụng ngôn ngữ từ phải sang trái. Do đó, android:layout_alignParentStart thuộc tính làm cho cạnh “bắt đầu” của chế độ xem này khớp với cạnh bắt đầu của phần tử cha. *bắt đầu* là cạnh trái của màn hình nếu tùy chọn là từ trái sang phải, hoặc là cạnh phải của màn hình nếu tùy chọn là từ phải sang trái.

5.Xóa bỏ android: layout_weight="1" thuộc tính từ `Hiển thị_số_lượt xem` và bớt điều đó không liên quan đến một Bối cục tương đối. Bản xem trước bối cục bây giờ trông giống như hình sau:



Mẹo: Bối cục tương đối giúp bạn dễ dàng đặt các thành phần UI vào một bối cục một cách nhanh chóng. Để tìm hiểu thêm về cách định vị các chế độ xem trong Bối cục tương đối, nhìn thấy "[Định vị chế độ xem](#)" trong chủ đề "Bối cục tương đối" của Hướng dẫn API.

Mã giải bài tập 3

Mã XML trong activity_main.xml:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ứng dụng="http://schemas.android.com/apk/res-auto"
    xmlns:công cụ="http://schemas.android.com/tools" android:bố
    cục_chiều rộng="phù hợp với cha mẹ"
    android:layout_height="match_parent" android:orientation="vertical" công
    cụ:context="com.example.android.hellotoast.MainActivity">

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

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

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

Bản tóm tắt

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

- Để xem trước bố cục ứng dụng theo hướng ngang trong trình chỉnh sửa bố cục, hãy nhấp vào **Định hướng trong Trình biên tập** cái nút  trong thanh công cụ trên cùng và chọn **Chuyển sang chế độ phong cảnh**. Chọn **Chuyển sang chế độ Chân dung** để trở về hướng dọc.
- Để tạo biến thể của bố cục khác với hướng ngang, hãy nhấp vào **Định hướng trong Trình biên tập** nút và chọn **Tạo biến thể cảnh quan**. Một cửa sổ biên tập mới mở ra với **đất/hoạt động_chính.xml** tab hiển thị bố cục cho hướng ngang (ngang).
- Để xem trước bố cục cho các thiết bị khác nhau mà không cần phải chạy ứng dụng trên thiết bị hoặc trình giả lập, nhấp vào **Thiết bị trong Trình chỉnh sửa** thiết bị nút  **Nexus 5**  trong thanh công cụ trên cùng và chọn một bấm.
- Để tạo biến thể của bố cục khác nhau cho máy tính bảng (màn hình lớn hơn), hãy nhấp vào **Định hướng trong Trình biên tập** nút và chọn **Tạo bố cục x-lớn Biến thể**. Một biên tập viên mới

cửa sổ mở ra với **xlarge/hoạt động_main.xml** tab hiển thị bố cục cho thiết bị có kích thước bằng máy tính bảng.

Sử dụng Bố cục ràng buộc:

- Để xóa tất cả các ràng buộc trong một bố cục với Bố cục ràng buộc gốc, nhấp vào **Xóa tất cả**  nút trên thanh công cụ.
- Bạn có thể căn chỉnh một phần tử UI có chứa văn bản, chẳng hạn như Xem văn bản hoặc Cái nút, với một người khác. Phần tử UI chứa văn bản. **Ràng buộc cơ sở** cho phép bạn giới hạn các thành phần sao cho đường cơ sở của văn bản khớp với nhau.
- Để tạo ràng buộc đường cơ sở, hãy di con trỏ của bạn qua phần tử UI cho đến khi đường cơ sở nút hạn chế  xuất hiện bên dưới phần tử.
- Nút gói  trên thanh công cụ cung cấp các tùy chọn để đóng gói hoặc mở rộng UI đã chọn các yếu tố. Bạn có thể sử dụng nó để sắp xếp đều Cái nút các yếu tố nằm ngang trên bố cục.

Sử dụng Bố cục tuyến tính:

- Bố cục tuyến tính** là một **Xem Nhóm** sắp xếp tập hợp các chế độ xem theo hàng ngang hoặc hàng dọc.
- MỘT Bố cục tuyến tính được yêu cầu phải có chiều rộng bố cục, chiều cao bố cục, và định hướng thuộc tính.
- khớp _ cha mẹ vì chiều rộng bố cục hoặc chiều cao bố cục: Mở rộng Xem để điền vào cha mẹ của nó bằng chiều rộng hoặc chiều cao. Khi Bố cục tuyến tính là gốc rõ Xem, nó mở rộng đến kích thước của màn hình (cha mẹ Xem).
- Nội dung gói vì chiều rộng bố cục hoặc chiều cao bố cục: Thu nhỏ các kích thước để Xem chỉ đủ lớn để bao gồm nội dung của nó. Nếu không có nội dung, Xem trở nên vô hình.
- Số lượng cố định của dp ([pixel không phụ thuộc vào mật độ](#)) vì chiều rộng bố cục hoặc chiều cao bố cục: Chỉ định kích thước cố định, được điều chỉnh theo mật độ màn hình của thiết bị. Ví dụ: 16dp có nghĩa là 16 pixel không phụ thuộc vào mật độ.
- Các định hướng cho một Bố cục tuyến tính có thể là nằm ngang để sắp xếp các phần tử từ trái sang phải, hoặc thẳng đứng để sắp xếp các thành phần từ trên xuống dưới.

- Chỉ định trọng lực và cân nặng các thuộc tính cung cấp cho bạn quyền kiểm soát bổ sung đối với việc sắp xếp chế độ xem và nội dung trong một bố cục tuyến tính.
- Các android:trọng lực thuộc tính chỉ định sự căn chỉnh của nội dung của một Xem trong vòng Xem chính nó.
- Các android:trọng lượng bố cục thuộc tính cho biết có bao nhiêu không gian bổ sung trong Bố cục tuyến tính sẽ được phân bổ cho Xem. Nếu chỉ có một Xem có thuộc tính này, nó sẽ có được tất cả không gian màn hình bổ sung. Đối với nhiều Xem các phần tử, không gian được chia theo tỷ lệ. Ví dụ, nếu hai Cái nút mỗi phần tử có trọng số là 1 và Xem văn bản 2, tổng cộng là 4, Cái nút các phần tử nhận được $\frac{1}{4}$ không gian mỗi phần tử và Xem văn bản một nửa.

Sử dụng Bố cục tương đối:

- Một Bố cục tương đối là một Xem Nhóm trong đó mỗi góc nhìn được định vị và căn chỉnh tương đối với các góc nhìn khác trong nhóm.
- Sử dụng android:layout_alignParentTop để căn chỉnh Xem lên đầu trang của cha mẹ.
- Sử dụng android:layout_alignParentLeft để căn chỉnh Xem ở phía bên trái của cha mẹ.
- Sử dụng android:layout_alignParentStart để làm cho cạnh bắt đầu của Xem phù hợp với sự khởi đầu cạnh của phần tử cha. Thuộc tính này hữu ích nếu bạn muốn ứng dụng của mình hoạt động trên các thiết bị sử dụng ngôn ngữ hoặc tùy chọn địa phương khác nhau. *bắt đầu* là cạnh trái của màn hình nếu tùy chọn là từ trái sang phải, hoặc là cạnh phải của màn hình nếu tùy chọn là từ phải sang trái.

Các khái niệm liên quan

Tài liệu khái niệm liên quan có trong [1.2: Bố cục và tài nguyên cho UI](#).

Tìm hiểu thêm

Tài liệu Android Studio:

- [Làm quen với Android Studio](#)
- [Tạo biểu tượng ứng dụng bằng Image Asset Studio](#)

Tài liệu dành cho nhà phát triển Android:

- [Bố cục](#)
- [Xây dựng giao diện người dùng với Layout Editor](#)
- [Xây dựng một giao diện người dùng đáp ứng với ConstraintLayout](#)
- [Bố cục](#)
- [Bố cục tuyến tính](#)
- [Bố cục tương đối](#)
- [Xem](#)
- [Cái nút](#)
- [Xem văn bản](#)
- [Hỗ trợ mật độ điểm ảnh khác nhau](#)

Khác:

- Codelab:[Sử dụng ConstraintLayout để thiết kế chế độ xem Android của bạn](#)
- [Từ vựng và khái niệm thuật ngữ](#)

Bài tập về nhà

Thay đổi ứng dụng

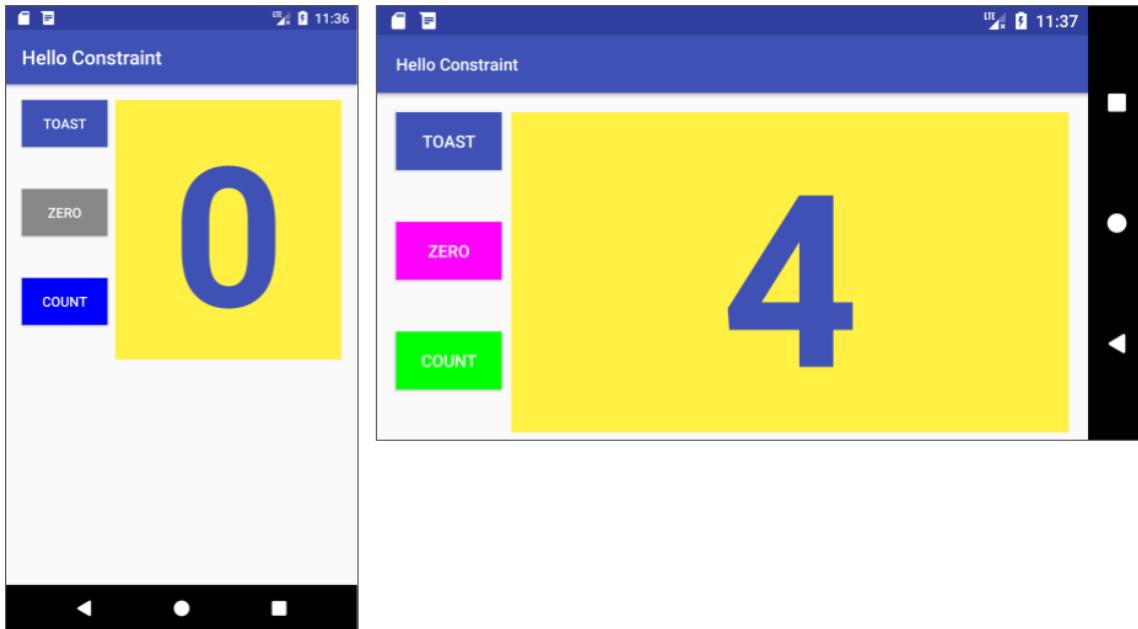
Mở [Xin chào Toast](#) ứng dụng.

1. Đổi tên dự án thành **Xin chào Constraint** và tái cấu trúc dự án để Xin chào Constraint. (Hoặc hướng dẫn về cách sao chép và tái cấu trúc một dự án, hãy xem [Phụ lục: Tiện ích](#).)
2. Sửa đổi hoạt động_main.xml bố trí để căn chỉnh **Nước** và **Đêm** Cái nút các yếu tố dọc theo phía bên trái của **Hiển thị_số_văn_bản** hiển thị "0". Tham khảo hình bên dưới để biết cách bố trí.
3. Bao gồm một phần ba Cái nút gọi điện **Số không** xuất hiện giữa **Nước** và **Đêm** Cái nút các yếu tố.
4. Phân phối Cái nút các phần tử theo chiều dọc giữa phần trên và phần dưới của **Hiển thị_số_lượt_xem_văn_bản**.
5. Đặt **Số không** Cái nút ban đầu có nền màu xám.
6. Hãy chắc chắn rằng bạn bao gồm **Số không** Cái nút cho hướng cảnh quan trong activity_main.xml (đất), và cũng cho một màn hình cỡ máy tính bảng activity_main (cực lớn).
7. Làm cho **Số không** Cái nút thay đổi giá trị trong **Hiển thị_số_văn_bản** đến 0.
8. Cập nhật trình xử lý nhấp chuột cho **Đêm** Cái nút để nó thay đổi sang màu nền, tùy thuộc vào số đếm mới là lẻ hay chẵn.

Gợi ý: Không sử dụng `findViewById` để tìm thấy **Đêm** Cái nút. Bạn có thể sử dụng cách nào khác không?

Hãy thoải mái sử dụng hằng số trong [Màu sắc](#) lớp cho hai màu nền khác nhau.

9. Cập nhật trình xử lý nhấp chuột cho **Đếm** Cái nút để thiết lập màu nền cho **Số không** Cái nút thành thứ gì đó khác ngoài màu xám để cho thấy nó hiện đang hoạt động. Gợi ý: Bạn có thể sử dụng `findViewById` trong trường hợp này.
10. Cập nhật trình xử lý nhấp chuột cho **Số không** Cái nút để thiết lập lại màu thành màu xám, sao cho màu xám khi số đếm bằng không.



Trả lời những câu hỏi này

Câu hỏi 1

Hai thuộc tính ràng buộc bối trí nào trên **Số không** Cái nút đặt nó theo chiều dọc cách đều hai cái kia Cái nút các nguyên tố? (Chọn 2 câu trả lời.)

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

Câu hỏi 2

Thuộc tính ràng buộc bố trí nào trên **Số không** Cái nút đặt nó theo chiều ngang thẳng hàng với hai cái kia Cái nút các yếu tố?

- Ứng dụng:layout_constraintLeft_toLeftOf="cha mẹ"
- Ứng dụng:layout_constraintBottom_toTopOf="@+id/button_count"
- android:layout_marginBottom="8dp"
- Ứng dụng:layout_constraintTop_toBottomOf="@+id/button_toast"

Câu hỏi 3

Chữ ký chính xác cho một phương pháp được sử dụng với là gì? android:onClick Thuộc tính XML?

- công khai void callMethod()
- public void callMethod(Xem chế độ xem)
- private void callMethod(Xem chế độ xem)
- public boolean callMethod(Xem chế độ xem)

Câu hỏi 4

Trình xử lý nhấp chuột cho **Đếm** Cái nút bắt đầu với chữ ký phương thức sau:

```
public void countUp(Xem chế độ xem)
```

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

- Sử dụng `tìmViewById` để tìm thấy **Đếm** Cái nút. Gán kết quả cho một `Xem biến`, và sau đó sử dụng thiết lập **Màu nền()**.
- Sử dụng `xem tham số` được truyền cho trình xử lý nhấp chuột với thiết lập **Màu nền()**: `view.setBackgroundColor()`

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Kiểm tra xem ứng dụng có các tính năng sau không:

- Nó hiển thị **Số không** cái nút.
- Các **Số không** nút nằm giữa **Nướng** và **Đếm** nút bấm.
- Ứng dụng bao gồm việc triển khai `activity_main.xml`, `activity_main.xml` (đất), `Vàactivity_main.xml` (cực lớn).
- Ứng dụng bao gồm một triển khai của phương pháp xử lý nhấp chuột cho **Số không** nút để thiết lập lại số đếm về 0. Phương pháp phải hiển thị số đếm bằng không trong `Hiển thị_số_lượt` xem văn bản. Trình xử lý nhấp chuột cũng phải thiết lập lại **Số không** màu nền của nút chuyển sang màu xám.
- Phương pháp xử lý nhấp chuột cho **Đếm** nút đã được cập nhật để thay đổi màu nền của riêng nó tùy thuộc vào việc số đếm mới là lẻ hay chẵn. Phương pháp này phải sử dụng `xemtham số` để truy cập vào nút. Phương pháp này cũng phải thay đổi nền của **Số không** nút có màu khác ngoài màu xám.

Bài 1.3: Văn bản và chế độ xem cuộn

Giới thiệu

Các [Xem văn bản](#) lớp là một lớp con của [Xem](#) lớp hiển thị văn bản trên màn hình. Bạn có thể kiểm soát cách văn bản xuất hiện bằng [Xem](#) văn bản thuộc tính trong tệp bố cục XML. Bài thực hành này cho thấy cách làm việc với nhiều [Xem](#) văn bản các thành phần, bao gồm một thành phần mà người dùng có thể cuộn nội dung theo chiều dọc.

Nếu bạn có nhiều thông tin hơn mức hiển thị trên màn hình của thiết bị, bạn có thể tạo một [chế độ xem cuộn](#) để người dùng có thể cuộn theo chiều dọc bằng cách vuốt lên hoặc xuống, hoặc theo chiều ngang bằng cách vuốt sang phải hoặc trái.

Bạn thường sử dụng chế độ xem cuộn cho các tin tức, bài viết hoặc bất kỳ văn bản dài nào không vừa hoàn toàn trên màn hình. Bạn cũng có thể sử dụng chế độ xem cuộn để cho phép người dùng nhập nhiều dòng văn bản hoặc kết hợp các thành phần UI (như trường văn bản và nút) trong chế độ xem cuộn.

Các [CuộnXem](#) Lớp này cung cấp bối cảnh cho chế độ xem cuộn.CuộnXem là một lớp con của [Bối cảnh khung](#). Chỉ đặt một xem như một phần tử con bên trong nó—một phần tử con chứa toàn bộ nội dung để cuộn. Bản thân phần tử con này có thể là một [XemNhóm](#) (chẳng hạn như [Bối cảnh tuyến tính](#)) chứa các thành phần UI.

Bối cảnh phức tạp có thể gặp vấn đề về hiệu suất với các chế độ xem con như hình ảnh. Một lựa chọn tốt cho Xem trong vòng một CuộnXem là một Bối cảnh tuyến tính được sắp xếp theo chiều dọc, hiển thị các mục mà người dùng có thể cuộn qua (chẳng hạn như Xem văn bản các yếu tố).

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

Những điều bạn nên biết

Bạn sẽ có thể:

- Tạo ứng dụng Hello World bằng Android Studio.
- Chạy ứng dụng trên trình giả lập hoặc thiết bị.
- Thực hiện một Xem văn bản trong bối cảnh của một ứng dụng.
- Tạo và sử dụng tài nguyên chuỗi.

Những gì bạn sẽ học được

- Cách sử dụng mã XML để thêm nhiều Xem văn bản các yếu tố.
- Cách sử dụng mã XML để xác định cuộn Xem.
- Cách hiển thị văn bản dạng tự do với một số thẻ định dạng HTML.
- Làm thế nào để tạo kiểu Xem văn bản màu nền và màu chữ.
- Cách chèn liên kết web vào văn bản.

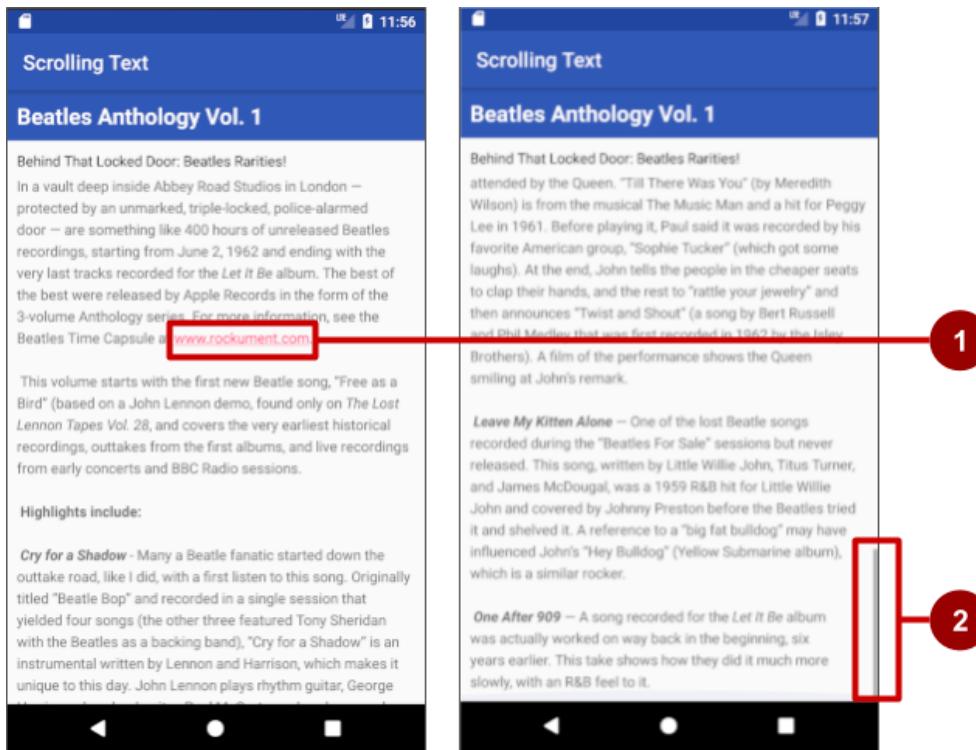
Bạn sẽ làm gì

- Tạo ứng dụng ScrollingText.
- Thay đổi ConstraintLayout ViewGroup Để Bối cảnh tương đối.
- Thêm hai Xem văn bản các thành phần cho tiêu đề và tiểu đề của bài viết.
- Sử dụng Văn bản Appearance Kiểu dáng và màu sắc cho tiêu đề và tiểu đề bài viết.

- Sử dụng thẻ HTML trong chuỗi văn bản để kiểm soát định dạng.
- Sử dụng khoảng cách dòng Extrathuộc tính để thêm khoảng cách dòng cho dễ đọc.
- Thêm một CuộnXem để bố trí để cho phép cuộn một Xem văn bản bảyếu tố.
- Thêm vào tự động liên kết thuộc tính cho phép URL trong văn bản hoạt động và có thể nhấp vào.

Tổng quan về ứng dụng

Ứng dụng Scrolling Text thể hiện [CuộnXem](#). Thành phần UI.CuộnXem là một Xem Nhóm trong ví dụ này có chứa một Chế độ xem văn bản.Nó hiển thị một trang văn bản dài—trong trường hợp này là bài đánh giá album nhạc—mà người dùng có thể cuộn theo chiều dọc để đọc bằng cách vuốt lên và xuống. Một thanh cuộn xuất hiện ở lề phải. Ứng dụng này cho thấy cách bạn có thể sử dụng văn bản được định dạng với các thẻ HTML tối thiểu để đặt văn bản thành in đậm hoặc in nghiêng và với các ký tự xuống dòng để phân tách các đoạn văn. Bạn cũng có thể bao gồm các liên kết web đang hoạt động trong văn bản.



Trong hình trên, nội dung sau xuất hiện:

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

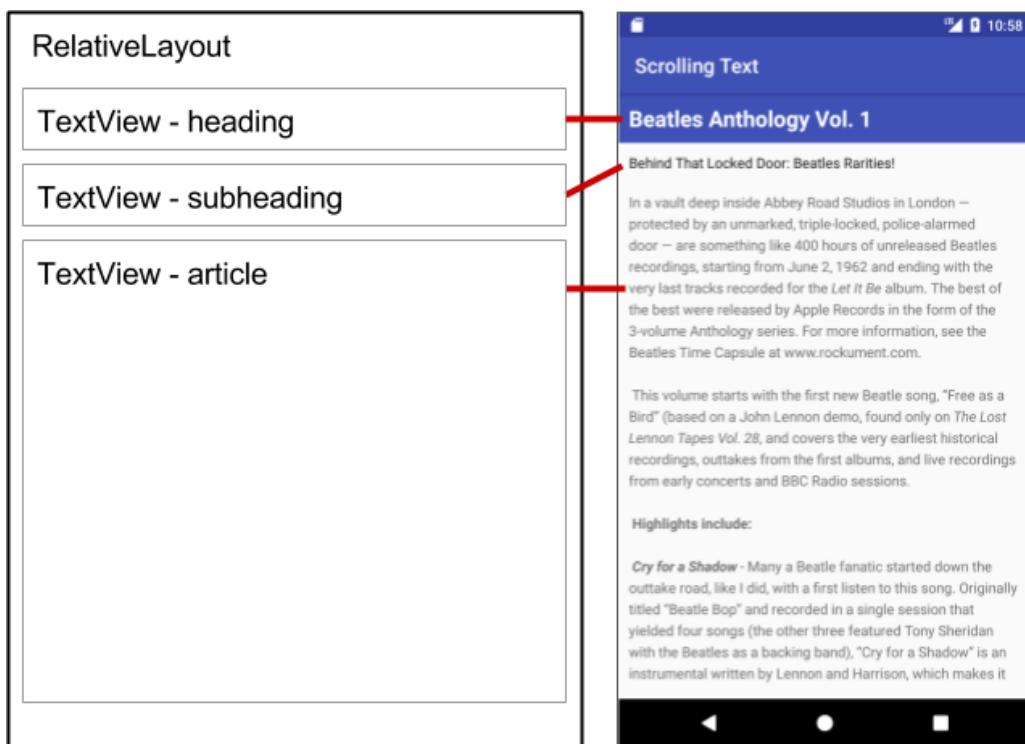
để biết thông tin cập nhật mới nhất.

1.Một liên kết web đang hoạt động được nhúng trong văn bản dạng tự do

2.Thanh cuộn xuất hiện khi cuộn văn bản

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

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



Bạn sẽ thực hiện tất cả những thay đổi này trong mã XML và trong strings.xml file. Bạn sẽ chỉnh sửa mã XML cho bố cục trong ngăn Văn bản, mà bạn hiển thị bằng cách nhấp vào **Chữ tab**, thay vì nhấp vào **Thiết kế tab** cho ngăn Thiết kế. Một số thay đổi đối với các thành phần và thuộc tính UI dễ thực hiện hơn trực tiếp trong ngăn Văn bản bằng cách sử dụng mã nguồn XML.

1.1 Tạo dự án và các thành phần TextView

Trong nhiệm vụ này, bạn sẽ tạo dự án và Xem văn bản các yếu tố và sử dụng Xem văn bản thuộc tính để tạo kiểu cho văn bản và nền.

Mẹo: Để tìm hiểu thêm về các thuộc tính này, hãy xem [Xem văn bản](#) tham quyền giải quyết.

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

Thuộc tính	Giá trị
Tên ứng dụng	Văn bản cuộn
Tên công ty	android.example.com (hoặc tên miền của riêng bạn)
SDK tối thiểu cho điện thoại và máy tính bảng	API15: Android 4.0.3 IceCreamSandwich
Bản mẫu	Hoạt động trống
Tạo hộp kiểm Tệp Bổ cục	Đã chọn
Hộp kiểm Tương thích ngược (AppCompat)	Đã chọn

2.Trong **Ứng dụng > res > bộ cục** thư mục trong **Dự án > Android** cửa sổ, mở **hoạt động_main.xml** tập tin và nhấp vào **Chữ** tab để xem mã XML.

Ở trên cùng, hoặc gốc rễ của Xem hệ thống phân cấp là **Bố cục ràng buộc** Nhóm xem:

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

3.Thay đổi điều nàyXem Nhóm ĐẾN Bố cục tương đối. Dòng mã thứ hai bây giờ trông giống như thế này:

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

Bố cục tương đối cho phép bạn đặt các thành phần UI tương đối với nhau hoặc tương đối với phần tử cha Bố cục tương đối chính nó.

Mặc định "Hello World" Xem văn bản phần tử được tạo bởi mẫu Bố cục trống vẫn có các thuộc tính ràng buộc (chẳng hạn như ứng dụng:layout_constraintBottom_toBottomOf="cha"). Để rõ ràng, bạn sẽ xóa chúng ở bước tiếp theo.

4.Xóa dòng mã XML sau đây có liên quan đến Bố cục ràng buộc:

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

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

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

5.Thêm một Xem văn bản phần tử phía trên "Hello World" Xem văn bản bằng cách nhập <Xem văn bản. MỘT Xem văn bản khôi xuất hiện kết thúc bằng /> và hiển thị chiều rộng bố cục. Và chiều cao bố cục các thuộc tính, được yêu cầu cho Chế độ xem văn bản.

6.Nhập các thuộc tính sau cho Chế độ xem văn bản. Khi bạn nhập từng thuộc tính và giá trị, các gợi ý sẽ xuất hiện để hoàn thiện tên thuộc tính hoặc giá trị.

Xem văn bản #1 thuộc tính	Giá trị
android:bố_cục_chiều_rộng	"phù hợp với cha mẹ"
android: layout_height	"bọc_nội_dung"
android:id	"@+id/tiêu đề bài viết"
android:nền	"@color/colorChính"
android:màu văn bản	"@android:màu/trắng"
android:đệm	"10dp"
android:textAppearance	"@android:style/TextAppearance.DeviceDefault.Large"
android:kiểu chữ	"in đậm"
android:văn bản	"Tiêu đề bài viết"

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

Đặt con trỏ vào chuỗi được mã hóa cứng, nhấn Alt-Enter (Option-Enter trên máy Mac) và chọn **Trích xuất tài nguyên chuỗi**. Hãy chắc chắn rằng **Tạo tài nguyên trong thư mục** tùy chọn được chọn, sau đó chỉnh sửa tên tài nguyên cho giá trị chuỗi thành **tiêu đề bài viết**.

Tài nguyên chuỗi được mô tả chi tiết trong [Tài nguyên chuỗi](#).

8.Trích xuất tài nguyên kích thước cho android:đệm chuỗi được mã hóa cứng của thuộc tính "10dp" trong TextView để tạo kích thước.xml và thêm một mục vào đó.

Đặt con trỏ vào chuỗi được mã hóa cứng, nhấn Alt-Enter (Option-Enter trên máy Mac) và chọn **Trích xuất tài nguyên kích thước**. Hãy chắc chắn rằng **Tạo tài nguyên trong thư mục** tùy chọn được chọn, sau đó chỉnh sửa tên Tài nguyên thành **đệm thường xuyên**.

9.Thêm một cái nữa Xem văn bản phần tử phía trên "Hello World" Xem văn bản và bên dưới Xem văn bản bạn đã tạo ở các bước trước. Thêm các thuộc tính sau vào Chế độ xem văn bản:

Xem văn bản #2 Thuộc tính	Giá trị
chiều rộng bố cục	"phù hợp với cha mẹ"

chiều cao bố cục	"bọc_nội_dung"
android:id	"@+id/tiêu đề phụ của bài viết"
android:bố_cục_bên_dưới	"@id/tiêu đề bài viết"
android:đệm	"@dimen/padding_regular"
android:textAppearance	"@android:style/TextAppearance.DeviceDefault"
android:văn bản	"Tiêu đề bài viết"

Bởi vì bạn đã trích xuất tài nguyên kích thước cho "10dp" chuỗi đến đệm thường xuyên trong những gì đã tạo ra trước đó. Chế độ xem văn bản, bạn có thể sử dụng "@kích thước/padding_regular" cho android:đệm thuộc tính này. Chế độ xem văn bản.

10. Trích xuất tài nguyên chuỗi cho android:văn bản. Chuỗi được mã hóa cứng của thuộc tính "Tiêu đề bài viết" trong Xem văn bản ĐẾN **tiêu đề bài viết**.

11. Trong "Xin chào thế giới" Xem văn bản phần tử, xóa ràng buộc bố cục thuộc tính:

ứng dụng:layout_constraintBottom_toBottomOf="cha mẹ"
 ứng dụng:layout_constraintLeft_toLeftOf="cha mẹ" ứng
 dụng:layout_constraintRight_toRightOf="cha mẹ" ứng
 dụng:layout_constraintTop_toTopOf="cha mẹ"

12. Thêm những điều sau đây Xem văn bản thuộc tính cho "Hello World" Xem văn bản phần tử và thay đổi android:văn bản thuộc tính:

Xem văn bản Thuộc tính	Giá trị
android:id	"@+id/bài viết"
android:bố_cục_bên_dưới	"@id/article_subheading"
android:lineSpacingExtra	"5sp"
android:đệm	"@dimen/padding_regular"
android:văn bản	Thay đổi thành "Văn bản bài viết"

13.Trích xuất tài nguyên chuỗi cho "Văn bản bài viết"ĐẾNvăn bản bài viếtvà trích xuất tài nguyên kích thước cho "5sp"ĐẾNkhoảng cách dòng.

14.Định dạng lại và căn chỉnh mã bằng cách chọnMã > Định dạng lại mã. Bạn nên định dạng lại và căn chỉnh mã của mình sao cho bạn và người khác có thể dễ hiểu hơn.

1.2 Thêm nội dung bài viết

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

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

1.Trong **ứng dụng > res > giá trị** thư mục, mở **chuỗi.xml**.

2.Mở bất kỳ tệp văn bản nào có nhiều văn bản hoặc mở [tệp strings.xml của ứng dụng ScrollingText đã hoàn thành](#).

3.Nhập giá trị cho các chuỗi tiêu đề bài viếtVà tiêu đề bài viết với tiêu đề và phụ đề được tạo sẵn hoặc sử dụng các giá trị trong tệp strings.xml của ứng dụng ScrollingText đã hoàn thiện. Làm cho các giá trị chuỗi thành văn bản một dòng không có thẻ HTML hoặc nhiều dòng.

4.Nhập hoặc sao chép và dán văn bản chovăn bản bài viêtsợi dây.

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

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

- Khi bạn nhập hoặc dán văn bản vào strings.xml file, các dòng văn bản không bao quanh dòng tiếp theo—chúng mở rộng ra ngoài lề phải. Đây là hành vi đúng—mỗi dòng văn bản mới bắt đầu từ lề trái đại diện cho toàn bộ một đoạn văn. Nếu bạn muốn văn bản trong chuỗi.xml để ngắt dòng, bạn có thể nhấn Return để kết thúc dòng cứng hoặc định dạng văn bản trước trong trình soạn thảo văn bản có kết thúc dòng cứng.
- Đi vào\Nđể biểu diễn sự kết thúc của một dòng, và một dòng khác\Nđể biểu diễn một dòng trống. Bạn cần thêm các ký tự kết thúc dòng để các đoạn văn không bị chồng lên nhau.

- Nếu bạn có dấu nháy đơn (`) trong văn bản của mình, bạn phải thoát khỏi nó bằng cách đặt trước nó một dấu gạch chéo ngược (`\`). Nếu văn bản của bạn có dấu ngoặc kép, bạn cũng phải thoát khỏi dấu ngoặc kép đó (`\"`). Bạn cũng phải thoát khỏi bất kỳ ký tự nào không phải ASCII. Xem [Định dạng và kiểu dáng](#) phần của [Tài nguyên chuỗi](#) để biết thêm chi tiết.
- Nhập mã HTML **Và** gắn thẻ xung quanh các từ cần được in đậm.
- Nhập mã HTML **<tôi>Và</tôi>** thẻ xung quanh các từ cần được in nghiêng. Nếu bạn sử dụng dấu nháy đơn cong trong cụm từ in nghiêng, hãy thay thế chúng bằng dấu nháy đơn thẳng.
- Bạn có thể kết hợp chữ in đậm và chữ in nghiêng bằng cách kết hợp các thẻ, như trong **<i>...tù...</i>**. Các thẻ HTML khác bị bỏ qua.
- Đặt toàn bộ văn bản trong **<chuỗi tên="article_text"> </string>** trong strings.xml filà.
- Bao gồm một liên kết web để kiểm tra, chẳng hạn như www.google.com. (Ví dụ dưới đây sử dụng (www.rockument.com).) *Đừng* sử dụng thẻ HTML, vì bất kỳ thẻ HTML nào ngoại trừ thẻ in đậm và in nghiêng đều bị bỏ qua và được hiển thị dưới dạng văn bản, đây không phải là điều bạn muốn.

```

activity_main.xml strings.xml MainActivity.java
Edit translations for all locales in the translations editor.
resources string
1 <resources>
2   <string name="app_name">Scrolling Text</string>
3   <string name="article_title">Beatles Anthology Vol. 1</string>
4   <string name="article_subtitle">Behind That Locked Door: Beatles Rarities!</string>
5   <string name="article_text">In a vault deep inside Abbey Road Studios in London – protected by an unmarked, triple-locked door – lay a time capsule containing a collection of rare Beatles recordings. This volume starts with the first new Beatle song, "Free as a Bird" (based on a John Lennon demo, found only on <i>The Lost Beatles Anthology</i>). Highlights include:</b>
6     <b>Cry for a Shadow</b> – Many a Beatle fanatic started down the outtake road, like I did, with a first listen to this one. <b>My Bonnie</b> and <b>Ain't She Sweet</b> – At the same session, the Beatles played on "My Bonnie" (the first song ever recorded by the band) and "Ain't She Sweet" (a Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular one for the Beatles). <b>Searchin'</b> – A Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular one for the Beatles. <b>Love Me Do</b> – An early version of the song, played a bit slower and with more of a blues feeling, and a cool bass line. <b>She Loves You – Till There Was You – Twist and Shout</b> – Live at the Princess Wales Theatre by Leicester Square. <b>Leave My Kitten Alone</b> – One of the lost Beatle songs recorded during the "Beatles For Sale" sessions but never released. <b>One After 909</b> – A song recorded for the <i>Let It Be</i> album was actually worked on way back in the beginning of 1968.
7   \n\n
8   This volume starts with the first new Beatle song, "Free as a Bird" (based on a John Lennon demo, found only on <i>The Lost Beatles Anthology</i>). Highlights include:</b>
9   \n\n
10  <b>Cry for a Shadow</b> – Many a Beatle fanatic started down the outtake road, like I did, with a first listen to this one. <b>My Bonnie</b> and <b>Ain't She Sweet</b> – At the same session, the Beatles played on "My Bonnie" (the first song ever recorded by the band) and "Ain't She Sweet" (a Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular one for the Beatles). <b>Searchin'</b> – A Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular one for the Beatles. <b>Love Me Do</b> – An early version of the song, played a bit slower and with more of a blues feeling, and a cool bass line. <b>She Loves You – Till There Was You – Twist and Shout</b> – Live at the Princess Wales Theatre by Leicester Square. <b>Leave My Kitten Alone</b> – One of the lost Beatle songs recorded during the "Beatles For Sale" sessions but never released. <b>One After 909</b> – A song recorded for the <i>Let It Be</i> album was actually worked on way back in the beginning of 1968.
11   \n\n
12   <b>For more information, see the Beatles Time Capsule at <a href="http://www.rockument.com">www.rockument.com.</b>
13   \n\n
14   This volume starts with the first new Beatle song, "Free as a Bird" (based on a John Lennon demo, found only on <i>The Lost Beatles Anthology</i>). Highlights include:</b>
15   \n\n
16   <b>Cry for a Shadow</b> – Many a Beatle fanatic started down the outtake road, like I did, with a first listen to this one. <b>My Bonnie</b> and <b>Ain't She Sweet</b> – At the same session, the Beatles played on "My Bonnie" (the first song ever recorded by the band) and "Ain't She Sweet" (a Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular one for the Beatles). <b>Searchin'</b> – A Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular one for the Beatles. <b>Love Me Do</b> – An early version of the song, played a bit slower and with more of a blues feeling, and a cool bass line. <b>She Loves You – Till There Was You – Twist and Shout</b> – Live at the Princess Wales Theatre by Leicester Square. <b>Leave My Kitten Alone</b> – One of the lost Beatle songs recorded during the "Beatles For Sale" sessions but never released. <b>One After 909</b> – A song recorded for the <i>Let It Be</i> album was actually worked on way back in the beginning of 1968.
17   \n\n
18   <b>For more information, see the Beatles Time Capsule at <a href="http://www.rockument.com">www.rockument.com.</b>
19   \n\n
20   This volume starts with the first new Beatle song, "Free as a Bird" (based on a John Lennon demo, found only on <i>The Lost Beatles Anthology</i>). Highlights include:</b>
21   \n\n
22   <b>Cry for a Shadow</b> – Many a Beatle fanatic started down the outtake road, like I did, with a first listen to this one. <b>My Bonnie</b> and <b>Ain't She Sweet</b> – At the same session, the Beatles played on "My Bonnie" (the first song ever recorded by the band) and "Ain't She Sweet" (a Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular one for the Beatles). <b>Searchin'</b> – A Jerry Leiber – Mike Stoller comedy song that was a hit for the Coasters in 1957, and a popular one for the Beatles. <b>Love Me Do</b> – An early version of the song, played a bit slower and with more of a blues feeling, and a cool bass line. <b>She Loves You – Till There Was You – Twist and Shout</b> – Live at the Princess Wales Theatre by Leicester Square. <b>Leave My Kitten Alone</b> – One of the lost Beatle songs recorded during the "Beatles For Sale" sessions but never released. <b>One After 909</b> – A song recorded for the <i>Let It Be</i> album was actually worked on way back in the beginning of 1968.
23   \n\n
24   <b>For more information, see the Beatles Time Capsule at <a href="http://www.rockument.com">www.rockument.com.</b>
25   \n\n
26   <b>For more information, see the Beatles Time Capsule at <a href="http://www.rockument.com">www.rockument.com.</b>
27   \n\n
</resources>

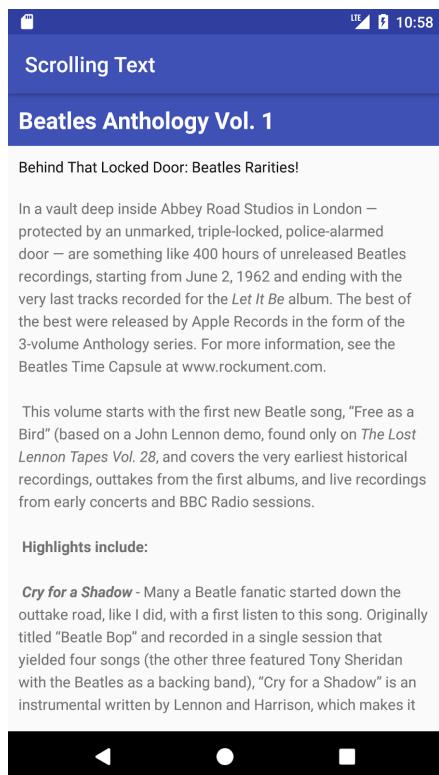
```

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

1.3 Chạy ứng dụng

Chạy ứng dụng. Bài viết xuất hiện, nhưng người dùng không thể cuộn bài viết vì bạn chưa thêm Cuộn xem (mà bạn sẽ thực hiện trong nhiệm vụ tiếp theo). Cũng lưu ý rằng việc chạm vào liên kết web hiện không có tác dụng gì. Bạn cũng sẽ sửa lỗi đó trong nhiệm vụ tiếp theo.



Mã giải bài tập 1

Tệp bố cục trống như sau:

```
<?xml phiên bản="1.0" mã hóa="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent" công  
    cụ:context="com.example.android.scrollingtext.MainActivity">  
  
<Xem văn bản  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/article_heading"  
    android:background="@color/colorPrimary"  
    android:padding="@dimen/padding_regular"  
    android:text="@string/article_title"  
    android:textAppearance=  
        "@android:style/TextAppearance.DeviceDefault.Large"  
    android:textColor="@android:color/white"  
    android:textStyle="bold" />  
  
<Xem văn bản  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/article_subheading"  
    android:layout_below="@+id/article_heading"  
    android:padding="@dimen/padding_regular"  
    android:text="@string/article_subtitle"  
    android:textAppearance=  
        "@android:style/TextAppearance.DeviceDefault" />  
  
<Xem văn bản  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:id="@+id/article"  
    android:layout_below="@+id/article_subheading"  
    android:lineSpacingExtra="@dimen/line_spacing"  
    android:padding="@dimen/padding_regular"  
    android:text="@string/article_text" />  
  
</Bối cục tương đối>
```

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

Trong nhiệm vụ trước, bạn đã tạo ứng dụng ScrollingText bằng Xem văn bản các thành phần cho tiêu đề bài viết, phụ đề và văn bản dài của bài viết. Bạn cũng đã bao gồm một liên kết web, nhưng liên kết vẫn chưa hoạt động. Bạn sẽ thêm mã để kích hoạt liên kết.

Ngoài ra, Xem văn bản thân nó không thể cho phép người dùng cuộn văn bản bài viết để xem toàn bộ. Bạn sẽ thêm một XemNhómđiềuCuộnXemvới bố cục XML sẽ tạo ra Xem văn bản có thể cuộn được.

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

Thêm vào `android:autoLink="web"` thuộc tính cho `bài viết TextView`. Mã XML cho điều này Xem văn bản bây giờ trông như thế này:

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

2.2 Thêm ScrollView vào bố cục

Để thực hiện một Xem (chẳng hạn như một Chế độ xem văn bản) có thể cuộn, nhưng Xem bên trong Một Chế độ xem cuộn.

1.Thêm một Cuộn Xem giữa các Tiêu đề phụ bài viết Xem văn bản và bài viết TextView.

Khi bạn nhập <Cuộn Xem>, Android Studio tự động thêm </Cuộn Xem> ở cuối và trình bày android:layout_height thuộc tính có gợi ý.

2.Chọn **nội dung bọc** từ những gợi ý cho cả hai thuộc tính.

Mã cho hai Xem văn bản các yếu tố và Cuộn Xem bằng giờ trông như thế này:

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

<Cuộn Xem
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"></ScrollView>

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

3. Di chuyển phần kết thúc </Cuộn Xem> mã số sau đây để bài viết TextView vì vậy mà bài viết TextView có thuộc tính hoàn toàn nằm bên trong Chế độ xem cuộn.

4. Xóa thuộc tính sau khỏi bài viết TextView và thêm nó vào Chế độ xem cuộn:

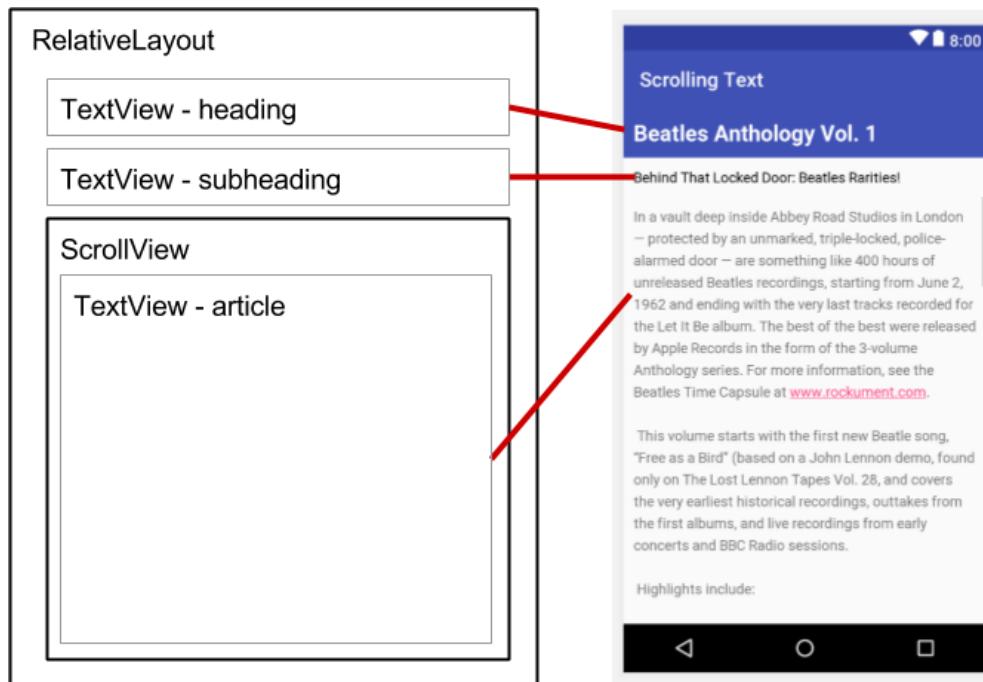
```
    android:layout_below="@+id/article_subheading"
```

Với thuộc tính trên,CuộnXemphần tử sẽ xuất hiện bên dưới tiêu đề phụ của bài viết. Bài viết nằm bên trongCuộnXemyếu tố.

5.ChọnMã > **Định dạng lại mã**để định dạng lại mã XML sao chobài viết TextView bây giờ xuất hiện thụt vào bên trong <CuộnXemmã số.

6.Nhấp vàoXem trướctab ở bên phải trình chỉnh sửa bối cục để xem bản xem trước của bối cục.

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



2.3 Chạy ứng dụng

Để kiểm tra cách văn bản cuộn:

Tác phẩm này được cấp phép theo mộtGiấy phép Creative Commons Ghi công 4.0 Quốc tế . PDF này là bản chụp nhanh một lần. Xemdev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

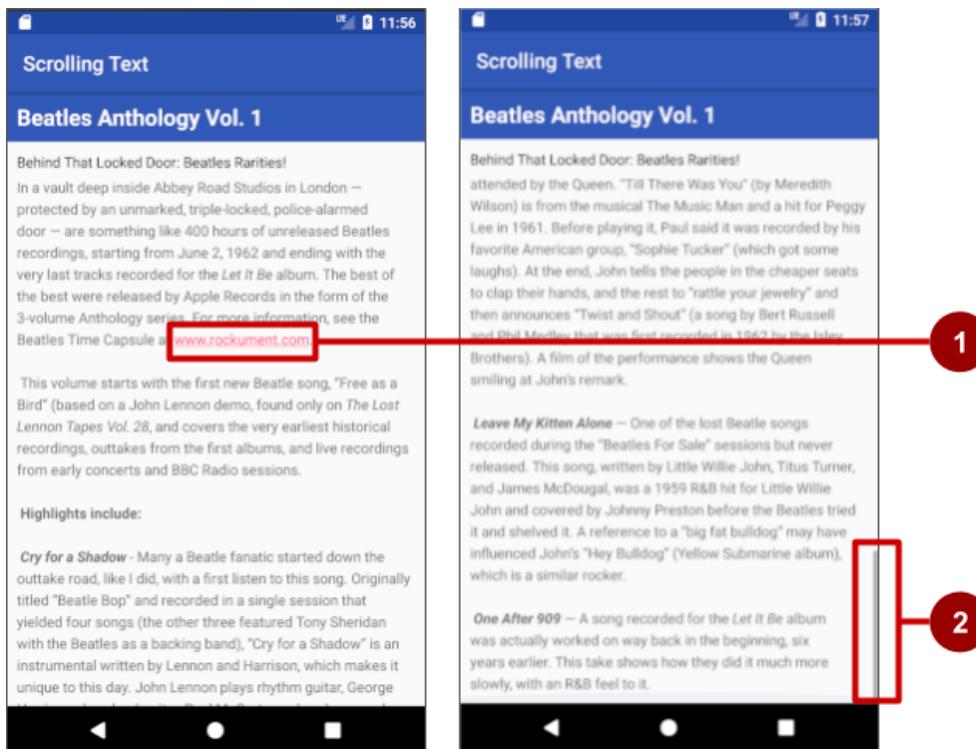
1.Chạy ứng dụng trên thiết bị hoặc trình giả lập.

Vuốt lên và xuống để cuộn bài viết. Thanh cuộn sẽ xuất hiện ở lề phải khi bạn cuộn.

Nhấn vào liên kết web để đi đến trang web.android:tự độngLiên kết thuộc tính chuyển bất kỳ URL nào có thể nhận dạng được trongXem văn bản (chẳng hạn nhưwww.rockument.com)vào một liên kết web.

2.Xoay thiết bị hoặc trình giả lập của bạn trong khi chạy ứng dụng. Lưu ý cách chế độ xem cuộn mở rộng để sử dụng toàn màn hình và vẫn cuộn đúng cách.

3.Chạy ứng dụng trên máy tính bảng hoặc trình giả lập máy tính bảng. Lưu ý cách chế độ xem cuộn mở rộng để sử dụng toàn màn hình và vẫn cuộn đúng cách.



Trong hình trên, nội dung sau xuất hiện:

1.Một liên kết web đang hoạt động được nhúng trong văn bản dạng tự do

2.Thanh cuộn xuất hiện khi cuộn văn bản

Mã giải bài tập 2

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

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

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

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

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

        <Xem văn bản
            android:id="@+id/article"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:autoLink="web"
```

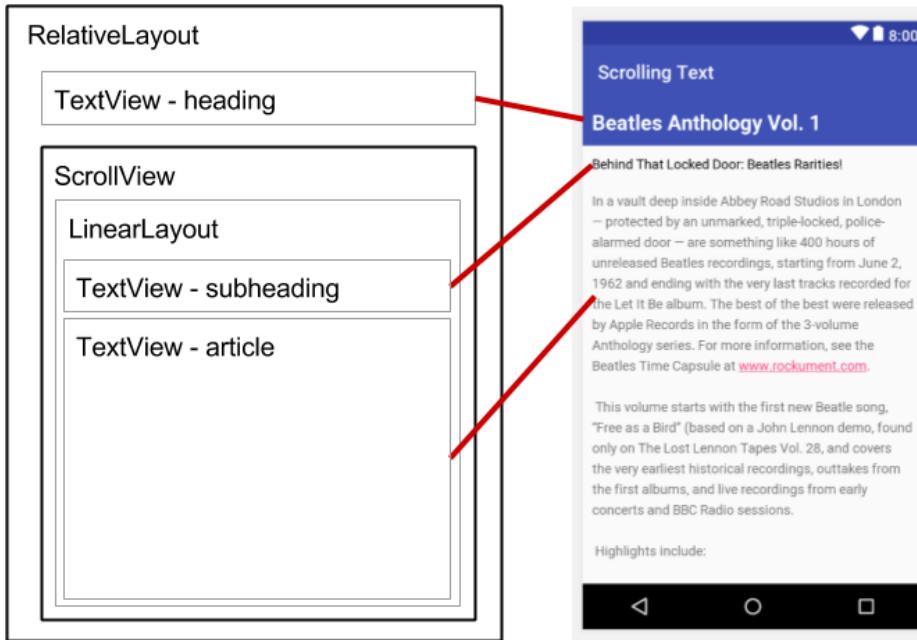
```
    android:lineSpacingExtra="@dimen/line_spacing"
    android:padding="@dimen/padding_regular"
    android:text="@string/article_text" />

</Cuộn Xem>
</Bố cục tương đối>
```

Nhiệm vụ 3: Cuộn nhiều phần tử

Như đã lưu ý trước đó, một Cuộn Xem chỉ có thể chứa một đứa trẻ Xem (chẳng hạn như bài viết TextView bạn đã tạo ra). Tuy nhiên, điều đó Xem có thể là một cái khác Xem Nhóm có chứa Xem các yếu tố, chẳng hạn như [Bố cục tuyến tính](#). Bạn có thể *tạo* Một Xem Nhóm chẳng hạn như Bố cục tuyến tính ở trong cái Cuộn xem, do đó cuộn mọi thứ bên trong Bố cục tuyến tính.

Ví dụ, nếu bạn muốn tiêu đề phụ của bài viết cuộn cùng với bài viết, hãy thêm Bố cục tuyến tính trong vòng Cuộn xem, và di chuyển tiêu đề phụ và bài viết vào Bố cục tuyến tính. Các Bố cục tuyến tính trở thành đứa con duy nhất Xem trong Cuộn Xem như thể hiện trong hình bên dưới và người dùng có thể cuộn toàn bộ Bố cục tuyến tính: tiêu đề phụ và bài viết.



3.1 Thêm LinearLayout vào ScrollView

- Mở **file main.xml** của dự án ứng dụng ScrollingText và chọn **Chữ** tab để chỉnh sửa mã XML (nếu chưa được chọn).
- Thêm một **Bố cục tuyến tính** phía trên bài viết TextView trong vòng **Chế độ xem cuộn**. Khi bạn nhập **<Bố cục tuyến tính**, Android Studio tự động thêm **</Bố cục tuyến tính>** đến cuối và trình bày **android:layout_width="wrap_content"** và **android:layout_height="wrap_content"** thuộc tính với các gợi ý. Chọn **khớp cha mẹ** **Và nội dung bọc** từ các gợi ý về chiều rộng và chiều cao của nó, tương ứng. Mã ở đầu Cuộn Xem bây giờ trông như thế này:

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

    <Bố cục tuyến tính
```

Tác phẩm này được cấp phép theo [một Giấy phép Creative Commons Ghi công 4.0 Quốc tế](#). PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

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

    <Xem văn bản
        android:id="@+id/bài viết"
```

Bạn sử dụng khόp_cha mẹ để phù hợp với chiều rộng của cha mẹ Nhóm xem. Bạn sử dụng nội dung bọc để thay đổi kích thước Bố cục tuyến tính vì vậy nó chỉ đủ lớn để chứa đồ bên trong.

3. Di chuyển phần kết thúc </Bố cục tuyến tính> mã số sau đó cái bài viết TextView. Nhưng trước sự đóng cửa </ScrollView>.

Các Bố cục tuyến tính bây giờ bao gồm bài viết TextView, và hoàn toàn nằm bên trong Chế độ xem cuộn.

4. Thêm vào android:orientation="dọc" thuộc tính cho Bố cục tuyến tính để thiết lập hướng của nó theo chiều dọc.

5. Chọn **Mã > Định dạng lại mã** để thực hiện mã đúng cách.

Các Bố cục tuyến tính bây giờ trông như thế này:

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

    <Bố cục tuyến tính
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

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

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

3.2 Di chuyển các thành phần UI trong LinearLayout

Các Bố cục tuyến tính bây giờ chỉ có một thành phần UI—bài viết TextView. Bạn muốn bao gồm Tiêu đề phụ bài viết Xem văn bản trong Bố cục tuyến tính để cả hai đều có thể cuộn.

1. Để di chuyển Tiêu đề phụ bài viết Xem văn bản, chọn mã, chọn **Chỉnh sửa > Cắt**, nhấp vào phía trên bài viết TextView bên trong Bố cục tuyến tính, và chọn **Sửa > Dán**.
2. Loại bỏ `android:layout_below="@+id/article_heading"` thuộc tính từ Chế độ xem văn bản của `article_subheading`. Bởi vì điều này Xem văn bản hiện đang ở trong Bố cục tuyến tính, thuộc tính này sẽ xung đột với Bố cục tuyến tính.
3. Thay đổi Cuộn Xem thuộc tính bố trí từ `android:layout_below="@+id/article_subheading"` ĐẾN `android:layout_below="@+id/article_heading"`. Bởi giờ tiêu đề phụ là một phần của Bố cục tuyến tính, cái Cuộn Xem phải được đặt bên dưới tiêu đề, không phải tiêu đề phụ.

Mã XML cho Cuộn Xem bây giờ khí như sau:

```
<Cuộn Xem  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_below="@+id/article_heading">  
  
    <Bố cục tuyến tính  
        android:layout_width="match_parent"  
        android:layout_height="wrap_content"  
        android:orientation="vertical">  
  
        <Xem văn bản  
            android:id="@+id/article_subheading"  
            android:layout_width="match_parent"  
            android:layout_height="wrap_content"
```

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

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

</Cuộn Xem>
```

4.Chạy ứng dụng.

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

Mã giải pháp

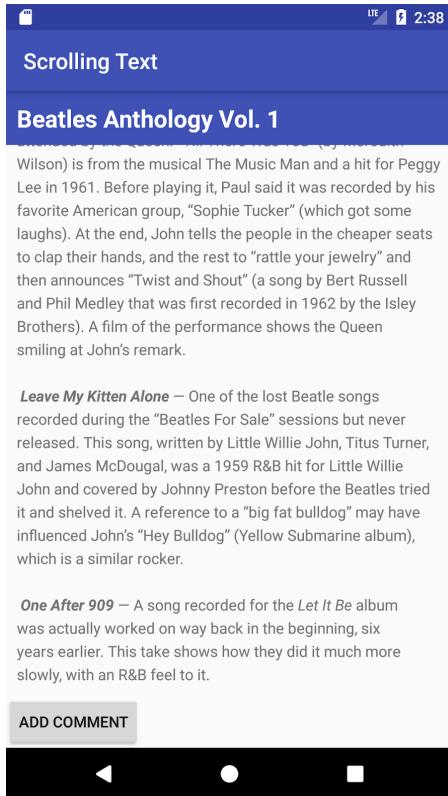
Dự án Android Studio:[Văn bản cuộn](#)

Thử thách mã hóa

Ghi chú:Mọi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách:Thêm một thành phần UI khác—a[Cái nút](#)—đếnBố cục tuyến tínhbên trongCuộnXemđể nó cuộn theo văn bản.

- Làm cho Cái nút xuất hiện bên dưới bài viết. Người dùng cuộn đến cuối bài viết để xem Cái nút.
- Sử dụng văn bản **Thêm bình luận** cho Cái nút. Đối với thử thách này, không cần phải tạo ra một phương pháp xử lý nút; tất cả những gì bạn phải làm là đặt Cái nút phàn tử ở đúng vị trí trong bố cục.



Mã giải pháp thách thức

Dự án Android Studio: [Thử thách cuộn văn bản](#)

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

Bản tóm tắt

- Sử dụng một [CuộnXem](#) để cuộn một đứa trẻ duy nhất Xem (chẳng hạn như một Xem văn bản). Một CuộnXem chỉ có thể giữ một đứa trẻ Xem hoặc Nhóm xem.
- Sử dụng một Xem Nhóm chẳng hạn như [Bố cục tuyến tính](#) khi còn là một đứa trẻ Xem trong vòng một CuộnXem để cuộn nhiều hơn một Xem phần tử. Bao gồm các phần tử trong Bố cục tuyến tính.
- Hiển thị văn bản dạng tự do trong một Xem văn bản với các thẻ định dạng HTML cho chữ in đậm và in nghiêng.
- Sử dụng \N như một ký tự kết thúc dòng trong văn bản dạng tự do để giữ cho một đoạn văn không chạy vào đoạn văn tiếp theo.
- Sử dụng android:autoLink="web" thuộc tính để làm cho các liên kết web trong văn bản có thể nhấp được.

Các khái niệm liên quan

Tài liệu khái niệm liên quan có trong [1.3 Văn bản và chế độ xem cuộn](#).

Tìm hiểu thêm

Tài liệu Android Studio:

- [Trang tải xuống Android Studio](#)
- [Làm quen với Android Studio](#)

Tài liệu dành cho nhà phát triển Android:

- [CuộnXem](#)
- [Bố cục tuyến tính](#)
- [Bố cục tương đối](#)
- [Xem](#)

- [Cái nút](#)
- [Xem văn bản](#)
- [Tài nguyên chuỗi](#)
- [Bố cục tương đối](#)

Khác:

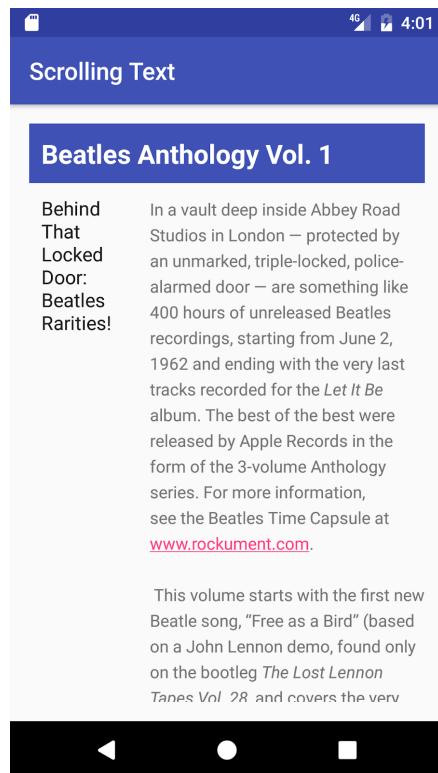
- Blog dành cho nhà phát triển Android:[Liên kết văn bản của bạn!](#)
- Đường dẫn mã:[Làm việc với TextView](#)

Bài tập về nhà

Thay đổi ứng dụng

Mở [Ứng dụng ScrollingText2](#) mà bạn đã tạo ra trong [Làm việc với các phần tử TextView](#) bài học.

1. Thay đổi tiêu đề phụ để nó nằm trong một cột bên trái có chiều rộng 100 dp, như hiển thị bên dưới.
2. Đặt văn bản của bài viết bên phải tiêu đề phụ như hình dưới đây.



Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

Trả lời những câu hỏi này

Câu hỏi 1

Bạn có thể sử dụng bao nhiêu chế độ xem trong một chế độ xem cuộn? Chọn một trong các mục sau:

- Chỉ có một chế độ xem
- Một chế độ xem hoặc một nhóm chế độ xem
- Nhiều như bạn cần

Câu hỏi 2

Bạn sử dụng thuộc tính XML nào trong bố cục tuyến tính để hiển thị các chế độ xem cạnh nhau? Chọn một:

- android:orientation="ngang"
- android:orientation="dọc"
- android:layout_width="wrap_content"

Câu hỏi 3

Bạn sử dụng thuộc tính XML nào để xác định chiều rộng của bố cục tuyến tính bên trong chế độ xem cuộn? Chọn một:

- android:layout_width="wrap_content"
- android:layout_width="phù hợp với cha mẹ"
- android:layout_width="200dp"

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Kiểm tra xem ứng dụng có các tính năng sau không:

- Bố cục hiển thị tiêu đề phụ ở cột bên trái và nội dung bài viết ở cột bên phải, như thể hiện trong hình trên.

- Các Cuộn Xem bao gồm một Bối cảnh tuyến tính với hai Xem văn bản các yếu tố.
- Các Bối cảnh tuyến tính hướng được thiết lập theo chiều ngang.

Bài 1.4: Học cách tự giúp mình

Giới thiệu

Những điều bạn nên biết

Bạn sẽ có thể:

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

Những gì bạn sẽ học được

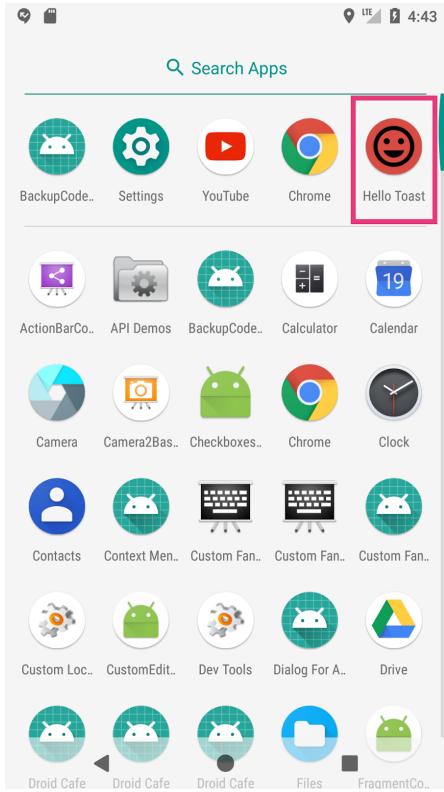
- Nơi tìm thông tin và tài nguyên dành cho nhà phát triển.
- Cách thêm biểu tượng trình khởi chạy vào ứng dụng của bạn.
- Cách tìm kiếm sự trợ giúp khi bạn đang phát triển ứng dụng Android.

Bạn sẽ làm gì

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

Tổng quan về ứng dụng

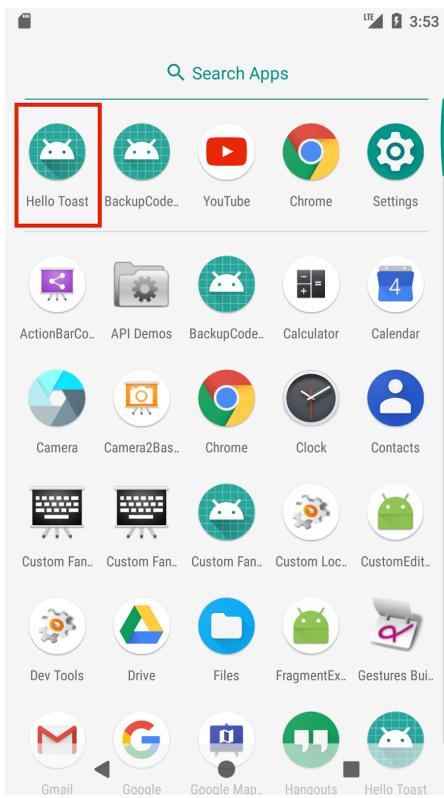
Bạn sẽ thêm biểu tượng trình khởi chạy vào ứng dụng HelloToast mà bạn đã tạo trước đó hoặc vào ứng dụng mới.



Nhiệm vụ 1: Thay đổi biểu tượng trình khởi chạy

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

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



Thay đổi biểu tượng trình khởi chạy là một quy trình từng bước đơn giản giới thiệu cho bạn các tính năng tài sản hình ảnh của Android Studio. Trong tác vụ này, bạn cũng tìm hiểu thêm về cách truy cập tài liệu Android chính thức.

1.1 Khám phá tài liệu chính thức của Android

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

Tài liệu này chứa rất nhiều thông tin được Google cập nhật thường xuyên.

16.Đi đến nhaphatnhanh.com/thietke/ .

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

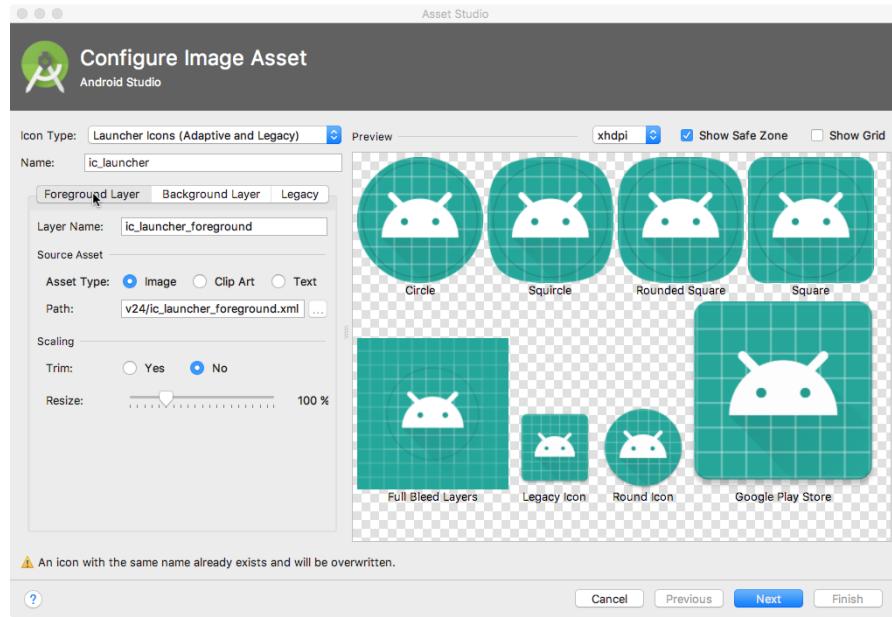
Phần này nói về Material Design, một triết lý thiết kế khái niệm phác thảo cách các ứng dụng nên trông như thế nào và hoạt động ra sao trên các thiết bị di động. Điều hướng các liên kết để tìm hiểu thêm về Material Design. Ví dụ, hãy truy cập[Phong cách](#) mục để tìm hiểu thêm về việc sử dụng màu sắc và các chủ đề khác.

- 17.Đi đến[nhà phát triển.android.com/docs/](#) để tìm thông tin về API, tài liệu tham khảo, hướng dẫn, hướng dẫn công cụ và mẫu mã.
- 18.Đi đến[nhà phát triển.android.com/distribute/](#) để tìm thông tin về việc đưa một ứng dụng vào[Google Chơi](#), Hệ thống phân phối kỹ thuật số của Google dành cho các ứng dụng được phát triển bằng Android SDK. Sử dụng[Bảng điều khiển Google Play](#) để phát triển cơ sở người dùng của bạn và bắt đầu[kiếm tiền](#).

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

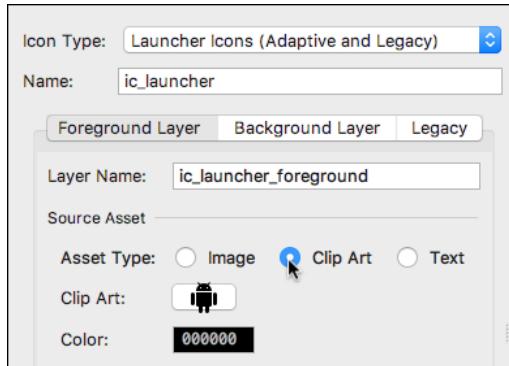
Để thêm hình ảnh clip-art làm biểu tượng trình khởi chạy, hãy làm theo các bước sau:

- 1.Mở dự án ứng dụng HelloToast từ bài học trước về cách sử dụng trình chỉnh sửa bối cảnh hoặc tạo một dự án ứng dụng mới.
- 2.Trong[Dự án > Android](#)cửa sổ,**nhấp chuột phải**(hoặc**Control-nhấp chuột**) các**độ phân giải** thư mục và chọn[Mới > Tài sản hình ảnh](#)Cửa sổ Cấu hình tài sản hình ảnh sẽ xuất hiện.



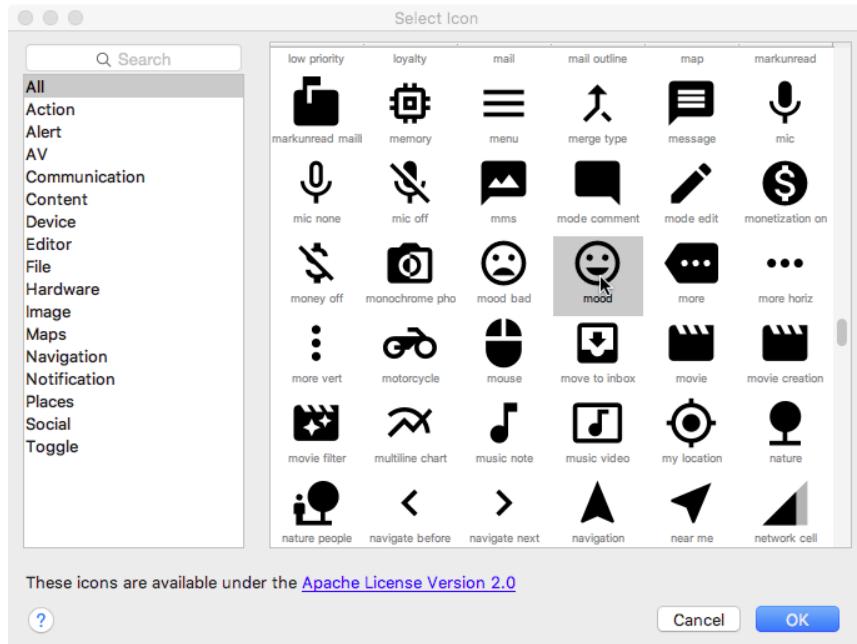
3.Trong **Loại biểu tượng** có lĩnh vực, chọn **Biểu tượng trình khởi chạy (Thích ứng & Cũ)** nếu nó chưa được chọn.

4.Nhấp vào **Lớp tiền cảnh** tab, chọn **Clip nghệ thuật** cho **Loại tài sản**.



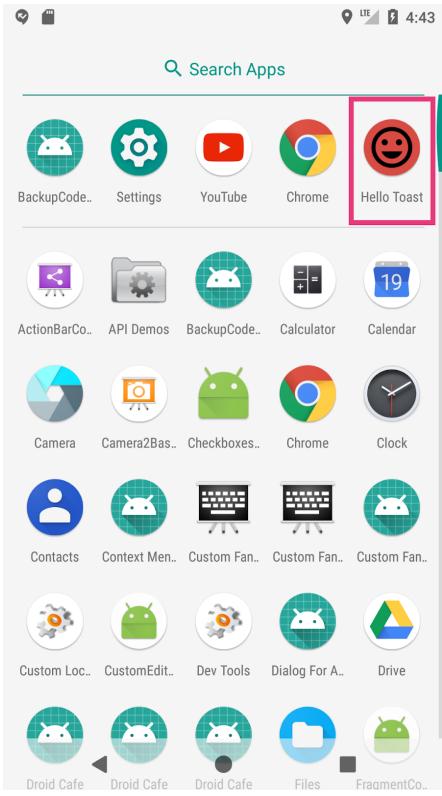
5.Nhấp vào biểu tượng trong **Clip nghệ thuật** cold. Các biểu tượng xuất hiện từ bộ biểu tượng thiết kế vật liệu.

6.Duyệt cửa sổ Chọn biểu tượng, chọn một biểu tượng thích hợp (chẳng hạn như biểu tượng tâm trạng để gợi ý tâm trạng tốt), sau đó nhấp vào **ĐƯỢC RỒI**.



- 7.Nhấp vào **Lớp nền** tab, chọn **Màu sắc** như là **Loại tài sản**, sau đó nhấp vào chip màu để chọn màu sử dụng làm lớp nền.
- 8.Nhấp vào **Di sản** tab và xem lại các thiết lập mặc định. Xác nhận rằng bạn muốn tạo biểu tượng legacy, round và Google Play Store. Nhấp vào **Kế tiếp** khi hoàn thành.
- 9.Chạy ứng dụng.

Android Studio tự động thêm hình ảnh trình khởi chạy vào **bản đồ mip** thư mục cho các mật độ khác nhau. Kết quả là, biểu tượng khởi chạy ứng dụng sẽ thay đổi thành biểu tượng mới sau khi bạn chạy ứng dụng, như hiển thị bên dưới.



Mẹo: Nhìn thấy [Biểu tượng Trình khởi chạy](#) để tìm hiểu thêm về cách thiết kế biểu tượng trình khởi chạy hiệu quả.

Nhiệm vụ 2: Sử dụng mẫu dự án

Android Studio cung cấp các mẫu cho các thiết kế ứng dụng và hoạt động phổ biến và được đề xuất. Sử dụng các mẫu tích hợp giúp tiết kiệm thời gian và giúp bạn tuân thủ các phương pháp thiết kế tốt nhất.

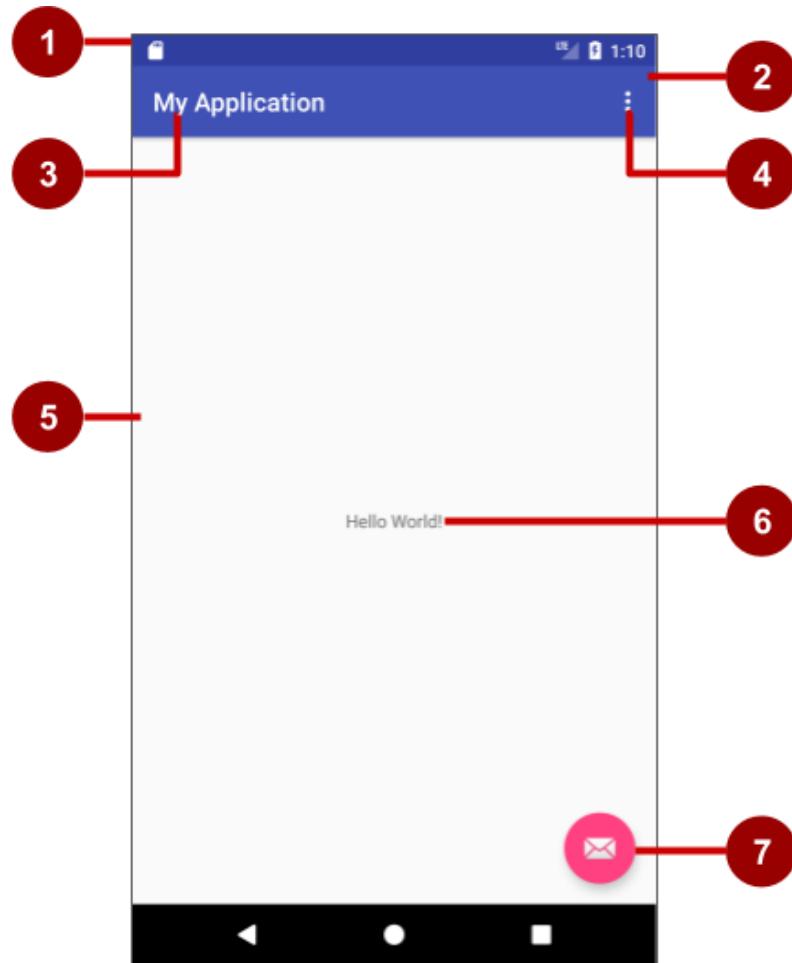
Mỗi mẫu kết hợp một hoạt động khung và giao diện người dùng. Bạn đã sử dụng mẫu Hoạt động trống. Mẫu Hoạt động cơ bản có nhiều tính năng hơn và kết hợp các tính năng ứng dụng được đề xuất, chẳng hạn như menu tùy chọn xuất hiện trên thanh ứng dụng.

2.1 Khám phá kiến trúc Hoạt động cơ bản

Mẫu Hoạt động cơ bản là mẫu đa năng do Android Studio cung cấp để hỗ trợ bạn bắt đầu phát triển ứng dụng.

- 1.Trong Android Studio, tạo một dự án mới với mẫu Hoạt động cơ bản.
- 2.Xây dựng và chạy ứng dụng.
- 3.Xác định các phần được gắn nhãn trong hình và bảng bên dưới. Tìm các phần tương đương của chúng trên màn hình thiết bị hoặc trình giả lập của bạn. Kiểm tra mã Java và tệp XML tương ứng được mô tả trong bảng.

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



Kiến trúc của mẫu Hoạt động cơ bản

#	Mô tả giao diện người dùng	Mã tham chiếu
1	Thanh trạng thái Hệ thống Android cung cấp và kiểm soát thanh trạng thái.	Không hiển thị trong mã mẫu. Bạn có thể truy cập nó từ hoạt động của mình. Ví dụ, bạn có thể ấn thanh trạng thái , nếu cần thiết.
2	AppBarLayout > Thanh công cụ Thanh ứng dụng (còn gọi là thanh hành động) cung cấp cấu trúc trực quan, các thành phần trực quan được chuẩn hóa và điều hướng. Đối với	TRONGhoat_dong_main.xml,tìm kiếm android.hỗ trợ.v7.widget.Thanh công cụ

Tác phẩm này được cấp phép theo một [Giấy phép Creative Commons Ghi công 4.0 Quốc tế](#). PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

	khả năng tương thích, AppBarLayout trong mẫu nhúng một Thanh công cụ với cùng chức năng như một Thanh hành động .	bên trong android.support.design.widget.AppBarLayout đang chờ xử lý. Thay đổi thanh công cụ để thay đổi giao diện của thanh công cụ cha, thanh ứng dụng. Ví dụ, hãy xem Hướng dẫn thanh ứng dụng .
3	Tên ứng dụng Tên này bắt nguồn từ tên gói của bạn nhưng bạn có thể chọn bất kỳ tên nào bạn muốn.	TRONGAndroidManifest.xml: android:label="@string/tên_Ứng_dung"
4	Nút tràn menu tùy chọn Các mục menu cho hoạt động, cũng như các tùy chọn toàn cầu, chẳng hạn như Tim kiếm và Cài đặt cho ứng dụng. Các mục menu ứng dụng của bạn sẽ được đưa vào menu này.	TRONGMainActivity.java: onOptionsItemSelected() thực hiện những gì xảy ra khi một mục menu được chọn. res > menu > menu_main.xml Tài nguyên chỉ định các mục menu cho menu tùy chọn.
5	Cách trình bày Xem Nhóm Các Bố cục điều phối XemNhóm là một bố cục giàu tính năng cung cấp các cơ chế cho Xem (UI) các thành phần tương tác. Giao diện người dùng của ứng dụng của bạn nằm bên trong content_main.xml file bao gồm trong này Nhóm xem.	TRONGactivity_main.xml: Không có chế độ xem nào được chỉ định trong bố cục này; thay vào đó, nó bao gồm một bố cục khác có bao gồm bố trí hướng dẫn để bao gồm @bố cục/nội dung chính/nơi các chế độ xem được chỉ định. Điều này tách biệt chế độ xem hệ thống khỏi chế độ xem dành riêng cho ứng dụng của bạn.
6	Xem văn bản Trong ví dụ, được sử dụng để hiển thị "Hello World". Thay thế bằng các thành phần UI cho ứng dụng của bạn.	TRONGnội dung_main.xml: Tất cả các thành phần UI của ứng dụng đều được xác định trong tệp này.
7	Nút hành động nổi (FAB)	TRONGhoạt động_main.xml dưới dạng một thành phần UI sử dụng biểu tượng clip-art.MainActivity.java bao gồm một phần còn lại trong khi tạo() điều đó đặt ra một khi nhấp vào() người nghe cho FAB.

2.2 Tùy chỉnh ứng dụng được tạo ra bởi mẫu

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

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem dev.android.com/courses/fundamentals-training/toc-v2

để biết thông tin cập nhật mới nhất.

cùng màu cơ bản). Bạn cũng có thể muốn xóa nút hành động nổi nếu bạn không định sử dụng nó.

- 1.Thay đổi màu của thanh ứng dụng (Thanh công cụ) TRONG `activity_main.xml` bằng cách thay đổi `android:background` thành "?thuộc tính/colorPrimaryDark", đặt màu thanh ứng dụng thành màu chính tối hơn phù hợp với thanh trạng thái:

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

2. Để xóa nút hành động nổi, hãy bắt đầu bằng cách xóa mã stub trong khi tạo() cái đó đặt một khi nhấp vào() người nghe cho nút. Mở **Hoạt động chính** và xóa khối mã sau:

```
Nút hành động nổi fab = (Nút hành động nổi)
    tìmViewById(R.id.fab);
fab.setOnClickListener( mới View.OnClickListener() {
    @Ghi đè
    public void onClick(Xem chế độ xem) {
        Snackbar.make(view, "Thay thế bằng hành động của riêng bạn",
                    Snackbar.LENGTH_LONG)
            .setAction("Hành động", null).show();
    }
});
```

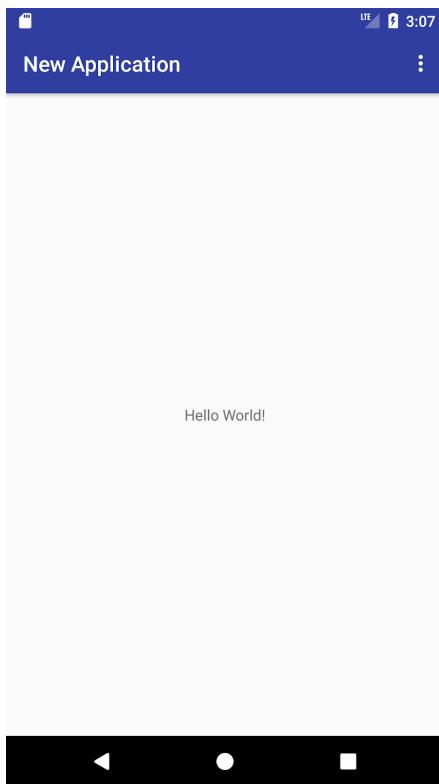
3. Để xóa nút hành động nổi khỏi bố cục, hãy xóa khối mã XML sau khỏi `activity_main.xml`:

```
<android.support.design.widget.FloatingActionButton
    android:id="@+id/fab"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="bottom|end"
    android:layout_margin="@dimen/fab_margin"
    app:srcCompat="@android:drawable/ic_dialog_email" />
```

4.Thay đổi tên ứng dụng được hiển thị trên thanh ứng dụng bằng cách thay đổi tài nguyên chuỗi app_name trong strings.xml thành tên sau:

```
<string name="app_name">Ứng dụng mới</string>
```

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



Mẹo: Nhìn thấy [Truy cập tài nguyên](#) để biết chi tiết về cú pháp XML để truy cập tài nguyên.

2.3 Khám phá cách thêm hoạt động bằng cách sử dụng mẫu

Đối với các bài thực hành cho đến nay, bạn đã sử dụng các mẫu Empty Activity và Basic Activity. Trong các bài học sau, các mẫu bạn sử dụng sẽ khác nhau, tùy thuộc vào nhiệm vụ.

Các mẫu hoạt động này cũng có sẵn từ bên trong dự án của bạn, để bạn có thể thêm nhiều hoạt động hơn vào ứng dụng của mình sau khi thiết lập dự án ban đầu. (Bạn tìm hiểu thêm về **Hoạt động** lớp học ở chương khác.)

- 1.Tạo một dự án ứng dụng mới hoặc chọn một dự án hiện có.
- 2.Trong **Dự án > Android** cửa sổ, **nhấp chuột phải** vào mục **cái java**.
- 3.Chọn **Mới > Hoạt động > Thư viện ảnh**.
- 4.Thêm một **Hoạt động**. Ví dụ, nhấp vào **Hoạt động ngăn kéo điều hướng** để thêm một **Hoạt động** có **ngăn kéo điều hướng** đến ứng dụng của bạn.
- 5.Nhấp đúp vào các tập tin bố trí cho **Hoạt động** để hiển thị chúng trong trình chỉnh sửa bố cục.

Nhiệm vụ 3: Học từ mã ví dụ

Android Studio và tài liệu Android cung cấp nhiều mẫu mã mà bạn có thể nghiên cứu, sao chép và kết hợp vào các dự án của mình.

3.1 Mẫu mã Android

Bạn có thể khám phá hàng trăm mẫu mã trực tiếp từ Android Studio.

- 1.Trong Android Studio, hãy chọn **Tệp > Mới > Nhập mẫu**.
- 2.Duyệt qua các mẫu.
- 3.Chọn một mẫu và nhấp vào **Kết nối**.
- 4.Chấp nhận các mặc định và nhấp vào **Hoàn thành**.

Ghi chú:Các mẫu có ở đây có ý nghĩa như một điểm khởi đầu cho sự phát triển tiếp theo. Chúng tôi khuyến khích bạn thiết kế và xây dựng ý tưởng của riêng bạn vào đó.

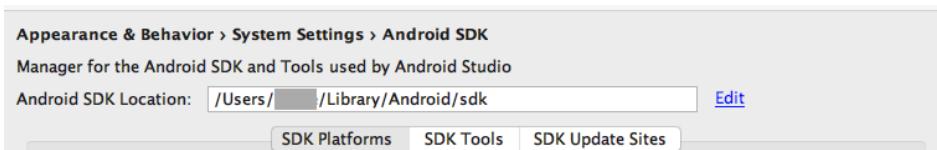
3.2 Sử dụng Trình quản lý SDK để cài đặt tài liệu ngoại tuyến

Cài đặt Android Studio cũng cài đặt các thành phần thiết yếu của Android SDK (Software Development Kit). Tuy nhiên, các thư viện và tài liệu bổ sung có sẵn và bạn có thể cài đặt chúng bằng Trình quản lý SDK.

1.Chọn**Công cụ > Android > Trình quản lý SDK**.

2.Ở cột bên trái, nhấp vào**Bộ công cụ phát triển Android**.

3.Chọn và sao chép đường dẫn đến Vị trí SDK Android ở đầu màn hình vì bạn sẽ cần đường dẫn này để tìm tài liệu trên máy tính của mình:



4.Nhấp vào**Nền tảng SDK**tab. Bạn có thể cài đặt các phiên bản bổ sung của hệ thống Android từ đây.

5.Nhấp vào**Các trang web cập nhật SDK**tab. Android Studio thường xuyên kiểm tra các trang web được liệt kê và chọn để cập nhật.

6.Nhấp vào**Công cụ SDK**tab. Bạn có thể cài đặt thêm các Công cụ SDK không được cài đặt theo mặc định, cũng như phiên bản ngoại tuyến của tài liệu dành cho nhà phát triển Android.

7.Chọn hộp kiểm "Tài liệu cho Android SDK" nếu nó chưa được cài đặt và nhấp vào**Áp dụng**.

8.Khi quá trình cài đặt hoàn tất, hãy nhấp vào**Hoàn thành**.

9.Điều hướng đến **SDK** thư mục bạn đã sao chép ở trên và mở **Tài liệu** thư mục.

10.Tìm thấy **index.html** và mở nó ra.

Nhiệm vụ 4: Nhiều tài nguyên hơn

- Các [Kênh YouTube dành cho nhà phát triển Android](#) là nguồn hướng dẫn và mèo tuyệt vời.
- Nhóm Android đăng tin tức và mèo trong [blog chính thức của Android](#).
- [Tràn Stack](#) là một cộng đồng lập trình viên giúp đỡ lẫn nhau. Nếu bạn gặp phải vấn đề, khả năng cao là ai đó đã đăng câu trả lời. Hãy thử đăng một câu hỏi như "Làm thế nào để thiết lập và sử dụng ADB qua WiFi?" hoặc "Những rò rỉ bộ nhớ phổ biến nhất trong quá trình phát triển Android là gì?"
- Và cuối cùng nhưng không kém phần quan trọng, hãy nhập câu hỏi của bạn vào Google tìm kiếm và công cụ tìm kiếm của Google sẽ thu thập các kết quả có liên quan từ tất cả các nguồn này. Ví dụ: "Phiên bản hệ điều hành Android phổ biến nhất ở Ấn Độ là gì?"

4.1 Tìm kiếm trên Stack Overflow bằng cách sử dụng thẻ

Đi đến [Tràn Stack](#) và gõ [android] trong hộp tìm kiếm. Dấu ngoặc vuông [] cho biết bạn muốn tìm kiếm các bài đăng được gắn thẻ là về Android.

Bạn có thể kết hợp các thẻ và thuật ngữ tìm kiếm để tìm kiếm cụ thể hơn. Hãy thử các tìm kiếm sau:

- [android] và [bối cảnh]
- [android] "Xin chào thế giới"

Để tìm hiểu thêm về nhiều cách bạn có thể tìm kiếm trên Stack Overflow, hãy xem [Trung tâm trợ giúp Stack Overflow](#).

Bản tóm tắt

- Tài liệu chính thức dành cho nhà phát triển Android: [nhà phát triển.android.com](#)
- Material Design là triết lý thiết kế khái niệm phác thảo cách ứng dụng sẽ trông như thế nào và hoạt động ra sao trên thiết bị di động.
- Các [Google Play](#) store là hệ thống phân phối kỹ thuật số của Google dành cho các ứng dụng được phát triển bằng Android SDK.

- Android Studio cung cấp các mẫu cho các thiết kế ứng dụng và hoạt động phổ biến và được khuyến nghị. Các mẫu này cung cấp mã hoạt động cho các trường hợp sử dụng phổ biến.
- Khi bạn tạo một dự án, bạn có thể chọn mẫu cho hoạt động đầu tiên của mình.
- Trong khi bạn tiếp tục phát triển ứng dụng của mình, các hoạt động và thành phần khác của ứng dụng có thể được tạo từ các mẫu có sẵn.
- Android Studio chứa nhiều mã mẫu mà bạn có thể nghiên cứu, sao chép và kết hợp vào các dự án của mình.

Khái niệm liên quan

Tài liệu khái niệm liên quan có trong [1.4: Tài nguyên giúp bạn học tập](#).

Tìm hiểu thêm

Tài liệu Android Studio:

- [Làm quen với Android Studio](#)
- [Cơ bản về quy trình làm việc của nhà phát triển](#)

Tài liệu dành cho nhà phát triển Android:

- [Trang web dành cho nhà phát triển Android](#)
- [Đào tạo nhà phát triển Google](#)
- [Bổ cục](#)
- [Tổng quan về tài nguyên ứng dụng](#)
- [Bổ cục](#)
- [Thực đơn](#)
- [Xem văn bản](#)
- [Tài nguyên chuỗi](#)
- [Bản kê khai ứng dụng](#)

Mẫu mã:

- [Mã nguồn cho các bài tập trên GitHub](#)
- [Mẫu mã Android dành cho nhà phát triển](#)

Video:

- [Kênh YouTube dành cho nhà phát triển Android](#)
- [Các khóa học trực tuyến của Udacity](#)

Khác:

- [Blog chính thức của Android](#)
- [Blog dành cho nhà phát triển Android](#)
- [Phòng thí nghiệm mã nguồn của Google Developers](#)
- [Tràn Stack](#)
- [Từ vựng Android](#)

Bài tập về nhà

Tải ứng dụng mẫu và khám phá tài nguyên

1. Tải một trong các ứng dụng mẫu vào Android Studio.
2. Mở một trong các tệp hoạt động Java trong ứng dụng. Tìm kiếm một lớp, kiểu hoặc quy trình mà bạn không quen thuộc và tra cứu trong tài liệu dành cho nhà phát triển Android.
3. Vào Stack Overflow và tìm kiếm các câu hỏi về cùng chủ đề.
4. Thay đổi biểu tượng trình khởi chạy. Sử dụng biểu tượng có sẵn trong phần tài sản hình ảnh của Android Studio.

Trả lời những câu hỏi này

Câu hỏi 1

Trong dự án Android Studio, bạn có thể sử dụng lệnh menu nào để mở danh sách các ứng dụng mẫu? Chọn một trong các lệnh sau:

- **Tệp > Mở**
- **Tệp > Mới > Nhập mẫu**

- **Tệp > Mới > Mô-đun nhập**
- **Tệp > Mới > Nhập dự án**

Câu hỏi 2

Mẫu Hoạt động cơ bản cung cấp những nút nào như một phần của UI? Chọn hai:

- Các nút điều hướng
- Nút tràn menu tùy chọn
- Nút hành động nổi
- Cái nút lõi có chữ "Nút"

Câu hỏi 3

Nguồn tài liệu nào là tài liệu chính thức dành cho nhà phát triển Android? Chọn một:

- stackoverflow.com
- officialandroid.blogspot.com
- nhà phát triển.android.com
- github.com

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Kết quả là một ứng dụng mới hoặc phiên bản Hello Toast, có biểu tượng trình khởi chạy mới xuất hiện trên màn hình Tìm kiếm ứng dụng của thiết bị Android.

Bài 2.1: Hoạt động và mục đích

Giới thiệu

MỘT Hoạt động đại diện cho một màn hình duy nhất trong ứng dụng của bạn mà người dùng có thể thực hiện một tác vụ duy nhất, tập trung như chụp ảnh, gửi email hoặc xem bản đồ. Một hoạt động thường được trình bày cho người dùng dưới dạng cửa sổ toàn màn hình.

Một ứng dụng thường bao gồm nhiều màn hình được liên kết lồng lèo với nhau. Mỗi màn hình là một hoạt động. Thông thường, một hoạt động trong ứng dụng được chỉ định là hoạt động "chính" (MainActivity.java), được trình bày cho người dùng khi ứng dụng được khởi chạy. Hoạt động chính sau đó có thể bắt đầu các hoạt động khác để thực hiện các hành động khác nhau.

Mỗi lần một hoạt động mới bắt đầu, hoạt động trước đó sẽ dừng lại, nhưng hệ thống vẫn giữ nguyên hoạt động trong một ngăn xếp ("ngăn xếp ngược"). Khi một hoạt động mới bắt đầu, hoạt động mới đó sẽ được đẩy vào ngăn xếp ngược và lấy sự tập trung của người dùng. Ngăn xếp ngược tuân theo logic ngăn xếp cơ bản "vào sau, ra trước". Khi người dùng hoàn tất hoạt động hiện tại và nhấn nút Quay lại, hoạt động đó sẽ được bật ra khỏi ngăn xếp và hủy, và hoạt động trước đó sẽ tiếp tục.

Một hoạt động được bắt đầu hoặc kích hoạt bằng một ý định. MỘT Ý định là một thông điệp không đồng bộ mà bạn có thể sử dụng trong hoạt động của mình để yêu cầu một hành động từ một hoạt động khác hoặc từ một số thành phần ứng dụng khác. Bạn sử dụng một ý định để bắt đầu một hoạt động từ một hoạt động khác và để truyền dữ liệu giữa các hoạt động.

MỘT Ý định có thể là rõ ràng hoặc ngầm hiểu.

- MỘT Ý định rõ ràng là một trong đó bạn biết mục tiêu của ý định đó. Nghĩa là bạn đã biết tên lớp đủ điều kiện của hoạt động cụ thể đó.
- MỘT Ý định ngầm là trường hợp bạn không có tên của thành phần mục tiêu, nhưng bạn có một hành động chung để thực hiện.

Trong phần thực hành này, bạn tạo ra các ý định rõ ràng. Bạn sẽ tìm hiểu cách sử dụng các ý định ngầm định trong phần thực hành sau.

Những điều bạn nên biết

Bạn sẽ có thể:

- Tạo và chạy ứng dụng trong Android Studio.
- Sử dụng trình chỉnh sửa bố cục để tạo bố cục trong Bố cục ràng buộc
- Chỉnh sửa mã XML bố cục.
- Thêm vào trên Click chức năng cho một Cái nút.

Những gì bạn sẽ học được

- Làm thế nào để tạo một cái mới Hoạt động trong Android Studio.
- Cách xác định hoạt động cha và hoạt động con cho điều hướng Lên.
- Làm thế nào để bắt đầu một Hoạt động với một cách rõ ràng Ý định.
- Làm thế nào để truyền dữ liệu giữa mỗi Hoạt động với một cách rõ ràng Ý định.

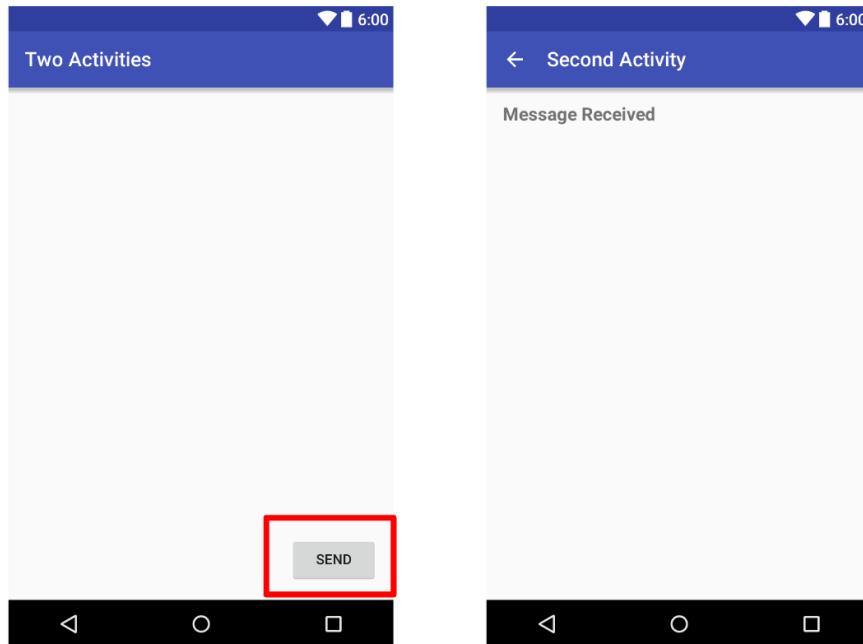
Bạn sẽ làm gì

- Tạo một ứng dụng Android mới với một chính Hoạt động và thứ hai Hoạt động.
- Truyền một số dữ liệu (một chuỗi) từ chính Hoạt động đến giây thứ hai sử dụng một Ý định, và hiển thị dữ liệu đó trong giây thứ hai Hoạt động.
- Gửi một bit dữ liệu khác thứ hai trở lại chính Hoạt động, cũng sử dụng một Ý định.

Tổng quan về ứng dụng

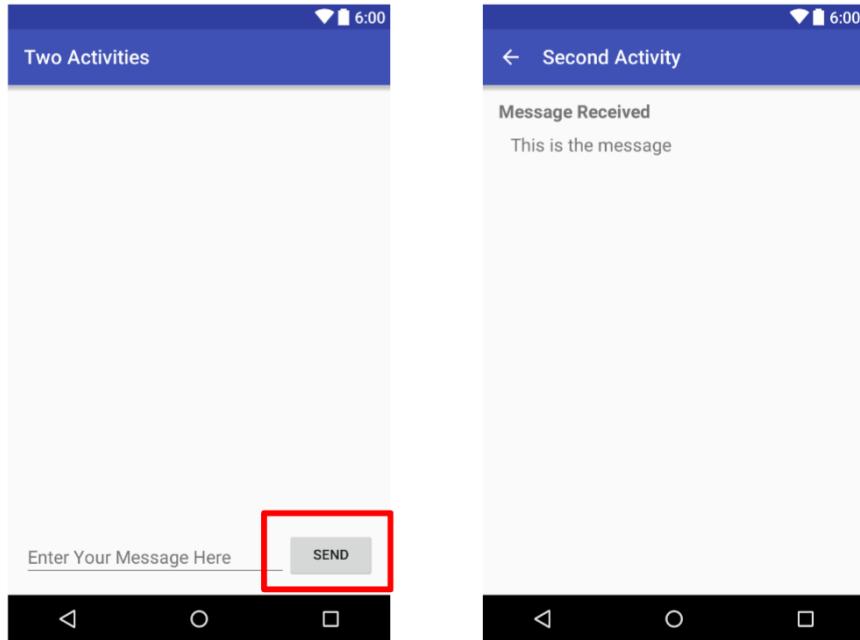
Trong chương này, bạn sẽ tạo và xây dựng một ứng dụng có tên là Hai hoạt động, không có gì ngạc nhiên khi ứng dụng này chứa hai Hoạt động triển khai. Bạn xây dựng ứng dụng theo ba giai đoạn.

Ở giai đoạn đầu tiên, bạn tạo một ứng dụng có hoạt động chính chứa một nút, **Gửi**. Khi người dùng nhấp vào nút này, hoạt động chính của bạn sẽ sử dụng ý định để bắt đầu hoạt động thứ hai.



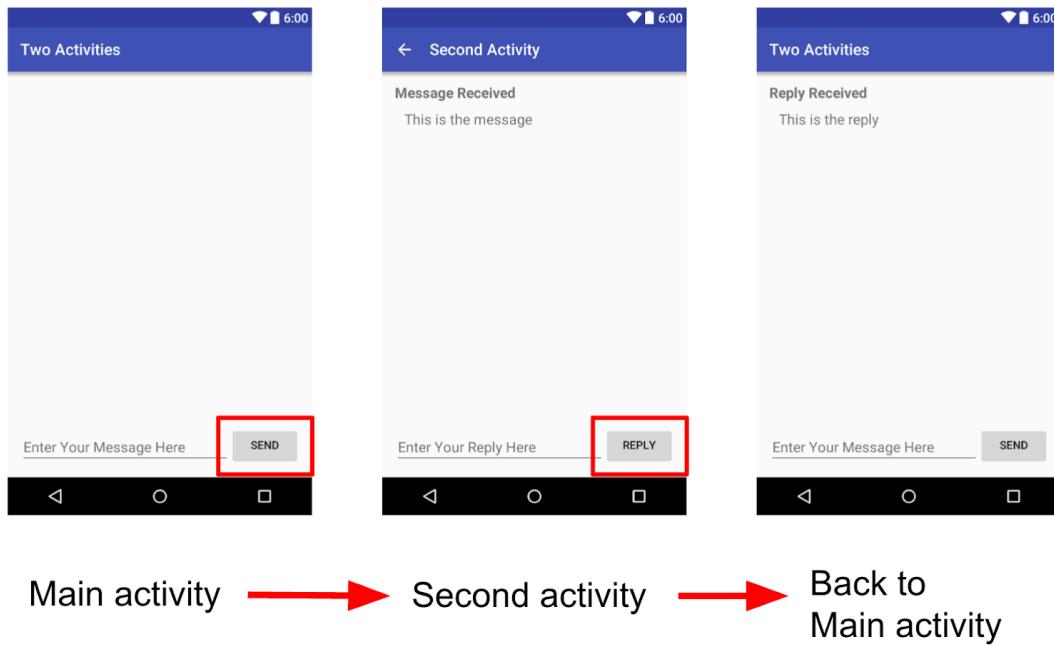
Main activity → Second activity

Ở giai đoạn thứ hai, bạn thêm một Sửa văn bản xem hoạt động chính. Người dùng nhập tin nhắn và nhấp vào **Gửi**. Hoạt động chính sử dụng một ý định để bắt đầu hoạt động thứ hai và gửi tin nhắn của người dùng đến hoạt động thứ hai. Hoạt động thứ hai hiển thị tin nhắn mà nó nhận được.



Main activity → Second activity

Ở giai đoạn cuối cùng của việc tạo ứng dụng Hai hoạt động, bạn thêm một **Sửa văn bản** và một **Hồi đáp** nút đến hoạt động thứ hai. Người dùng hiện có thể nhập tin nhắn trả lời và chạm vào **Hồi đáp** và phản hồi được hiển thị trên hoạt động chính. Tại thời điểm này, bạn sử dụng ý định để chuyển phản hồi từ hoạt động thứ hai trở lại hoạt động chính.



Nhiệm vụ 1: Tạo dự án TwoActivities

Trong nhiệm vụ này, bạn thiết lập dự án ban đầu với một dự án chính Hoạt động, xác định bố cục và xác định phương pháp khung chơtrênClicksự kiện nút.

1.1 Tạo dự án TwoActivities

- 1.Khởi động Android Studio và tạo một dự án Android Studio mới.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

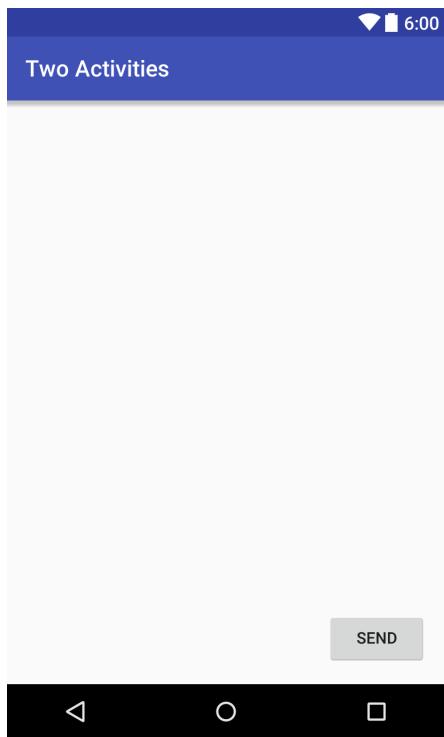
để biết thông tin cập nhật mới nhất.

Đặt tên cho ứng dụng của bạn **Hai hoạt động** và chọn giống nhau **Điện thoại và máy tính bảng** thiết lập mà bạn đã sử dụng trong các bài thực hành trước đó. Thư mục dự án được tự động đặt tên **Hai hoạt động**, và tên ứng dụng xuất hiện trên thanh ứng dụng sẽ là "Hai hoạt động".

- 2.Chọn **Hoạt động** cho **Hoạt động** mẫu. Nhấp vào **Kế tiếp**.
- 3.Chấp nhận mặc định **Hoạt động** tên (Hoạt động chính). Hãy chắc chắn rằng **Tạo tệp Bố cục** và **Khả năng tương thích ngược (AppCompat)** các tùy chọn được kiểm tra.
- 4.Nhấp chuột **Hoàn thành**.

1.2 Xác định bố cục cho Hoạt động chính

- 1.Mở **res > bố cục > activity_main.xml** trong **Dự án > Android** ngăn. Trình chỉnh sửa bố cục xuất hiện.
- 2.Nhấp vào **Thiết kế** tab nếu nó chưa được chọn và xóa Xem văn bản (cái có nội dung "Xin chào thế giới") trong **Cây thành phần** khung cửa sổ.
- 3.Với Autoconnect được bật (cài đặt mặc định), hãy kéo một Cái nút từ **Bảng màu** gần ở góc dưới bên phải của bố cục. Tự động kết nối tạo ra các ràng buộc cho Cái nút.
- 4.Trong **Thuộc tính** ngăn, thiết lập **NHẬN ĐẶNG ĐỀN** nút **nhấn chính**, các **chiều rộng** bố cục và **chiều cao** bố cục **ĐẾN nội dung bọc** và nhập **Gửi** cho trường Văn bản. Bố cục bây giờ sẽ trông như thế này:



5.Nhấp vào **Chữ tab** để chỉnh sửa mã XML. Thêm thuộc tính sau vào Cái nút:

```
    android:onClick="launchHoạt động thứ hai"
```

Giá trị thuộc tính được gạch chân màu đỏ vì khởi chạy Hoạt động thứ hai() phương pháp vẫn chưa được tạo. Bỏ qua lỗi này ngay bây giờ; bạn sửa nó trong tác vụ tiếp theo.

6.Trích xuất tài nguyên chuỗi, như đã mô tả trong bài thực hành trước, cho "Gửi" và sử dụng tên nút_chính cho tài nguyên.

Mã XML cho Cái nút sẽ trông giống như sau:

```
<Nút
```

```
    android:id="@+id/button_main"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:layout_marginRight="16dp" android:text="@string/
button_main" android:onClick="launchSecondActivity"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintRight_toRightOf="parent" />
```

1.3 Xác định hành động của Nút

Trong nhiệm vụ này bạn thực hiện `launchSecondActivity()` mphương pháp bạn đã đề cập đến trong bối cảnh cho `android:onClick` thuộc tính.

- 1.Nhấp vào "**khởi chạy Hoạt động thứ hai**" trong `activity_main.xml` XML.
- 2.Nhấn Alt+Enter (Tùy chọn+Enter trên máy Mac) và chọn **Tạo 'launchSecondActivity(View)' trong 'MainActivity'**.

Các Hoạt động chính file mở ra và Android Studio tạo ra một phương thức khung cho khởi chạy Hoạt động thứ hai() người xử lý.

- 3.Bên trong `khởi chạy Hoạt động thứ hai()`, thêm một Nhập ký câu lệnh có nội dung "Nút đã được nhấp!"

```
Log.d(LOG_TAG, "Đã nhấp vào nút!");
```

NHẬT KÝ_THÊSẽ hiển thị màu đỏ. Bạn thêm định nghĩa cho biến đó ở bước sau.

- 4.Ở phía trên cùng của `Hoạt động chính` lớp, thêm một hằng số cho `NHẬT KÝ_THÊ` biến:

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

Hàng số này sử dụng tên của chính lớp làm thẻ.

5.Chạy ứng dụng của bạn. Khi bạn nhấp vào **Gửi** bạn thấy thông báo "Nút đã nhấp!" trong **Logcat**. Nếu có quá nhiều đầu ra trên màn hình, hãy nhập **Hoạt động chính** vào hộp tìm kiếm và **Logcat** này sẽ chỉ hiển thị những dòng khớp với thẻ đó.

Mã cho **Hoạt động chính** trông như sau:

```
goi com.example.android.twoactivities;

nhap khau android.support.v7.app.AppCompatActivity;
nhap khau android.os.Bundle;
nhap khau android.util.Log;
nhap khau android.view.View;

lớp công khai MainActivity mở rộng AppCompatActivity {
    private static final String LOG_TAG =
        MainActivity.class.getSimpleName();

    @Ghi đè
    được bảo vệ void onCreate(Gói savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void launchSecondActivity(Xem chế độ xem) {
        Log.d(LOG_TAG, "Đã nhấp vào nút!");
    }
}
```

Nhiệm vụ 2: Tạo và khởi chạy Hoạt động thứ hai

Mỗi hoạt động mới bạn thêm vào dự án của mình đều có bố cục và tệp Java riêng, tách biệt với hoạt động chính. Chúng cũng có <hoạt động>các yếu tố trongAndroidManifest.xmlle. Cũng giống như hoạt động chính, các triển khai hoạt động mới mà bạn tạo trong Android Studio cũng mở rộng từHoạt động tương thích của ứng dụnglớp học.

Mỗi hoạt động trong ứng dụng của bạn chỉ được kết nối lỏng lẻo với các hoạt động khác. Tuy nhiên, bạn có thể định nghĩa một hoạt động là cha mẹ của một hoạt động khác trongAndroidManifest.xmlle. Mỗi quan hệ cha-con này cho phép Android thêm các gợi ý điều hướng như mũi tên hướng sang trái vào thanh tiêu đề cho mỗi hoạt động.

Một hoạt động giao tiếp với các hoạt động khác (trong cùng một ứng dụng và trên các ứng dụng khác nhau) bằng một mục đích. MộtÝ địnhcó thể làrõ rànghoặccngầm hiểu.

- MỘTÝ định rõ rànglà lớp mà bạn biết mục tiêu của ý định đó; nghĩa là bạn đã biết tên lớp đủ điều kiện của hoạt động cụ thể đó.
- MỘTÝ định ngầmlà loại lệnh mà bạn không có tên của thành phần mục tiêu nhưng có một hành động chung để thực hiện.

Trong nhiệm vụ này, bạn thêm một hoạt động thứ hai vào ứng dụng của chúng tôi, với bố cục riêng của nó. Bạn sửa đổi AndroidManifest.xmlle để xác định hoạt động chính là hoạt động cha của hoạt động thứ hai. Sau đó, bạn sửa đổi khởi chạy Hoạt động thứ hai()phương pháp trongHoạt động chínhđể bao gồm một ý định khởi chạy hoạt động thứ hai khi bạn nhấp vào nút.

2.1 Tạo Hoạt động thứ hai

- 1.Nhấp vàoỨng dụngthư mục cho dự án của bạn và chọnTệp > Mới > Hoạt động > Hoạt động trống.
- 2.Đặt tên mớiHoạt độngHoạt động thứ hai. Hãy chắc chắnTạo tệpbố cụcVàNgược lạiKhả năng tương thích (AppCompat)được kiểm tra. Tên bố cục được điền vào nhưhoạt động_thứ_hai. LÀMkhôngkiểm traHoạt động của Launcherlựa chọn.
- 3.Nhấp chuộtHàn thànhs. Android Studio bổ sung cả hai tính năng mớiHoạt độngcách trình bày (hoạt động_thứ_hai.xml)và một tệp Java mới (SecondActivity.java)cho dự án của bạn cho cái mớiHoạt động.Nó cũng cập nhậtAndroidManifest.xmlle để bao gồm cái mớiHoạt động.

2.2 Sửa đổi tệp AndroidManifest.xml

1.Mởbiểu hiện > **AndroidManifest.xml**.

2.Tìm <hoạt động>phần tử mà Android Studio đã tạo cho phần thứ haiHoạt động.

```
<hoạt động android:name=".SecondActivity"></hoạt động>
```

3.Thay thế toàn bộ <hoạt động>phần tử có nội dung sau:

```
<hoạt động android:name=".SecondActivity"
    android:label = "Hoạt động thứ hai"
    android:parentActivityName=".MainActivity">
    <meta-data
        android:ên="android.hỗ trợ.HOẠT ĐỘNG_PHỤ HUYNH"
        android:giá_trị=
            "com.example.android.twoactivities.MainActivity" />
</hoạt động>
```

Cácnhãnthuộc tính thêm tiêu đề củaHoạt độngvào thanh ứng dụng.

VớiTên hoạt động cha mèthuộc tính, bạn chỉ ra rằng hoạt động chính là hoạt động cha của hoạt động thứ hai. Mỗi quan hệ này được sử dụng để điều hướng Lên trong ứng dụng của bạn: thanh ứng dụng cho hoạt động thứ hai sẽ có mũi tên hướng sang trái để người dùng có thể điều hướng "lên trên" đến hoạt động chính.

Với <siêu dữ liệu>phần tử, bạn cung cấp thông tin tùy ý bổ sung về hoạt động dưới dạng cặp khóa-giá trị. Trong trường hợp này, các thuộc tính siêu dữ liệu thực hiện cùng một việc như android:parentActivityName thuộc tính—chúng xác định mối quan hệ giữa hai hoạt động để điều hướng lên trên. Các thuộc tính siêu dữ liệu này là bắt buộc đối với các phiên bản Android cũ hơn, vì android:parentActivityName thuộc tính này chỉ khả dụng cho API cấp 16 trở lên.

4.Trích xuất một tài nguyên chuỗi cho "Hoạt động thứ hai" trong mã ở trên và sử dụnghoạt động2_tênnhư tên tài nguyên.

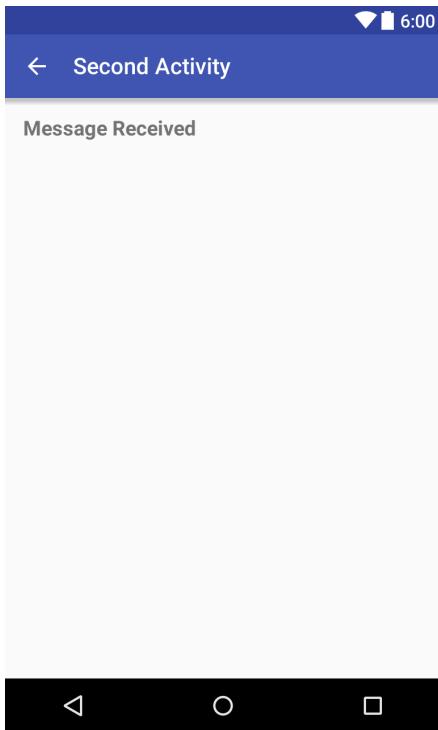
2.3 Xác định bố cục cho Hoạt động thứ hai

- 1.Mở **hoạt động_thứ hai.xml** và nhấp vào **Thiết kế** tab nếu nó chưa được chọn.
- 2.Kéo một **Xem văn bản** từ **Bảng màu** gần ở góc trên bên trái của bố cục và thêm ràng buộc ở phía trên và phía bên trái của bố cục. Đặt các thuộc tính của nó trong **Thuộc tính khung** như sau:

Thuộc tính	Giá trị
nhận dạng	tiêu đề văn bản
Lề trên cùng	16
Lề trái	8
chiều rộng bố cục	nội dung bọc
chiều cao bố cục	nội dung bọc
chữ	Tin nhắn đã nhận
văn bảnNgoại hình	AppCompat.Medium
Kiểu văn bản	B (in đậm)

Giá trị của **văn bảnNgoại hình** là một thuộc tính chủ đề Android đặc biệt xác định kiểu phông chữ cơ bản. Bạn sẽ tìm hiểu thêm về chủ đề trong bài học sau.

Bố cục bây giờ sẽ trông như thế này:



- 3.Nhấp vào **Chữ tab** để chỉnh sửa mã XML và trích xuất chuỗi "Tin nhắn đã nhận" vào một tài nguyên có tên tiêu đề văn bản.
- 4.Thêm vào `android:layout_marginLeft="8dp"` thuộc tính cho Xem văn bản để bổ sung cho `layout_margin` Bắt đầu thuộc tính dành cho các phiên bản Android cũ hơn.

Mã XML cho hoạt động thứ hai.xml phải như sau:

```
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:ứng dụng="http://schemas.android.com/apk/res-auto"  
    xmlns:công cụ="http://schemas.android.com/tools" android:bố  
    cục_chiều rộng="phù hợp với cha mẹ"  
    android:layout_height="match_parent" công  
    cụ:context="com.example.android.twoactivities.SecondActivity">  
  
<Xem văn bản  
    android:id="@+id/tiêu đề văn bản">
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginLeft="8dp"
    android:layout_marginTop="16dp"
    android:text="@string/text_header"
    android:textAppearance=
        "@style/TextAppearance.AppCompat.Medium"
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" /> </
    android.support.constraint.ConstraintLayout>
```

2.4 Thêm Intent vào Hoạt động chính

Trong nhiệm vụ này bạn thêm một cách rõ ràng Ý định đến chính Hoạt động. Cái này Ý định được sử dụng để kích hoạt thứ hai Hoạt động khi **Gửi** nút được nhấp vào.

1. Mở Hoạt động chính.

2. Tạo một cái mới Ý định trong khởi chạy Hoạt động thứ hai() phương pháp.

Trình xây dựng Intent lấy hai đối số cho một Mục đích: một ứng dụng Bối cảnh và thành phần cụ thể sẽ nhận được điều đó Ý định. Ở đây bạn nên sử dụng cái này như là Bối cảnh, Và Lớp Hoạt động thứ hai như lớp cụ thể:

```
Ý định intent = new Intent(this, SecondActivity.class);
```

3. Gọi chobắt đầu Hoạt động() phương pháp với cái mới Ý định như là một lập luận.

```
startActivity(ý định);
```

4.Chạy ứng dụng.

Khi bạn nhấp vào **Gửi** cái nút, Hoạt động chính gửi ý định và hệ thống Android ra mắt Hoạt động thứ hai, xuất hiện trên màn hình. Để quay lại Hoạt động chính, nhấp vào **Hướng lên** (mũi tên bên trái trên thanh ứng dụng) hoặc nút Quay lại ở cuối màn hình.

Nhiệm vụ 3: Gửi dữ liệu từ Activity chính đến Activity thứ hai

Trong nhiệm vụ cuối cùng, bạn đã thêm một ý định rõ ràng vào Hoạt động chính để ra mắt Hoạt động thứ hai. Bạn cũng có thể sử dụng ý định để gửi dữ liệu từ hoạt động này sang hoạt động khác khi khởi chạy nó.

Đối tượng ý định của bạn có thể truyền dữ liệu đến hoạt động mục tiêu theo hai cách: trong trường dữ liệu hoặc trong ý định *thêm vào*. Dữ liệu ý định là một URI chỉ ra mục đích cụ thể dữ liệu cần được xử lý. Nếu thông tin bạn muốn chuyển đến một hoạt động thông qua một ý định không phải là URI hoặc bạn có nhiều hơn một thông tin bạn muốn gửi, bạn có thể đưa thông tin bổ sung đó vào *thêm vào* thay vì.

Ý định *thêm vào* là cặp khóa/giá trị trong một **Bộ**. MỘT Bộ là một tập hợp dữ liệu, được lưu trữ dưới dạng cặp khóa/giá trị. Để truyền thông tin từ hoạt động này sang hoạt động khác, bạn đặt khóa và giá trị vào mục đích bổ sung Bộ từ hoạt động gửi, sau đó lấy lại chúng trong hoạt động nhận.

Trong nhiệm vụ này, bạn sửa đổi ý định rõ ràng trong Hoạt động chính để bao gồm dữ liệu bổ sung (trong trường hợp này là chuỗi do người dùng nhập) trong mục đích bổ sung Bộ. Sau đó bạn sửa đổi Hoạt động thứ hai để lấy lại dữ liệu đó từ mục đích bổ sung Bộ và hiển thị trên màn hình.

3.1 Thêm EditText vào bố cục MainActivity

1.Mở hoạt động_main.xml.

2.Kéo một **Văn bản thuần túy** (Chỉnh sửa văn bản) phần tử từ **Bảng màu** xuống dưới cùng của bố cục, và thêm các ràng buộc vào phía bên trái của bố cục, phía dưới của bố cục và phía bên trái của **Gửi** Cái nút. Đặt các thuộc tính của nó trong **Thuộc tính khung** như sau:

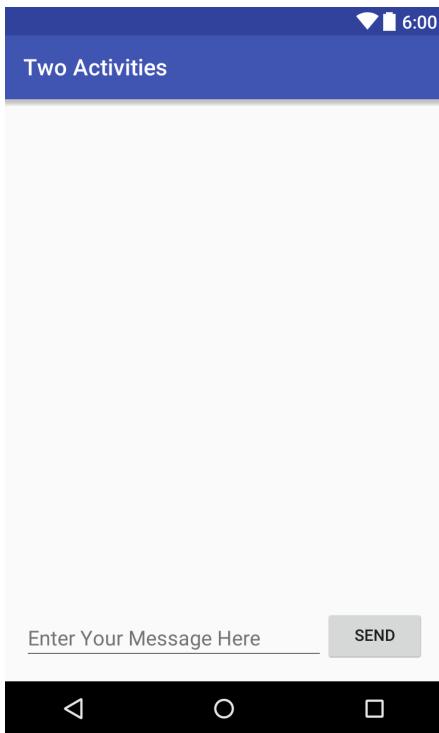
Thuộc tính	Giá trị
nhân dạng	editText_main
Lề phải	8

Tác phẩm này được cấp phép theo một [Giấy phép Creative Commons Ghi công 4.0 Quốc tế](#). PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

Lề trái	8
Lề dưới	16
chiều rộng bối cảnh	ràng buộc_phù hợp
chiều cao bối cảnh	nội dung bọc
Kiểu đầu vào	văn bảnLongMessage
gợi ý	Nhập tin nhắn của bạn ở đây
chữ	(Xóa bất kỳ văn bản nào trong trường này)

Bối cảnh mới trong hoạt động_main.xml trông giống thế này:



3.Nhấp vào **Chữ** tab để chỉnh sửa mã XML và trích xuất chuỗi "Nhập tin nhắn của bạn tại đây" vào một tài nguyên có tên **chỉnh sửa Văn bản_chính**.

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

XML mã cho bố cục sẽ trông giống như sau.

```
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ứng dụng="http://schemas.android.com/apk/resauto"
    xmlns:công cụ="http://schemas.android.com/tools" android:bố
    cục_chiều rộng="phù hợp với cha mẹ"
    android:layout_height="match_parent" công
    cụ:context="com.example.android.twoactivities.MainActivity">

    <Nút
        android:id="@+id/button_main"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginBottom="16dp"
        android:layout_marginRight="16dp" android:text="@string/
        button_main" android:onClick="launchSecondActivity"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintRight_toRightOf="parent" />

    <Sửa đổi văn bản
        android:id="@+id/editText_main"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginBottom="16dp"
        android:layout_marginEnd="8dp"
        android:layout_marginStart="8dp"
        android:ems="10"
        android:hint="@string/editText_main"
        android:inputType="textLongMessage"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toStartOf="@+id/button2"
        app:layout_constraintStart_toStartOf="parent" /> </
    android.support.constraint.ConstraintLayout>
```

3.2 Thêm một chuỗi vào phần bổ sung Intent

Các **Ý định** là cặp khóa/giá trị trong một **Bộ**. Bộ là một tập hợp dữ liệu, được lưu trữ dưới dạng cặp khóa/giá trị. Để truyền thông tin từ một **Hoạt động** với một cái khác, bạn đặt khóa và giá trị vào **Ý định** từ việc gửi **Hoạt động**, và sau đó lấy chúng ra ngoài một lần nữa trong quá trình tiếp nhận **Hoạt động**.

1. Mở **Hoạt động chính**.

2. Thêm một công cộng **hằng số** ở đầu lớp để xác định khóa cho **Ý định**:

```
công khai tĩnh cuối cùng Chuỗi EXTRA_MESSAGE =  
    "com.example.android.twoactivities.extra.MESSAGE";
```

3. Thêm một biến riêng tư ở đầu lớp để giữ **Chỉnh sửa văn bản**:

```
riêng tư EditText mMessageEditText;
```

4. Trong khi tạo() phương pháp, sử dụng **tìmViewById()** để có được một tham chiếu đến **Sửa văn bản** và gán nó vào biến riêng tư đó:

```
mMessageEditText = findViewById(R.id.editText_main);
```

5. Trong khởicloud chạy **Hoạt động thứ hai**() phương pháp, ngay dưới cái mới **Ý định**, lấy văn bản từ **Sửa văn bản** dưới dạng một chuỗi:

```
Chuỗi tin nhắn = mMessageEditText.getText().toString();
```

6.Thêm chuỗi đó vào Ý định như một phần bổ sung với TIN_NHẮN_THÊM hằng số làm khóa và chuỗi làm giá trị:

```
ý định.putExtra(THÔNG_TIN_THÊM, tin nhắn);
```

Các khi tạo() phương pháp trong Hoạt động chính bấy giờ sẽ trông như sau:

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

Các khi chạy Hoạt động thứ hai() phương pháp trong Hoạt động chính bấy giờ sẽ trông như sau:

```
public void launchSecondActivity(Xem chế độ xem) {  
    Log.d(LOG_TAG, "Đã nhấp vào nút!");  
    Ý định intent = new Intent(this, SecondActivity.class); Chuỗi message  
    = mMessageEditText.getText().toString();  
    intent.putExtra(EXTRA_MESSAGE, message);  
    startActivity(Ý định);  
}
```

3.3 Thêm TextView vào SecondActivity cho tin nhắn

1.Mở **hoạt động_thứ hai.xml**.

2.Kéo một cái khác **Xem văn bản** để bố trí bên dưới **text_header** Xem văn bản, và thêm các ràng buộc vào phía bên trái của bố cục và vào phía dưới tiêu đề văn bản.

3.Đặt cái mới **Xem văn bản** các thuộc tính trong **Thuộc tính khung** như sau:

Thuộc tính	Giá trị
nhận dạng	tin nhắn văn bản
Lề trên cùng	8
Lề trái	8
chiều rộng bố cục	nội dung bọc
chiều cao bố cục	nội dung bọc
chữ	(Xóa bất kỳ văn bản nào trong trường này)
văn bản Ngoại hình	AppCompat.Medium

Bố cục mới trông giống như trong nhiệm vụ trước đó, vì **Xem văn bản** (chưa) chứa bất kỳ văn bản nào và do đó không hiển thị trên màn hình.

XML mã cho **hoạt động_thứ hai.xml** Bố cục phải trông giống như sau:

```
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ứng dụng="http://schemas.android.com/apk/resauto"
    xmlns:công cụ="http://schemas.android.com/tools" android:bố
    cục_chiều rộng="phù hợp với cha mẹ"
    android:layout_height="match_parent" công
    cụ:context="com.example.android.twoactivities.SecondActivity">

    <Xem văn bản
        android:id="@+id/text_header"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
```

```
    android:layout_marginTop="16dp"
    android:text="@string/text_header"
    android:textAppearance=
        "@style/TextAppearance.AppCompat.Medium"
    android:textStyle="bold" Ứng
    dụng:layout_constraintStart_toStartOf="cha mẹ" Ứng
    dụng:layout_constraintTop_toTopOf="cha mẹ" />

<Xem văn bản
    android:id="@+id/text_message" android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/text_header" </
    android.support.constraint.ConstraintLayout>
/>
```

3.4 Sửa đổi SecondActivity để có thêm nội dung và hiển thị thông báo

1. Mở **Hoạt động thứ hai** để thêm mã vào khi tạo() phương pháp.
2. Nhận được Ý định đã kích hoạt điều này **Hoạt động**:

```
Ý định intent = getIntent();
```

3. Lấy chuỗi chứa tin nhắn từ Ý định phần bổ sung sử dụng **Hoạt động chính.EXTRA_MESSAGE** biến tĩnh làm khóa:

```
Chuỗi message = intent.getStringExtra(MainActivity.EXTRA_MESSAGE);
```

4.Sử dụng `tìmViewByID()` để có được một tham chiếu đến Xem văn bản cho thông điệp từ bố cục:

```
TextView textView = findViewById(R.id.text_message);
```

5.Đặt văn bản của Xem văn bản đến chuỗi từ Ý định thêm:

```
textView.setText(tin nhắn);
```

6.Chạy ứng dụng. Khi bạn nhập tin nhắn vào Hoạt động chính và nhấp vào **Gửi**, Hoạt động thứ hai khởi chạy và hiển thị thông báo.

Các Hoạt động thứ hai khi tạo() phương pháp này sẽ như sau:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_second); Ý định intent =  
    getIntent();  
    Chuỗi message = intent.getStringExtra(MainActivity.EXTRA_MESSAGE); TextView textView =  
    findViewById(R.id.text_message);  
    textView.setText(tin nhắn);  
}
```

Nhiệm vụ 4: Trả dữ liệu trả lại Hoạt động chính

Bây giờ bạn đã có một ứng dụng khởi chạy một hoạt động mới và gửi dữ liệu đến hoạt động đó, bước cuối cùng là trả dữ liệu từ hoạt động thứ hai trả lại hoạt động chính. Bạn cũng sử dụng một ý định và ý định *thêm vào* cho nhiệm vụ này.

4.1 Thêm EditText và Button vào layout SecondActivity

1. Mở `chuỗi.xml` và thêm tài nguyên chuỗi cho Cái nút văn bản và gợi ý cho Sửa văn bản mà bạn sẽ thêm vào Hoạt động thứ hai:

```
<string name="button_second">Trả lời</string>
<string name="editText_second">Nhập câu trả lời của bạn ở đây</string>
```

2. Mở `hoạt động_main.xml` và `hoạt động_thứ hai.xml`.

3. Sao chép cái Sửa văn bản và Cái nút từ `hoạt động_main.xml` và Dán họ vào `hoạt động_thứ hai.xml` cách trình bày.

4. TRONG `hoạt động_thứ hai.xml`, sửa đổi các giá trị thuộc tính cho Cái nút như sau:

Giá trị thuộc tính cũ	Giá trị thuộc tính mới
<code>android:id="@+id/button_main"</code>	<code>android:id="@+id/button_second"</code>
<code>android:onClick="khởi chạy Hoạt động thứ hai"</code>	<code>android:onClick="trả lời"</code>
<code>android:văn bản="@string/button_main"</code>	<code>android:văn bản="@string/button_second"</code>

5. TRONG `hoạt động_thứ hai.xml`, sửa đổi các giá trị thuộc tính cho Sửa văn bản như sau:

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

Giá trị thuộc tính cũ	Giá trị thuộc tính mới
android:id="@+id/editText_main"	android:id="@+id/editText_giây"
ứng dụng:layout_constraintEnd_toStartOf="@+ id/nút"	ứng dụng:layout_constraintEnd_toStartOf="@+ id/button_second"
android:gợi ý="@string/editText_main"	android:gợi ý="@string/editText_giây"

6.Trong trình soạn thảo bố cục XML, hãy nhấp vào **trả lời**, nhấn Alt+Enter (Tùy chọn+Trả về trên máy Mac) và chọn **Tạo 'returnReply(View)' trong 'SecondActivity'**.

Android Studio tạo ra một phương pháp khung xương chotrả về Trả lời(handler. Bạn triển khai phương pháp này trong tác vụ tiếp theo.

Bố cục mới cho hoạt động_thứ hai.xml trông giống thế này:



Mã XML cho hoạt động thứ hai.xml Tệp bố cục như sau:

```
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ứng dụng="http://schemas.android.com/apk/res-auto"
    xmlns:công cụ="http://schemas.android.com/tools" android:bo
    cục_chiều rộng="phù hợp với cha mẹ"
    android:layout_height="match_parent" công
    cụ:context="com.example.android.twoactivities.SecondActivity">

    <Xem văn bản
```

```
    android:id="@+id/text_header" android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:layout_marginStart="8dp"
    android:layout_marginLeft="8dp" android:layout_marginTop="16dp"
    android:text="@string/text_header" android:textAppearance="@style/
    TextAppearance.AppCompat.Medium" android:textStyle="bold"
```

```
    ứng dụng:layout_constraintStart_toStartOf="cha mẹ"
    ứng dụng:layout_constraintTop_toTopOf="cha mẹ" />
```

<Xem văn bản

```
    android:id="@+id/text_message" android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp" android:layout_marginLeft="8dp"
    android:layout_marginTop="8dp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/text_header" />
```

<Nút

```
    android:id="@+id/button_second"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:layout_marginRight="16dp" android:text="@string/
    button_second" android:onClick="returnReply"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintRight_toRightOf="parent" />
```

<Sửa đổi văn bản

```
    android:id="@+id/editText_second"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:ems="10"
    android:hint="@string/editText_second"
    android:inputType="textLongMessage"
    app:layout_constraintBottom_toBottomOf="parent"
```

```
    ứng dụng:layout_constraintEnd_toStartOf="@+id/button_second"  
    ứng dụng:layout_constraintStart_toStartOf="parent" /> </  
    android.support.constraint.ConstraintLayout>
```

4.2 Tạo Intent phản hồi trong Activity thứ hai

Dữ liệu phản hồi từ thứ hai Hoạt động quay lại với chính Hoạt động được gửi trong một Ý định thêm. Bạn xây dựng sự trả lại này Ý định và đưa dữ liệu vào đó theo cách tương tự như bạn làm khi gửi Ý định.

1. Mở **Hoạt động thứ hai**.

2. Ở đầu lớp, thêm một hằng số công khai để xác định khóa cho Ý định thêm:

```
public static final String EXTRA_REPLY =  
    "com.example.android.twoactivities.extra.REPLY";
```

3. Thêm một biến riêng tư ở đầu lớp để giữ Chỉnh sửa văn bản.

```
riêng tư EditText mReply;
```

4. Trong khi tạo() phương pháp, trước khi Ý định m, sử dụng `findViewById()` để có được một tham chiếu đến Sửa văn bản và gán nó vào biến riêng tư đó:

```
mReply = findViewById(R.id.editText_second);
```

5.Trong trả về Trả lời() phương pháp, lấy văn bản của Sửa văn bản dưới dạng một chuỗi:

```
Trả lời chuỗi = mReply.getText().toString();
```

6.Trong trả về Trả lời() phương pháp, tạo một ý định mới cho phản hồi—đừng tái sử dụng Ý định đối tượng mà bạn nhận được từ yêu cầu ban đầu.

```
Ý định replyIntent = new Intent();
```

7.Thêm vào hồi đáp chuỗi từ Sửa văn bản với ý định mới như một Ý định thêm. Bởi vì thêm vào là cặp khóa/giá trị, ở đây khóa là TRẢ LỜI THÊM, và giá trị là hồi đáp:

```
replyIntent.putExtra(TRẢ LỜI_EXTRA, trả lời);
```

8.Đặt kết quả thành KẾT QUẢ_OK để chỉ ra rằng phản hồi đã thành công.[Hoạt động](#) lớp xác định mã kết quả, bao gồm KẾT QUẢ_OK và KẾT QUẢ_ĐÃ_HỦY.

```
đặt Kết quả(KẾT QUẢ_OK, Ý định trả lời);
```

9.Gọi hoàn thành() để đóng Hoạt động và quay trở lại Hoạt động chính.

```
hoàn thành();
```

Mã cho Hoạt động thứ hai bấy giờ sẽ như sau:

```
lớp công khai SecondActivity mở rộng AppCompatActivity {
    public static final String EXTRA_REPLY =
        "com.example.android.twoactivities.extra.REPLY";
    riêng tư EditText mReply;

    @Ghi đè
    được bảo vệ void onCreate(Gói savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second); mReply =
        findViewById(R.id.editText_second); Ý định intent =
        getIntent();
        Chuỗi message = intent.getStringExtra(MainActivity.EXTRA_MESSAGE); TextView
        textView = findViewById(R.id.text_message);
        textView.setText(tin nhắn);
    }

    public void returnReply(Xem chế độ xem) {
        Chuỗi trả lời = mReply.getText().toString(); Ý định
        replyIntent = new Intent();
        replyIntent.putExtra(EXTRA_REPLY, trả lời); đặt Kết
        quả(RESULT_OK, replyIntent); kết thúc();
    }
}
```

4.3 Thêm các phần tử TextView để hiển thị câu trả lời

Hoạt động chính cần một cách để hiển thị câu trả lời đó. Hoạt động thứ hai gửi. Trong nhiệm vụ này bạn thêm Xem văn bản các yếu tố để hoạt động_main.xml bố trí để hiển thị câu trả lời trong Hoạt động chính.

Để thực hiện nhiệm vụ này dễ dàng hơn, bạn sao chép Xem văn bản các yếu tố bạn đã sử dụng trong Hoạt động thứ hai.

1. Mở **chuỗi.xml** và thêm một chuỗi tài nguyên cho tiêu đề trả lời:

```
<string name="text_header_reply">Đã nhận được phản hồi</string>
```

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

2.Mở **hoạt động_main.xml**và **hoạt động_thứ hai.xml**.

3.Sao chép **hai**Xem văn bản các yếu tố từ **hoạt động_thứ hai.xml**tập tin bố trí và dán chúng vào **hoạt động_main.xml**bố trí phía trênCái nút.

4.TRONG **hoạt động_main.xml**,sửa đổi các giá trị thuộc tính cho lần đầu tiênXem văn bản như sau:

Giá trị thuộc tính cũ	Giá trị thuộc tính mới
android:id="@+id/tiêu đề văn bản"	android:id="@+id/text_header_reply"
android:văn bản=@string/text_header"	android:văn bản=@string/text_header_trả lời"

5.TRONG **hoạt động_main.xml**,sửa đổi các giá trị thuộc tính cho **thứ hai**Xem văn bản như sau:

Giá trị thuộc tính cũ	Giá trị thuộc tính mới
android:id="@+id/tin_nhắn_văn_bản"	android:id="@+id/tin_nhắn_văn_bản_trả_lời"
ứng dụng:layout_constraintTop_toBottomOf="@+id/text_header"	ứng dụng:layout_constraintTop_toBottomOf="@+id/text_header_reply"

6.Thêm vào **android:khả năng hiển thị** thuộc tính cho mỗiXem văn bản để làm cho chúng ban đầu vô hình.(Việc hiển thị chúng trên màn hình nhưng không có nội dung nào có thể gây nhầm lẫn cho người dùng.)

android:visibility="vô hình"

Bạn sẽ làm những điều nàyXem văn bản các phần tử có thể nhìn thấy sau khi dữ liệu phản hồi được truyền lại từ phần thứ hai Hoạt động.

Cách hoạt động_main.xml bối cục trông giống như trong nhiệm vụ trước đó—mặc dù bạn đã thêm hai mục mới Xem văn bản các thành phần vào bối cục. Vì bạn đặt các thành phần này ở chế độ vô hình nên chúng không xuất hiện trên màn hình.

Sau đây là mã XML cho activity_main.xml file:

```
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:ứng dụng="http://schemas.android.com/apk/res-auto"
    xmlns:công cụ="http://schemas.android.com/tools" android:bối
    cục_chiều rộng="phù hợp với cha mẹ"
    android:layout_height="match_parent" công
    cụ:context="com.example.android.twoactivities.MainActivity">

    <Xem văn bản
        android:id="@+id/text_header_reply" android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:layout_marginStart="8dp"
        android:layout_marginLeft="8dp" android:layout_marginTop="16dp"
        android:text="@string/text_header_reply" android:textAppearance="@style/
        TextAppearance.AppCompat.Medium" android:textStyle="bold"

        android:visibility="invisible" ứng
        dụng:layout_constraintStart_toStartOf="cha mẹ" ứng
        dụng:layout_constraintTop_toTopOf="cha mẹ" />

    <Xem văn bản
        android:id="@+id/text_message_reply" android:layout_width="wrap_content"
        android:layout_height="wrap_content" android:layout_marginStart="8dp"
        android:layout_marginLeft="8dp" android:layout_marginTop="8dp"
        android:visibility="invisible" app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/text_header_reply" />

    <Nút
        android:id="@+id/button2"
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:layout_marginRight="16dp" android:text="@string/
button_main" android:onClick="launchSecondActivity"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintRight_toRightOf="parent" />

<Sửa đổi văn bản
    android:id="@+id/editText_main"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginBottom="16dp"
    android:layout_marginEnd="8dp"
    android:layout_marginStart="8dp"
    android:ems="10"
    android:hint="@string/editText_main"
    android:inputType="textLongMessage"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/button2"
    app:layout_constraintStart_toStartOf="parent" /> </
    android.support.constraint.ConstraintLayout>
```

4.4 Nhận phản hồi từ Intent extra và hiển thị nó

Khi bạn sử dụng một cách rõ ràng để bắt đầu một cái khác Hoạt động, bạn có thể không mong đợi lấy lại bất kỳ dữ liệu nào — bạn chỉ đang kích hoạt dữ liệu đó Hoạt động. Trong trường hợp đó, bạn sử dụng `bắt đầu Hoạt động()` để bắt đầu cái mới Hoạt động, như bạn đã làm trước đó trong bài thực hành này. Nếu bạn muốn lấy lại dữ liệu từ Hoạt động, tuy nhiên, bạn cần phải bắt đầu nó bằng `bắt đầu Hoạt động Từ Kết quả()`.

Trong nhiệm vụ này, bạn sửa đổi ứng dụng để bắt đầu Hoạt động thứ hai mong đợi một kết quả, để trích xuất dữ liệu trả về đó từ Ý định, và để hiển thị dữ liệu đó trong Xem văn bản các thành phần bạn đã tạo ở nhiệm vụ trước.

1. Mở Hoạt động chính.

2. Thêm một hằng số công khai vào đầu lớp để xác định khóa cho loại phản hồi cụ thể mà bạn quan tâm:

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

```
công khai tĩnh cuối cùng int TEXT_REQUEST = 1;
```

3.Thêm hai biến riêng tư để giữ tiêu đề trả lời và trả lờiXem văn bản các yếu tố:

```
TextView riêng tư mReplyHeadTextView;  
TextView riêng tư mReplyTextView;
```

4.Trong khi tạo()phương pháp, sử dụng `findViewById()`để có được các tham chiếu từ bố cục đến tiêu đề trả lời và trả lờiXem văn bản các phần tử. Gán các thể hiện chế độ xem đó cho các biến riêng tư:

```
mReplyHeadTextView = findViewById(R.id.text_header_reply);  
mReplyTextView = findViewById(R.id.text_message_reply);
```

Đầy đủ khi tạo()phương pháp bây giờ sẽ trông như thế này:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main); mMessageEditText =  
    findViewById(R.id.editText_main); mReplyHeadTextView =  
    findViewById(R.id.text_header_reply); mReplyTextView =  
    findViewById(R.id.text_message_reply);  
}
```

5.Trong khi chạy Hoạt động thứ hai()phương pháp, thay đổi cuộc gọi đến bắt đầu Hoạt động()để được bắt đầu Hoạt động Cho Kết quả(), và bao gồm YÊU CẦU VĂN BẢN chìa khóa như một đối số:

```
startActivityForResult(ý định, YÊU CẦU VĂN BẢN);
```

6.Ghi đè trên `onActivityResult()` phương thức gọi lại với chữ ký này:

```
@Ghi đè  
công khai void onActivityResult(int requestCode,  
                                int resultCode, Dữ liệu ý định) {  
}
```

Ba đối số để trên `onActivityResult()` chứa tất cả thông tin bạn cần để xử lý dữ liệu trả về: Mã yêu cầu bạn thiết lập khi bạn khởi chạy Hoạt động với bắt đầu Hoạt động Cho Kết quả(), cái Mã kết quả đặt trong ra mắt Hoạt động (thường là một trong KẾT QUẢ_OK hoặc KẾT QUẢ_ĐÃ HỦY), và Dữ liệu ý định chứa dữ liệu được trả về từ lần khởi chạy Hoạt động.

7.Bên trong `onActivityResult()`, gọi `super.onActivityResult()`:

```
super.onActivityResult(mã yêu cầu, mã kết quả, dữ liệu);
```

8.Thêm mã để kiểm tra YÊU CẦU VĂN BẢN để đảm bảo bạn xử lý đúng Ý định kết quả, trong trường hợp có nhiều. Cũng kiểm tra KẾT QUẢ_OK, để đảm bảo yêu cầu thành công:

```
nhếu (requestCode == TEXT_REQUEST) {  
    nhếu (resultCode == RESULT_OK) {}  
}
```

Các Hoạt động lớp định nghĩa mã kết quả. Mã có thể là KẾT QUẢ_OK (yêu cầu đã thành công), KẾT QUẢ_ĐÃ_HỦY (người dùng đã hủy bỏ thao tác), hoặc KẾT QUẢ_NGUỒI DÙNG ĐẦU TIÊN (để xác định mã kết quả của riêng bạn).

9.Bên trong khối if bên trong, lấy Ý định thêm từ phản hồi Ý định (dữ liệu).Ở đây chìa khóa cho phần bổ sung là TRẢ LỜI THÊM hằng số từ Hoạt động thứ hai:

```
Trả lời chuỗi = data.getStringExtra(SecondActivity.EXTRA_REPLY);
```

10.Đặt khả năng hiển thị của tiêu đề trả lời thành đúng:

```
mReplyHeadTextView.setVisibility(View.VISIBLE);
```

11.Đặt câu trả lời Xem văn bản văn bản để nhồi đáp,và đặt khả năng hiển thị của nó thành đúng:

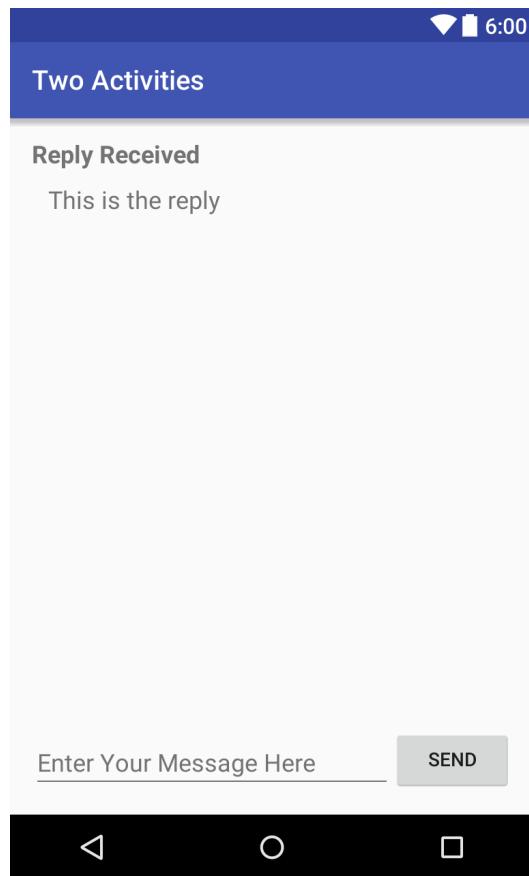
```
mReplyTextView.setText(trả lời);
mReplyTextView.setVisibility(View.VISIBLE);
```

Đầy đủ trên `onActivityResult()` phương pháp bây giờ sẽ trông như thế này:

```
@Ghi đè
công khai void onActivityResult(int requestCode,
                               int resultCode, Dữ liệu ý định) {
    super.onActivityResult(requestCode, resultCode, data); nếu
    (requestCode == TEXT_REQUEST) {
        nếu (resultCode == RESULT_OK) {
            Chuỗi trả lời =
                data.getStringExtra(SecondActivity.EXTRA_REPLY);
            mReplyHeadTextView.setVisibility(View.VISIBLE);
            mReplyTextView.setText(trả lời);
            mReplyTextView.setVisibility(View.VISIBLE);
        }
    }
}
```

12.Chạy ứng dụng.

Bây giờ, khi bạn gửi tin nhắn đến người thứ hai Hoạt động và nhận được câu trả lời, chính Hoạt động cập nhật để hiển thị câu trả lời.



Mã giải pháp

Dự án Android Studio:[HaiHoat Đông](#)

Thử thách mã hóa

Ghi chú:Mỗi thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách:Tạo một ứng dụng với baCái nút các yếu tố được dán nhãn**Văn bản Một,Văn bản Hai, VàVăn bản Ba**. Khi bất kỳ điều nào trong số nàyCái nút các yếu tố được nhấp, khởi chạy một thứ haiHoạt động.Giây phút đóHoạt động nên chứa mộtCuộnXemhiển thị một trong ba đoạn văn bản (bạn có thể bao gồm đoạn văn bạn chọn). Sử dụngÝ địnhđể khởi chạy thứ haiHoạt độngvới phần bổ sung để chỉ ra đoạn văn nào trong ba đoạn văn sẽ được hiển thị.

Bản tóm tắt

Tổng quan:

- MỘTHoạt độnglà một thành phần ứng dụng cung cấp một màn hình duy nhất tập trung vào một tác vụ của người dùng.
- MỗiHoạt độngcó tệp bố cục giao diện người dùng riêng.
- Bạn có thể chỉ địnhHoạt độngtriển khai mối quan hệ cha/con để cho phép điều hướng Lên trong ứng dụng của bạn.
- MỘTXemcó thể được làm cho có thể nhìn thấy hoặc vô hình vớiandroid:khả năng hiển thị thuộc tính.

Để thực hiện một Hoạt động:

- Chọn **Tệp > Mới > Hoạt động** để bắt đầu từ một mẫu và tự động thực hiện các bước sau.
- Nếu không bắt đầu từ một mẫu, hãy tạo một Hoạt động Lớp Java, triển khai giao diện người dùng cơ bản cho Hoạt động trong một tệp bố cục XML được liên kết và khai báo tệp mới Hoạt động trong `AndroidManifest.xml`.

Mục đích:

- Một **Ý định** cho phép bạn yêu cầu một hành động từ một thành phần khác trong ứng dụng của bạn, ví dụ, để bắt đầu một Hoạt động từ một người khác. Một **Ý định** có thể rõ ràng hoặc ngầm hiểu.
- Với một cách rõ ràng **Ý định** bạn chỉ định thành phần mục tiêu cụ thể để nhận dữ liệu.
- Với một hàm **Ý định** bạn chỉ định chức năng bạn muốn nhưng không chỉ định thành phần mục tiêu.
- Một **Ý định** có thể bao gồm dữ liệu để thực hiện hành động (như URI) hoặc thông tin bổ sung như **Ý định thêm vào**.
- **Ý định thêm vào** là cặp khóa/giá trị trong một **Bộ** được gửi kèm theo **Ý định**.

Khái niệm liên quan

Tài liệu khái niệm liên quan có trong [2.1: Hoạt động và mục đích](#).

Tìm hiểu thêm

Tài liệu Android Studio:

- [Làm quen với Android Studio](#)

Tài liệu dành cho nhà phát triển Android:

- [Cơ sở ứng dụng](#)
- [Các hoạt động](#)
- [Ý định và Bộ lọc ý định](#)

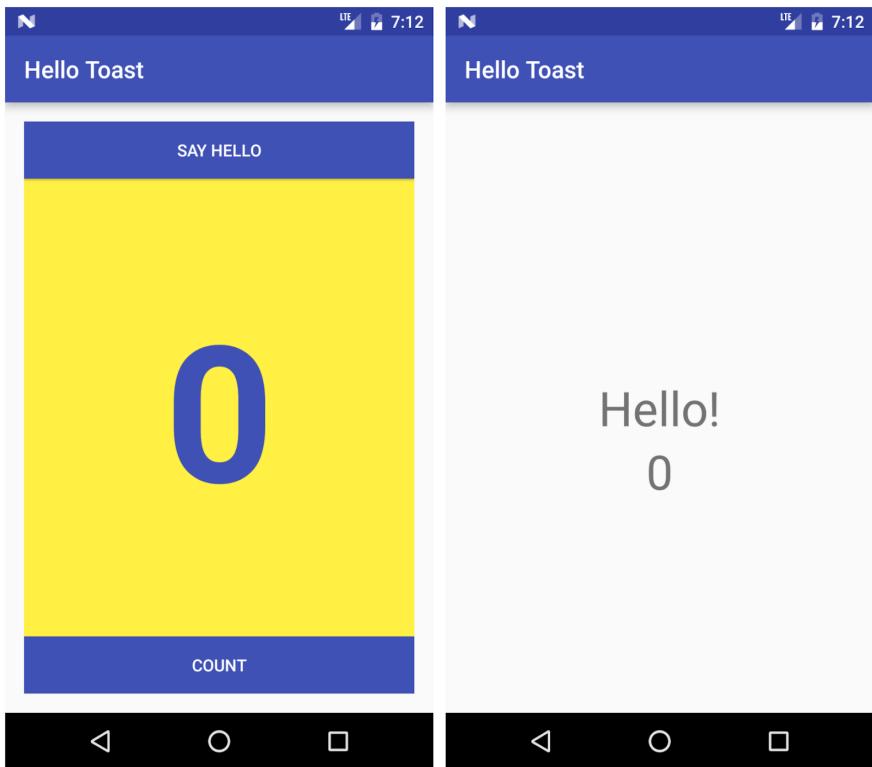
- [Thiết kế điều hướng Quay lại và Lên](#)
- [Hoạt động](#)
- [Ý định](#)
- [CuộnXem](#)
- [Xem](#)
- [Cái nút](#)
- [Xem văn bản](#)
- [Tài nguyên chuỗi](#)

Bài tập về nhà

Xây dựng và chạy một ứng dụng

Mở [Xin chàoToast](#) ứng dụng mà bạn đã tạo trong phòng thí nghiệm mã thực hành trước đó.

1. Sửa đổi **Nút** để nó khởi chạy một cái mới **Hoạt động** để hiển thị từ "Xin chào!" và số lượng hiện tại, như hiển thị bên dưới.
2. Thay đổi văn bản trên **Nút** để **Nói Xin chào**.



Trả lời những câu hỏi này

Câu hỏi 1

Những thay đổi nào được thực hiện khi bạn thêm một thứ hai? Hoạt động vào ứng dụng của bạn bằng cách chọn **Tệp > Mới > Hoạt động** và một **Hoạt động** mẫu? Chọn một:

- Thứ hai Hoạt động được thêm vào như một lớp Java. Bạn vẫn cần phải thêm tệp bố cục XML.
- Thứ hai Hoạt động Tệp bố cục XML được tạo và thêm lớp Java. Bạn vẫn cần phải xác định chữ ký lớp.
- Thứ hai Hoạt động được thêm vào như một lớp Java, tệp bố cục XML được tạo và AndroidManifest.xmlle được thay đổi để khai báo thứ hai Hoạt động.

- Thứ haiHoạt độngTệp bối cục XML được tạo ra vàAndroidManifest.xmlle được thay đổi để khai báo thứ haiHoạt động.

Câu hỏi 2

Điều gì xảy ra nếu bạn loại bỏ android:parentActivityNamevà <siêu dữ liệu>các yếu tố từ thứ haiHoạt độngtuyên bố trongAndroidManifest.xmlle? Chọn một trong các lựa chọn sau:

- Thứ haiHoạt độngkhông còn xuất hiện nữa khi bạn cố gắng bắt đầu nó bằng một lệnh rõ ràng Ý định.
- Thứ haiHoạt độngTệp bối cục XML đã bị xóa.
- Nút Quay lại không còn hoạt động trong giây thứ haiHoạt độngđể gửi người dùng trở lại trang chínhHoạt động.
- Nút Lên trên thanh ứng dụng không còn xuất hiện ở giây thứ haiHoạt độngđể gửi người dùng trở lại cha mẹHoạt động.

Câu hỏi 3

Bạn sử dụng phương thức xây dựng nào để tạo một đối tượng rõ ràng mới?Ý định?Chọn một trong các mục sau:

- Ý định mới()
- new Intent(Context context, Class<?> class)
- new Intent(String action, Uri uri)
- new Intent(Chuỗi hành động)

Câu hỏi 4

Trong bài tập về ứng dụng HelloToast, làm thế nào để bạn thêm giá trị hiện tại của số đếm vàoÝ định? Chọn một trong các mục sau:

- Như làÝ địnhđữ liệu
- Như làÝ định YÊU CẦU VĂN BẢN
- Là mộtÝ địnhhoạt động
- Là mộtÝ địnhthêm

Câu hỏi 5

Trong bài tập về ứng dụng HelloToast, làm thế nào để hiển thị số đếm hiện tại trong "Hello" thứ hai? Hoạt động? Chọn một trong các mục sau:

- Nhận được Ý định rằng Hoạt động đã được ra mắt với.
- Lấy giá trị đếm hiện tại ra khỏi Ý định.
- Cập nhật Xem văn bản để đếm.
- Tất cả những điều trên.

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

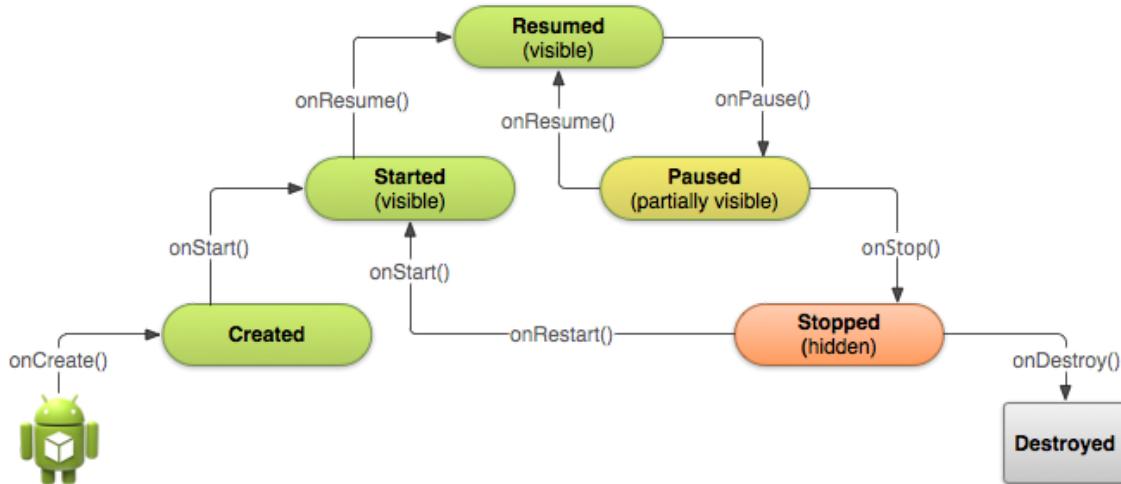
Kiểm tra xem ứng dụng có các tính năng sau không:

- Nó hiển thị **Nói Xin chào** nút thay vì **Nhấn** cái nút.
- Thứ hai Hoạt động bắt đầu khi **Nói Xin chào** nút được nhấn và nó sẽ hiển thị thông báo "Xin chào!" và số đếm hiện tại từ Hoạt động chính.
- Thứ hai Hoạt động Các tệp bố cục Java và XML đã được thêm vào dự án.
- Tệp bố cục XML cho thứ hai Hoạt động chứa hai Xem văn bản các phần tử, một phần tử có chuỗi "Xin chào!" và phần tử thứ hai có đếm.
- Nó bao gồm việc triển khai phương pháp xử lý nhấp chuột cho **Nói Xin chào** nút (trong Hoạt động chính).
- Nó bao gồm việc thực hiện khi tạo() phương pháp cho thứ hai Hoạt động và cập nhật số lượng Xem văn bản với đếm từ Hoạt động chính.

Bài 2.2: Vòng đời và trạng thái của hoạt động

Giới thiệu

Trong phần thực hành này, bạn sẽ tìm hiểu thêm về *vòng đời hoạt động*. Vòng đời là tập hợp các trạng thái mà một hoạt động có thể có trong toàn bộ vòng đời của nó, từ khi nó được tạo ra cho đến khi nó bị hủy và hệ thống lấy lại tài nguyên của nó. Khi người dùng điều hướng giữa các hoạt động trong ứng dụng của bạn (cũng như vào và ra khỏi ứng dụng của bạn), các hoạt động sẽ chuyển đổi giữa các trạng thái khác nhau trong vòng đời của chúng.



Mỗi giai đoạn trong vòng đời của một hoạt động đều có phương thức gọi lại tương ứng: `onCreate()`, `onStart()`, `onPause()`, và vân vân. Khi một hoạt động thay đổi trạng thái, phương thức gọi lại liên quan sẽ được gọi. Bạn đã thấy một trong những phương thức sau: khi Tạo(). Bằng cách ghi đè bất kỳ phương thức gọi lại vòng đời nào trong `Hoạt động` lớp, bạn có thể thay đổi hành vi mặc định của hoạt động để phản hồi lại hành động của người dùng hoặc hệ thống.

Trạng thái hoạt động cũng có thể thay đổi để phản hồi các thay đổi về cấu hình thiết bị, ví dụ khi người dùng xoay thiết bị từ chế độ dọc sang chế độ ngang. Khi những thay đổi về cấu hình này xảy ra, hoạt động sẽ bị hủy và được tạo lại ở trạng thái mặc định, và người dùng có thể mất thông tin mà họ đã nhập vào hoạt động. Để tránh gây nhầm lẫn cho người dùng, điều quan trọng là bạn phải phát triển ứng dụng của mình để ngăn ngừa mất dữ liệu bất ngờ. Sau đó trong phần thực hành này, bạn sẽ thử nghiệm các thay đổi về cấu hình và tìm hiểu cách bảo toàn trạng thái của hoạt động để phản hồi các thay đổi về cấu hình thiết bị và các sự kiện vòng đời hoạt động khác.

Trong bài thực hành này, bạn thêm các câu lệnh ghi nhật ký vào ứng dụng TwoActivities và quan sát các thay đổi trong vòng đời hoạt động khi bạn sử dụng ứng dụng. Sau đó, bạn bắt đầu làm việc với những thay đổi này và khám phá cách xử lý đầu vào của người dùng trong những điều kiện này.

Những điều bạn nên biết

Bạn sẽ có thể:

- Tạo và chạy dự án ứng dụng trong Android Studio.
- Thêm các câu lệnh nhật ký vào ứng dụng của bạn và xem các nhật ký đó trong **Logcat** khung cửa sổ.
- Hiểu và làm việc với một **Hoạt động** và một **Ý định**, và thoái mái khi tương tác với họ.

Những gì bạn sẽ học được

- Làm thế nào **Hoạt động** vòng đời hoạt động.
- Khi một **Hoạt động** bắt đầu, tạm dừng, dừng hẳn và bị phá hủy.
- Về các phương thức gọi lại vòng đời liên quan đến **Hoạt động** thay đổi.
- Tác động của các hành động (chẳng hạn như thay đổi cấu hình) có thể dẫn đến **Hoạt động** sự kiện vòng đời.
- Làm thế nào để giữ lại **Hoạt động** trạng thái trong suốt các sự kiện vòng đời.

Bạn sẽ làm gì

- Thêm mã vào ứng dụng **TwoActivities** từ bài thực hành trước để triển khai các chức năng khác nhau **Hoạt động** gọi lại vòng đời để bao gồm các câu lệnh ghi nhật ký.
- Quan sát các thay đổi trạng thái khi ứng dụng của bạn chạy và khi bạn tương tác với từng ứng dụng **Hoạt động** trong ứng dụng của bạn.
- Sửa đổi ứng dụng của bạn để giữ nguyên trạng thái phiên bản của **Hoạt động** được tạo lại bất ngờ để phản hồi hành vi của người dùng hoặc thay đổi cấu hình trên thiết bị.

Tổng quan về ứng dụng

Trong phần thực hành này bạn thêm vào [Hai Hoạt Động](#) ứng dụng. Ứng dụng trông và hoạt động gần giống như trong codelab trước. Nó chứa hai **Hoạt động** triển khai và cung cấp cho người dùng khả năng

Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế. PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

gửi giữa chúng. Những thay đổi bạn thực hiện đối với ứng dụng trong phần thực hành này sẽ không ảnh hưởng đến hành vi người dùng có thể nhìn thấy.

Nhiệm vụ 1: Thêm các lệnh gọi lại vòng đời vào TwoActivities

Trong nhiệm vụ này bạn sẽ thực hiện tất cả các **Hoạt động** phương pháp gọi lại vòng đời để in thông báo tới logcat khi các phương pháp đó được gọi. Các thông báo nhật ký này sẽ cho phép bạn xem khi nào **Hoạt động** vòng đời thay đổi trạng thái và cách những thay đổi trạng thái vòng đời đó ảnh hưởng đến ứng dụng của bạn khi chạy.

1.1 (Tùy chọn) Sao chép dự án TwoActivities

Đối với các nhiệm vụ trong phần thực hành này, bạn sẽ sửa đổi các mục hiện có [Hai Hoạt Động](#) dự án bạn đã xây dựng trong bài thực hành cuối cùng. Nếu bạn muốn giữ nguyên dự án TwoActivities trước đó, hãy làm theo các bước trong [Phụ lục: Tiện ích](#) để tạo một bản sao của dự án.

1.2 Triển khai các lệnh gọi lại vào MainActivity

1. Mở dự án TwoActivities trong Android Studio và mở **Hoạt động chính** trong **Dự án > Android** khung cửa sổ.
2. Trong khi tạo() phương pháp, thêm các câu lệnh nhật ký sau:

```
Nhật ký.d(LOG_TAG, "");  
Nhật ký.d(LOG_TAG, "khi tạo");
```

3. Thêm ghi đè cho khi **Bắt đầu()** gọi lại, với một câu lệnh vào nhật ký cho sự kiện đó:

```
@Ghi đè  
công khai void onStart(){  
    super.onStart();  
    Nhật ký.d(LOG_TAG, "onStart");
```

}

Để có một phím tắt, hãy chọn **Mã > Ghi đè phương thức** trong Android Studio. Một hộp thoại xuất hiện với tất cả các phương thức có thể bạn có thể ghi đè trong lớp của mình. Việc chọn một hoặc nhiều phương thức gọi lại từ danh sách sẽ chèn một mẫu hoàn chỉnh cho các phương thức đó, bao gồm cả lệnh gọi bắt buộc đến siêu lớp.

4.Sử dụng khi **Bắt đầu()** phương pháp như một khuôn mẫu để thực hiện **onPause()**, **onRestart()**, **onResume()**, **onStop()**. Và khi **Hủy Bỏ()** gọi lại vòng đời.

Tất cả các phương thức gọi lại đều có cùng chữ ký (trừ tên). Nếu bạn **Sao chép** và **Dán** khi **Bắt đầu()** để tạo các phương thức gọi lại khác này, đừng quên cập nhật nội dung để gọi đúng phương thức trong siêu lớp và ghi lại phương thức đúng.

1.Chạy ứng dụng của bạn.

2.Nhấp vào **Logcat** tab ở cuối Android Studio để hiển thị **Logcat** ngắn. Bạn sẽ thấy ba thông báo nhật ký hiển thị ba trạng thái vòng đời Hoạt động đã chuyển tiếp như lúc bắt đầu:

D/Hoạt động chính: ----- D/
Hoạt động chính: onCreate D/
Hoạt động chính: onStart D/
Hoạt động chính: onResume

1.3 Triển khai các cuộc gọi lại vòng đời trong SecondActivity

Bây giờ bạn đã triển khai các phương thức gọi lại vòng đời cho Hoạt động chính, làm tương tự cho Hoạt động thứ hai.

1.Mở **Hoạt động thứ hai**.

2.Ở đầu lớp, thêm một hằng số cho **NHẬT KÝ_THẺ** biến:

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

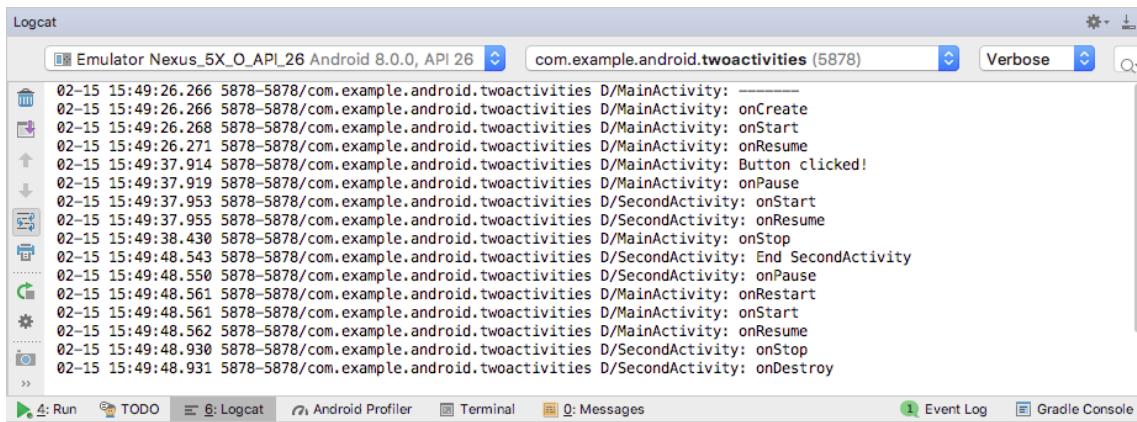
3.Thêm các lệnh gọi lại vòng đời và các câu lệnh nhật ký vào thứ haiHoạt động. (Bạn có thể**Sao chép**Và **Dán**các phương thức gọi lại từHoạt động chính.)

4.Thêm một câu lệnh nhật ký và trả vềTrả lời()phương pháp ngay trước khi hoàn thành()phương pháp:

```
Log.d(LOG_TAG, "Kết thúc hoạt động thứ hai");
```

1.4 Quan sát nhật ký khi ứng dụng chạy

- 1.Chạy ứng dụng của bạn.
- 2.Nhấp vào**Logcat**tab ở cuối Android Studio để hiển thị**Logcat**khung cửa sổ.
- 3.Đi vào**Hoạt động**trong hộp tìm kiếm.
Nhật ký Android có thể rất dài và lộn xộn. Bởi vì NHẬT KÝ_ THÊbiến trong mỗi lớp chứa các từ Hoạt động chínhhoặc Hoạt động thứ hai,Tùy khóa này cho phép bạn lọc nhật ký chỉ để tìm những thông tin bạn quan tâm.



Tác phẩm này được cấp phép theo một Giấy phép Creative Commons Ghi công 4.0 Quốc tế . PDF này là bản chụp nhanh một lần. Xem [dev.android.com/courses/fundamentals-training/toc-v2](https://developer.android.com/courses/fundamentals-training/toc-v2)

để biết thông tin cập nhật mới nhất.

Thử nghiệm bằng ứng dụng của bạn và lưu ý rằng các sự kiện vòng đời xảy ra để phản hồi các hành động khác nhau. Cụ thể, hãy thử những điều sau:

- Sử dụng ứng dụng bình thường (gửi tin nhắn, trả lời bằng tin nhắn khác).
- Sử dụng nút Quay lại để quay lại từ thứ haiHoạt độngđến chínhHoạt động.
- Sử dụng mũi tên Lên trên thanh ứng dụng để quay lại từ thứ haiHoạt độngvào những thời điểm khác nhau trong ứng dụng của bạn và quan sát những gì xảy ra trong nhật ký và trên màn hình.
- Nhấn nút tổng quan (nút hình vuông bên phải Trang chủ) và đóng ứng dụng (chạm vàoX).
- Trở lại màn hình chính và khởi động lại ứng dụng.

MẸO:Nếu bạn đang chạy ứng dụng của mình trong trình giả lập, bạn có thể mô phỏng chuyển động quay bằngĐiều khiển+F11 hoặc Control+Function+F11.

Mã giải bài tập 1

Các đoạn mã sau đây hiển thị mã giải pháp cho nhiệm vụ đầu tiên.

Hoạt động chính

Các đoạn mã sau đây hiển thị mã đã thêm vàoHoạt động chính,nhưng không phải toàn bộ lớp.

Cáckhi tạo()phương pháp:

```
@Ghi đè  
được bảo vệ void onCreate(Gói savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    // Ghi lại thời điểm bắt đầu của phương thức  
    onCreate(). Log.d(LOG_TAG, "-----");  
    Nhật ký.d(LOG_TAG, "onCreate");
```

```
// Khởi tạo tất cả các biến dạng xem. mMessageEditText =
findViewById(R.id.editText_main); mReplyHeadTextView =
findViewById(R.id.text_header_reply); mReplyTextView =
findViewById(R.id.text_message_reply);
}
```

Các phương pháp vòng đời khác:

```
@Ghi đè
được bảo vệ void onStart() {
    super.onStart();
    Nhật ký.d(LOG_TAG, "onStart");
}

@Ghi đè
được bảo vệ void onPause() {
    super.onPause();
    Nhật ký.d(LOG_TAG, "onPause");
}

@Ghi đè
được bảo vệ void onRestart() {
    super.onRestart();
    Nhật ký.d(LOG_TAG, "onRestart");
}

@Ghi đè
được bảo vệ void onResume() {
    super.onResume();
    Nhật ký.d(LOG_TAG, "onResume");
}

@Ghi đè
được bảo vệ void onStop() {
    super.onStop();
    Nhật ký.d(LOG_TAG, "onStop");
}

@Ghi đè
được bảo vệ void onDestroy()
```

Khóa học Cơ bản dành cho nhà phát triển Android (V2) – Đơn vị 1

Gửi ứng dụng của bạn để chấm điểm

Hướng dẫn cho người chấm điểm

Không có ứng dụng nào để nộp bài tập về nhà này.

Kết thúc đơn vị