

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.....	3
Bài 1)	Tạo ứng dụng đầu tiên.....	3
1.1)	Android Studio và Hello World.....	3
1.2)	Giao diện người dùng tương tác đầu tiên.....	5
1.3)	Trình chỉnh sửa bố cục.....	11
1.4)	Văn bản và các chế độ cuộn.....	11
1.5)	Tài nguyên có sẵn.....	11
Bài 2)	Activities.....	11
2.1)	Activity và Intent.....	11
2.2)	Vòng đời của Activity và trạng thái.....	11
2.3)	Intent ngầm định.....	11
Bài 3)	Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ.....	11
3.1)	Trình gỡ lỗi.....	11
3.2)	Kiểm thử đơn vị.....	11
3.3)	Thư viện hỗ trợ.....	11
CHƯƠNG 2.	Trải nghiệm người dùng.....	12
Bài 1)	Tương tác người dùng.....	12
1.1)	Hình ảnh có thể chọn.....	12
1.2)	Các điều khiển nhập liệu.....	12
1.3)	Menu và bộ chọn.....	12
1.4)	Điều hướng người dùng.....	12
1.5)	RecyclerView.....	12
Bài 2)	Trải nghiệm người dùng thú vị.....	12
2.1)	Hình vẽ, định kiểu và chủ đề.....	12
2.2)	Thẻ và màu sắc.....	12
2.3)	Bố cục thích ứng.....	12
Bài 3)	Kiểm thử giao diện người dùng.....	12

3.1)	Espresso cho việc kiểm tra UI.....	12
CHƯƠNG 3. Làm việc trong nền.....		12
Bài 1)	Các tác vụ nền.....	12
1.1)	AsyncTask.....	12
1.2)	AsyncTask và AsyncTaskLoader.....	12
1.3)	Broadcast receivers.....	12
Bài 2)	Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền.....	12
2.1)	Thông báo.....	12
2.2)	Trình quản lý cảnh báo.....	12
2.3)	JobScheduler.....	12
CHƯƠNG 4. Lưu dữ liệu người dùng.....		13
Bài 1)	Tùy chọn và cài đặt.....	13
1.1)	Shared preferences.....	13
1.2)	Cài đặt ứng dụng.....	13
Bài 2)	Lưu trữ dữ liệu với Room.....	13
2.1)	Room, LiveData và ViewModel.....	13
2.2)	Room, LiveData và ViewModel.....	13

3.1) Trình gỡ lỗi 13

CHƯƠNG 1. LÀM QUEN

Bài 1) Tạo ứng dụng đầu tiên

1.1) Android Studio và Hello World

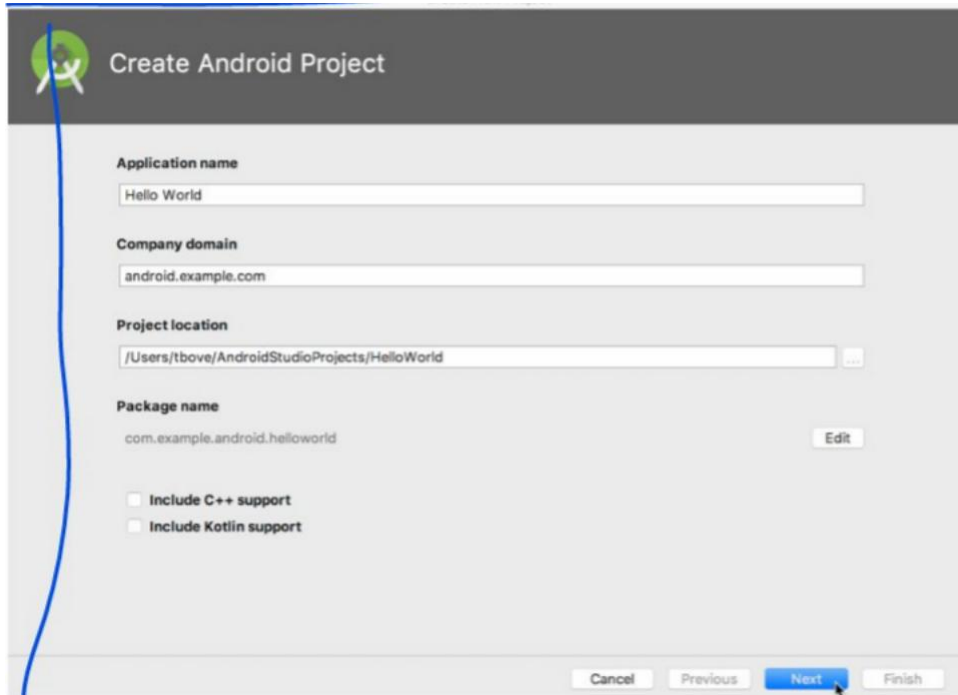
Giới thiệu

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

Những gì Bạn nên biết

Bạn nên có khả năng:

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



Những gì Bạn sẽ cần:

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

Những gì bạn sẽ học

- Cách cài đặt và sử dụng IDE Android Studio.
- Cách sử dụng quy trình phát triển để xây dựng ứng dụng Android.
- Cách tạo một dự án Android từ một 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**

1.2) Giao diện người dùng tương tác đầu tiên

Giới thiệu

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

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

Mã Java hiển thị và điều khiển giao diện người dùng (UI) được chứa trong một lớp mở rộng từ **Activity**. Một **Activity** thường được liên kết với một bố cục UI được định nghĩa trong một tệp XML (**eXtended Markup Language**). Tệp XML này thường được đặt tên theo **Activity** tương ứng và xác định cách sắp xếp các phần tử **View** trên màn hình.

Ví dụ, mã **MainActivity** trong ứng dụng **Hello World** hiển thị một bố cục được định nghĩa trong tệp **activity_main.xml**, trong đó có một **TextView** với nội dung văn bản "Hello World".

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

Trong bài thực hành này, bạn sẽ học cách tạo ứng dụng tương tác đầu tiên một ứng dụng cho phép người dùng tương tác. Bạn sẽ tạo một ứng dụng bằng mẫu **Empty Activity**. Ngoài ra, bạn cũng sẽ học cách sử dụng trình chỉnh sửa bố cục (**layout editor**) để thiết kế giao diện và chỉnh sửa bố cục.

Những gì bạn nên biết

Bạn nên làm quen với:

- Cách cài đặt và mở Android Studio
- Cách tạo ứng dụng Hello World
- Cách chạy ứng dụng Hello World

Những gì bạn sẽ học

- Cách tạo một ứng dụng có hành vi tương tác
- Cách sử dụng layout editor để thiết kế giao diện
- Cách chỉnh sửa bố cục trong XML
- Nhiều thuật ngữ mới

Những gì bạn sẽ làm

- Tạo một ứng dụng và thêm hai Button và một TextView
- Điều chỉnh từng phần trong ConstraintLayout, ràng buộc chúng với lề và các phần tử khác
- Thay đổi thuộc tính của các phần tử giao diện người dùng (UI elements)
- Chỉnh sửa bố cục của ứng dụng trong XML.
- Trích xuất các chuỗi mã cứng (**hardcoded strings**) thành tài nguyên chuỗi (**string resources**).
- Triển khai phương thức xử lý sự kiện nhấp (**click-handler methods**) để hiển thị thông báo khi người dùng nhấn vào mỗi **Button**.

Tổng quan về ứng dụng

Ứng dụng **HelloToast** bao gồm hai phần tử **Button** và một **TextView**. Khi người dùng nhấn vào **Button** đầu tiên, một thông báo ngắn (**Toast**) sẽ hiển thị trên màn hình. Nhấn vào **Button** thứ hai sẽ tăng giá trị của bộ đếm số lần nhấp (**click counter**) hiển thị trong **TextView**, bắt đầu từ số **0**.

Dưới đây là giao diện của ứng dụng sau khi hoàn thành:


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

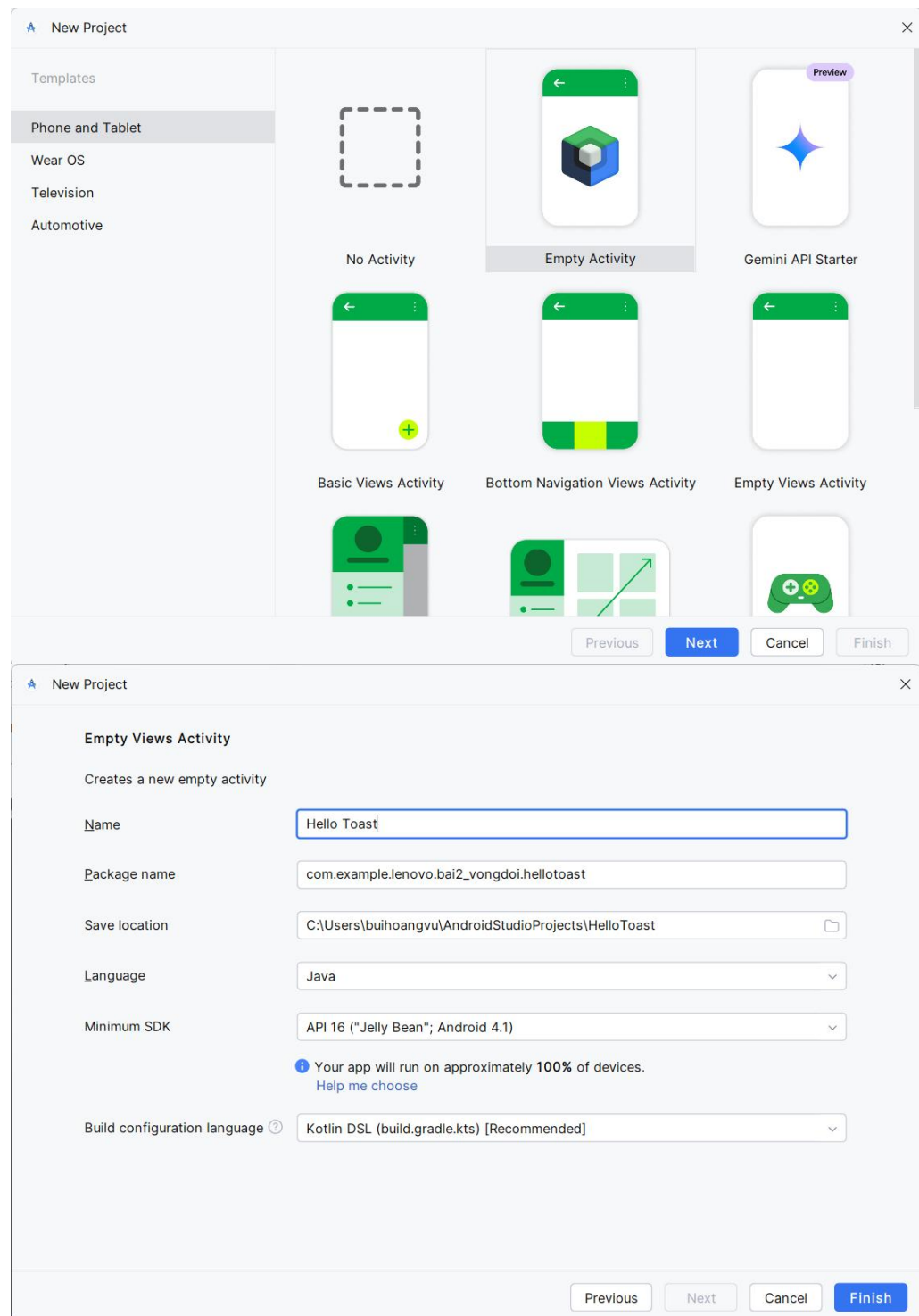
Trong bài thực hành này, bạn cần thiết kế và triển khai một dự án cho ứng dụng HelloToast.

1.1. Tạo dự án Android Studio

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

Attribute	Value
Application Name	Hello Toast
Company Name	com.example.android (or your own domain)
Phone and Tablet Minimum SDK	API15: Android 4.0.3 IceCreamSandwich
Template	Empty Activity
Generate Layout file box Selected	Selected
Backwards Compatibility box Selected	Selected

- Chọn Run > Run app hoặc nhấn và biểu tượng  trên thanh công cụ để biên dịch và chạy ứng dụng trên trình giả lập (**emulator**) hoặc thiết bị của bạn

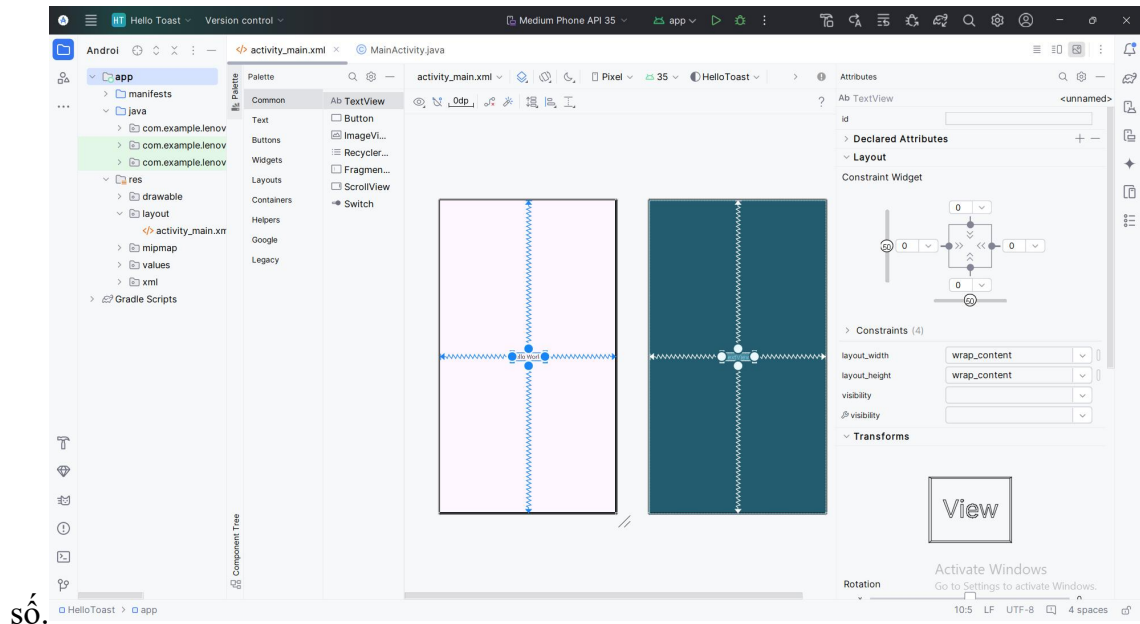


1.2. Khám phá trình chỉnh sửa bố cục

Android Studio cung cấp **trình chỉnh sửa bố cục (layout editor)** để nhanh chóng xây dựng giao diện người dùng (UI) của ứng dụng. Trình chỉnh sửa này cho phép

bạn kéo thả các phần tử vào chế độ xem thiết kế trực quan và bản vẽ (**blueprint view**), định vị chúng trong bố cục, thêm ràng buộc (**constraints**) và thiết lập thuộc tính. **Ràng buộc** xác định vị trí của một phần tử giao diện người dùng trong bố cục. Một **ràng buộc** thể hiện một kết nối hoặc căn chỉnh với một phần tử khác, bố cục cha, hoặc một đường hướng dẫn vô hình.

Khám phá trình chỉnh sửa bố cục và tham khảo hình minh họa bên dưới khi bạn thực hiện theo các bước được đánh



số.



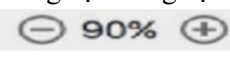
1. Trong bảng **Project > Android**, điều hướng đến thư mục **app > res > layout**, sau đó nhấp đúp vào tệp **activity_main.xml** để mở, nếu nó chưa được mở.
2. Nhấp vào tab **Design** nếu nó chưa được chọn. Bạn sử dụng tab **Design** để thao tác với các phần tử và bố cục, còn tab **Text** để chỉnh sửa mã XML của bố cục.
3. **Pane Palettes** hiển thị các phần tử giao diện người dùng (**UI elements**) mà bạn có thể sử dụng trong bố cục ứng dụng.
4. **Pane Component Tree** hiển thị hệ thống phân cấp của các phần tử giao diện người dùng. Các phần tử được tổ chức theo dạng cây, trong đó **con** kế thừa thuộc tính của **cha**. Trong hình minh họa, **TextView** là một phần tử con của **ConstraintLayout**. Bạn sẽ tìm hiểu thêm về các phần tử này trong bài học sau.
5. **Pane thiết kế và bản vẽ (Design & Blueprint panes)** trong trình chỉnh sửa bố cục hiển thị các phần tử UI trong bố cục. Trong hình minh họa, bố cục chỉ có một phần tử duy nhất: **TextView** hiển thị dòng chữ "Hello World".
6. **Tab Attributes** hiển thị bảng thuộc tính (**Attributes pane**) để thiết lập các thuộc tính cho một phần tử UI.

Nhiệm vụ 2: Thêm vào các phần tử View trong chỉnh sửa bố cục



Trong nhiệm vụ này, bạn sẽ tạo bố cục giao diện người dùng cho ứng dụng HelloToast trong trình chỉnh sửa bố cục bằng cách sử dụng các tính năng của ConstraintLayout. Bạn có thể tạo ràng buộc thủ công, như sẽ được trình bày sau, hoặc tự động bằng công cụ Autoconnect.

2.1. Kiểm tra các ràng buộc của phần tử

Thực hiện các bước sau:

1. Mở activity_main.xml từ ngăn Project > Android nếu tệp chưa được mở. Nếu tab Design chưa được chọn, hãy nhấp vào . Nếu không thấy Blueprint, nhấn vào nút Select Design Surface trên thanh công cụ và chọn Design + Blueprint.
2. Công cụ Autoconnect  cũng nằm trên thanh công cụ và được bật theo mặc định. Ở bước này, hãy đảm bảo công cụ không bị tắt.
3. Nhấn nút Zoom  để phóng to bảng thiết kế và bảng Blueprint để xem chi tiết hơn.
4. Chọn TextView trong ngăn Component Tree. TextView có nội dung "Hello World" sẽ được tô sáng trong bảng thiết kế và bảng Blueprint, đồng thời các ràng buộc (constraints) của phần tử cũng sẽ hiển thị.
5. Thực hiện theo hình động bên dưới cho bước này. Nhấp vào chấm tròn ở bên phải của TextView để xóa ràng buộc ngang đang liên kết nó với cạnh phải của bố cục. Khi đó, TextView sẽ dịch chuyển sang trái do không còn bị ràng buộc vào cạnh phải. Để thêm lại ràng buộc ngang, nhấp vào cùng một chấm tròn và kéo một đường đến cạnh phải của bố cục.

Trong bảng **Blueprint** hoặc **Design**, các điều khiển sau sẽ xuất hiện trên thẻ TextView:

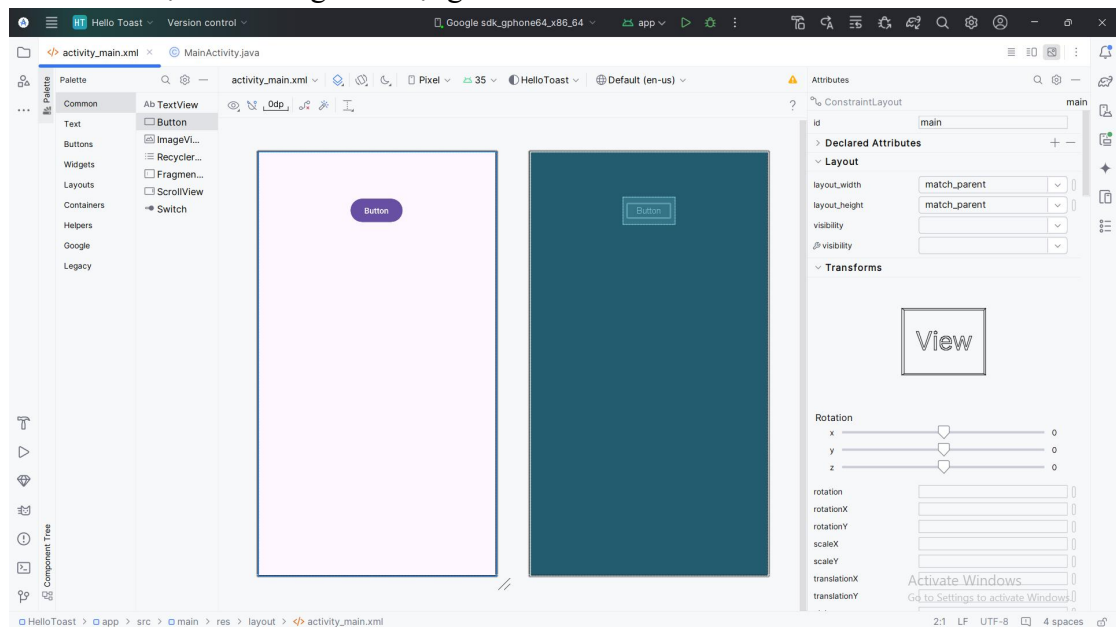
-  **Constraint handle:** Để tạo một ràng buộc như trong hình động trên, hãy nhấp vào một tay cầm ràng buộc, được hiển thị dưới dạng vòng tròn ở cạnh của một phần tử. Sau đó, kéo tay cầm này đến một tay cầm ràng buộc khác hoặc đến ranh giới của phần tử cha. Một đường ziczac sẽ đại diện cho ràng buộc đó.
-  **Resizing handle:** Để thay đổi kích thước phần tử, hãy kéo các tay cầm hình vuông. Khi bạn kéo, tay cầm sẽ chuyển thành một góc xiên.

2.2. Thêm một button vào bố cục

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

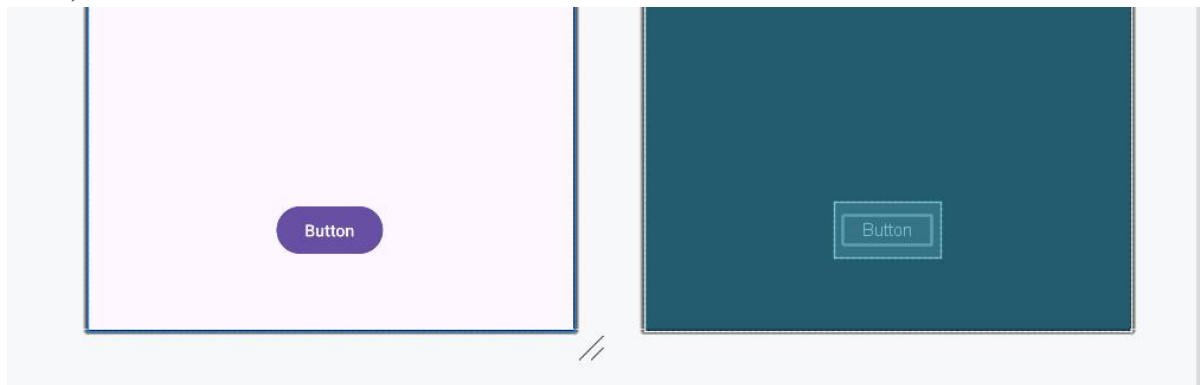
Thực hiện các bước sau để thêm một **Button**:

1. **Bắt đầu với một bố cục trống:** Phần tử **TextView** không cần thiết, vì vậy khi nó vẫn đang được chọn, nhấn phím **Delete** hoặc chọn **Edit > Delete**. Lúc này, bạn sẽ có một bố cục hoàn toàn trống.
2. Kéo một **Button** từ ngăn **Palette** vào bất kỳ vị trí nào trong bố cục. Nếu bạn thả **Button** vào khu vực giữa phía trên của bố cục, có thể các ràng buộc sẽ tự động xuất hiện. Nếu không, bạn có thể kéo các ràng buộc đến cạnh trên, cạnh trái và cạnh phải của bố cục như trong hình động bên dưới.



2.3. Thêm nút thứ 2 vào bố cục

1. Kéo một Button khác từ ngăn Palette vào giữa bố cục như trong hình động bên dưới. Autoconnect có thể tự động tạo các ràng buộc ngang cho bạn (nếu không, bạn có thể tự kéo chúng).
2. Kéo một ràng buộc dọc từ Button xuống cạnh dưới của bố cục (tham khảo hình bên dưới).



Bạn có thể xóa ràng buộc khỏi một phần tử bằng cách chọn phần tử đó và di chuột qua để hiển thị nút Clear Constraints. Nhấp vào nút này để xóa tất cả ràng buộc của phần tử đã chọn.

Để xóa một ràng buộc cụ thể, hãy nhấp vào tay cầm của ràng buộc đó.

Để xóa tất cả ràng buộc trong toàn bộ bố cục, nhấp vào công cụ Clear All Constraints trên thanh công cụ. Công cụ này rất hữu ích nếu bạn muốn thiết lập lại toàn bộ ràng buộc trong bố cục của mình.

Nhiệm vụ 3: Thay đổi thành phần UI

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

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

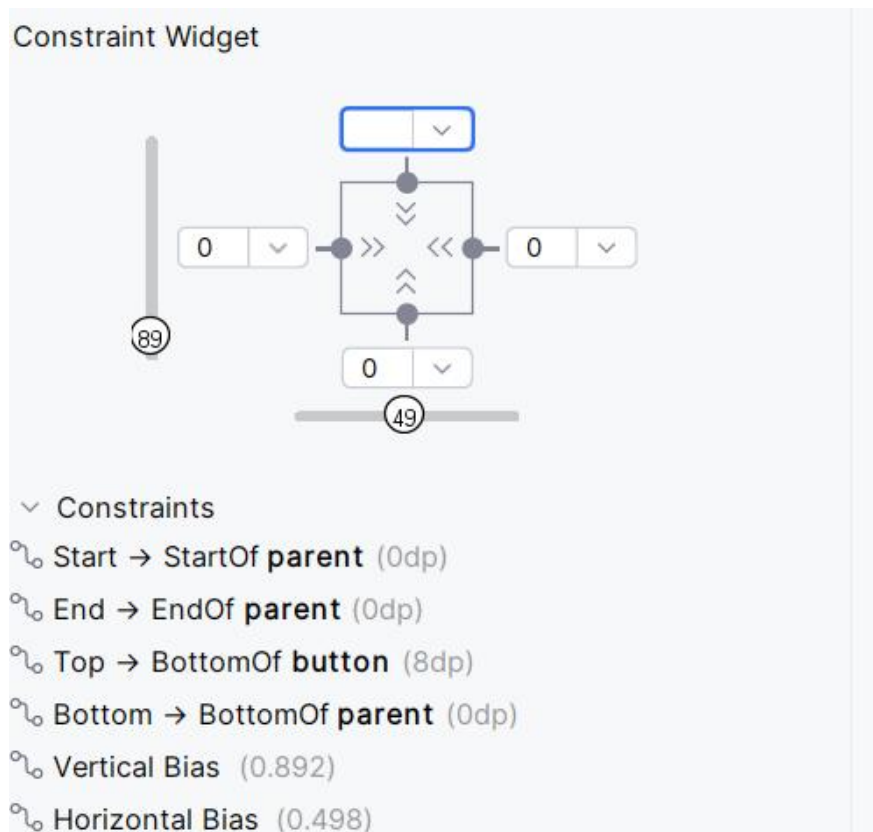
3.1 Thay đổi kích thước Button

Trình chỉnh sửa bố cục cung cấp các tay cầm thay đổi kích thước ở bốn góc của một **View**, giúp bạn có thể điều chỉnh kích thước nhanh chóng. Bạn có thể kéo các tay cầm này để thay đổi kích thước của **View**, nhưng cách này sẽ đặt kích thước cố định (**hardcoded**).

Lưu ý: Tránh đặt kích thước cố định cho hầu hết các phần tử **View**, vì chúng không thể tự động điều chỉnh theo nội dung và kích thước màn hình khác nhau.

Thay vào đó, hãy sử dụng ngăn **Attributes** ở bên phải trình chỉnh sửa bố cục để chọn một chế độ kích thước không sử dụng giá trị cố định.

- Ngăn **Attributes** bao gồm một bảng kích thước hình vuông gọi là **view inspector** ở phía trên.
- Các biểu tượng bên trong hình vuông này đại diện cho các cài đặt chiều cao và chiều rộng như sau:



Trong hình trên:

1. Điều khiển chiều cao (Height control):

- Điều khiển này xác định thuộc tính **layout_height** và xuất hiện dưới dạng hai đoạn ở cạnh trên và cạnh dưới của hình vuông.

- Nếu các góc được bo tròn, điều đó có nghĩa là thuộc tính này được đặt thành **wrap_content**, tức là **View** sẽ mở rộng theo chiều dọc tùy theo nội dung bên trong.
- Số **8** hiển thị trên điều khiển này cho biết một khoảng lề tiêu chuẩn là **8dp**.

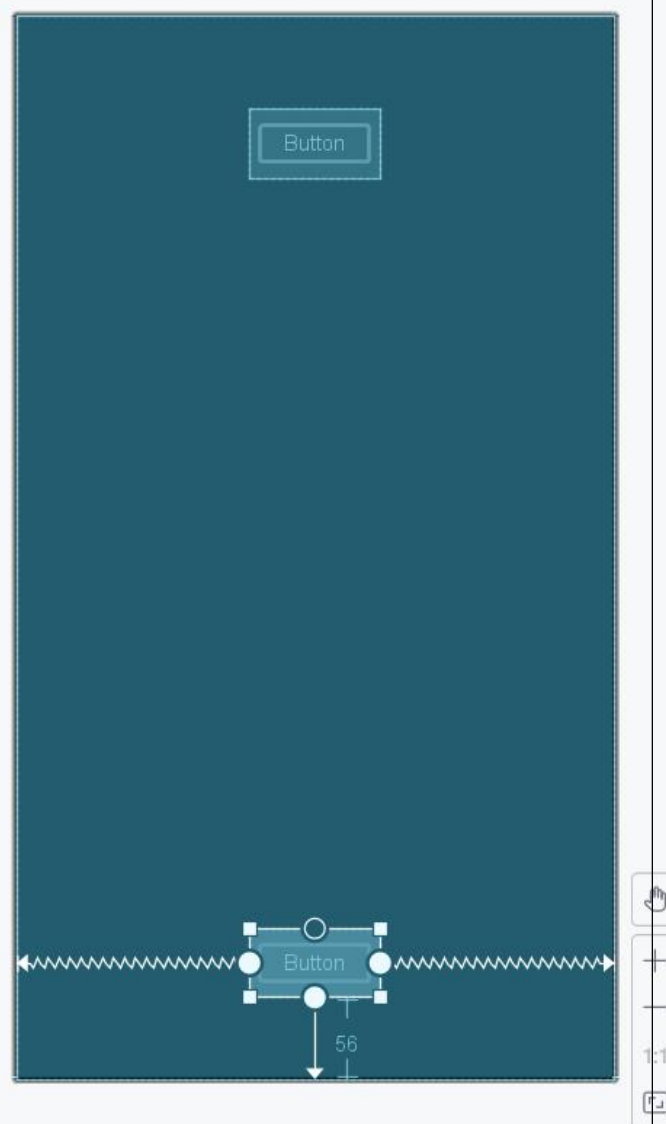
2. Điều khiển chiều rộng (Width control):

- Điều khiển này xác định thuộc tính **layout_width** và xuất hiện dưới dạng hai đoạn ở cạnh trái và cạnh phải của hình vuông.
- Nếu các góc được bo tròn, điều đó có nghĩa là thuộc tính này được đặt thành **wrap_content**, tức là **View** sẽ mở rộng theo chiều ngang tùy theo nội dung bên trong nghĩa là **View** sẽ mở rộng theo chiều ngang tùy theo nội dung bên trong, với khoảng lề tối đa là **8dp**.

2. Nút đóng ngăn Attributes (Attributes pane close button): Nhấp vào nút này để đóng ngăn **Attributes**.

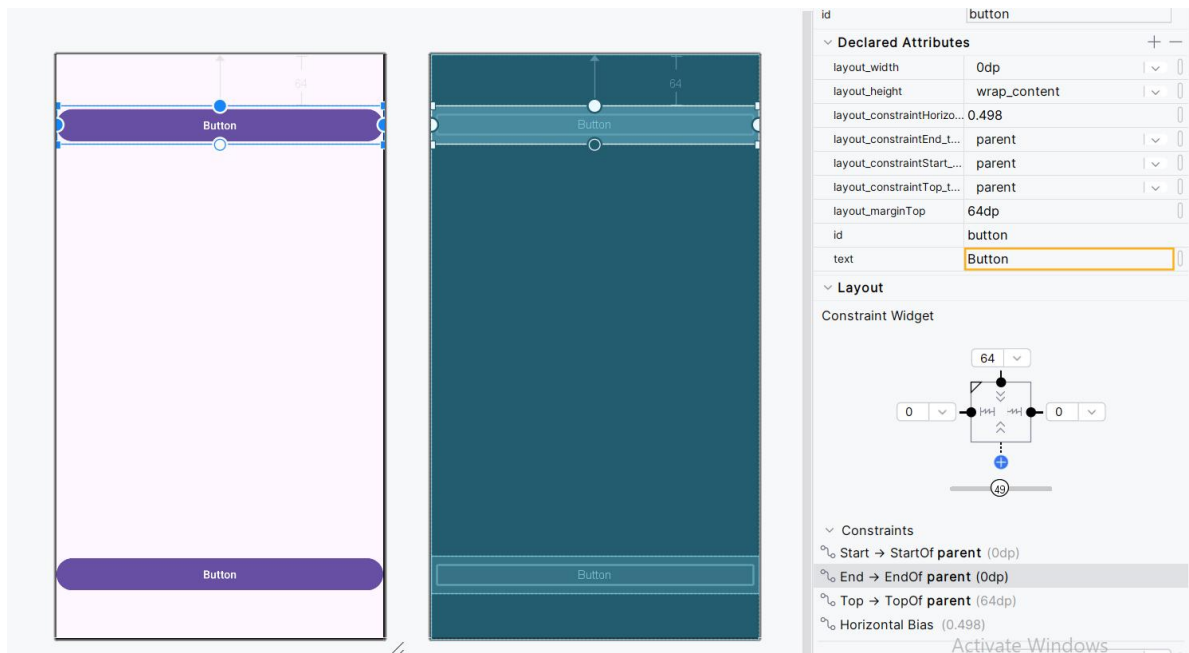
Thực hiện các bước sau:

1. Chọn **Button** ở phía trên trong ngăn **Component Tree**.
2. Nhấp vào tab **Attributes** ở bên phải cửa sổ trình chỉnh sửa bố cục.
3. Nhấp vào **điều khiển chiều rộng (width control)** hai lần:
 - Lần nhấp đầu tiên sẽ thay đổi nó thành **Fixed** (Cố định), được biểu thị bằng các đường thẳng.
 - Lần nhấp thứ hai sẽ thay đổi nó thành **Match Constraints** (Khớp ràng buộc), được biểu thị bằng các lò xo, như trong hình động bên dưới.



Do thay đổi điều khiển chiều rộng, thuộc tính **layout_width** trong ngăn **Attributes** hiển thị giá trị **match_constraint**, và phần tử **Button** sẽ kéo giãn theo chiều ngang để lấp đầy khoảng trống giữa cạnh trái và cạnh phải của bố cục.

4. Chọn **Button** thứ hai và thực hiện các thay đổi tương tự cho **layout_width** như ở bước trước, như hiển thị trong hình bên dưới.

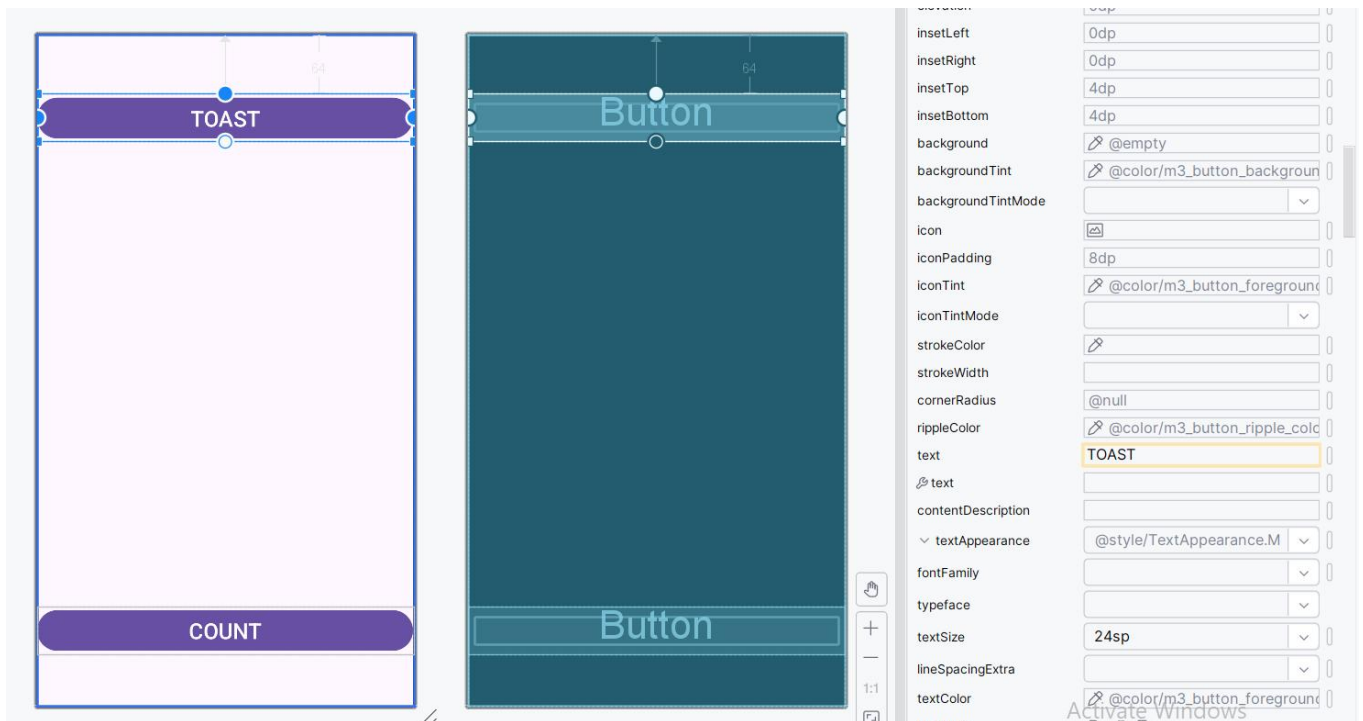


Như đã thấy trong các bước trước, các thuộc tính **layout_width** và **layout_height** trong ngăn **Attributes** sẽ thay đổi khi bạn điều chỉnh các điều khiển chiều cao và chiều rộng trong **inspector**. Các thuộc tính này có thể nhận một trong ba giá trị trong **ConstraintLayout**:

- **match_constraint** mở rộng phần tử **View** để lấp đầy phần tử cha theo chiều rộng hoặc chiều cao, tối đa đến khoảng lề nếu có. Trong trường hợp này, phần tử cha là **ConstraintLayout**. Bạn sẽ tìm hiểu thêm về **ConstraintLayout** trong nhiệm vụ tiếp theo.
- **wrap_content** thu nhỏ kích thước của phần tử **View** sao cho vừa đủ chứa nội dung bên trong. Nếu không có nội dung, phần tử **View** sẽ trở nên vô hình.
- Để chỉ định kích thước cố định nhưng vẫn điều chỉnh theo kích thước màn hình của thiết bị, hãy sử dụng một số cố định với đơn vị **dp** (**density-independent pixels**). Ví dụ, **16dp** có nghĩa là **16 pixel độc lập với mật độ màn hình**.

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

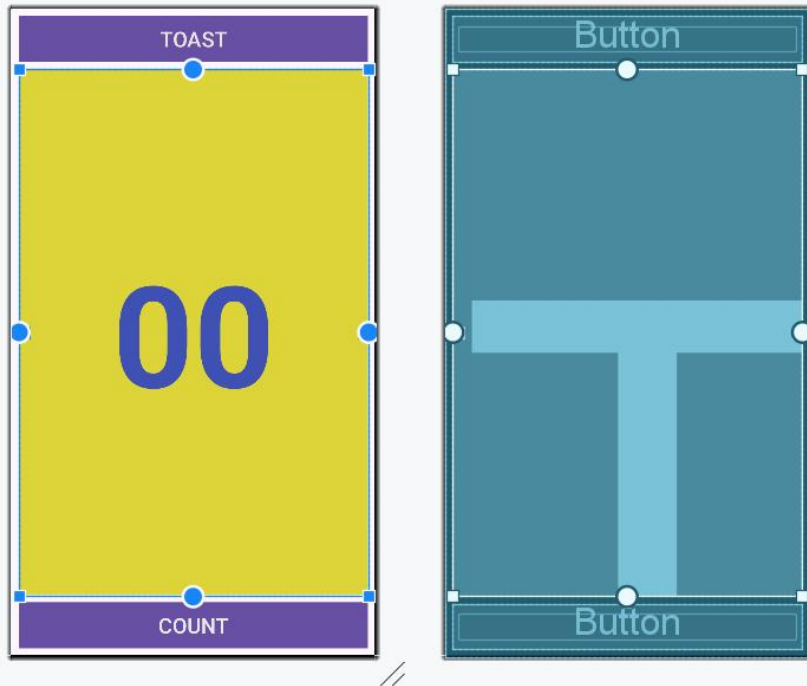
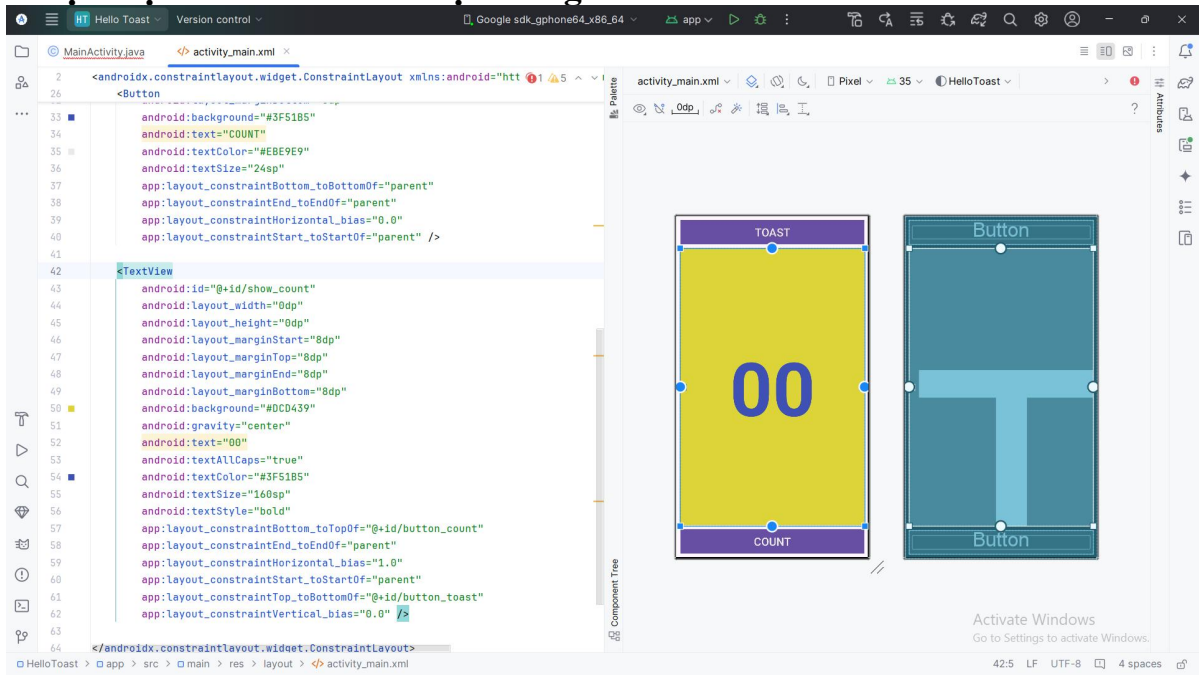
Một trong những lợi ích của **ConstraintLayout** là khả năng căn chỉnh hoặc ràng buộc các phần tử so với các phần tử khác. Trong nhiệm vụ này, bạn sẽ thêm một **TextView** vào giữa bố cục và ràng buộc nó theo chiều ngang với các lề và theo chiều dọc với hai **Button**. Sau đó, bạn sẽ thay đổi các thuộc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



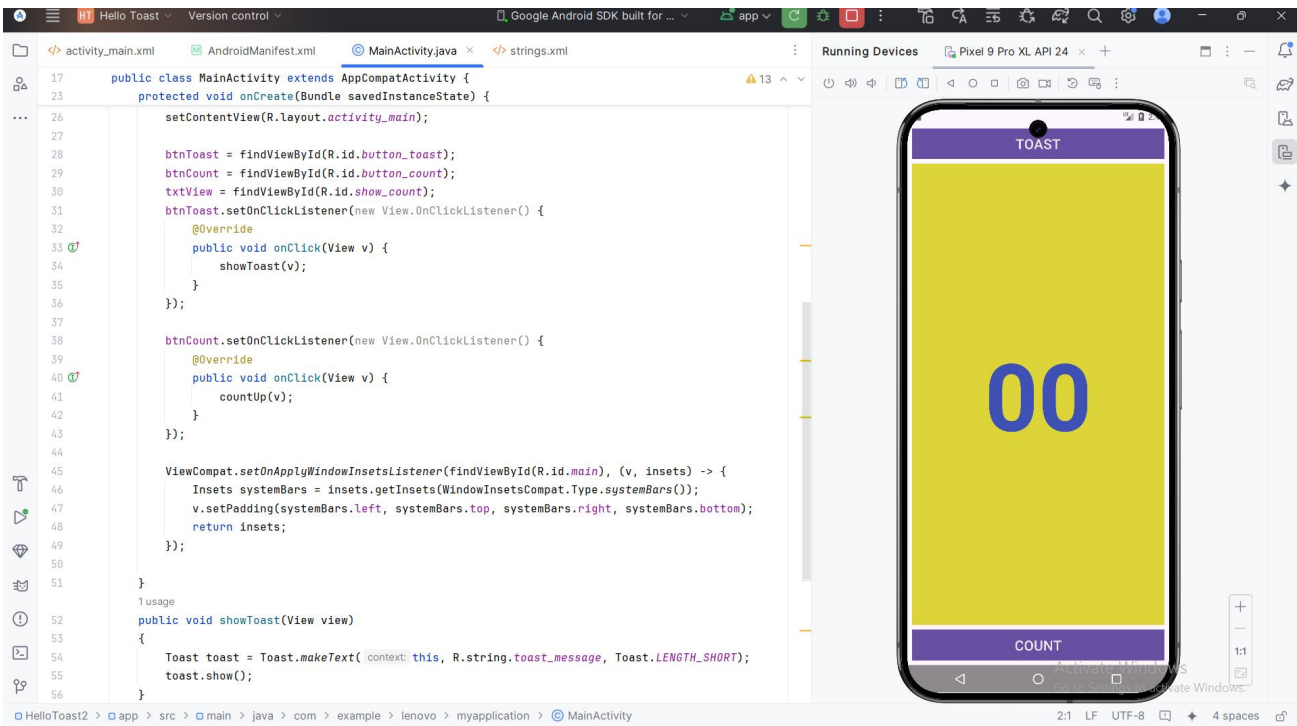
4.1 Thêm một TextView và thiết lập ràng buộc

1. Như trong hình động bên dưới, kéo một **TextView** từ ngăn **Palette** vào phần trên của bố cục, sau đó kéo một ràng buộc từ cạnh trên của **TextView** đến tay cầm ở cạnh dưới của nút **Toast Button**. Việc này sẽ ràng buộc **TextView** nằm ngay bên dưới **Button**.
2. Như trong hình động bên dưới, kéo một ràng buộc từ cạnh dưới của **TextView** đến tay cầm ở cạnh trên của nút **Count Button**, và kéo các ràng buộc từ hai cạnh của **TextView** đến hai cạnh của bố cục. Việc này sẽ ràng buộc **TextView** nằm ở giữa bố cục, giữa hai **Button**.

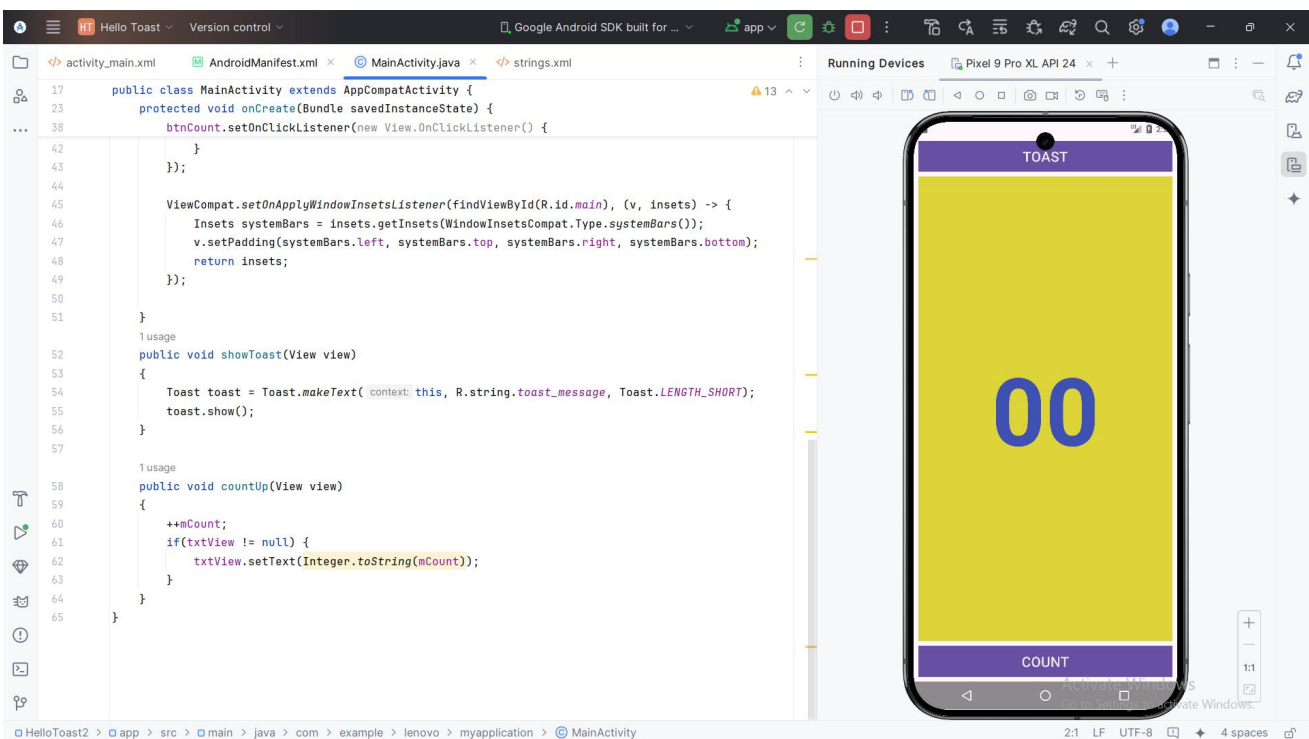
Nhiệm vụ 5: Chỉnh sửa bố cục trong XML



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



Mã giải pháp



1.3) Trình chỉnh sửa bố cục

1.4) Văn bản và các chế độ cuộn

1.5) Tài nguyên có sẵn

Bài 2) Activities

2.1) Activity và Intent

2.2) Vòng đời của Activity và trạng thái

2.3) Intent ngầm định

Bài 3) Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ

3.1) Trình gỡ lỗi

3.2) Kiểm thử đơn vị

3.3) Thư viện hỗ trợ

CHƯƠNG 2. TRẢI NGHIỆM NGƯỜI DÙNG

Bài 1) Tương tác người dùng

- 1.1) Hình ảnh có thể chọn**
- 1.2) Các điều khiển nhập liệu**
- 1.3) Menu và bộ chọn**
- 1.4) Điều hướng người dùng**
- 1.5) RecyclerView**

Bài 2) Trải nghiệm người dùng thú vị

- 2.1) Hình vẽ, định kiểu và chủ đề**
- 2.2) Thẻ và màu sắc**
- 2.3) Bố cục thích ứng**

Bài 3) Kiểm thử giao diện người dùng

- 3.1) Espresso cho việc kiểm tra UI**

CHƯƠNG 3. LÀM VIỆC TRONG NỀN

Bài 1) Các tác vụ nền

- 1.1) AsyncTask**
- 1.2) AsyncTask và AsyncTaskLoader**
- 1.3) Broadcast receivers**

Bài 2) Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền

- 2.1) Thông báo**
- 2.2) Trình quản lý cảnh báo**
- 2.3) JobScheduler**

CHƯƠNG 4. LƯU DỮ LIỆU NGƯỜI DÙNG

Bài 1) Tùy chọn và cài đặt

1.1) Shared preferences

1.2) Cài đặt ứng dụng

Bài 2) Lưu trữ dữ liệu với Room

2.1) Room, LiveData và ViewModel

2.2) Room, LiveData và ViewModel