

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	17
1.3)	Trình chỉnh sửa bố cục	17
1.4)	Văn bản và các chế độ cuộn	17
1.5)	Tài nguyên có sẵn	17
Bài 2)	Activities	17
2.1)	Activity và Intent	17
2.2)	Vòng đời của Activity và trạng thái	17
2.3)	Intent ngầm định	17
Bài 3)	Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ	17
3.1)	Trình gỡ lỗi	17
3.2)	Kiểm thử đơn vị	17
3.3)	Thư viện hỗ trợ	17
CHƯƠNG 2.	Trải nghiệm người dùng	18
Bài 1)	Tương tác người dùng	18
1.1)	Hình ảnh có thể chọn	18
1.2)	Các điều khiển nhập liệu	18
1.3)	Menu và bộ chọn	18
1.4)	Điều hướng người dùng	18
1.5)	RecyclerView	18
Bài 2)	Trải nghiệm người dùng thú vị	18
2.1)	Hình vẽ, định kiểu và chủ đề	18
2.2)	Thẻ và màu sắc	18
2.3)	Bố cục thích ứng	18
Bài 3)	Kiểm thử giao diện người dùng	18

3.1)	Espresso cho việc kiểm tra UI	18
CHƯƠNG 3. Làm việc trong nền		18
Bài 1)	Các tác vụ nền	18
1.1)	AsyncTask	18
1.2)	AsyncTask và AsyncTaskLoader	18
1.3)	Broadcast receivers	18
Bài 2)	Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền	18
2.1)	Thông báo	18
2.2)	Trình quản lý cảnh báo	18
2.3)	JobScheduler	18
CHƯƠNG 4. Lưu dữ liệu người dùng		19
Bài 1)	Tùy chọn và cài đặt	19
1.1)	Shared preferences	19
1.2)	Cài đặt ứng dụng	19
Bài 2)	Lưu trữ dữ liệu với Room	19
2.1)	Room, LiveData và ViewModel	19
2.2)	Room, LiveData và ViewModel	19
3.1)	Trình gowx lỗi	

CHƯƠNG 1. LÀM QUEN

Bài 1) Tạo ứng dụng đầu tiên

1.1) Android Studio và Hello World

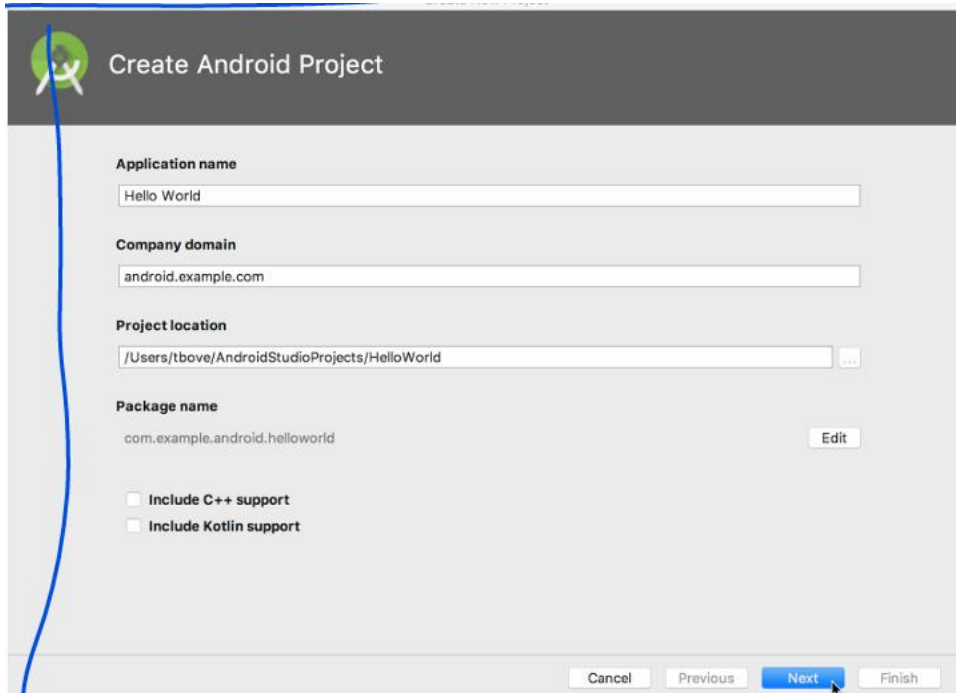
Giới thiệu

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.

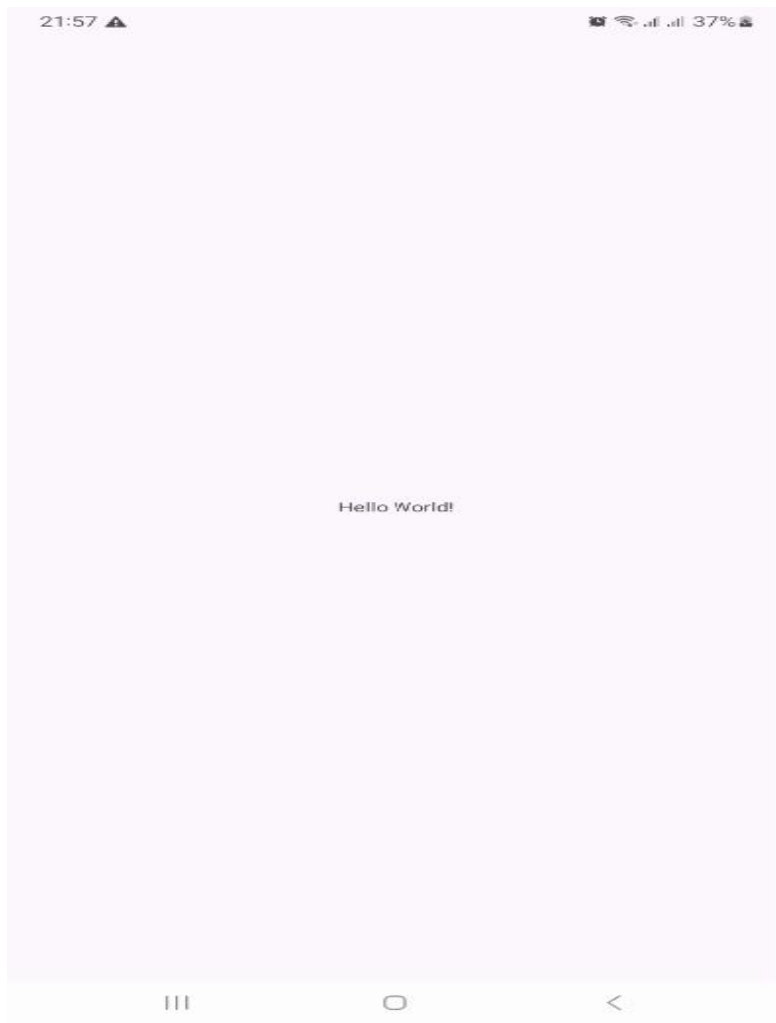
Những gì bạn sẽ làm

- 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**

Tổng quan về ứng dụng

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

Đây là giao diện của ứng dụng sau khi hoàn thành:



Nhiệm vụ 1: Cài đặt Android Studio

Android Studio cung cấp một **môi trường phát triển tích hợp (IDE)** hoàn chỉnh, bao gồm **trình soạn thảo mã nâng cao** và một bộ **mẫu ứng dụng**. Ngoài ra, nó còn chứa các công cụ hỗ trợ **phát triển, gỡ lỗi, kiểm thử và tối ưu hóa hiệu suất**, giúp quá trình phát triển ứng dụng nhanh hơn và dễ dàng hơn.

Bạn có thể:

- ✓ **Kiểm thử ứng dụng** trên nhiều **trình giả lập** đã được cấu hình sẵn hoặc trên **thiết bị di động thực tế**.
- ✓ **Xây dựng ứng dụng** để phát hành chính thức.
- ✓ **Đăng tải ứng dụng** lên **Google Play Store**.

Lưu ý: **Android Studio** liên tục được cập nhật và cải tiến. Để biết thông tin mới nhất về **yêu cầu hệ thống** và **hướng dẫn cài đặt**, hãy xem tại [Android Studio](#).

Android Studio có sẵn cho **Windows, Linux** và **macOS**. Phiên bản mới nhất của **OpenJDK (Java Development Kit)** cũng được tích hợp sẵn trong **Android Studio**.

Các bước cài đặt **Android Studio**:

- ✓ **Bước 1:** Truy cập trang web dành cho [nhà phát triển Android](#) và làm theo hướng dẫn để tải xuống & cài đặt **Android Studio**.
- ✓ **Bước 2:** Chấp nhận **cấu hình mặc định** cho tất cả các bước và đảm bảo rằng **tất cả các thành phần cần thiết** được chọn để cài đặt.
- ✓ **Bước 3:** Sau khi hoàn tất cài đặt, **Setup Wizard** sẽ tải xuống và cài đặt **các thành phần bổ sung**, bao gồm **Android SDK**. Quá trình này có thể mất một khoảng thời gian tùy thuộc vào tốc độ Internet của bạn. Một số bước có thể trông giống nhau, nhưng hãy kiên nhẫn.
- ✓ **Bước 4:** Khi quá trình tải xuống hoàn tất, **Android Studio** sẽ tự động khởi động, và bạn đã sẵn sàng tạo **dự án đầu tiên**.

Nếu gặp vấn đề trong quá trình cài đặt, hãy kiểm tra **ghi chú phát hành của Android Studio** hoặc nhờ sự trợ giúp từ **giảng viên** của bạn.

Nhiệm vụ 2: Tạo ứng dụng Hello World

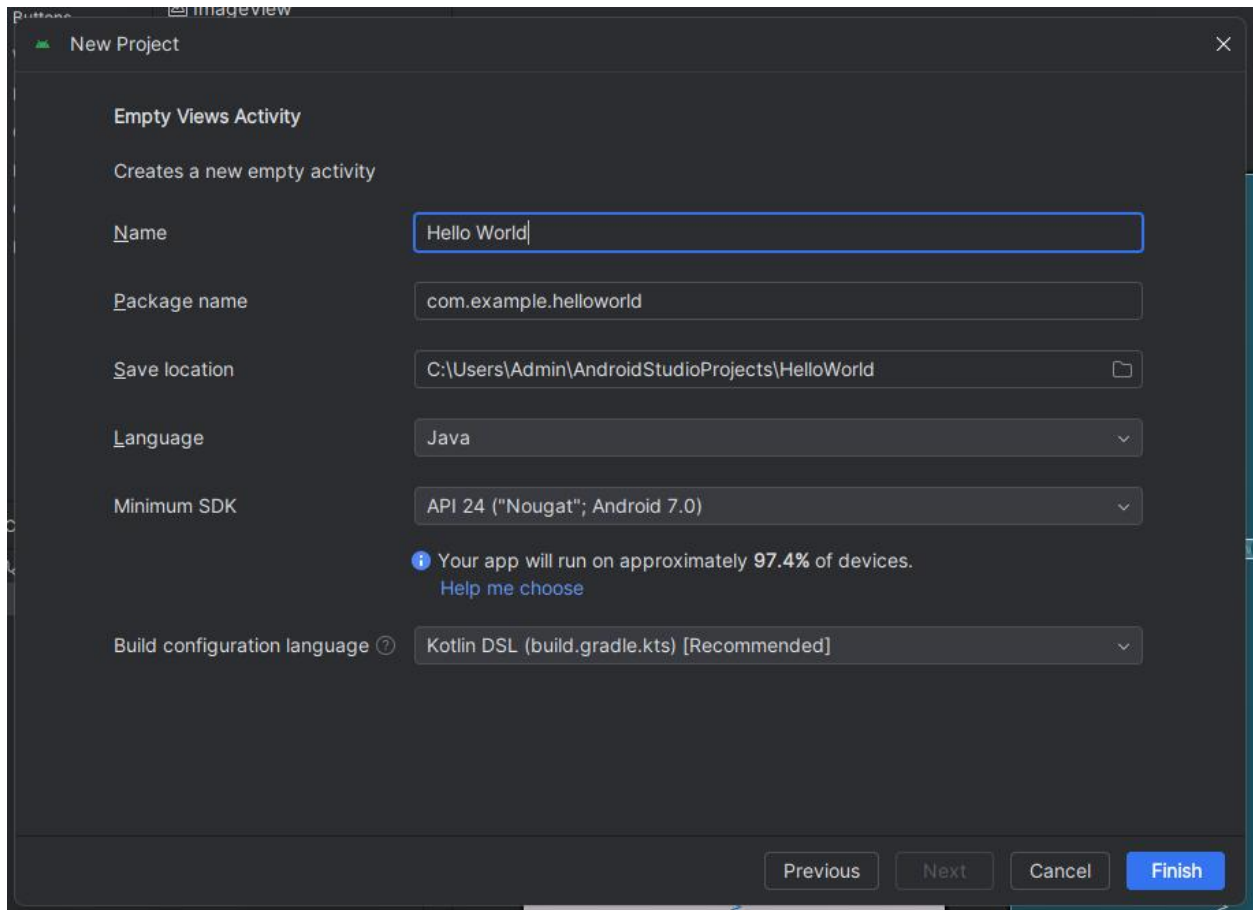
Trong nhiệm vụ này, bạn sẽ tạo một ứng dụng hiển thị **"Hello World"** để xác minh rằng **Android Studio** đã được cài đặt đúng cách và làm quen với các bước phát triển ứng dụng cơ bản.

2.1 Tạo dự án ứng dụng

✓ **Bước 1:** Mở **Android Studio** (nếu chưa mở).

✓ **Bước 2:** Trong cửa sổ **Welcome to Android Studio**, nhấp vào **"Start a new Android Studio project"** để tạo một dự án mới.

✓ **Bước 3:** Trong cửa sổ **Create Android Project**, nhập **Hello World** vào ô **Application name**.



✓ **Bước 4:** Xác minh rằng thư mục trong **Project location** là nơi bạn muốn lưu trữ ứng dụng **Hello World** và các dự án Android Studio khác. Nếu cần, hãy thay đổi sang thư mục bạn mong muốn.

✓ **Bước 5:** Chấp nhận giá trị mặc định **android.example.com** cho **Company Domain**, hoặc nhập một tên miền riêng nếu bạn có.

- Nếu không có kế hoạch xuất bản ứng dụng, bạn có thể giữ nguyên mặc định.
- Hãy lưu ý rằng **thay đổi tên package** sau này sẽ tốn công hơn.

✓ **Bước 6:** Bỏ chọn các tùy chọn **Include C++ support** và **Include Kotlin support**, sau đó nhấp **Next**.

✓ **Bước 7:** Ở màn hình **Target Android Devices**, đảm bảo **Phone and Tablet** được chọn.

- Kiểm tra **Minimum SDK** được đặt thành **API 15: Android 4.0.3 IceCreamSandwich**. Nếu chưa đúng, hãy sử dụng menu thả xuống để chọn.

Hoàn tất bước tạo dự án Hello World

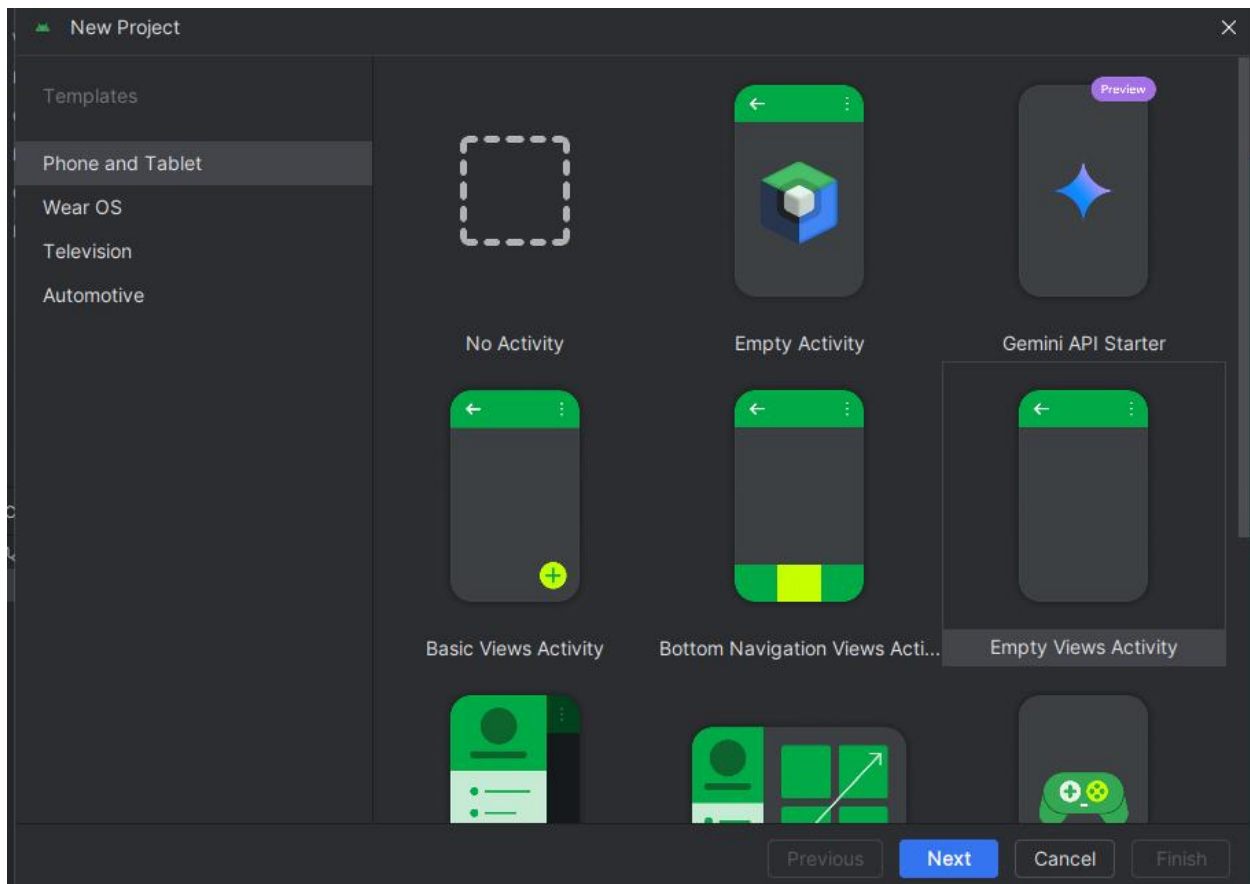
✓ **Bước 8:** Bỏ chọn **Include Instant App support** và tất cả các tùy chọn khác, sau đó nhấp **Next**.

- Nếu dự án của bạn yêu cầu thêm thành phần cho **SDK đã chọn, Android Studio** sẽ tự động cài đặt.

✓ **Bước 9:** Màn hình **Add an Activity** sẽ xuất hiện.

- Một **Activity** là một phần quan trọng trong bất kỳ ứng dụng Android nào. Nó đại diện cho một tác vụ cụ thể mà người dùng có thể thực hiện.
- **Activity** thường có một **layout** đi kèm để xác định cách hiển thị các thành phần giao diện (**UI elements**) trên màn hình.
- Android Studio cung cấp nhiều mẫu **Activity** để giúp bạn bắt đầu nhanh chóng.

✓ Đối với dự án **Hello World**, hãy chọn **Empty Activity** như trong hình minh họa, sau đó nhấp **Next**.



✓ **Bước 10:** Khi màn hình **Configure Activity** xuất hiện, bạn sẽ thấy các tùy chọn cấu hình khác nhau (tùy thuộc vào mẫu **Activity** bạn đã chọn trước đó).

- Theo mặc định, Activity trống (**Empty Activity**) sẽ được đặt tên là **MainActivity**.
- Bạn có thể đổi tên nếu muốn, nhưng bài học này sử dụng **MainActivity**, vì vậy hãy giữ nguyên mặc định.

✓ **Bước 11:** Đảm bảo rằng tùy chọn **Generate Layout file** được chọn.

- *Tên mặc định của tệp layout là **activity_main.xml**.*
- *Bạn có thể đổi tên, nhưng bài học này sử dụng **activity_main.xml**, nên hãy giữ nguyên.*

✓ **Bước 12:** Đảm bảo rằng tùy chọn **Backwards Compatibility (App Compat)** được chọn.

- Tùy chọn này giúp ứng dụng *tương thích ngược* với các phiên bản Android cũ hơn.

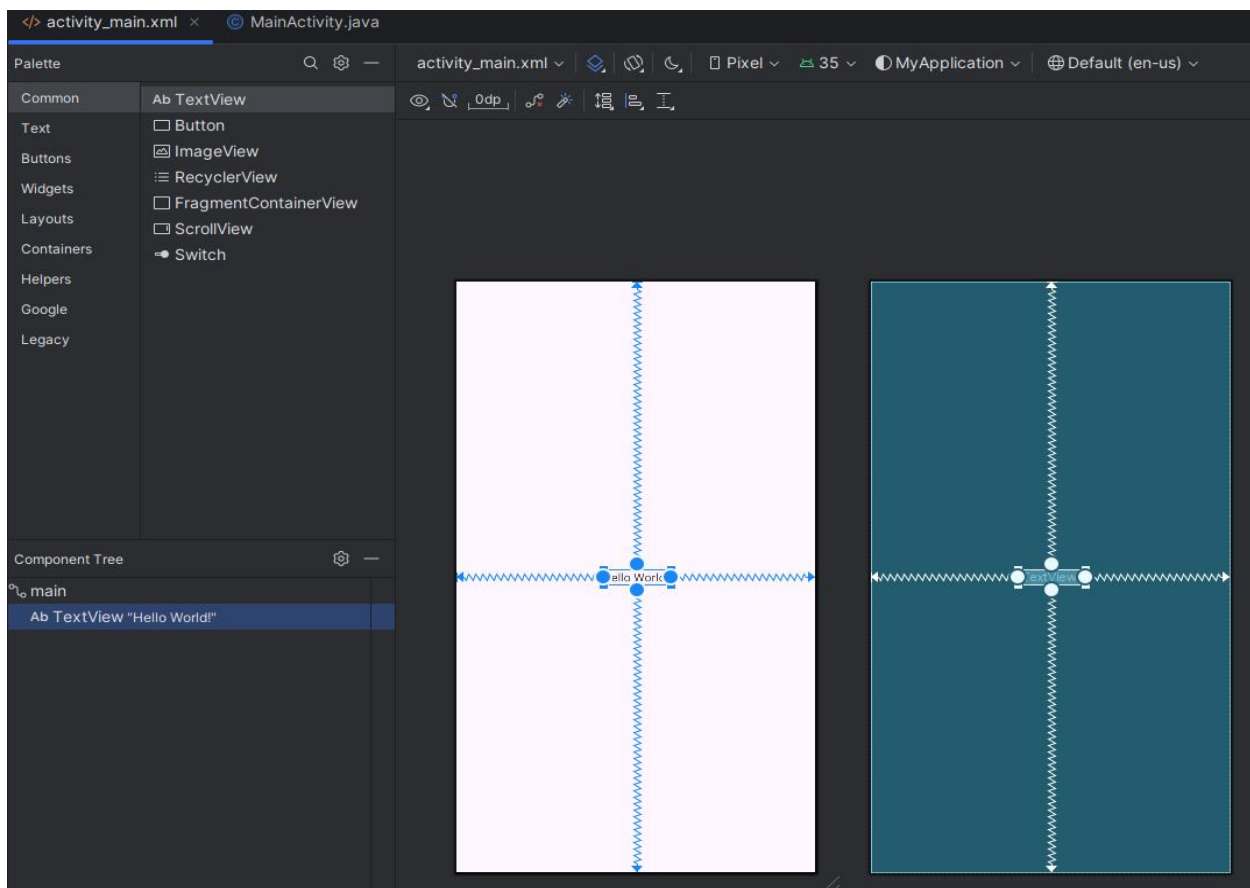
✓ **Bước 13:** Nhấp **Finish** để hoàn tất quá trình tạo dự án.

- *Android Studio* sẽ tạo thư mục dự án và *biên dịch* dự án bằng *Gradle* (quá trình này có thể mất vài phút).
- Nếu xuất hiện hộp thoại "*Tip of the day*", bạn có thể đọc hoặc nhấp *Close* để tắt.

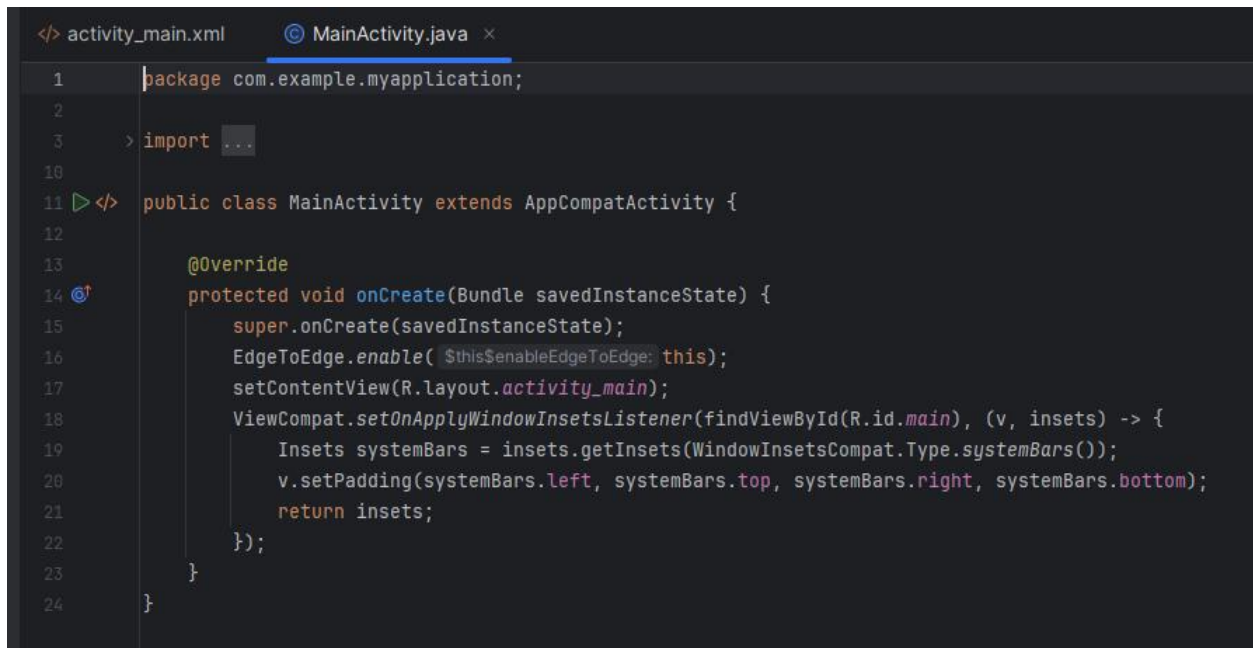
Biên tập viên Android Studio xuất hiện. Thực hiện theo các bước sau:

✓ **Bước 1:** Nhấp vào tab **activity_main.xml** để mở trình chỉnh sửa layout.

✓ **Bước 2:** Nhấp vào tab **Design** (nếu chưa được chọn) để xem **bản xem trước giao diện đồ họa**.



✓ **Bước 3:** Nhấp vào tab `MainActivity.java` để mở trình chỉnh sửa mã nguồn.

A screenshot of the Android Studio code editor showing the MainActivity.java file. The code is in Java and defines the MainActivity class, which extends AppCompatActivity. It includes an onCreate method that calls super.onCreate, enables edge-to-edge, sets the content view to R.layout.activity_main, and sets up window insets. The code is as follows:

```
1 package com.example.myapplication;
2
3 > import ...
10
11 </> public class MainActivity extends AppCompatActivity {
12
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         EdgeToEdge.enable( $this$enableEdgeToEdge: this);
17         setContentView(R.layout.activity_main);
18         ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
19             Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
20             v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
21             return insets;
22         });
23     }
24 }
```

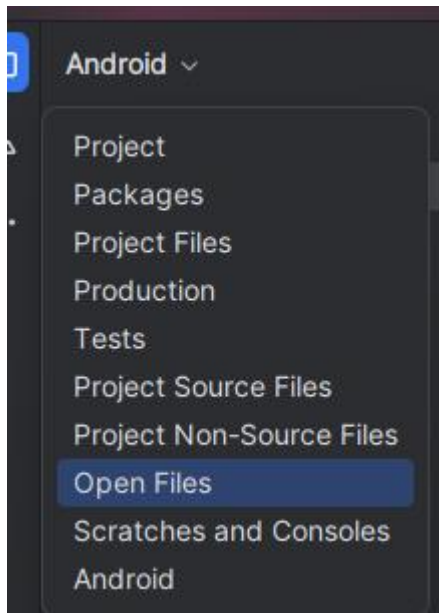
2.2 Khám phá mục Project > Android trong Android Studio

Trong phần này, bạn sẽ tìm hiểu cách tổ chức dự án trong **Android Studio**.

✓ **Bước 1:** Nếu chưa được chọn, nhấp vào tab **Project** ở cột tab dọc bên trái của cửa sổ **Android Studio**.

- Khi đó, khung *Project pane* sẽ xuất hiện, hiển thị cấu trúc của dự án.

✓ **Bước 2:** Ở menu thả xuống trên cùng của **Project pane**, chọn **Android** để hiển thị cấu trúc chuẩn của một dự án Android.

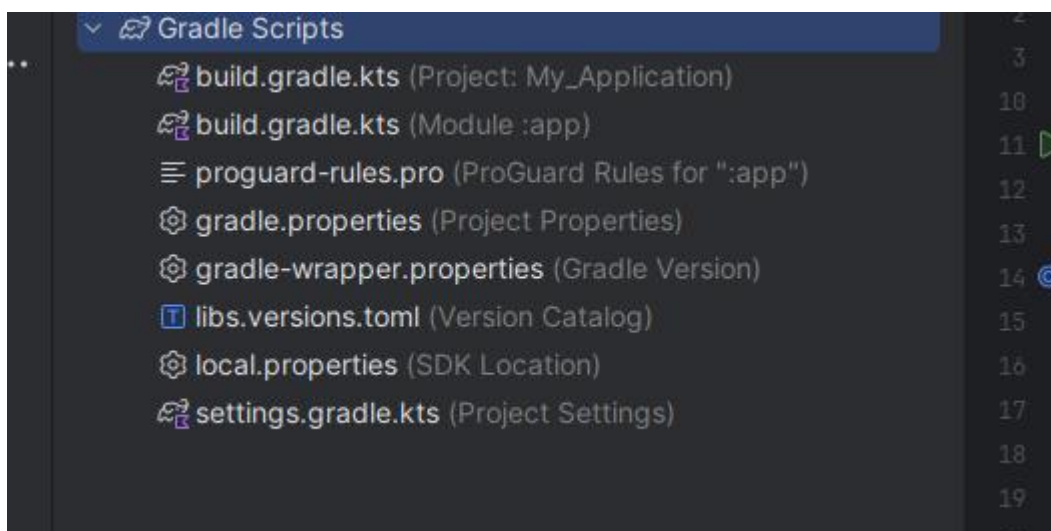


2.3 Khám phá thư mục Gradle Scripts trong Android Studio

Gradle là hệ thống build của **Android Studio**, giúp bạn dễ dàng thêm **thư viện bên ngoài** hoặc **các module khác** vào dự án của mình.

✓ **Lưu ý:** Khi **Project pane** được đặt ở chế độ **Android**, nó được gọi là **Project > Android pane** trong tài liệu hướng dẫn.

✓ Khi bạn tạo một dự án mới, **Gradle Scripts** sẽ được **mở rộng tự động** trong **Project > Android pane**.



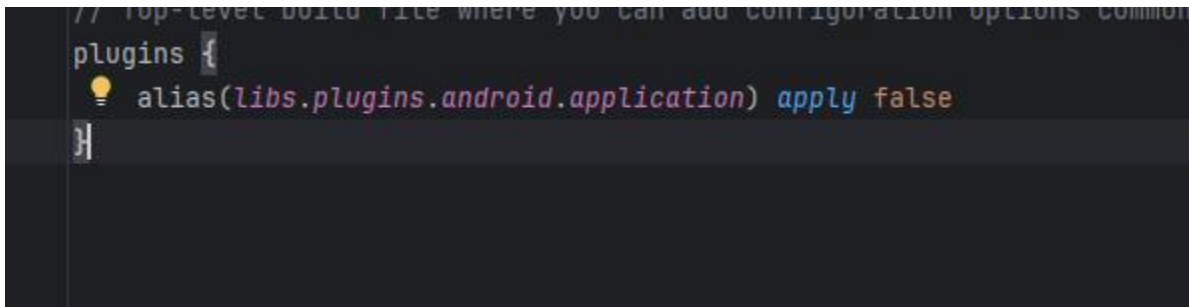
Làm theo các bước sau để tìm hiểu hệ thống **Gradle**:

✓ **Bước 1:** Nếu thư mục **Gradle Scripts** chưa được mở rộng, hãy **nhấp vào biểu tượng tam giác** để mở nó.

- *Thư mục này chứa tất cả các tệp cần thiết cho hệ thống build.*

✓ **Bước 2:** Tìm tệp **build.gradle (Project: HelloWorld)**.

- *Đây là nơi chứa các tùy chọn cấu hình chung cho tất cả các module trong dự án.*
- *Mỗi dự án Android Studio chỉ có một tệp build Gradle cấp cao nhất.*
- *Thông thường, bạn không cần chỉnh sửa tệp này, nhưng việc hiểu nội dung của nó sẽ hữu ích.*



✓ **Bước 3:** Tìm tệp **build.gradle (Module: app)** trong thư mục **Gradle Scripts**.

- *Tệp này dùng để cấu hình cài đặt build cho từng module cụ thể (trong dự án HelloWorld, chỉ có một module duy nhất).*
- *Bạn có thể tùy chỉnh các tùy chọn đóng gói, như build types (loại build) hoặc product flavors (biến thể sản phẩm).*
- *Có thể ghi đè một số cài đặt trong AndroidManifest.xml hoặc build.gradle cấp dự án.*

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

You can use the Project Structure dialog to view and edit your project configuration

```
1 |plugins {
2 |    ⚡ alias(libs.plugins.android.application)
3 |}
4 |
5 |android {
6 |    namespace = "com.example.myapplication"
7 |    compileSdk = 35
8 |
9 |    defaultConfig {
10 |        applicationId = "com.example.myapplication"
11 |        minSdk = 24
12 |        targetSdk = 35
13 |        versionCode = 1
14 |        versionName = "1.0"
15 |
16 |        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
17 |    }
18 |
19 |    buildTypes {
20 |        release {
21 |            isMinifyEnabled = false
22 |            proguardFiles(
23 |                getDefaultProguardFile( name: "proguard-android-optimize.txt"),
24 |                "proguard-rules.pro"
25 |            )
26 |        }
27 |    }
28 |
29 |    compileOptions {
30 |        sourceCompatibility = JavaVersion.VERSION_11
31 |        targetCompatibility = JavaVersion.VERSION_11
32 |    }
33 |}
34 |dependencies {
35 |
36 |    implementation(libs.appcompat)
37 |    implementation(libs.material)
38 |    implementation(libs.activity)
39 |    implementation(libs.constraintlayout)
40 |    testImplementation(libs.junit)
```

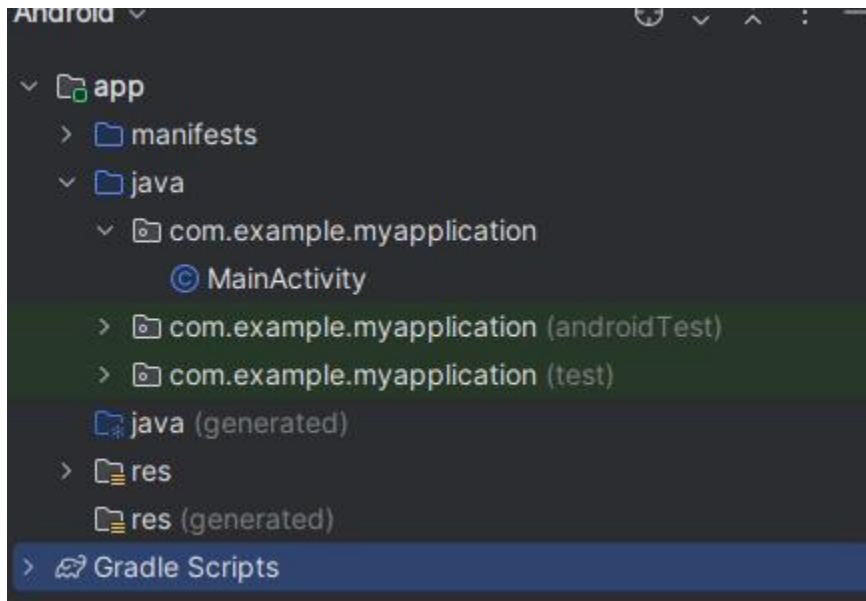
✓ Bước 4: Nhấp vào tam giác để đóng các tập lệnh Gradle

2.4 Khám phá thư mục `app` và `res` trong Android Studio

Tất cả **mã nguồn** và **tài nguyên** của ứng dụng đều nằm trong thư mục `app` và `res`.

✓ **Bước 1:** • Mở rộng thư mục **app**, thư mục **java**, và thư mục **com.example.android.helloworld** để xem tệp **MainActivity.java**.

• Nhấp đúp vào tệp để mở nó trong trình chỉnh sửa mã (code editor).



Thư mục **java** bao gồm các tệp lớp Java trong ba thư mục con, như hình minh họa ở trên. Thư mục **com.example.hello.helloworld** (hoặc tên miền mà bạn đã chỉ định) chứa tất cả các tệp của một gói ứng dụng. Hai thư mục còn lại được sử dụng để kiểm thử và sẽ được mô tả trong một bài học khác.

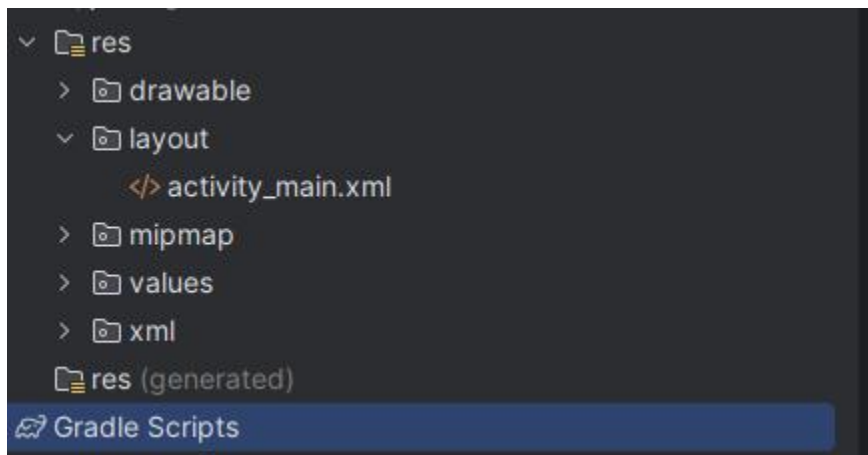
Đối với ứng dụng **Hello World**, chỉ có một gói và nó chứa tệp **MainActivity.java**. Tên của **Activity đầu tiên** (màn hình đầu tiên mà người dùng nhìn thấy), đồng thời khởi tạo các tài nguyên dùng chung trong ứng dụng, theo thông lệ sẽ được đặt là **MainActivity** (phần mở rộng tệp bị ẩn trong **Project > Android pane**).

✓ **Bước 2:** Mở rộng thư mục **res**, sau đó mở rộng thư mục **layout**, và nhấp đúp vào tệp **activity_main.xml** để mở nó trong trình chỉnh sửa giao diện (layout editor).

Thư mục **res** chứa các tài nguyên như **bố cục (layouts)**, **chuỗi văn bản (strings)**, và **hình ảnh (images)**.

Một **Activity** thường được liên kết với một **bố cục giao diện người dùng (UI views)**, được định nghĩa trong một tệp **XML**.

Tệp này thường được đặt tên theo **tên của Activity** mà nó liên kết.



Thư mục **res** chứa các tài nguyên như **bố cục (layouts)**, **chuỗi văn bản (strings)**, và **hình ảnh (images)**. Một **Activity** thường được liên kết với một **bố cục giao diện người dùng (UI views)**, được định nghĩa trong một tệp **XML**. Tệp này thường được đặt tên theo **tên của Activity** mà nó liên kết.

2.5 Khám phá thư mục `manifests`

Thư mục **manifests** chứa các tệp cung cấp thông tin quan trọng về ứng dụng cho **hệ thống Android**. Hệ thống cần thông tin này trước khi có thể chạy bất kỳ mã nào của ứng dụng.

Các bước thực hiện:

1 Mở rộng thư mục `manifests`.

2 Nhấp đúp vào tệp `AndroidManifest.xml` để mở nó.

Giới thiệu về `AndroidManifest.xml`

- Đây là tệp mô tả **tất cả các thành phần** của ứng dụng Android.
- Mọi **thành phần** của ứng dụng (chẳng hạn như mỗi **Activity**) đều **phải được khai báo** trong tệp này.
- Trong các bài học tiếp theo, bạn sẽ sửa đổi tệp này để **thêm tính năng và cấp quyền** cho ứng dụng. Tham khảo thêm: [App Manifest Overview](#)

Task 3: Sử dụng thiết bị ảo (emulator)

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

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

1.2) Giao diện người dùng tương tác đầu tiên

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