**TRƯỜNG ĐẠI HỌC THỦY LỢI**

**KHOA CÔNG NGHỆ THÔNG TIN**



**GIÁO TRÌNH**

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

Hà Nội, 2.2025

MỤC LỤC

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

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

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

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

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

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

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

[Bài 2) Activities 5](#_Toc190855154)

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

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

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

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

[3.1) Trình gỡ lỗi 5](#_Toc190855159)

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

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

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

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

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

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

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

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

[1.5) RecycleView 6](#_Toc190855168)

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

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

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

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

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

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

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

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

[1.1) AsyncTask 6](#_Toc190855177)

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

[1.3) Broadcast receivers 6](#_Toc190855179)

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

[2.1) Thông báo 6](#_Toc190855181)

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

[2.3) JobScheduler 6](#_Toc190855183)

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

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

[1.1) Shared preferences 7](#_Toc190855186)

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

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

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

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

3.1) Trinfh gowx loi ……………………………………………………………………...

# LÀM QUEN

## Tạo ứng dụng đầu tiên

### Android Studio và Hello World

Giới thiệu

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

Những gì Bạn nên biết

Bạn nên có khả năng:

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

A screenshot of a computer

AI-generated content may be incorrect.

Những gì Bạn sẽ cần:

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

Những gì bạn sẽ học

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

Những gì bạn sẽ làm

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

### Giao diện người dùng tương tác đầu tiên

### Trình chỉnh sửa bố cục

### Văn bản và các chế độ cuộn

### Tài nguyên có sẵn

## Activities

### Activity và Intent

### Vòng đời của Activity và trạng thái

### Intent ngầm định

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

### Trình gỡ lỗi

### Kiểm thử đơn vị

### Thư viện hỗ trợ

# TRẢI NGHIỆM NGƯỜI DÙNG

## Tương tác người dùng

### Hình ảnh có thể chọn

Giao diện người dùng xuất hiện trên các thiết bị hỗ trợ hệ điều hành Android bao gồm giao diện phân cấp các vật thể được gọi là các view. Mỗi thành phần trên màn hình là 1 view

Lớp view biểu diễn khối xây dựng cơ bản của tất cả các thành phần UI (Giao diện người dùng). View là lớp cơ bản cho các lớp khác thứ mà cung cấp những thành phần UI có thể tương tác, như thành phần nút bấm (Button). Nút bấm là 1 nguyên tố UI mà người dùng có thể nhấn để thực hiện 1 hành động

Bạn có thể chuyển bất cứ View nào, như View hình ảnh, thành 1 nguyên tố UI có thể nhấn hoặc click. Bạn phải lưu trữ hình ảnh cho View hình ảnh trong thư mục có thể vẽ được trong dự án của bạn.

Trong bài này, bạn sẽ học cách sử dụng hình ảnh như 1 nguyên tố mà người dùng có thể nhấn hoặc click chuột

**Những thứ bạn nên biết**

* Bạn nên có khả năng:
* Tạo một dự án Android Studio từ một mẫu có sẵn và tạo bố cục chính.
* Chạy ứng dụng trên trình giả lập hoặc thiết bị được kết nối.
* Tạo và chỉnh sửa các thành phần UI bằng trình chỉnh sửa bố cục và code XML.
* Truy cập các thành phần UI từ code của bạn bằng findViewById().
* Xử lý sự kiện nhấn nút.
* Hiển thị thông báo Toast.
* Thêm hình ảnh vào thư mục có thể vẽ của dự án.

**Những thứ bạn sẽ được học**

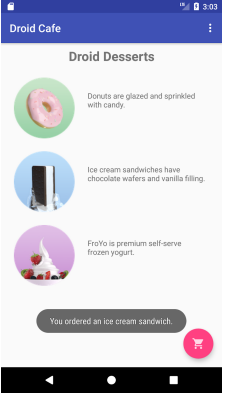
* Cách sử dụng hình ảnh như 1 thành phần có thể tương tác được để thực hiện 1 hành động
* Cách thiết lập các thuộc tính cho các thành phần ImageView trong trình chỉnh sửa bố cục
* Cách thêm phương thức onClick() để hiển thị thông báo Toast

**Những thứ bạn sẽ làm được**

* Tạo 1 dự án Android mới cho ứng dụng đặt hàng bánh tráng miệng mà sử dụng hình ảnh như những thuộc tính tương tác được
* Thiết lập phương thức onClick cho hình ảnh để hiển thị các thông báo Toast khác nhau
* Thay đổi nút tác vụ nổi được cung cấp bởi mẫu có sẵn để nó hiển thị các biểu tượng khác nhau và thực hiện các hành động khác nhau

**Tổng quan ứng dụng**

Trong bài thực hành này, bạn sẽ tạo và xây dựng một ứng dụng mới dựa trên mẫu Basic Activity, mô phỏng một ứng dụng đặt bánh tráng miệng. Người dùng có thể nhấn vào hình ảnh để thực hiện một hành động—trong trường hợp này là hiển thị một thông báo Toast—như minh họa trong hình dưới đây. Ngoài ra, người dùng có thể nhấn vào nút giỏ hàng để chuyển sang Activity tiếp theo.

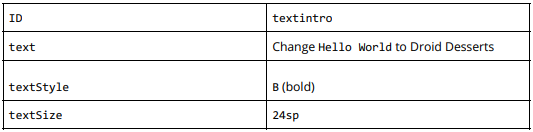
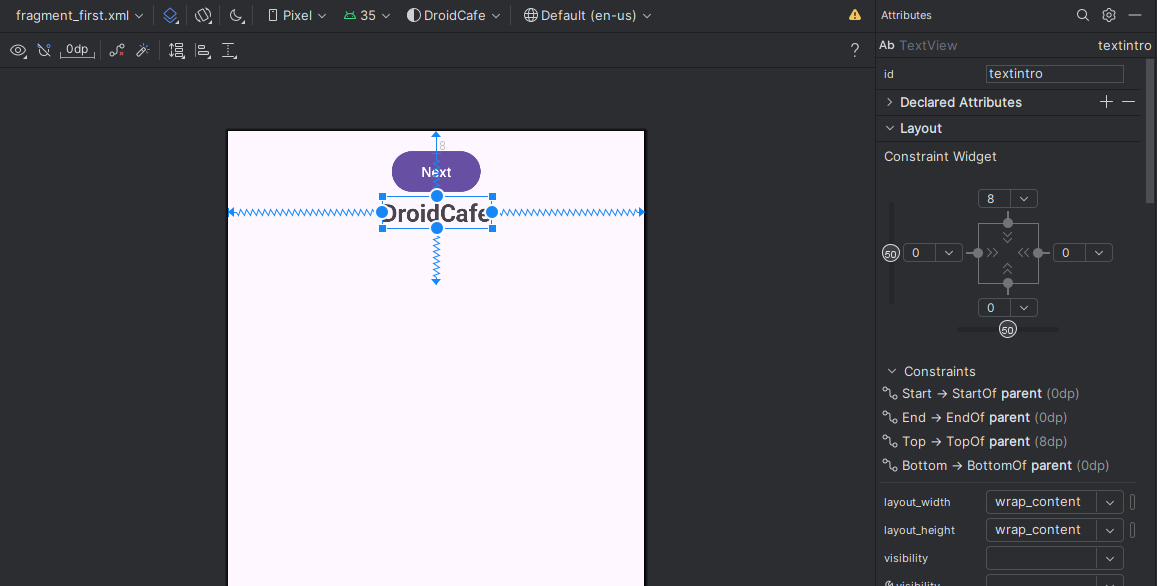


**Bước 1: Thêm hình ảnh vào giao diện**

Bạn có thể làm cho một View có thể nhấn, giống như một nút, bằng cách thêm thuộc tính android:onClick trong tệp XML của layout. Ví dụ, bạn có thể làm cho một hình ảnh hoạt động như một nút bằng cách thêm android:onClick vào ImageView.

Trong bước này, bạn sẽ tạo một nguyên mẫu ứng dụng đặt món tráng miệng từ một quán cà phê. Sau khi khởi tạo một dự án mới dựa trên mẫu Basic Activity, bạn sẽ chỉnh sửa TextView mặc định "Hello World" thành văn bản phù hợp và thêm các hình ảnh mà người dùng có thể nhấn để tương tác.

**1.1 Bắt đầu dự án mới**

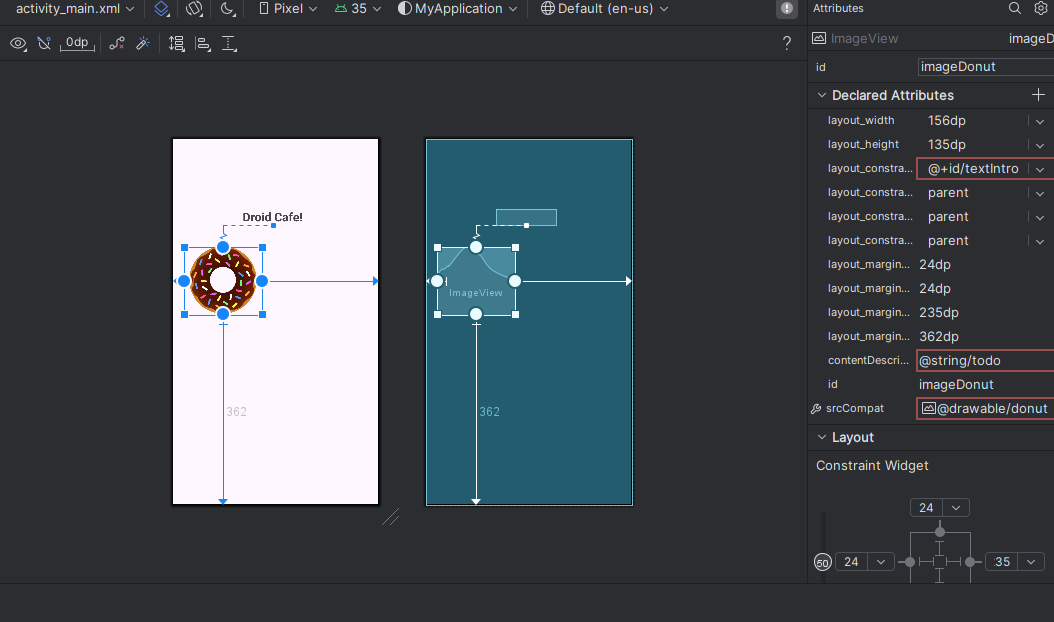
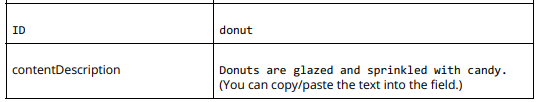
* Tạo một dự án mới trong Android Studio với tên ứng dụng là Droid Cafe.
* Chọn mẫu Basic Activity và giữ nguyên tên MainActivity mặc định.Đảm bảo đã chọn Generate Layout File và Backwards Compatibility (AppCompat).
* Nhấn Finish để tạo dự án.
* Dự án sẽ mở ra với hai tệp layout trong thư mục res > layout:,activity\_main.xml: Chứa thanh App Bar và Floating Action Button (không thay đổi trong bước này). content\_main.xml: Chứa các thành phần giao diện khác.
* Mở content\_main.xml, chọn tab Design (nếu chưa được chọn) để hiển thị Layout Editor.
* Chọn TextView "Hello World" trong layout, sau đó mở bảng Attributes.
* Thay đổi các thuộc tính textintro như sau:
* Thay đổi các thuộc tính của text như sau 
* Xóa ràng buộc kéo dài từ cạnh dưới của TextView (textintro) đến cạnh dưới của layout, để TextView tự động dính lên phía trên của layout, và đặt lề trên (top margin) là 8dp như hình dưới đây. z
* Trong bài học trước, bạn đã học cách trích xuất một chuỗi tài nguyên từ một chuỗi văn bản cố định. Nhấn vào tab \*\*Text\*\* để chuyển sang mã XML, sau đó trích xuất chuỗi `"Droid Desserts"` vào tài nguyên chuỗi.

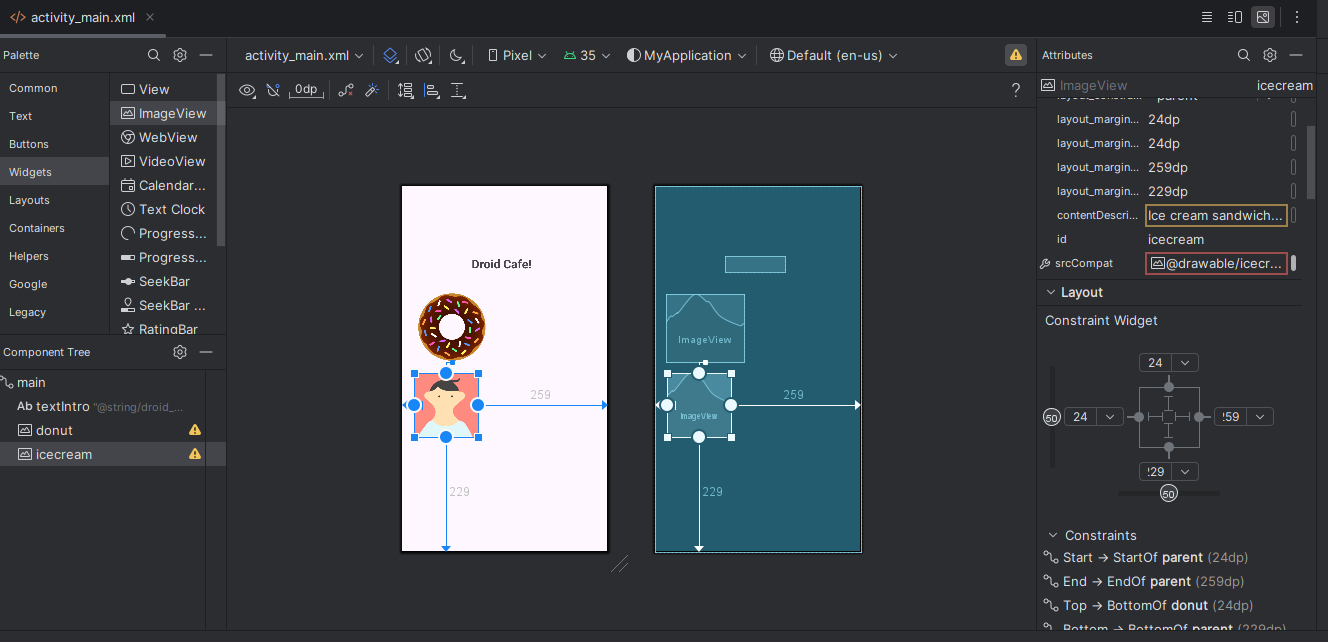
**1.2 Thêm hình ảnh**

Ba hình ảnh (donut\_circle.png, froyo\_circle.png, và icecream\_circle.png) được cung cấp cho ví dụ này. Bạn có thể tải xuống hoặc thay thế bằng hình ảnh của riêng mình dưới dạng tệp PNG, nhưng chúng phải có kích thước khoảng 113 x 113 pixels để sử dụng trong ví dụ này.

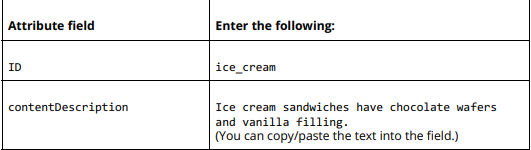
Bước này cũng giới thiệu một kỹ thuật mới trong trình chỉnh sửa giao diện (Layout Editor): sử dụng nút Fix trong thông báo cảnh báo để trích xuất chuỗi tài nguyên.

Các bước thực hiện:

* Đóng dự án.
* Sao chép các tệp hình ảnh vào thư mục drawable của dự án theo đường dẫn
* Mở lại dự án.
* Mở tệp content\_main.xml và chọn tab Design (nếu chưa được chọn).
* Kéo một ImageView vào giao diện, chọn hình ảnh donut\_circle cho nó, rồi thiết lập ràng buộc: Căn ImageView vào TextView phía trên. Căn ImageView vào cạnh trái của layout. Đặt margin là 24dp cho cả hai ràng buộc. 
* Trong mục thuộc tính, điền các giá trị thuộc tính sau: 
* Kéo một ImageView thứ hai vào giao diện, chọn hình ảnh icecream\_circle cho nó, rồi thiết lập ràng buộc:Căn ImageView này vào cạnh dưới của ImageView đầu tiên. Căn ImageView vào cạnh trái của layout. Đặt margin là 24dp cho cả hai ràng buộc.



* Trong mục thuộc tính, điền các giá trị thuộc tính sau:



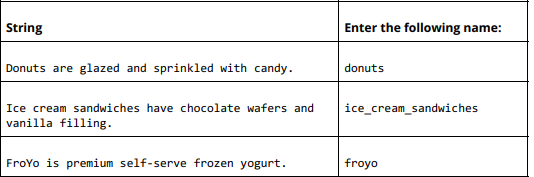
* Kéo một ImageView thứ ba vào giao diện, chọn hình ảnh froyo\_circle cho nó, rồi thiết lập ràng buộc:Căn ImageView này vào cạnh dưới của ImageView thứ hai. Căn ImageView vào cạnh trái của layout. Đặt margin là 24dp cho cả hai ràng buộc.
* Trong bảng \*\*Attributes\*\*, nhập các giá trị sau cho các thuộc tính:

ID:Froyo

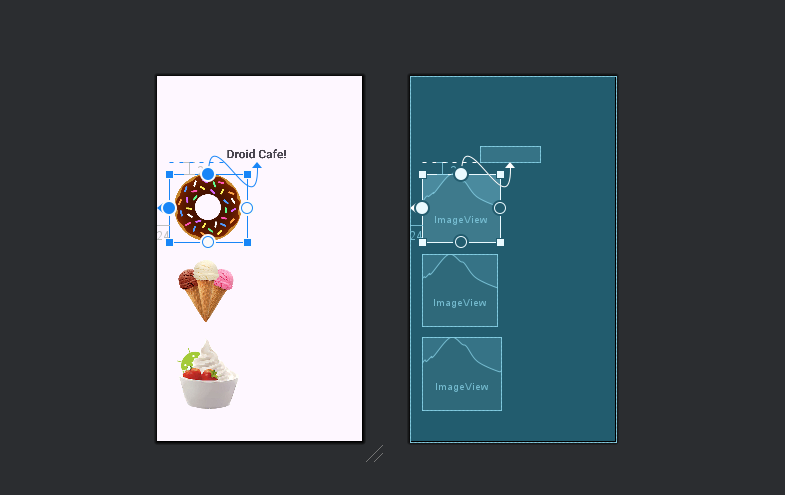
contentDescription: FroYo is premium self‑serve frozen yogurt.

(You can copy/paste the text into the field.)

* Nhấp vào biểu tượng cảnh báo ở góc trên bên trái của Layout Editor để mở bảng cảnh báo. Bảng này sẽ hiển thị các cảnh báo về việc sử dụng văn bản cố định (hardcoded text).
* Mở rộng từng cảnh báo về Hardcoded text, cuộn xuống cuối thông báo cảnh báo, và nhấn nút Fix như minh họa bên dưới. Việc sửa từng cảnh báo về Hardcoded text sẽ trích xuất chuỗi thành tài nguyên chuỗi (String Resource). Hộp thoại Extract Resource sẽ xuất hiện, cho phép bạn nhập tên cho tài nguyên chuỗi. Nhập các tên sau cho từng chuỗi tài nguyên:



Bố cục nên trông giống như hình dưới

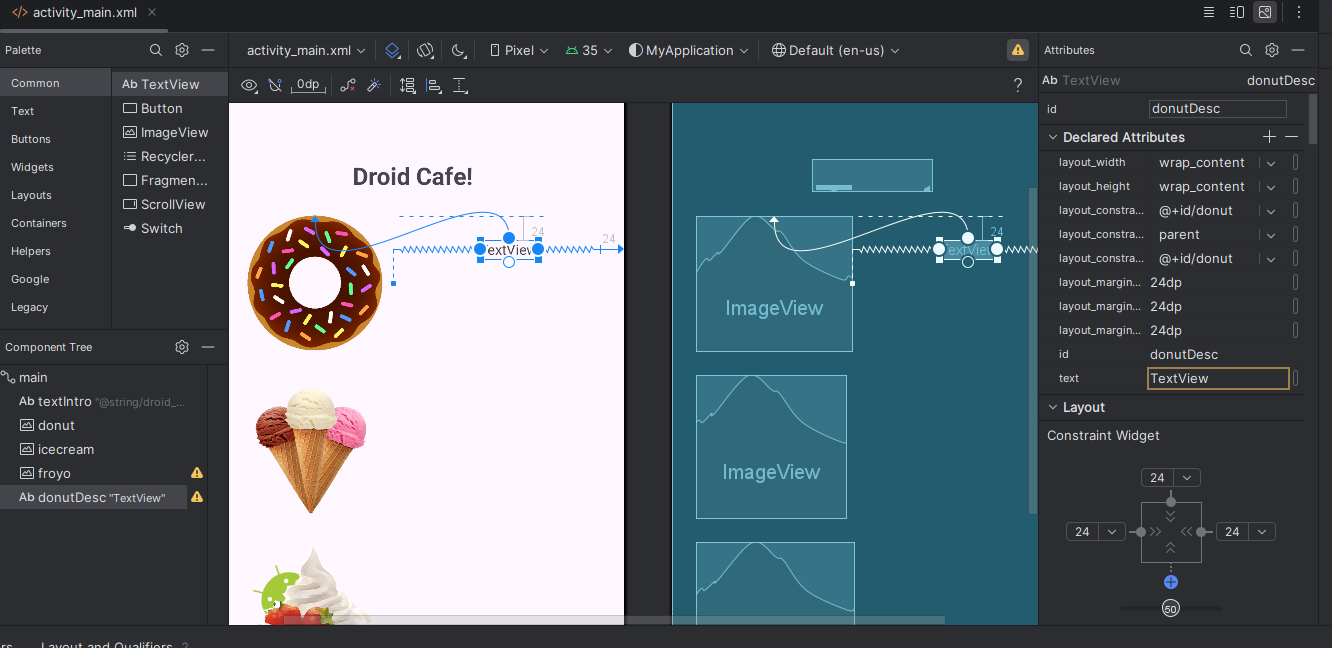


1.3 Thêm mô tả văn bản

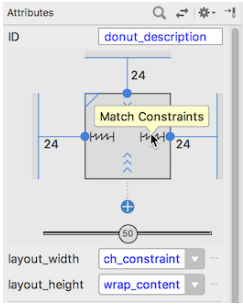
Trong bước này, bạn sẽ thêm một TextView mô tả cho từng món tráng miệng. Vì bạn đã trích xuất tài nguyên chuỗi cho thuộc tính contentDescription của các ImageView, bạn có thể sử dụng lại các tài nguyên chuỗi này cho từng TextView mô tả.

**Các bước thực hiện:**

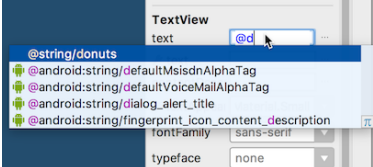
* Kéo một TextView vào giao diện.
* Thiết lập ràng buộc: Căn cạnh trái của TextView vào cạnh phải của ImageView (donut\_circle). Căn cạnh trên của TextView vào cạnh trên của ImageView (donut\_circle). Đặt margin là 24dp cho cả hai ràng buộc trên.
* Căn TextView vào cạnh phải của layout, với margin là 24dp.
* Đặt ID của TextView là donut\_description trong bảng Attributes.



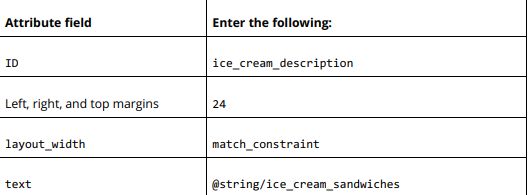
4. Trong bảng Attributes, thay đổi thuộc tính width trong Inspector pane thành Match Constraints (`0dp`).



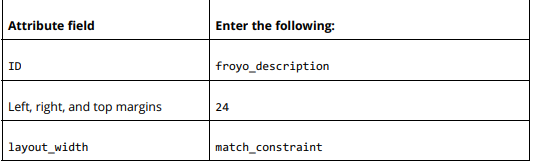
* Trong bảng Attributes, bắt đầu nhập tài nguyên chuỗi cho trường văn bản bằng cách đặt ký tự @ phía trước: `@d`. Sau đó, chọn tên tài nguyên chuỗi (@string/donuts) xuất hiện trong gợi ý.



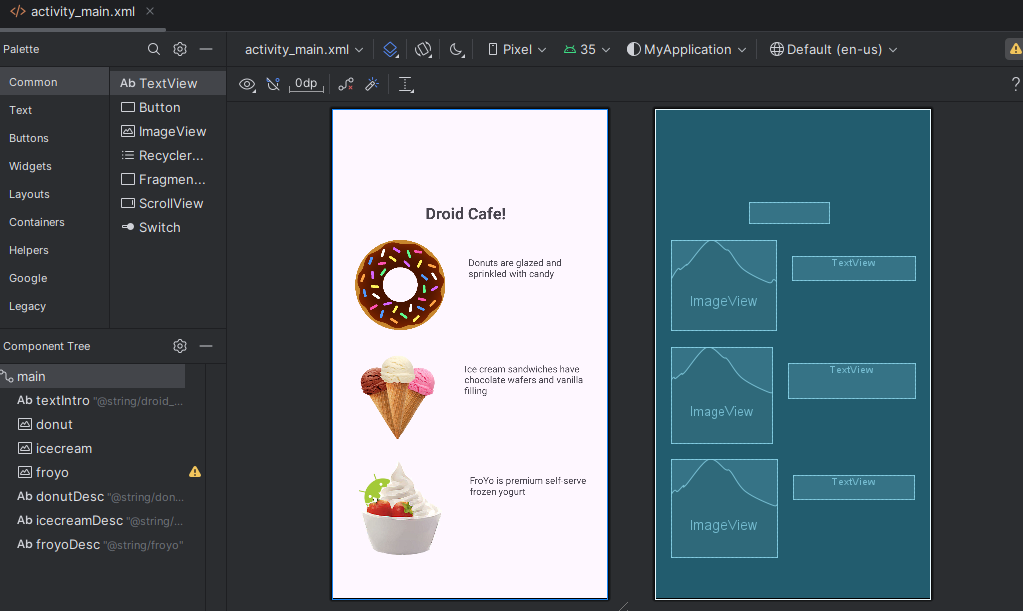
* Lặp lại các bước trên để thêm một TextView thứ hai, với các ràng buộc sau: \*Cạnh trái của `TextView` căn vào cạnh phải của ImageView`(`icecream\_circle`). \*\*Cạnh trên của `TextView` căn vào cạnh trên của `ImageView` (`icecream\_circle`). Cạnh phải của `TextView` căn vào cạnh phải của layout. Margin là 24dp cho tất cả các ràng buộc. Sau đó, nhập các giá trị cần thiết trong bảng Attributes.



* Lặp lại các bước trên để thêm một TextView thứ ba, căn cạnh trái vào cạnh phải của `ImageView` (`froyo\_circle`), cạnh trên vào cạnh trên của `ImageView` (`froyo\_circle`), và cạnh phải vào cạnh phải của layout, với margin 24dp cho tất cả các ràng buộc. Sau đó, nhập các giá trị cần thiết trong bảng \*\*Attributes\*\*.

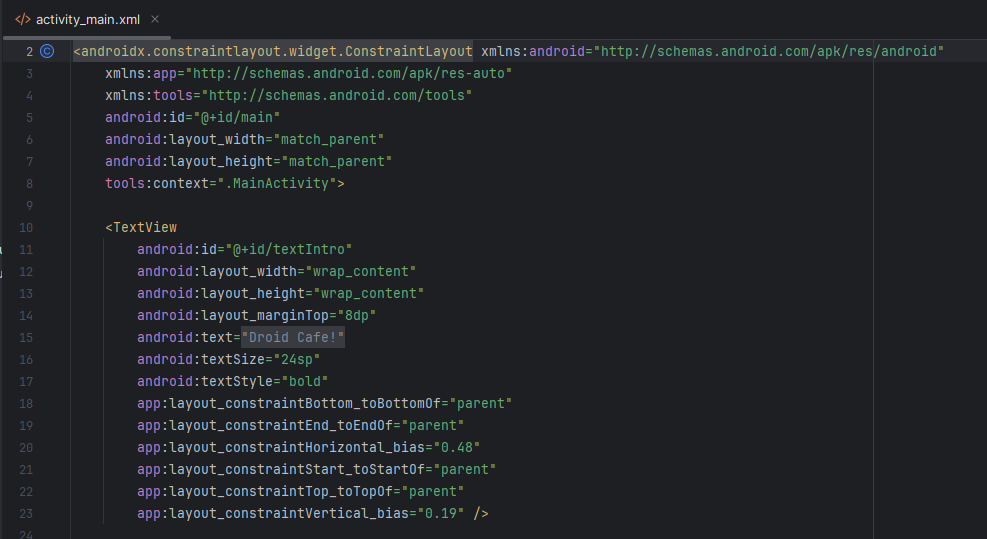


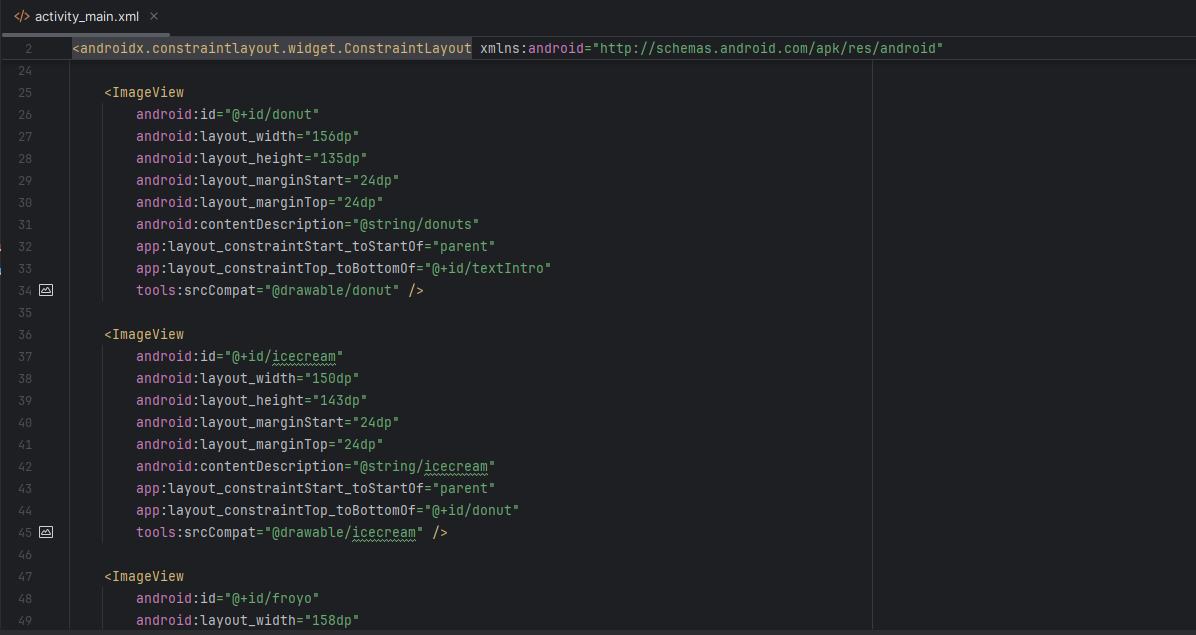
Bố cục nên trông giống như sau

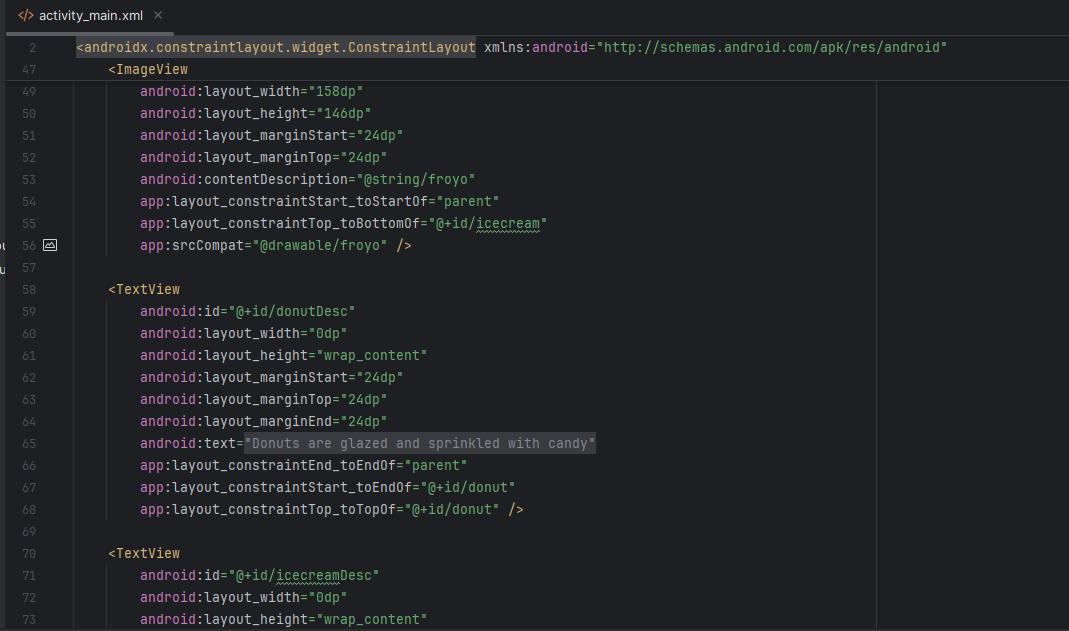


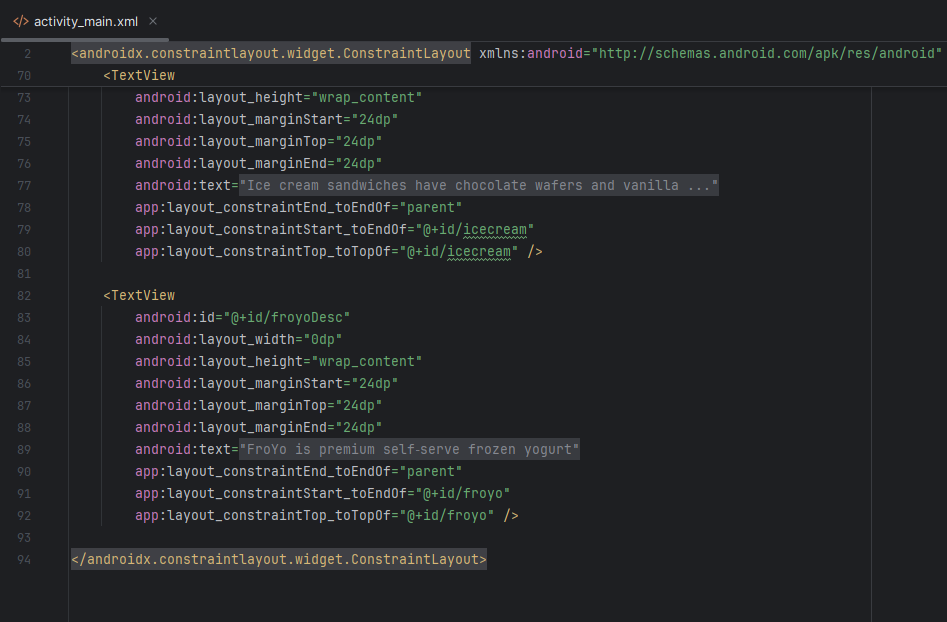
Task1: Mã XML

Mã XML của giao diện trong tệp content.xml sau khi hoàn thành Task 1 được hiển thị dưới đây.







x

**Task 2: Thêm phương thức onClick cho hình ảnh**

Để làm cho một View có thể nhấn được, bạn cần thêm thuộc tính android:onClick vào XML layout và chỉ định trình xử lý sự kiện khi nhấn. Ví dụ, bạn có thể làm cho ImageView hoạt động như một Button đơn giản bằng cách thêm android:onClick vào ImageView. Trong nhiệm vụ này, bạn sẽ làm cho các hình ảnh trong giao diện trở nên có thể nhấn được.

**2.1 Tạo phương thức hiển thị Toast**

Trong bước này, bạn sẽ thêm từng phương thức xử lý sự kiện onClick để gọi khi mỗi hình ảnh được nhấn. Các phương thức này sẽ hiển thị một Toast message thông báo hình ảnh nào đã được nhấn. (Trong một chương khác, các phương thức này sẽ được sửa đổi để chuyển sang một Activity khác.)

* Để sử dụng tài nguyên chuỗi trong mã Java, trước tiên bạn cần thêm chúng vào tệp strings.xml.Mở Project > Android trong Android Studio, mở thư mục res > values, và mở tệp strings.xml. Sau đó, thêm các tài nguyên chuỗi sau để hiển thị trong Toast message:

<string name="donut\_order\_message">You ordered a donut.</string>

<string name="ice\_cream\_order\_message">You ordered an ice cream sandwich.</string>

<string name="froyo\_order\_message">You ordered a FroYo.</string>

* Mở tệp `MainActivity.java` và thêm phương thức `displayToast()` vào cuối lớp `MainActivity`, ngay trước dấu ngoặc nhọn đóng (`}`) của lớp:

public void displayToast(String message) {

Toast.makeText(getApplicationContext(), message,

Toast.LENGTH\_SHORT).show();

}

Mặc dù bạn có thể thêm phương thức này ở bất kỳ vị trí nào trong MainActivity, nhưng theo best practice, bạn nên đặt các phương thức của riêng mình bên dưới các phương thức mặc định đã được cung cấp bởi template trong MainActivity.

**2.2 Tạo trình xử lý sự kiện onClick**

Mỗi hình ảnh có thể nhấn cần một click handler—một phương thức để android:onClick gọi đến. Khi được gọi từ thuộc tính android:onClick, phương thức xử lý sự kiện phải có các đặc điểm sau:public,return void,chỉ nhận một tham số kiểu View. Thêm phương thức xử lý sự kiện cho hình ảnh bánh donut

* Mở MainActivity.java và thêm phương thức sau để xử lý sự kiện khi người dùng nhấn vào hình ảnh bánh donut:

public void showDonutOrder(View view) {

displayToast(getString(R.string.donut\_order\_message));

}

Ba dòng đầu tiên là Javadoc comment, giúp mã dễ hiểu hơn và hỗ trợ tạo tài liệu tự động. Đây là một best practice và bạn nên thêm chú thích này vào mọi phương thức mới.

* Thêm các phương thức sau vào cuối MainActivity để xử lý sự kiện khi người dùng nhấn vào từng món tráng miệng:

public void showIceCreamOrder(View view) {

displayToast(getString(R.string.ice\_cream\_order\_message));

}

/\*\*

\* Shows a message that the froyo image was clicked.

\*/

public void showFroyoOrder(View view) {

displayToast(getString(R.string.froyo\_order\_message));

}

* (Tùy chọn) Chọn Code > Reformat Code để định dạng lại mã trong MainActivity, giúp mã tuân theo tiêu chuẩn và dễ đọc hơn.

**2.3 Thêm thuộc tính onClick**

Trong bước này, bạn sẽ thêm thuộc tính android:onClick vào từng phần tử ImageView trong tệp content\_main.xml. Thuộc tính android:onClick sẽ gọi trình xử lý sự kiện khi người dùng nhấn vào từng hình ảnh.

* Mở tệp content\_main.xml, nhấn vào tab Text trong trình chỉnh sửa giao diện để hiển thị mã XML.
* Thêm thuộc tính android:onClick vào ImageView của bánh donut. Khi nhập, các gợi ý sẽ xuất hiện, cho phép bạn chọn trình xử lý sự kiện showDonutOrder. Mã XML sau khi chỉnh sửa sẽ trông như sau:

<ImageView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:padding="10dp"

android:id="@+id/donut"

android:layout\_below="@id/choose\_dessert"

android:contentDescription="@string/donut"

android:src="@drawable/donut\_circle"

android:onClick="showDonutOrder"/>

Dòng cuối cùng (android:onClick="showDonutOrder") gán trình xử lý sự kiện (showDonutOrder) cho ImageView.

* (Tùy chọn) Chọn Code > Reformat Code để định dạng lại mã XML trong content\_main.xml, giúp mã tuân theo tiêu chuẩn và dễ đọc hơn. Android Studio sẽ tự động di chuyển thuộc tính android:onClick lên trên để nhóm cùng với các thuộc tính khác có tiền tố android:.
* Lặp lại quy trình tương tự để thêm thuộc tính android:onClick cho các phần tử ImageView của kem và sữa chua, chọn các trình xử lý sự kiện showIceCreamOrder và showFroyoOrder. Bạn cũng có thể chọn Code > Reformat Code để định dạng lại mã XML.Mã XML sau khi chỉnh sửa sẽ trông như sau:

<ImageView

android:id="@+id/ice\_cream"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginStart="@dimen/margin\_wide"

android:layout\_marginTop="@dimen/margin\_wide"

android:contentDescription="@string/ice\_cream\_sandwiches"

android:onClick="showIceCreamOrder"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/donut"

app:srcCompat="@drawable/icecream\_circle" />

<ImageView

android:id="@+id/froyo"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginStart="@dimen/margin\_wide"

android:layout\_marginTop="@dimen/margin\_wide"

android:contentDescription="@string/froyo"

android:onClick="showFroyoOrder"

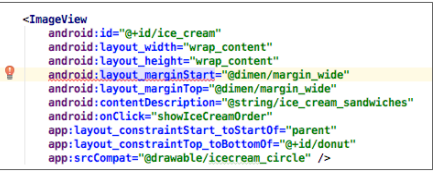
app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/ice\_cream"

app:srcCompat="@drawable/froyo\_circle" />

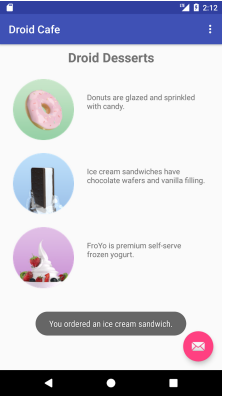
Lưu ý rằng thuộc tính android:layout\_marginStart trong mỗi ImageView bị gạch chân màu đỏ. Thuộc tính này xác định khoảng cách lề "bắt đầu" cho ImageView, tức là nằm ở bên trái đối với hầu hết các ngôn ngữ nhưng sẽ nằm ở bên phải đối với các ngôn ngữ đọc từ phải sang trái (RTL).

* Nhấp vào phần tiền tố android: của thuộc tính android:layout\_marginStart, và một biểu tượng bóng đèn đỏ sẽ xuất hiện bên cạnh nó, như được minh họa trong hình dưới đây.



* Để đảm bảo ứng dụng của bạn tương thích với các phiên bản Android trước đó, nhấp vào biểu tượng bóng đèn đỏ bên cạnh mỗi thuộc tính android:layout\_marginStart, sau đó chọn Set layout\_marginLeft… để đặt giá trị layout\_marginLeft thành @dimen/margin\_wide.
* Chạy ứng dụng.

Khi nhấn vào hình ảnh bánh donut, kem hoặc sữa chua, một thông báo Toast sẽ hiển thị nội dung về đơn hàng, như minh họa trong hình dưới đây.



**Task 3: Thay đổi nút hành động nổi (Floating Action Button - FAB)**

Khi bạn nhấn vào nút hành động nổi có biểu tượng email ở góc dưới màn hình, mã trong MainActivity sẽ hiển thị một thông báo ngắn gọn trong một snackbar. Snackbar này xuất hiện từ dưới lên trên trên điện thoại thông minh hoặc từ góc dưới bên trái trên các thiết bị lớn hơn, sau đó tự động biến mất sau vài giây. Snackbar thường được sử dụng để cung cấp phản hồi về một thao tác nào đó.Bạn có thể tham khảo cách các ứng dụng khác sử dụng nút hành động nổi. Ví dụ:Ứng dụng Gmail sử dụng FAB để tạo một email mới.Ứng dụng Danh bạ sử dụng FAB để thêm một liên hệ mới.Trong nhiệm vụ này, bạn sẽ thay đổi biểu tượng của FloatingActionButton thành giỏ hàng và thay đổi hành động của nó để mở một Activity mới.

**3.1 Thêm một biểu tượng mới**

Như bạn đã học trước đây, Android Studio cung cấp một tập hợp các biểu tượng sẵn có. Thực hiện theo các bước sau để thêm biểu tượng giỏ hàng:

1. Mở **Project > Android pane**, mở rộng thư mục res, sau đó **nhấp chuột phải** (Control + click trên Mac) vào thư mục **drawable**.
2. Chọn **New > Image Asset**. Hộp thoại **Configure Image Asset** sẽ xuất hiện.
3. Trong danh sách **Asset Type**, chọn **Action Bar and Tab Icons**. (Lưu ý: **Action bar** chính là **app bar**).
4. Trong trường **Name**, thay đổi tên từ ic\_action\_name thành **ic\_shopping\_cart**.
5. Nhấp vào hình **clip art** (biểu tượng Android bên cạnh **Clipart:**) để chọn một biểu tượng từ thư viện.
6. Trong danh sách biểu tượng xuất hiện, tìm và chọn biểu tượng **giỏ hàng (shopping cart)**.
7. Trong mục **Theme**, chọn **HOLO\_DARK**. Cài đặt này sẽ giúp biểu tượng hiển thị màu trắng trên nền tối hoặc đen.
8. Nhấp **Next**, sau đó nhấp **Finish** trong hộp thoại **Confirm Icon Path**.

**3.2 Thêm một Activity mới**

Như bạn đã học trước đó, **Activity** đại diện cho một màn hình trong ứng dụng, nơi người dùng có thể thực hiện một tác vụ cụ thể. Hiện tại, ứng dụng chỉ có một Activity là MainActivity.java. Trong bước này, bạn sẽ thêm một **Activity mới** có tên OrderActivity.java.

1. **Các bước thực hiện:**
2. Trong cột bên trái, **nhấp chuột phải** (Control + click trên Mac) vào thư mục com.example.android.droidcafe.
3. Chọn **New > Activity > Empty Activity**.
4. Đặt tên **Activity Name** là **OrderActivity**, và **Layout Name** là **activity\_order**.
5. Giữ nguyên các tùy chọn khác và nhấp **Finish**.

Sau khi hoàn thành, bạn sẽ thấy:

* **OrderActivity.java** xuất hiện trong thư mục **java**.
* **activity\_order.xml** xuất hiện trong thư mục **layout**.
* Android Studio đã tự động tạo các file này dựa trên mẫu **Empty Activity**.

**3.3 Thay đổi hành động của FloatingActionButton**

Trong bước này, bạn sẽ sửa đổi hành động của **FloatingActionButton** để mở OrderActivity.

**Các bước thực hiện:**

1. Mở **MainActivity.java**.
2. Tìm phương thức onClick(View view), sau đó thay đổi nó để tạo một **explicit intent** để khởi chạy OrderActivity:

@Override

public void onClick(View view) {

Intent intent = new Intent(MainActivity.this, OrderActivity.class);

startActivity(intent);

}

1. Chạy ứng dụng và kiểm tra kết quả. Chạy ứng dụng trong Android Studio bằng cách nhấn Run (hoặc Shift + F10).Nhấn vào nút Floating Action Button (hiển thị biểu tượng giỏ hàng 🛒).Màn hình mới (OrderActivity) sẽ mở ra (hiện tại đang trống).Nhấn nút Back trên thiết bị hoặc trình giả lập để quay lại MainActivity.

**Thử thách lập trình**

Bạn đã học trong một bài trước về cách gửi dữ liệu từ một Activity sang một Activity khác. Hãy thay đổi ứng dụng để gửi thông điệp đơn hàng của món tráng miệng đã chọn từ MainActivity đến một TextView mới nằm ở đầu giao diện của OrderActivity.

* **Thêm một TextView** vào phần đầu của giao diện OrderActivity với id là order\_textview.
* **Tạo một biến thành viên (mOrderMessage)** trong MainActivity để lưu thông điệp đơn hàng sẽ xuất hiện trong Toast.
* **Chỉnh sửa các phương thức xử lý click** (showDonutOrder(), showIceCreamOrder(), showFroyoOrder()) để gán giá trị cho mOrderMessage trước khi hiển thị Toast:

mOrderMessage = getString(R.string.donut\_order\_message);

displayToast(mOrderMessage);

* **Thêm một hằng số public static final String** có tên EXTRA\_MESSAGE vào đầu MainActivity để làm khóa cho intent.putExtra:

public static final String EXTRA\_MESSAGE =

"com.example.android.droidcafe.extra.MESSAGE";

* **Chỉnh sửa phương thức onClick()** để thêm intent.putExtra trước khi khởi chạy OrderActivity.

public void onClick(View view) {

Intent intent =

new Intent(MainActivity.this, OrderActivity.class);

intent.putExtra(EXTRA\_MESSAGE, mOrderMessage);

startActivity(intent);

}

* **Trong OrderActivity**, cập nhật phương thức onCreate() để nhận Intent, lấy dữ liệu từ nó và hiển thị thông điệp trong TextView.

Intent intent = getIntent();

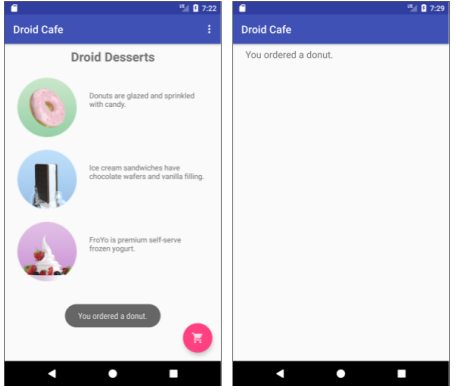
String message = "Order: " +

intent.getStringExtra(MainActivity.EXTRA\_MESSAGE);

TextView textView = findViewById(R.id.order\_textview);

textView.setText(message);

* **Chạy ứng dụng**. Sau khi chọn một hình ảnh món tráng miệng, nhấn nút nổi (FloatingActionButton) để mở OrderActivity, nơi thông điệp đơn hàng sẽ hiển thị trên màn hình.



* **Tóm tắt**
* Để sử dụng hình ảnh trong dự án, hãy sao chép hình ảnh vào thư mục drawable của dự án (project\_name > app > src > main > res > drawable).
* Định nghĩa một ImageView bằng cách kéo ImageView vào giao diện và chọn hình ảnh cho nó.
* Thêm thuộc tính android:onClick để làm cho ImageView có thể nhấp giống như một nút và chỉ định tên của trình xử lý sự kiện nhấp.
* Tạo một trình xử lý sự kiện nhấp (click handler) trong Activity để thực hiện hành động mong muốn.
* Chọn một biểu tượng: Mở rộng thư mục res trong bảng điều hướng Project > Android, nhấp chuột phải vào thư mục drawable, chọn **New > Image Asset**, chọn **Action Bar and Tab Icons**, sau đó nhấp vào biểu tượng Clipart (biểu tượng Android) để chọn hình ảnh làm biểu tượng.
* Thêm một Activity mới: Trong bảng điều hướng Project > Android, nhấp chuột phải vào thư mục chứa tên gói trong thư mục java, chọn **New > Activity**, sau đó chọn một mẫu Activity (chẳng hạn như Empty Activity).
* Hiển thị thông báo Toast.

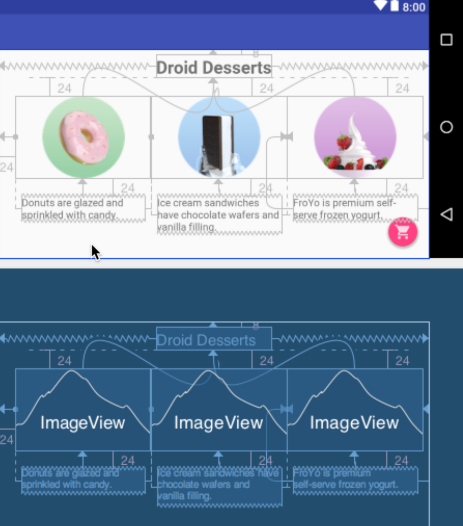
Toast.makeText(getApplicationContext(), message,

Toast.LENGTH\_SHORT).show();

**Bài tập về nhà: Thay đổi ứng dụng**

Ứng dụng **DroidCafe** hoạt động tốt khi thiết bị hoặc trình giả lập được đặt theo chiều dọc. Tuy nhiên, nếu chuyển sang chế độ ngang, hai hình ảnh cuối cùng không xuất hiện.

1. Mở (hoặc tải về) dự án ứng dụng **DroidCafe**.
2. Tạo một biến thể bố cục cho chế độ ngang: **content\_main.xml (land)**.
3. Xóa tất cả ràng buộc (constraints) khỏi ba hình ảnh và ba mô tả văn bản.
4. Chọn cả ba hình ảnh trong biến thể bố cục và nhấp vào nút **Pack**, sau đó chọn **Expand Horizontally** để phân bố đều các hình ảnh trên màn hình (như hình minh họa).
5. Ràng buộc các mô tả văn bản vào các cạnh và đáy của hình ảnh tương ứng (như hình minh họa).



**Trả lời các Câu hỏi sau**

**Câu 1**

Làm cách nào để thêm hình ảnh vào một dự án trong Android Studio? Chọn một:

- Kéo từng hình ảnh vào trình chỉnh sửa bố cục.

- Sao chép các tệp hình ảnh vào thư mục drawable của dự án.

- Kéo một ImageButton vào trình chỉnh sửa bố cục.

- Chọn \*\*New > Image Asset\*\*, sau đó chọn tệp hình ảnh.

**Câu 2**

Làm thế nào để làm cho một ImageView có thể nhấp vào giống như một Button? Chọn một:

- Thêm thuộc tính android:contentDescription vào ImageView trong bố cục và sử dụng nó để gọi trình xử lý sự kiện trong Activity.

- Thêm thuộc tính android:src vào ImageView trong bố cục và sử dụng nó để gọi trình xử lý sự kiện trong Activity.

- Thêm thuộc tính android:onClick vào ImageView trong bố cục và sử dụng nó để gọi trình xử lý sự kiện trong Activity.

- Thêm thuộc tính android:id vào ImageView trong bố cục và sử dụng nó để gọi trình xử lý sự kiện trong Activity.

Câu 3

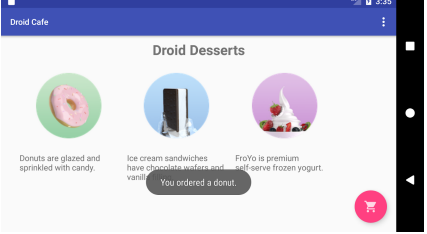
Quy tắc nào áp dụng cho một click handler được gọi từ thuộc tính trong bố cục? Chọn một:

- Phương thức click handler phải bao gồm event listener View.OnClickListener, một giao diện trong lớp View.

- Phương thức click handler phải là public, trả về kiểu void và nhận một tham số duy nhất là View.

- Trình xử lý sự kiện nhấp chuột phải tùy chỉnh lớp View.OnClickListener và ghi đè phương thức xử lý sự kiện để thực hiện một hành động nào đó.

- Phương thức click handler phải là private và trả về một View.



### Các điều khiển nhập liệu

**Giới thiệu**

Để cho phép người dùng nhập văn bản hoặc số, bạn sử dụng phần tử **EditText**. Một số điều khiển nhập liệu là các thuộc tính của **EditText**, giúp xác định loại bàn phím xuất hiện để làm cho việc nhập dữ liệu dễ dàng hơn. Ví dụ, bạn có thể chọn **phone** cho thuộc tính **android:inputType** để hiển thị bàn phím số thay vì bàn phím chữ và số.

Các điều khiển nhập khác giúp người dùng dễ dàng đưa ra lựa chọn. Ví dụ, các phần tử **RadioButton** cho phép người dùng chọn một (và chỉ một) mục từ một tập hợp các tùy chọn.

Trong bài thực hành này, bạn sẽ sử dụng các thuộc tính để kiểm soát giao diện bàn phím trên màn hình và đặt loại dữ liệu nhập cho **EditText**. Bạn cũng sẽ thêm các nút radio vào ứng dụng **DroidCafe** để người dùng có thể chọn một mục từ danh sách.

**Những gì bạn cần biết trước**

Bạn nên có khả năng:

* Tạo một dự án Android Studio từ mẫu và tạo bố cục chính.
* Chạy ứng dụng trên trình giả lập hoặc thiết bị được kết nối.
* Tạo và chỉnh sửa các phần tử UI bằng trình chỉnh sửa bố cục và mã XML.
* Truy cập các phần tử UI từ mã bằng **findViewById()**.
* Chuyển đổi văn bản trong **View** thành chuỗi bằng **getText()**.
* Tạo trình xử lý sự kiện khi người dùng nhấp vào **Button**.
* Hiển thị thông báo **Toast**.

**Những gì bạn sẽ học**

* Cách thay đổi phương thức nhập để bật gợi ý, tự động viết hoa và ẩn mật khẩu.
* Cách thay đổi bàn phím trên màn hình chung thành bàn phím số hoặc các bàn phím chuyên biệt khác.
* Cách thêm **radio buttons** để người dùng chọn một tùy chọn.
* Cách thêm **spinner** để hiển thị menu thả xuống với các giá trị mà người dùng có thể chọn.

**Những gì bạn sẽ làm**

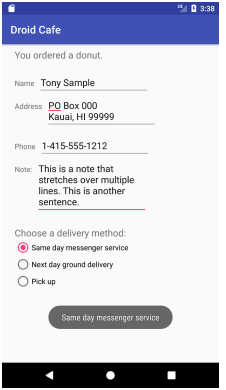
* Hiển thị bàn phím để nhập địa chỉ email.
* Hiển thị bàn phím số để nhập số điện thoại.
* Cho phép nhập văn bản nhiều dòng với tự động viết hoa câu.
* Thêm các nút radio để chọn một tùy chọn.
* Thiết lập trình xử lý sự kiện **onClick** cho các nút radio.
* Thêm **spinner** vào trường số điện thoại để chọn một giá trị từ danh sách.

**Tổng quan về ứng dụng**

Trong bài thực hành này, bạn sẽ thêm nhiều tính năng hơn vào ứng dụng DroidCafe, dựa trên bài học về cách sử dụng hình ảnh có thể nhấp.

Trong OrderActivity, bạn sẽ thử nghiệm với thuộc tính android:inputType cho các phần tử EditText. Bạn sẽ thêm các EditText để nhập tên và địa chỉ của người dùng, đồng thời sử dụng các thuộc tính để xác định các trường nhập một dòng hoặc nhiều dòng, giúp hiển thị gợi ý khi nhập văn bản. Ngoài ra, bạn cũng sẽ thêm một EditText hiển thị bàn phím số để nhập số điện thoại.

Các loại điều khiển nhập khác bao gồm các phần tử tương tác giúp người dùng đưa ra lựa chọn. Bạn sẽ thêm radio buttons vào ứng dụng DroidCafe để chọn một trong số các tùy chọn giao hàng. Bên cạnh đó, bạn cũng sẽ cung cấp một spinner để người dùng chọn nhãn cho số điện thoại (Home, Work, Other, Custom).



**Tác vụ 1: Thử nghiệm với các thuộc tính nhập văn bản**

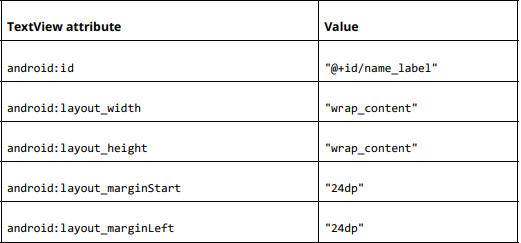
Khi chạm vào một trường văn bản có thể chỉnh sửa (**EditText**), con trỏ sẽ xuất hiện trong trường đó và bàn phím ảo sẽ tự động hiển thị để người dùng nhập dữ liệu.

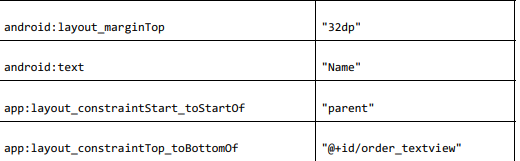
Mỗi trường **EditText** yêu cầu một loại dữ liệu đầu vào cụ thể, chẳng hạn như văn bản thuần túy, địa chỉ email, số điện thoại hoặc mật khẩu. Việc xác định loại đầu vào cho từng trường là rất quan trọng để hệ thống hiển thị phương thức nhập phù hợp, ví dụ như bàn phím chữ cái cho văn bản hoặc bàn phím số cho số điện thoại.

**1.1 Thêm một EditText để nhập tên**

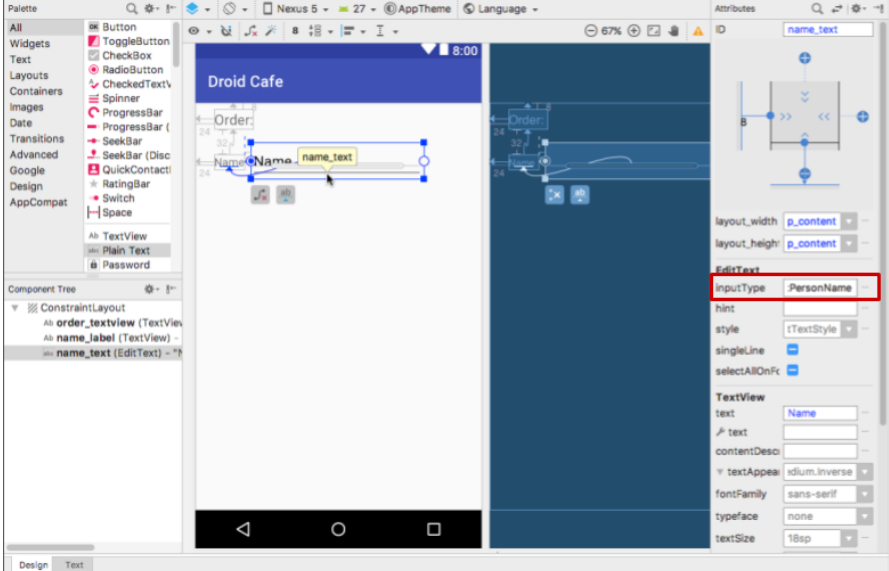
Ở bước này, bạn sẽ thêm một **TextView** và một **EditText** vào bố cục **OrderActivity** trong ứng dụng **DroidCafe** để người dùng có thể nhập tên.

1. Sao chép ứng dụng **DroidCafe** từ bài học về hình ảnh có thể nhấp và đổi tên bản sao thành **DroidCafeInput**. Nếu bạn chưa hoàn thành thử thách lập trình trong bài học đó, hãy tải xuống dự án **DroidCafeChallenge** và đổi tên thành **DroidCafeInput**.
2. Mở tệp bố cục **activity\_order.xml**, tệp này sử dụng **ConstraintLayout**.
3. Thêm một **TextView** vào **ConstraintLayout** trong **activity\_order.xml**, đặt bên dưới phần tử **order\_textview** đã có sẵn. Sử dụng các thuộc tính sau cho **TextView** mới:

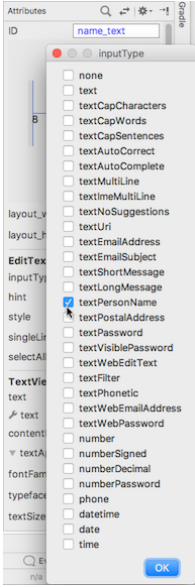




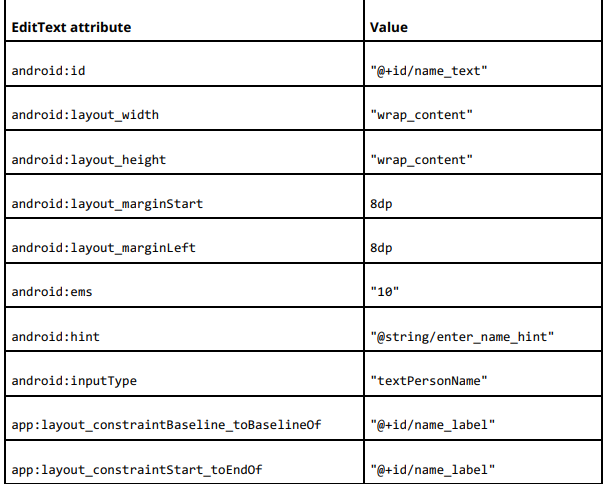
1. Trích xuất chuỗi cho thuộc tính `android:text` của TextView vào strings.xml, sau đó thêm EditText có ID `name\_text`, đặt cạnh TextView, căn chỉnh baseline và ràng buộc cạnh trái với cạnh phải của TextView.
2. Thêm một **EditText** bằng cách kéo một **Plain Text** từ **Palette** vào cạnh **TextView** name\_label, đặt ID là name\_text, và ràng buộc cạnh trái cùng baseline của nó với cạnh phải và baseline của name\_label.



1. Hình trên làm nổi bật trường **inputType** trong bảng **Attributes**, cho thấy Android Studio tự động gán kiểu textPersonName. Nhấp vào trường **inputType** để xem các tùy chọn khác.

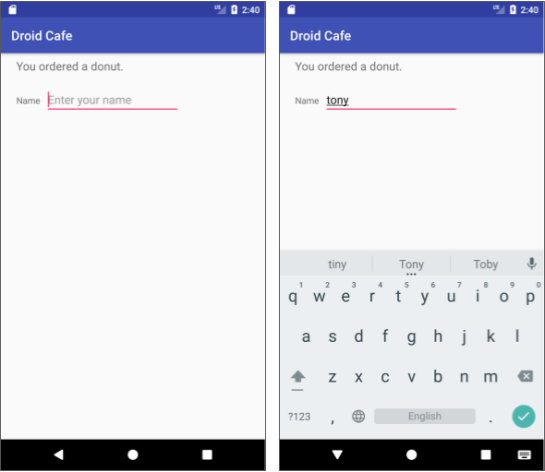


1. Thêm gợi ý nhập văn bản, chẳng hạn như **"Enter your name"**, vào trường **hint** trong bảng **Attributes**, rồi xóa nội dung trong trường **text**. Văn bản gợi ý sẽ hiển thị mờ bên trong **EditText**.
2. Mở tab **Text** để kiểm tra mã XML của layout, sau đó trích xuất chuỗi tài nguyên cho thuộc tính android:hint với giá trị **enter\_name\_hint**. Đảm bảo các thuộc tính cần thiết được thiết lập cho **EditText**, bao gồm cả layout\_marginLeft để tương thích với các phiên bản Android cũ hơn.



Như bạn có thể thấy trong mã XML, thuộc tính android:inputType được đặt thành **textPersonName**.

1. Chạy ứng dụng, nhấn vào hình bánh donut trên màn hình đầu tiên, sau đó nhấn nút **floating action button** để chuyển sang **Activity** tiếp theo. Nhấn vào ô nhập văn bản để hiển thị bàn phím và nhập văn bản, như minh họa trong hình dưới đây.



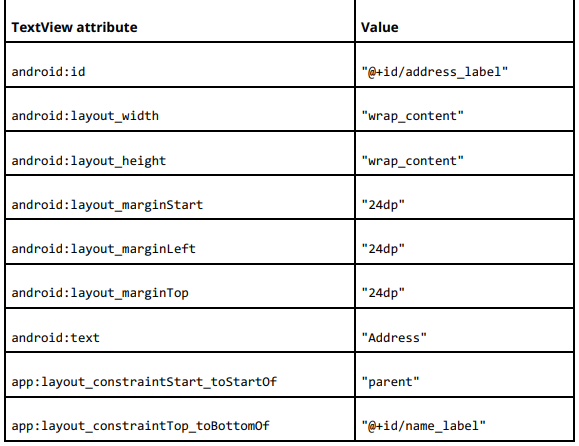
Lưu ý rằng các gợi ý tự động xuất hiện cho các từ bạn nhập. Nhấn vào một gợi ý để sử dụng nó. Đây là một trong những đặc tính của giá trị **textPersonName** cho thuộc tính **android:inputType**. Thuộc tính **inputType** kiểm soát nhiều tính năng khác nhau, bao gồm bố cục bàn phím, viết hoa và hỗ trợ nhiều dòng văn bản.

1. Để đóng bàn phím, hãy nhấn vào biểu tượng dấu kiểm trong vòng tròn màu xanh lá, xuất hiện ở góc dưới bên phải của bàn phím. Đây được gọi là phím **Done**.

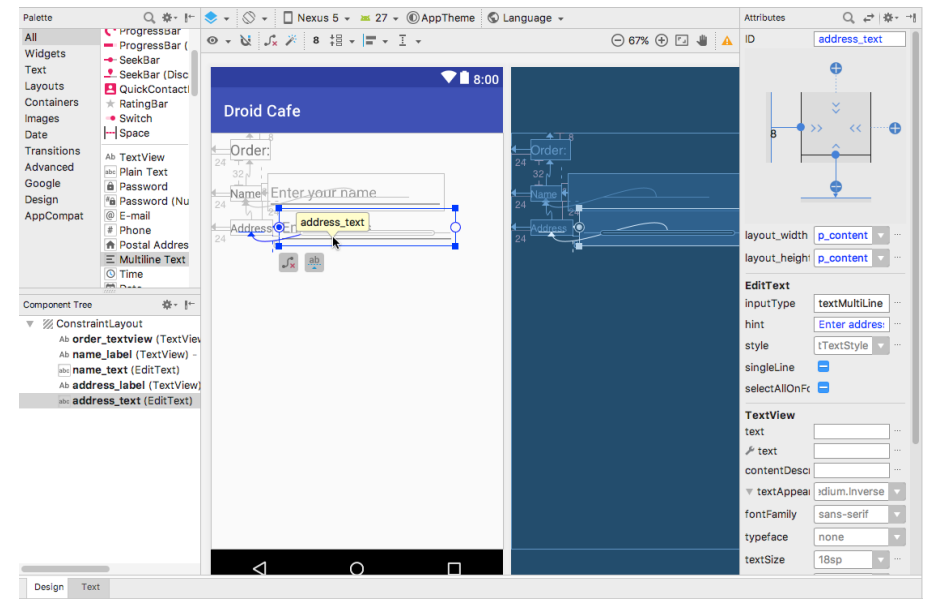
**1.2 Thêm một EditText nhiều dòng**

Trong bước này, bạn sẽ thêm một **EditText** khác vào **OrderActivity** trong ứng dụng **DroidCafe** để người dùng có thể nhập địa chỉ với nhiều dòng.

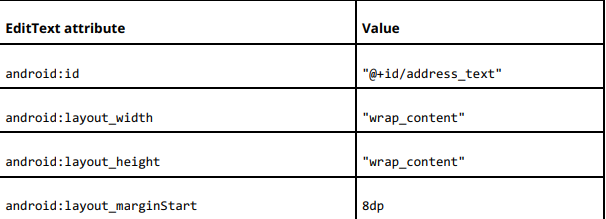
1. Mở tệp **activity\_order.xml** nếu chưa mở.
2. Thêm một **TextView** bên dưới phần tử **name\_label** đã có trong bố cục. Sử dụng các thuộc tính sau cho **TextView** mới:

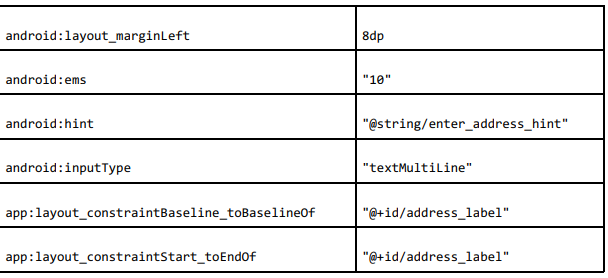


1. Trích xuất giá trị thuộc tính **android:text** thành một **string resource** với tên **address\_label\_text** trong **strings.xml**.
2. Thêm một phần tử **EditText**: Để sử dụng trình chỉnh sửa bố cục trực quan, kéo một phần tử **Multiline Text** từ **Palette** vào vị trí bên cạnh **address\_label TextView**. Sau đó, đặt **ID** là **address\_text**, và ràng buộc cạnh trái và đường cơ sở của phần tử này với cạnh phải và đường cơ sở của **address\_label**, như trong hình minh họa.

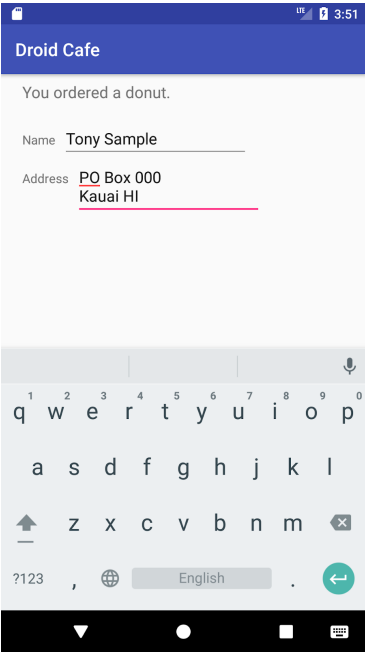


1. Thêm gợi ý nhập văn bản, chẳng hạn như **Enter address**, vào trường **hint** trong **Attributes pane**. Văn bản **"Enter address"** sẽ hiển thị mờ bên trong **EditText** như một gợi ý cho người dùng.
2. Kiểm tra mã XML của bố cục bằng cách nhấn vào tab **Text**. Trích xuất giá trị thuộc tính **android:hint** thành một **string resource** với tên **enter\_address\_hint**. Các thuộc tính sau nên được thiết lập cho **EditText** mới (thêm **layout\_marginLeft** để tương thích với các phiên bản Android cũ hơn).





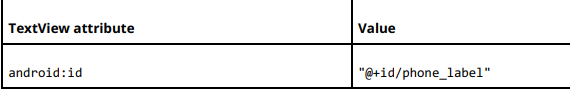
1. Chạy ứng dụng. Nhấn vào một hình ảnh trên màn hình đầu tiên, sau đó nhấn vào nút **Floating Action Button** để chuyển đến **Activity** tiếp theo.
2. Nhấn vào trường nhập văn bản **"Address"** để hiển thị bàn phím và nhập văn bản như trong hình minh họa bên dưới. Sử dụng phím **Return** (còn gọi là **Enter** hoặc **New Line**) ở góc dưới bên phải bàn phím để xuống dòng. Phím **Return** sẽ xuất hiện nếu bạn đặt thuộc tính **android:inputType** thành **textMultiLine**.

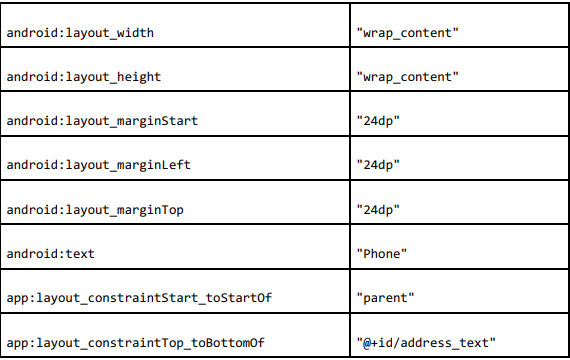


1. Để đóng bàn phím, nhấn vào nút mũi tên xuống xuất hiện thay cho nút **Back** trong hàng nút dưới cùng.

1.3 Sử dụng bàn phím số cho số điện thoại  
Trong bước này, bạn thêm một **EditText** vào bố cục **OrderActivity** trong ứng dụng **DroidCafe** để người dùng có thể nhập số điện thoại bằng bàn phím số.

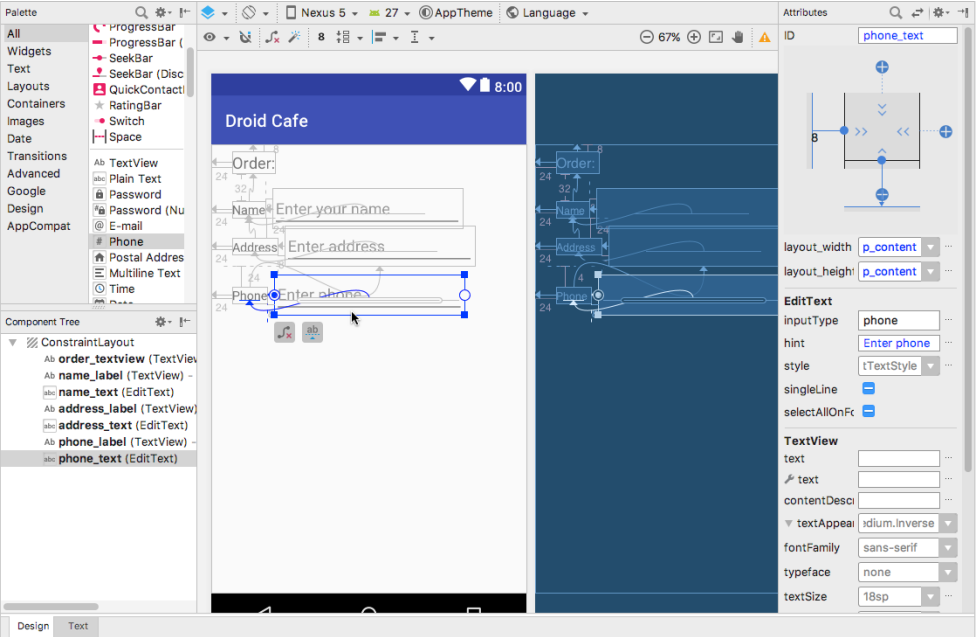
1. Mở tệp **activity\_order.xml** nếu chưa mở.
2. Thêm một **TextView** bên dưới phần tử **address\_label** đã có trong bố cục. Sử dụng các thuộc tính sau cho **TextView** mới:



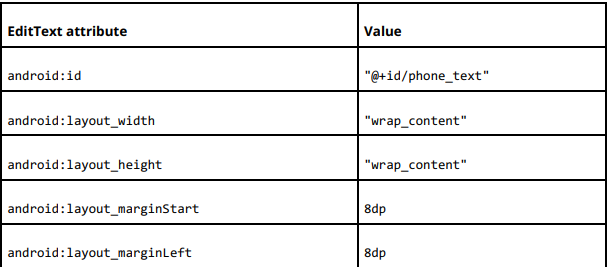


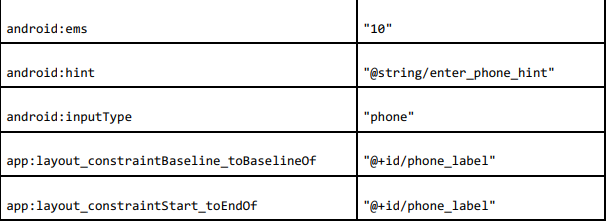
Lưu ý rằng **TextView** này được ràng buộc vào phía dưới của **EditText nhiều dòng** (**address\_text**). Điều này là do **address\_text** có thể mở rộng thành nhiều dòng, và **TextView** này cần xuất hiện bên dưới nó.

1. Trích xuất giá trị chuỗi cho thuộc tính **android:text** để tạo một mục mới trong **strings.xml** với tên **phone\_label\_text**.
2. Thêm một phần tử **EditText**. Để sử dụng trình chỉnh sửa bố cục trực quan, kéo một phần tử **Phone** từ **Palette** vào vị trí bên cạnh **TextView phone\_label**. Sau đó đặt **phone\_text** làm **ID**, và ràng buộc cạnh trái và đường cơ sở của phần tử này với cạnh phải và đường cơ sở của **phone\_label**, như minh họa trong hình dưới đây.

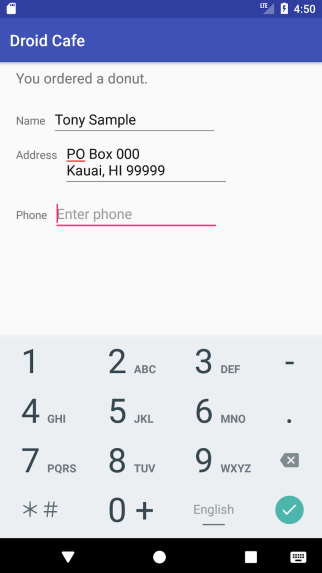


1. Thêm gợi ý nhập văn bản, chẳng hạn như **Enter phone**, vào trường **hint** trong **Attributes pane**. Gợi ý này sẽ xuất hiện dưới dạng văn bản mờ bên trong **EditText**.
2. Kiểm tra mã XML của bố cục bằng cách nhấp vào tab **Text**. Trích xuất giá trị chuỗi cho thuộc tính **android:hint** vào **strings.xml** với tên **enter\_phone\_hint**. Các thuộc tính sau đây nên được đặt cho **EditText** mới (thêm **layout\_marginLeft** để đảm bảo tương thích với các phiên bản Android cũ hơn).





1. Chạy ứng dụng. Nhấn vào một hình ảnh trên màn hình đầu tiên, sau đó nhấn vào nút hành động nổi để mở **OrderActivity**.
2. Nhấn vào trường **"Phone"** để hiển thị bàn phím số. Bạn có thể nhập số điện thoại như trong hình minh họa bên dưới.



1. Để đóng bàn phím, nhấn vào phím **Done**.

Bạn có thể thử nghiệm với thuộc tính **android:inputType** bằng cách thay đổi giá trị của nó trong **EditText** để thấy kết quả:

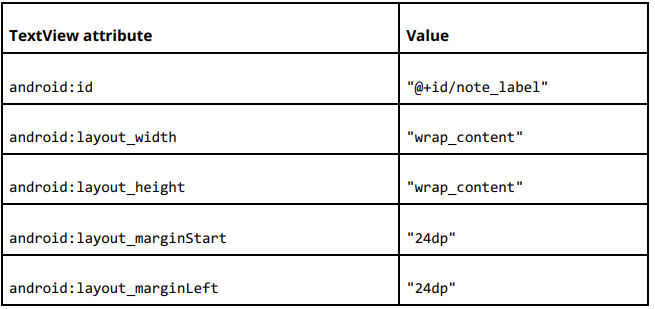
* **textEmailAddress**: Khi nhấn vào trường nhập, bàn phím sẽ hiển thị với ký hiệu **@** gần phím cách.
* **textPassword**: Các ký tự nhập vào sẽ được hiển thị dưới dạng dấu chấm để che mật khẩu.

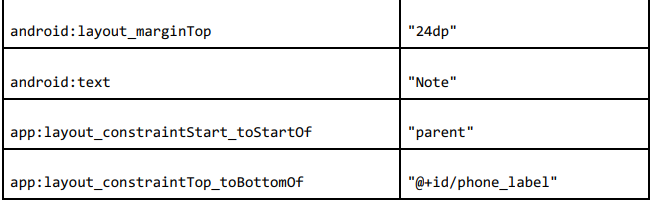
**1.4 Kết hợp nhiều kiểu nhập trong một EditText**

Bạn có thể kết hợp các giá trị của **inputType** nếu chúng không xung đột với nhau. Ví dụ, có thể kết hợp **textMultiLine** và **textCapSentences** để tạo trường nhập nhiều dòng, trong đó mỗi câu bắt đầu bằng chữ hoa.

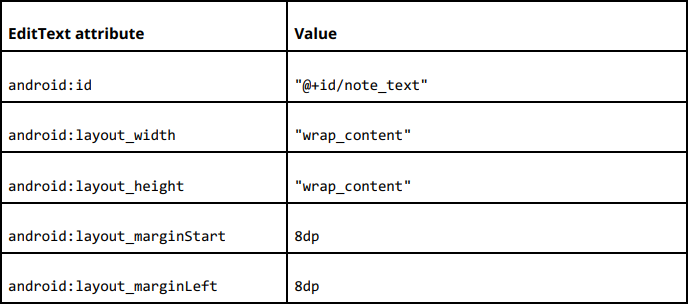
**Các bước thực hiện:**

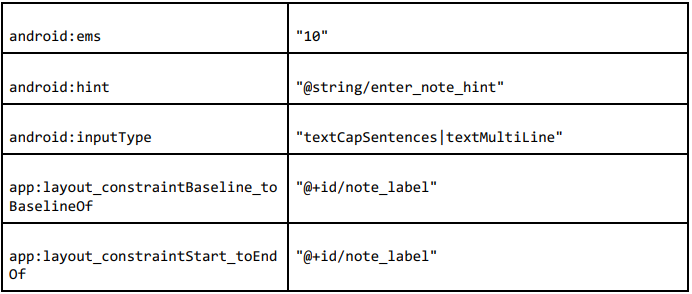
1. Mở tệp **activity\_order.xml** nếu chưa mở.
2. Thêm một **TextView** bên dưới phần tử **phone\_label** trong layout, với các thuộc tính sau:





1. Trích xuất giá trị thuộc tính **android:text** thành một resource trong **strings.xml** với tên **note\_label\_text**.
2. Thêm một **EditText**: Dùng trình chỉnh sửa trực quan, kéo một phần tử **Multiline Text** từ bảng **Palette** vào vị trí bên cạnh **note\_label TextView**. Đặt **ID** là **note\_text** và ràng buộc cạnh trái và baseline của nó với **note\_label**, giống như đã làm với các **EditText** trước đó.
3. Thêm một gợi ý nhập liệu, chẳng hạn **Enter note**, vào trường **hint** trong bảng **Attributes**.
4. Nhấn vào trường **inputType** trong bảng **Attributes**. Giá trị **textMultiLine** đã được chọn sẵn. Ngoài ra, chọn thêm **textCapSentences** để kết hợp hai thuộc tính này.
5. Kiểm tra mã XML của layout bằng cách nhấp vào tab **Text**. Trích xuất giá trị thuộc tính **android:hint** thành resource **enter\_note\_hint**. Các thuộc tính sau nên được thiết lập cho **EditText** mới (thêm **layout\_marginLeft** để đảm bảo tương thích với các phiên bản Android cũ hơn):

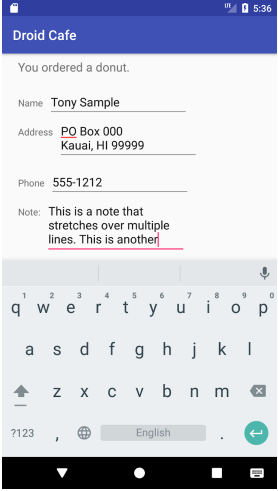




Để kết hợp nhiều giá trị cho thuộc tính **android:inputType**, hãy nối chúng bằng ký tự **|** (pipe).

1. Chạy ứng dụng. Nhấn vào một hình ảnh trên màn hình đầu tiên, sau đó nhấn nút **Floating Action Button** để chuyển sang **Activity** tiếp theo.

9. Nhấn vào ô nhập **"Note"** và nhập các câu hoàn chỉnh như hình minh họa. Sử dụng phím **Return** để tạo dòng mới hoặc chỉ cần tiếp tục nhập để văn bản tự xuống dòng.



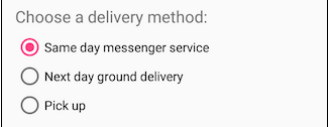
**Nhiệm vụ 2: Sử dụng các nút radio**

Các điều khiển nhập liệu là các phần tử tương tác trong giao diện ứng dụng của bạn cho phép người dùng nhập dữ liệu. Radio Button là một loại điều khiển nhập liệu hữu ích khi bạn muốn người dùng chỉ chọn một tùy chọn trong một tập hợp các tùy chọn.

Trong nhiệm vụ này, bạn sẽ thêm một nhóm các radio button vào ứng dụng DroidCafeInput để thiết lập tùy chọn giao hàng cho đơn hàng tráng miệng.

**2.1 Thêm một RadioGroup và các nút radio**

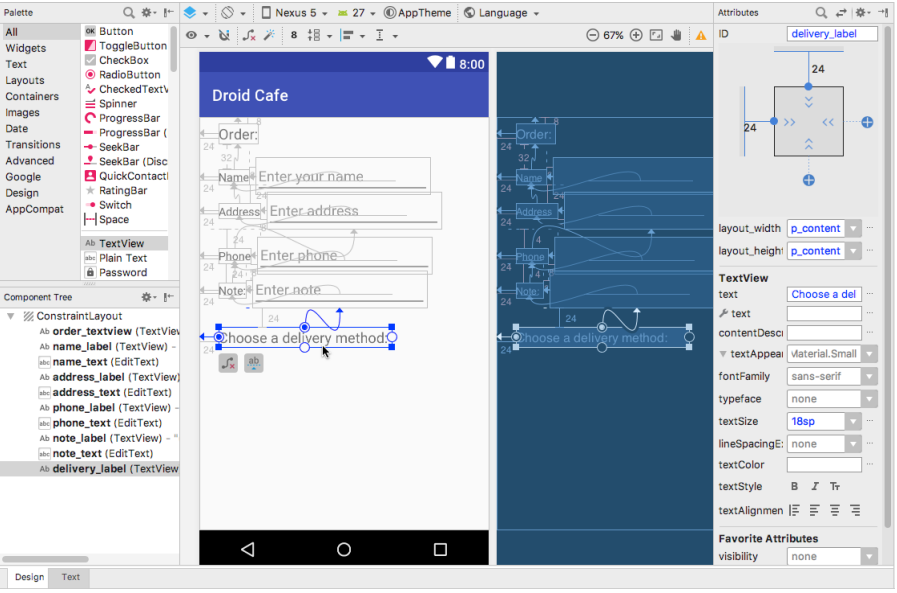
Để thêm radio button vào OrderActivity trong ứng dụng DroidCafeInput, bạn cần tạo các phần tử RadioButton trong tệp activity\_order.xml. Sau khi chỉnh sửa tệp giao diện, bố cục của các nút radio trong OrderActivity sẽ giống như hình minh họa bên dưới.



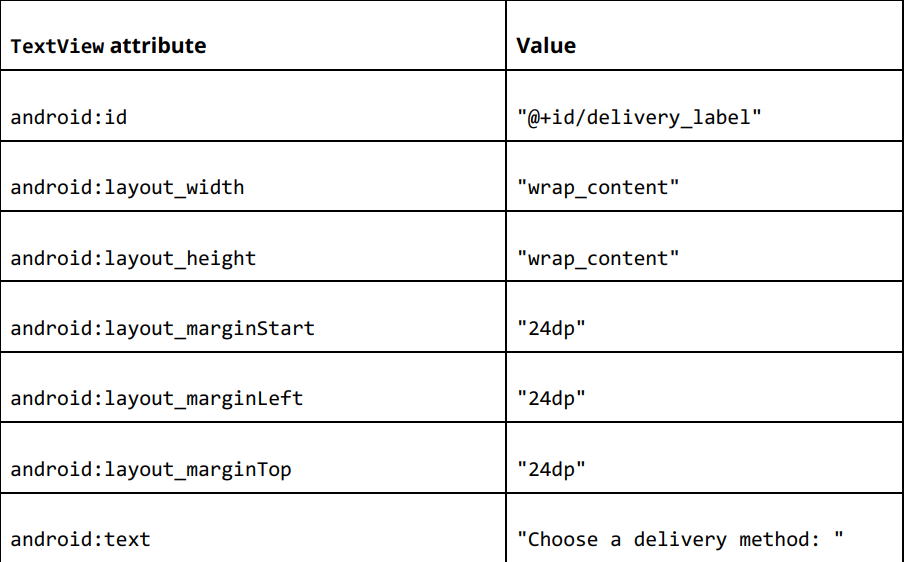
Vì các lựa chọn của **radio button** là loại trừ lẫn nhau, bạn cần nhóm chúng lại bên trong một **RadioGroup**. Việc nhóm lại như vậy giúp hệ thống Android đảm bảo rằng chỉ có một **radio button** được chọn tại một thời điểm.

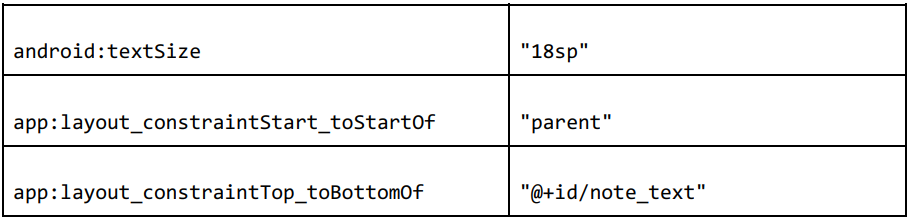
**Lưu ý:** Thứ tự liệt kê các phần tử **RadioButton** trong XML sẽ quyết định thứ tự hiển thị của chúng trên màn hình.

1. Mở **activity\_order.xml**, thêm một phần tử **TextView**, đặt nó bị ràng buộc (constrained) bên dưới phần tử **note\_text** đã có trong bố cục và căn lề trái, như trong hình minh họa bên dưới.



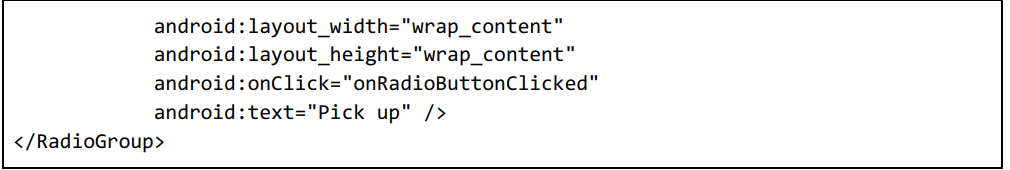
1. Chuyển sang chỉnh sửa XML và đảm bảo rằng **TextView** mới có các thuộc tính sau được thiết lập:





1. Trích xuất chuỗi tài nguyên cho "Choose a delivery method:" và đặt tên là **choose\_delivery\_method**.
2. Để thêm các nút radio, bao bọc chúng bên trong một **RadioGroup**. Thêm **RadioGroup** vào bố cục ngay bên dưới **TextView** vừa thêm, chứa ba phần tử **RadioButton** như trong đoạn mã XML dưới đây:





Thuộc tính **android:onClick** của mỗi **RadioButton** sẽ bị gạch chân màu đỏ cho đến khi bạn thêm phương thức **onRadioButtonClicked** ở bước tiếp theo.

1. Trích xuất ba chuỗi tài nguyên của thuộc tính **android:text** và đặt tên lần lượt là **same\_day\_messenger\_service**, **next\_day\_ground\_delivery**, và **pick\_up** để dễ dàng dịch thuật.

**2.2 Thêm trình xử lý sự kiện cho radio button**

Thuộc tính android:onClick của mỗi radio button xác định phương thức onRadioButtonClicked() để xử lý sự kiện khi người dùng nhấn vào. Vì vậy, bạn cần thêm phương thức onRadioButtonClicked() mới trong lớp OrderActivity.

1. Mở **activity\_order.xml** (nếu chưa mở) và tìm một trong các giá trị onRadioButtonClicked của thuộc tính android:onClick đang bị gạch đỏ.
2. Nhấp vào giá trị onRadioButtonClicked, sau đó nhấp vào biểu tượng bóng đèn đỏ ở lề trái.
3. Chọn **Create onRadioButtonClicked(View) in OrderActivity** từ menu bóng đèn.
   1. Android Studio sẽ tạo phương thức onRadioButtonClicked(View view) trong OrderActivity:

public void onRadioButtonClicked(View view) {

}

1. Để hiển thị radio button nào được chọn (tức là phương thức giao hàng mà người dùng chọn), sử dụng một **Toast message**. Mở OrderActivity và thêm phương thức displayToast:
   1. private void displayToast(String message) {
   2. Toast.makeText(this, message, Toast.LENGTH\_SHORT).show();
   3. }
2. Trong phương thức onRadioButtonClicked(), thêm một khối switch-case để kiểm tra radio button nào được chọn và gọi displayToast() với thông báo phù hợp.Mã nguồn sử dụng phương thức isChecked() của giao diện Checkable, trả về true nếu radio button được chọn.Ngoài ra, phương thức getId() của View được dùng để lấy ID của radio button đã chọn.

public void onRadioButtonClicked(View view) {

// Is the button now checked?

boolean checked = ((RadioButton) view).isChecked();

// Check which radio button was clicked.

switch (view.getId()) {

case R.id.sameday:

if (checked)

// Same day service

displayToast(getString(R.string.same\_day\_messenger\_service));

break;

case R.id.nextday:

if (checked)

// Next day delivery

displayToast(getString(R.string.next\_day\_ground\_delivery));

break;

case R.id.pickup:

if (checked)

// Pick up

displayToast(getString(R.string.pick\_up));

break;

default:

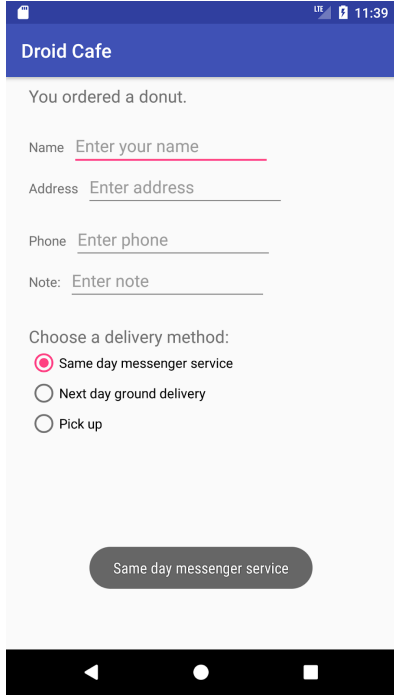
// Do nothing.

break;

}

}

1. Nhấn vào một hình ảnh để mở **OrderActivity**, nơi hiển thị các tùy chọn giao hàng. Khi chọn một tùy chọn giao hàng, một **Toast message** sẽ xuất hiện ở cuối màn hình hiển thị lựa chọn của bạn, như trong hình minh họa.



**Task 3: Sử dụng Spinner cho lựa chọn của người dùng**

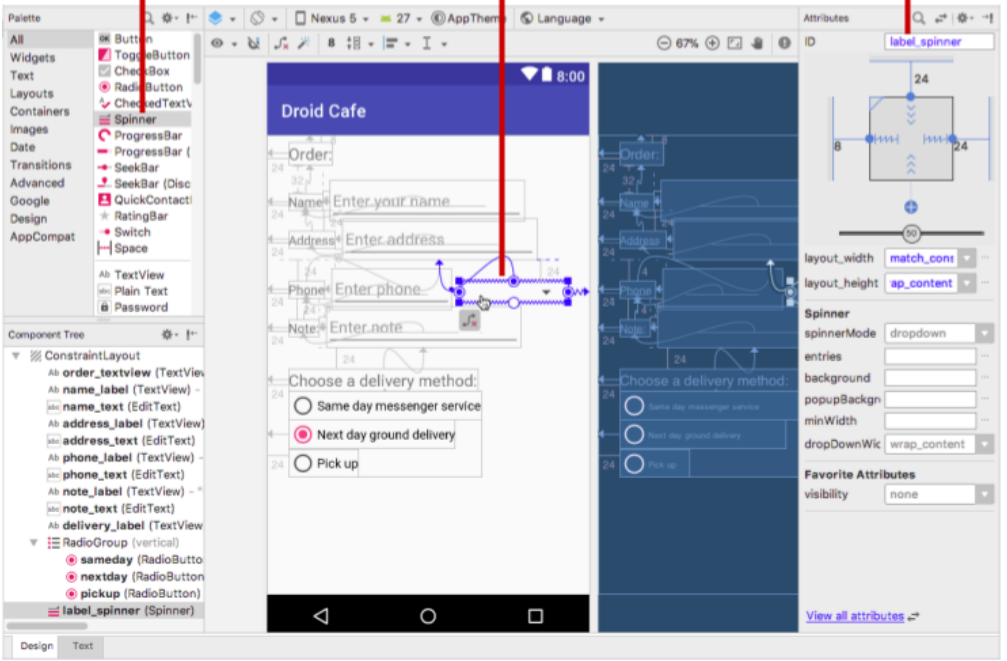
Spinner cung cấp cách nhanh chóng để chọn một giá trị từ một tập hợp. Khi chạm vào Spinner, một danh sách thả xuống hiển thị tất cả các giá trị có sẵn để người dùng chọn. Nếu chỉ có hai hoặc ba lựa chọn, bạn có thể sử dụng radio buttons nếu có đủ không gian trong giao diện. Tuy nhiên, nếu có hơn ba lựa chọn, Spinner là lựa chọn tốt vì nó có thể cuộn để hiển thị các mục và chiếm ít không gian trong giao diện.

**3.1 Thêm Spinner vào giao diện**

Để thêm **Spinner** vào giao diện **OrderActivity** trong ứng dụng **DroidCafe**, thực hiện các bước sau:

1. Mở **activity\_order.xml** và kéo **Spinner** từ bảng **Palette** vào giao diện.
2. Căn chỉnh **Spinner** như sau:
   * Đặt **top** của **Spinner** vào **bottom** của **address\_text**.
   * Đặt **right** của **Spinner** vào **right** của giao diện.
   * Đặt **left** của **Spinner** vào **phone\_text**.
3. Để căn chỉnh ngang **Spinner** và **phone\_text**, chọn cả hai trong **Component Tree**, nhấn nút **Pack**, và chọn **Expand Horizontally** để đặt chiều rộng cố định cho cả hai.
4. Trong bảng **Attributes**, đặt:
   * **ID** của **Spinner** là label\_spinner.
   * **Top margin** và **Right margin** là 24dp, **Left margin** là 8dp.
   * **layout\_width**: match\_constraint.
   * **layout\_height**: wrap\_content.

Sau khi hoàn thành, giao diện sẽ giống như hình minh họa. Bạn cũng có thể thử nghiệm các giá trị khác cho **layout\_width** của **phone\_text** (ví dụ: 134dp).



Xem mã XML của activity\_order.xml

Nhấn vào tab Text trong Android Studio để xem mã XML của giao diện.

Spinner nên có các thuộc tính sau:

<Spinner

android:id="@+id/label\_spinner"

android:layout\_width="0dp"

android:layout\_height="wrap\_content"

android:layout\_marginEnd="24dp"

android:layout\_marginRight="24dp"

android:layout\_marginStart="8dp"

android:layout\_marginLeft="8dp"

android:layout\_marginTop="24dp"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toEndOf="@+id/phone\_text"

app:layout\_constraintTop\_toBottomOf="@+id/address\_text" />

<EditText

android:id="@+id/phone\_text"

android:layout\_width="134dp"

android:layout\_height="wrap\_content"

android:layout\_marginLeft="8dp"

android:layout\_marginStart="8dp"

android:ems="10"

android:hint="@string/enter\_phone\_hint"

android:inputType="phone"

app:layout\_constraintBaseline\_toBaselineOf="@+id/phone\_label"

app:layout\_constraintStart\_toEndOf="@+id/phone\_label" />

**3.2 Thêm mã để kích hoạt Spinner và trình lắng nghe (Listener)**

Các lựa chọn cho Spinner là các chuỗi tĩnh đã được xác định trước như "Home", "Work", vì vậy bạn có thể sử dụng một mảng chuỗi (string array) được định nghĩa trong strings.xml để chứa các giá trị này.

Để kích hoạt Spinner và trình lắng nghe của nó, bạn cần triển khai AdapterView.OnItemSelectedListener, đồng thời thêm hai phương thức onItemSelected() và onNothingSelected().

1. Mở strings.xml và định nghĩa các giá trị có thể chọn (\*\*Home, Work, Mobile, Other\*\*) dưới dạng mảng chuỗi labels\_array.<string‑array name="labels\_array">

<item>Home</item>

<item>Work</item>

<item>Mobile</item>

<item>Other</item>

</string‑array>

1. Để xác định **callback** xử lý lựa chọn cho **Spinner**, hãy sửa đổi **OrderActivity** để triển khai giao diện **AdapterView.OnItemSelectedListener** như sau:

public class OrderActivity extends AppCompatActivity implements

AdapterView.OnItemSelectedListener {

Khi bạn nhập AdapterView. trong câu lệnh trên, **Android Studio** sẽ tự động nhập **AdapterView** vào tệp mã của bạn.Lý do cần **AdapterView** là vì bạn cần một **Adapter**—cụ thể là **ArrayAdapter**—để gán mảng dữ liệu vào **Spinner**.**Adapter** đóng vai trò kết nối dữ liệu (ở đây là mảng các mục trong **Spinner**) với **Spinner**. Đây là một mẫu lập trình phổ biến trong **Android**, và bạn sẽ tìm hiểu sâu hơn về nó trong các bài thực hành khác.Dòng sau sẽ xuất hiện trong phần **import** của bạn:

import android.widget.AdapterView;

Sau khi nhập OnItemSelectedListener trong câu lệnh trên, đợi vài giây cho biểu tượng **bóng đèn đỏ** xuất hiện ở lề trái.

1. Nhấp vào bóng đèn đỏ và chọn Thực hiện phương thức. Các phương thức onItemSelected() và onNothingSelected(), là các phương thức bắt buộc cho OnItemSelectedListener, sẽ được đánh dấu, và tùy chọn “Chèn @Override” sẽ được chọn. Nhấp vào OK.  
   Bước này tự động thêm các phương thức gọi lại onItemSelected() và onNothingSelected() trống vào cuối lớp OrderActivity. Cả hai phương thức đều sử dụng tham số AdapterView là một ký hiệu đại diện cho kiểu Java, cho phép phương thức đủ linh hoạt để chấp nhận bất kỳ loại AdapterView nào làm đối số.
2. Khởi tạo một Spinner trong phương thức onCreate() bằng cách sử dụng phần tử label\_spinner trong bố cục và thiết lập trình lắng nghe của nó (spinner.setOnItemSelectedListener) trong phương thức onCreate(), như được hiển thị trong đoạn mã sau:

@Override

protected void onCreate(Bundle savedInstanceState) {

// ... Rest of onCreate code ...

// Create the spinner.

Spinner spinner = findViewById(R.id.label\_spinner);

if (spinner != null) {

spinner.setOnItemSelectedListener(this);

}

// Create ArrayAdapter using the string array and default spinner layout.

1. Tiếp tục chỉnh sửa phương thức onCreate(), thêm một câu lệnh tạo ArrayAdapter với mảng chuỗi (labels\_array) bằng cách sử dụng bố cục Spinner do Android cung cấp cho từng mục (layout.simple\_spinner\_item):

// Create ArrayAdapter using the string array and default spinner layout.

ArrayAdapter<CharSequence>adapter=ArrayAdapter.createFromResource(this,R.array.labels\_array, android.R.layout.simple\_spinner\_item); // Specify the layout to use when the list of choices appears.

Bố cục simple\_spinner\_item được sử dụng trong bước này và bố cục simple\_spinner\_dropdown\_item được sử dụng trong bước tiếp theo là các bố cục mặc định được Android cung cấp trong lớp R.layout. Bạn nên sử dụng các bố cục này trừ khi bạn muốn tự định nghĩa bố cục riêng cho các mục trong Spinner và cách hiển thị của nó.

1. Xác định bố cục cho danh sách lựa chọn trong Spinner là simple\_spinner\_dropdown\_item, sau đó áp dụng adapter cho Spinner:

// Specify the layout to use when the list of choices appears.

adapter.setDropDownViewResource

(android.R.layout.simple\_spinner\_dropdown\_item);

// Apply the adapter to the spinner.

if (spinner != null) {

spinner.setAdapter(adapter);

}

// ... End of onCreate code ...

**3.3 Add code to respond to Spinner selections**

When the user selects an item in the Spinner, the Spinner receives an on-item-selected event. To handle this event, you already implemented the AdapterView.OnItemSelectedListener interface in the previous step, adding empty onItemSelected() and onNothingSelected() callback methods.

In this step you fill in the code for the onItemSelected() method to retrieve the selected item in the Spinner, using getItemAtPosition(), and assign the item to the spinnerLabel variable:

1. Add code to the empty onItemSelected() callback method, as shown below, to retrieve the user’s selected item using getItemAtPosition(), and assign it to spinnerLabel. You can also add a call to the displayToast() method you already added to OrderActivity:

public void onItemSelected(AdapterView<?> adapterView, View view, int

i, long l) {

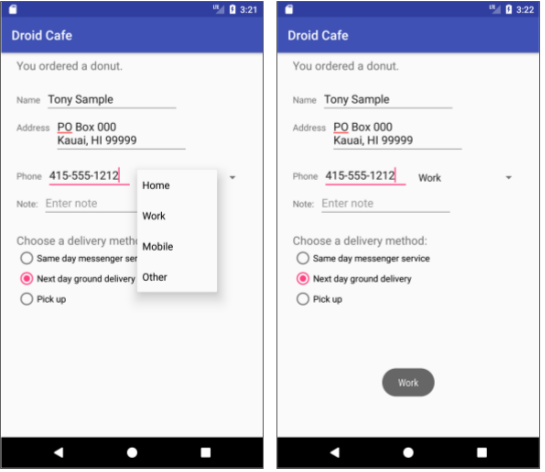
spinnerLabel = adapterView.getItemAtPosition(i).toString();

displayToast(spinnerLabel);

}

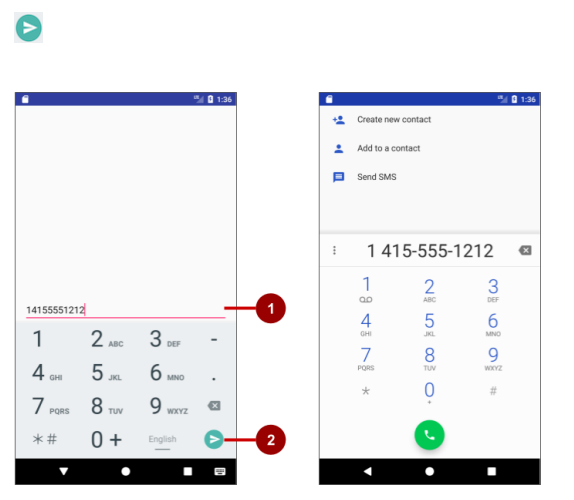
2. Chạy ứng dụng.

`Spinner` xuất hiện bên cạnh trường nhập số điện thoại và hiển thị lựa chọn đầu tiên (`Home`). Khi nhấn vào `Spinner`, tất cả các lựa chọn sẽ hiển thị như hình bên trái trong hình minh họa bên dưới. Khi chọn một mục trong `Spinner`, một thông báo `Toast` sẽ xuất hiện với lựa chọn đó, như hình bên phải.



**Thử thách Coding**

Thử thách: Viết mã để thực hiện một hành động trực tiếp từ bàn phím bằng cách nhấn phím "Gửi" (Send), chẳng hạn như quay số điện thoại.



1. Nhập số điện thoại vào trường EditText.

2. Nhấn phím "Gửi" (`Send`) để mở trình quay số điện thoại. Trình quay số sẽ xuất hiện ở phía bên phải của hình minh họa.

Để thực hiện thử thách này, hãy tạo một dự án ứng dụng mới và thêm một `EditText` với thuộc tính `android:inputType` đặt thành `phone`.

Sử dụng thuộc tính `android:imeOptions` cho phần tử `EditText` với giá trị `actionSend`.

android:imeOptions="actionSend"

Người dùng bây giờ có thể nhấn phím "Gửi" (Send) để quay số điện thoại, như trong hình minh họa ở trên.

Trong phương thức onCreate() của Activity này, bạn có thể sử dụng setOnEditorActionListener() để thiết lập trình lắng nghe cho EditText, nhằm phát hiện khi phím được nhấn:

EditText editText = findViewById(R.id.editText\_main);

if (editText != null)

editText.setOnEditorActionListener(new TextView.OnEditorActionListener() {

// Nếu view được tìm thấy, đặt trình lắng nghe cho editText.

});

Để được hướng dẫn về cách thiết lập trình lắng nghe, hãy xem phần **Chỉ định loại phương thức nhập** (*Specify the input method type*).

Bước tiếp theo là ghi đè phương thức onEditorAction() và sử dụng hằng số IME\_ACTION\_SEND trong lớp EditorInfo để phản hồi khi phím được nhấn. Trong ví dụ dưới đây, phím này được sử dụng để gọi phương thức dialNumber(), giúp quay số điện thoại:

@Override

public boolean onEditorAction(TextView textView, int actionId, KeyEvent keyEvent) {

boolean handled = false;

if (actionId == EditorInfo.IME\_ACTION\_SEND) {

dialNumber();

handled = true;

}

return handled;

}

Để hoàn thành thử thách, hãy tạo phương thức dialNumber(), sử dụng **implicit intent** với ACTION\_DIAL để chuyển số điện thoại đến một ứng dụng khác có thể thực hiện cuộc gọi. Phương thức này có thể trông như sau:

private void dialNumber() {

// Find the editText\_main view.

EditText editText = findViewById(R.id.editText\_main);

String phoneNum = null;

// If the editText field is not null,

// concatenate "tel: " with the phone number string.

if (editText != null) phoneNum = "tel:" +

editText.getText().toString();

// Optional: Log the concatenated phone number for dialing.

Log.d(TAG, "dialNumber: " + phoneNum);

// Specify the intent.

Intent intent = new Intent(Intent.ACTION\_DIAL);

// Set the data for the intent as the phone number.

intent.setData(Uri.parse(phoneNum));

// If the intent resolves to a package (app),

// start the activity with the intent.

if (intent.resolveActivity(getPackageManager()) != null) {

startActivity(intent);

} else {

Log.d("ImplicitIntents", "Can't handle this!");

}

}

**Tóm tắt**

Các giá trị thuộc tính android:inputType ảnh hưởng đến giao diện bàn phím trên màn hình:

* **textAutoCorrect**: Gợi ý sửa lỗi chính tả.
* **textCapSentences**: Viết hoa chữ cái đầu tiên của mỗi câu.
* **textPersonName**: Hiển thị một dòng văn bản với gợi ý khi người dùng nhập, kèm theo phím **Done** để kết thúc.
* **textMultiLine**: Cho phép nhập nhiều dòng văn bản và có phím **Return** để xuống dòng.
* **textPassword**: Ẩn mật khẩu khi nhập.
* **textEmailAddress**: Hiển thị bàn phím nhập email thay vì bàn phím tiêu chuẩn.
* **phone**: Hiển thị bàn phím số điện thoại thay vì bàn phím tiêu chuẩn.

Bạn có thể thiết lập android:inputType trong tệp XML của **EditText**. Để kết hợp nhiều giá trị, sử dụng dấu gạch đứng (|).

* **Radio Button**

Radio Button là điều khiển đầu vào giúp chọn một tùy chọn duy nhất trong một tập hợp:

* Nhóm các **RadioButton** trong một **RadioGroup** để đảm bảo chỉ có một nút được chọn tại một thời điểm.
* Thứ tự khai báo các **RadioButton** trong nhóm quyết định thứ tự hiển thị trên màn hình.
* Sử dụng thuộc tính android:onClick để chỉ định phương thức xử lý sự kiện khi nhấn nút.
* Để kiểm tra xem một nút có được chọn hay không, sử dụng phương thức isChecked() từ giao diện **Checkable**, nếu được chọn sẽ trả về true.
* **Spinner**

Spinner cung cấp một menu thả xuống:

* Thêm **Spinner** vào giao diện.
* Sử dụng **ArrayAdapter** để gán một mảng chuỗi làm mục chọn cho **Spinner**.
* Cài đặt giao diện **AdapterView.OnItemSelectedListener**, trong đó cần triển khai hai phương thức callback onItemSelected() và onNothingSelected() để kích hoạt **Spinner** và bộ lắng nghe của nó.
* Dùng phương thức onItemSelected() để lấy giá trị mục đã chọn trong **Spinner**, sử dụng getItemAtPosition().

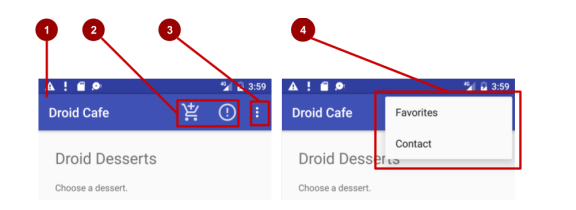
### Menu và bộ chọn

**Giới thiệu**

Thanh ứng dụng (app bar, còn gọi là action bar) là một không gian chuyên dụng nằm ở đầu mỗi màn hình của Activity. Khi bạn tạo một Activity từ mẫu Basic Activity, Android Studio sẽ tự động thêm một thanh ứng dụng.

Menu tùy chọn trong thanh ứng dụng thường cung cấp các lựa chọn để điều hướng, chẳng hạn như chuyển đến một Activity khác trong ứng dụng. Menu cũng có thể chứa các tùy chọn ảnh hưởng đến cách sử dụng ứng dụng, chẳng hạn như thay đổi cài đặt hoặc thông tin hồ sơ, thường được thực hiện trong một Activity riêng biệt.

Trong bài thực hành này, bạn sẽ tìm hiểu về cách thiết lập thanh ứng dụng và menu tùy chọn trong ứng dụng của mình, như minh họa trong hình bên dưới.



**Trong hình trên:**

1. **Thanh ứng dụng (App bar):** Gồm tiêu đề ứng dụng, menu tùy chọn và nút tràn (overflow button).
2. **Menu tùy chọn:** Các mục menu xuất hiện trong thanh ứng dụng.
3. **Nút tràn (Overflow button):** Nút ba chấm dọc mở menu hiển thị thêm các mục tùy chọn.
4. **Menu tùy chọn tràn:** Khi nhấn vào nút tràn, các mục tùy chọn bổ sung sẽ xuất hiện trong menu tràn.

Các mục trong **menu tùy chọn** thường xuất hiện trong menu tràn (overflow menu). Tuy nhiên, bạn có thể đặt một số mục trực tiếp trên thanh ứng dụng nếu có đủ không gian. Việc sử dụng thanh ứng dụng để hiển thị menu giúp ứng dụng của bạn đồng bộ với các ứng dụng Android khác, giúp người dùng dễ dàng làm quen và có trải nghiệm tốt hơn.

**Mẹo:** Để mang lại trải nghiệm người dùng quen thuộc và nhất quán, hãy sử dụng **Menu API** để hiển thị các thao tác và tùy chọn trong **Activity**. Xem tài liệu **Menus** để biết thêm chi tiết.

Bạn cũng sẽ tạo một ứng dụng hiển thị **hộp thoại** để yêu cầu người dùng đưa ra lựa chọn, chẳng hạn như cảnh báo yêu cầu nhấn **OK** hoặc **Cancel**. Một **hộp thoại (dialog)** là cửa sổ xuất hiện trên màn hình hoặc che phủ toàn bộ màn hình, làm gián đoạn luồng hoạt động. Android cung cấp sẵn các **hộp thoại chọn lựa (pickers)** để chọn **ngày** hoặc **giờ**, giúp đảm bảo người dùng nhập đúng định dạng và phù hợp với múi giờ địa phương. Trong bài học này, bạn sẽ tạo một ứng dụng có **Date Picker**.

**Những gì bạn cần biết trước:**

Bạn cần có khả năng:

* Tạo và chạy ứng dụng trong Android Studio.
* Tạo và chỉnh sửa giao diện người dùng (UI) bằng trình chỉnh sửa layout.
* Chỉnh sửa mã XML layout và truy cập các phần tử từ mã Java.
* Thêm **trình xử lý sự kiện click** (click handler) cho **Button**.

**Những gì bạn sẽ học:**

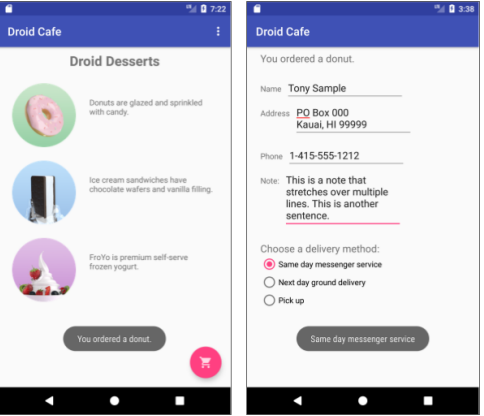
* Cách thêm **mục menu** vào menu tùy chọn.
* Cách hiển thị một số mục menu trên thanh ứng dụng.
* Cách thêm **trình xử lý click** cho các mục menu.
* Cách tạo hộp thoại cảnh báo (**AlertDialog**).
* Cách sử dụng **Date Picker** để nhập ngày.

**Những gì bạn sẽ làm:**

* Tiếp tục bổ sung tính năng cho **Droid Cafe** từ bài thực hành trước.
* Thêm mục menu vào menu tùy chọn.
* Kết nối sự kiện click của menu với **trình xử lý sự kiện**.
* Sử dụng hộp thoại cảnh báo để yêu cầu người dùng xác nhận.
* Thêm **Date Picker** để nhập ngày.

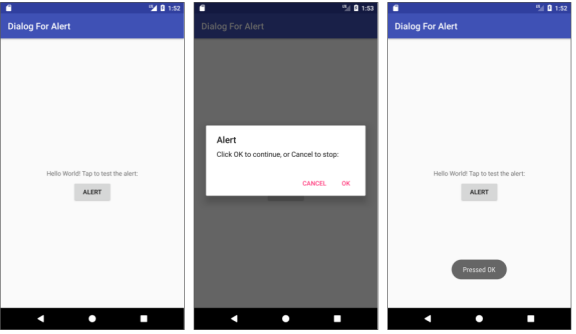
**Tổng quan ứng dụng:**

Trong bài thực hành trước, bạn đã tạo ứng dụng **Droid Cafe** bằng mẫu **Basic Activity**, như minh họa trong hình bên dưới. Mẫu này cũng bao gồm một menu tùy chọn cơ bản trên thanh ứng dụng ở đầu màn hình.

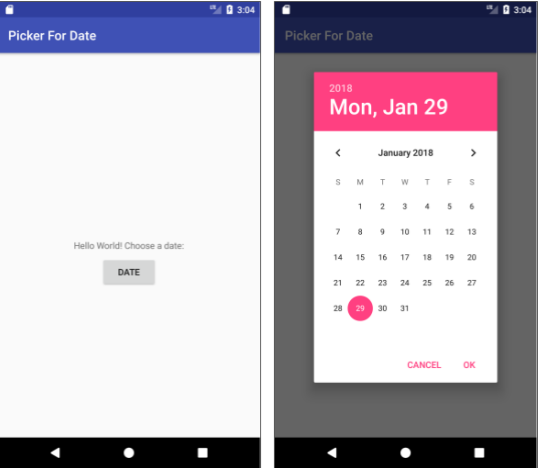


Trong bài tập này, bạn sẽ sử dụng **Toolbar** từ thư viện hỗ trợ **v7 appcompat** làm thanh ứng dụng (**app bar**). Toolbar này tương thích với nhiều thiết bị nhất và cho phép bạn tùy chỉnh thanh ứng dụng khi ứng dụng phát triển. Để tìm hiểu thêm về các yếu tố thiết kế khi sử dụng thanh ứng dụng, hãy xem tài liệu **Responsive layout grid** trong **Material Design**.

Bạn sẽ tạo một ứng dụng mới hiển thị **hộp thoại cảnh báo (alert dialog)**. Hộp thoại này sẽ làm gián đoạn luồng làm việc của người dùng và yêu cầu họ đưa ra lựa chọn.



Bạn cũng sẽ tạo một ứng dụng cung cấp một **Button** để hiển thị **chọn ngày (date picker)** và chuyển đổi ngày đã chọn thành chuỗi để hiển thị trong **Toast message**.



**Task 1: Thêm các mục vào menu tùy chọn**

Trong nhiệm vụ này, bạn sẽ mở dự án **DroidCafeInput** từ bài thực hành trước và thêm các mục vào menu tùy chọn trong thanh ứng dụng (app bar) ở đầu màn hình.

**1.1 Xem mã nguồn**

**Mở dự án DroidCafeInput** và xem các tệp layout sau trong thư mục **res > layout**:

* + **activity\_main.xml**: Layout chính cho **MainActivity**, màn hình đầu tiên người dùng thấy.
  + **content\_main.xml**: Layout cho nội dung của màn hình **MainActivity**, được bao gồm trong **activity\_main.xml**.
  + **activity\_order.xml**: Layout cho **OrderActivity**, bạn đã thêm trong bài thực hành trước về việc sử dụng các điều khiển nhập liệu.

1. **Mở tệp content\_main.xml** và nhấp vào thẻ **Text** để xem mã XML. Thuộc tính **app:layout\_behavior** cho **ConstraintLayout** được đặt là **@string/appbar\_scrolling\_view\_behavior**, điều khiển cách màn hình cuộn liên quan đến thanh ứng dụng ở trên cùng. (Chuỗi này được định nghĩa trong tệp **values.xml**, tệp này bạn không nên chỉnh sửa.)

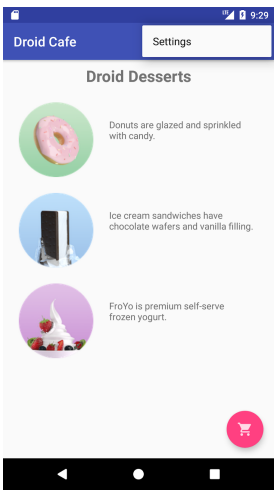
Để tìm hiểu thêm về hành vi cuộn, xem **Android Design Support Library** trong **Android Developers Blog**. Đối với các phương pháp thiết kế liên quan đến menu cuộn, xem **Scrolling** trong **Material Design specification**.

1. **Mở tệp activity\_main.xml** và nhấp vào thẻ **Text** để xem mã XML cho layout chính, sử dụng **CoordinatorLayout** với một layout **AppBarLayout** được nhúng. Các thẻ **CoordinatorLayout** và **AppBarLayout** yêu cầu tên đầy đủ để chỉ định **android.support.design**, đây là thư viện hỗ trợ thiết kế Android.

**AppBarLayout** tương tự như một **LinearLayout** theo chiều dọc. Nó sử dụng lớp **Toolbar** trong thư viện hỗ trợ, thay vì **ActionBar** gốc, để triển khai thanh ứng dụng. **Toolbar** trong layout này có id là **toolbar**, và cũng được chỉ định với tên đầy đủ (android.support.v7.widget).

Thanh ứng dụng là một phần trên cùng của màn hình hiển thị có thể hiển thị tiêu đề hoạt động, điều hướng và các mục tương tác khác. **ActionBar** gốc có hành vi khác nhau tùy thuộc vào phiên bản Android trên thiết bị. Vì lý do này, khi thêm menu tùy chọn, bạn nên sử dụng **Toolbar** trong thư viện hỗ trợ **v7 appcompat** như một thanh ứng dụng. Việc sử dụng **Toolbar** giúp dễ dàng thiết lập thanh ứng dụng hoạt động trên nhiều thiết bị và cũng cho phép bạn tùy chỉnh thanh ứng dụng sau này khi ứng dụng phát triển. **Toolbar** bao gồm các tính năng mới nhất và hoạt động trên bất kỳ thiết bị nào có thể sử dụng thư viện hỗ trợ.

1. **Chạy ứng dụng**. Lưu ý rằng thanh ở đầu màn hình hiển thị tên ứng dụng (**Droid Cafe**). Nó cũng hiển thị nút **overflow** (ba dấu chấm dọc) ở bên phải. Nhấn vào nút overflow để xem menu tùy chọn, tại thời điểm này, menu chỉ có một mục, đó là **Settings**.



1. Xem tệp **AndroidManifest.xml**. Hoạt động **MainActivity** được thiết lập để sử dụng chủ đề **NoActionBar**. Chủ đề này được định nghĩa trong tệp **styles.xml** (mở **app > res > values > styles.xml** để xem). **Styles** được đề cập trong một bài học khác, nhưng bạn có thể thấy rằng chủ đề **NoActionBar** đặt thuộc tính **windowActionBar** thành **false** (không có app bar cửa sổ), và thuộc tính **windowNoTitle** thành **true** (không có tiêu đề). Những giá trị này được thiết lập vì bạn đang định nghĩa app bar với **AppBarLayout**, thay vì sử dụng **ActionBar**. Sử dụng một trong các chủ đề **NoActionBar** ngăn ứng dụng sử dụng lớp **ActionBar** gốc để cung cấp app bar.
2. Xem **MainActivity**, lớp này kế thừa từ **AppCompatActivity** và bắt đầu với phương thức **onCreate()**, trong đó thiết lập nội dung của màn hình thành layout **activity\_main.xml** và thiết lập **toolbar** là **Toolbar** được định nghĩa trong layout. Sau đó, nó gọi **setSupportActionBar()** và truyền **toolbar** cho nó, thiết lập **toolbar** làm app bar cho Activity.

**1.2 Thêm các mục menu vào menu tùy chọn**

Bạn sẽ thêm các mục menu sau vào menu tùy chọn:

● \*\*Order\*\*: Chuyển hướng đến \*\*OrderActivity\*\* để xem đơn đặt hàng món tráng miệng.

● \*\*Status\*\*: Kiểm tra trạng thái của đơn hàng.

● \*\*Favorites\*\*: Hiển thị các món tráng miệng yêu thích.

● \*\*Contact\*\*: Liên hệ với quán cà phê. Vì bạn không cần mục \*\*Settings\*\* hiện có, bạn sẽ thay đổi \*\*Settings\*\* thành \*\*Contact\*\*.

Android cung cấp một định dạng XML chuẩn để định nghĩa các mục menu. Thay vì xây dựng menu trong mã Activity của bạn, bạn có thể định nghĩa một menu và tất cả các mục menu của nó trong một tài nguyên menu XML. Bạn có thể sau đó nạp tài nguyên menu này (tải nó như một đối tượng Menu) trong Activity của bạn:

Mở rộng \*\*res > menu\*\* trong Project > Android pane, và mở \*\*menu\_main.xml\*\*. Mục menu duy nhất được cung cấp từ mẫu là \*\*action\_settings\*\* (tùy chọn Cài đặt), được định nghĩa như sau:

<item

android:id="@+id/action\_settings"

android:orderInCategory="100"

android:title="@string/action\_settings"

app:showAsAction="never" />

1. Change the following attributes of the action\_settings item to make it the action\_contact item (don’t change the existing android:orderInCategory attribute):

### Điều hướng người dùng

### RecycleView

## Trải nghiệm người dùng thú vị

### Hình vẽ, định kiểu và chủ đề

### Thẻ và màu sắc

### Bố cục thích ứng

## Kiểm thử giao diện người dùng

### Espresso cho việc kiểm tra UI

# LÀM VIỆC TRONG NỀN

## Các tác vụ nền

### AsyncTask

### AsyncTask và AsyncTaskLoader

### Broadcast receivers

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

### Thông báo

### Trình quản lý cảnh báo

### JobScheduler

# LƯU DỮ LIỆU NGƯỜI DÙNG

## Tùy chọn và cài đặt

### Shared preferences

### Cài đặt ứng dụng

## Lưu trữ dữ liệu với Room

### Room, LiveData và ViewModel

### Room, LiveData và ViewModel