# Layout

Dưới đây là giải thích chi tiết và ví dụ về các Layout Managers trong Android Studio để giúp bạn hiểu rõ hơn:

### 1. ConstraintLayout:

- \*\*Mục đích:\*\* Được thiết kế để tạo ra giao diện linh hoạt và dễ dàng thay đổi trên nhiều thiết bị và kích thước màn hình khác nhau.

Documentation](https://developer.android.com/reference/androidx/constraintlayout/widget/ConstraintLayout)

### 2. LinearLayout:

- \*\*Mục đích:\*\* Sắp xếp các phần tử theo chiều ngang hoặc chiều dọc.

- \*\*Ví dụ:\*\*

```xml

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Hello World!"/>

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Click me"/>

</LinearLayout>

```

- \*\*Tài liệu:\*\* [LinearLayout Documentation](https://developer.android.com/guide/topics/ui/layout/linear)

### 3. TableLayout:

- \*\*Mục đích:\*\* Sắp xếp các phần tử trong bảng có hàng và cột.

- \*\*Ví dụ:\*\*

```xml

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<TableRow>

<TextView

android:text="Name"

android:padding="5dp"/>

<TextView

android:text="Age"

android:padding="5dp"/>

</TableRow>

<TableRow>

<TextView

android:text="John"

android:padding="5dp"/>

<TextView

android:text="25"

android:padding="5dp"/>

</TableRow>

</TableLayout>

```

- \*\*Tài liệu:\*\* [TableLayout Documentation](https://developer.android.com/guide/topics/ui/layout/tablelayout)

### 5. RelativeLayout:

- \*\*Mục đích:\*\* Đặt các phần tử dựa trên mối quan hệ tương đối với các phần tử khác.

- \*\*Ví dụ:\*\*

```xml

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<Button

android:id="@+id/button"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Click me"

android:layout\_alignParentTop="true"

android:layout\_alignParentStart="true"/>

</RelativeLayout>

```

- \*\*Tài liệu:\*\* [RelativeLayout Documentation](https://developer.android.com/guide/topics/ui/layout/relative)

### 8. CoordinatorLayout:

- \*\*Mục đích:\*\* Phối hợp các phần tử con để tạo ra hiệu ứng tương tác và giao diện người dùng đáng chú ý.

- \*\*Ví dụ:\*\*

```xml

<androidx.coordinatorlayout.widget.CoordinatorLayout

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

android:layout

# Cấu trúc của dự án Android

Thư mục và File Quan Trọng:

1. app Thư mục:

Mục Đích: Chứa tất cả các mã nguồn, tài nguyên và các tệp tin liên quan đến ứng dụng của bạn.

Công Dụng: Bạn sẽ thực hiện phần lớn công việc phát triển ứng dụng trong thư mục này.

2. res Thư mục:

Mục Đích: Chứa tất cả các tài nguyên (như hình ảnh, xml, values, layout) của ứng dụng.

Công Dụng: Là nơi bạn đặt các tài nguyên đồ họa, các tệp tin XML, và các tài nguyên khác cần thiết cho ứng dụng của bạn.

3. values Thư mục:

Mục Đích: Chứa các tệp tin XML định nghĩa các giá trị, chẳng hạn như màu sắc, kích thước, strings.

Công Dụng: Giúp quản lý các giá trị như màu sắc, kích thước, text trong một nơi và làm cho code dễ bảo trì hơn.

4. layout Thư mục:

Mục Đích: Chứa các tệp tin XML định nghĩa giao diện người dùng (UI) của ứng dụng.

Công Dụng: Là nơi bạn đặt các layout cho các Activity hoặc Fragment.

5. java (hoặc kotlin) Thư mục trong app Thư mục:

Mục Đích: Chứa mã nguồn của ứng dụng, được tổ chức theo gói (package).

Công Dụng: Bạn sẽ viết mã nguồn Java hoặc Kotlin trong các gói này.

6. AndroidManifest.xml File:

Mục Đích: Định nghĩa cấu trúc cơ bản của ứng dụng, bao gồm các thành phần như Activities, Services, Permissions.

Công Dụng: Cung cấp thông tin quan trọng về ứng dụng và là nơi bạn đặt các khai báo quan trọng như permissions, intent filters.

7. res/drawable Thư mục:

Mục Đích: Chứa hình ảnh và drawable của ứng dụng.

Công Dụng: Là nơi bạn đặt hình ảnh sử dụng trong ứng dụng.

8. res/values Thư mục:

Mục Đích: Chứa các tệp tin XML định nghĩa giá trị như màu sắc (colors.xml), strings (strings.xml), kích thước (dimens.xml).

Công Dụng: Dễ quản lý và sửa đổi giá trị như màu sắc, chuỗi, kích thước mà không phải sửa trực tiếp trong mã nguồn.

Thư Mục Khác:

9. build Thư mục:

Mục Đích: Chứa tất cả các tệp tin tạm thời và các tệp tin liên quan đến quá trình xây dựng ứng dụng.

Công Dụng: Không cần quan tâm nhiều trong quá trình phát triển.

10. gradle Thư mục:

Mục Đích: Chứa các tệp tin cấu hình của Gradle.

Công Dụng: Gradle là hệ thống xây dựng cho Android, và các tệp tin trong thư mục này cấu hình quá trình xây dựng và các dependencies.

11. build.gradle File:

Mục Đích: Cấu hình dự án và module bằng cách sử dụng Gradle DSL.

Công Dụng: Quyết định các dependencies, plugin, và cấu hình khác cho dự án.

Bước 1: tạo file xử lý logic MainActivity và file giao diện xml trong folder “layout”

Bước 2: thiết kế giao diện bằng các layout có sẵn(Contrain/Relative/Linner/Table Layout, thêm id cho các phần tử

ArrayAdapter<Object> -> hiện danh sách của 1 Object, lấy dữ liệu từ bộ dữ liệu và tạo ra các đối tượng View dựa trên dữ liệu đó rồi gắn lên bất kỳ Adapter View mà ràng buộc với Adapter

**Tạo một ArrayAdapter**

Để tạo một Adapter, bạn cần những thứ sau đây:

một tập hợp dữ liệu

một tập tin có chứa Layout của các đối tượng View được tạo ra

Ngoài ra, vì lớp ArrayAdapter chỉ có thể làm việc với chuỗi, nên bạn cần phải chắc chắn rằng Layout của các đối tượng View được tạo ra có chứa ít nhất một TextView.

Bước 3: xác định các biến cần dùng trong MainActivity -> ánh xạ(findViewById) ->

Tạo event cho button

1. Ở file xml(giao diện), tạo thuộc tính “onCLick”=”name?” -> click vào bóng đèn bên trái để tạo phương thức ở file activity rồi viết code
2. File Activity: ánh xạ button -> gọi phương thức setOnClickListener -> new View.OnClickListener()

**Để thiết lập số cột trong Grid, hãy gọi phương thức setNumColumns() của nó. Tôi sẽ cho nó hai cột.**

cheeseGrid.setNumColumns(2);

Thông thường, bạn sẽ cần điều chỉnh độ rộng của các cột và khoảng cách giữa chúng bằng cách sử dụng các phương thức setColumnWidth(), setVerticalSpacing() và setHorizontalSpacing(). Lưu ý rằng những phương thức này sử dụng pixel làm đơn vị của chúng.

cheeseGrid.setColumnWidth(60);

cheeseGrid.setVerticalSpacing(20);

cheeseGrid.setHorizontalSpacing(20);