

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	30
1.3)	Trình chỉnh sửa bố cục	38
1.4)	Văn bản và các chế độ cuộn	38
1.5)	Tài nguyên có sẵn	38
Bài 2)	Activities	38
2.1)	Activity và Intent	38
2.2)	Vòng đời của Activity và trạng thái	38
2.3)	Intent ngầm định.....	38
Bài 3)	Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ.....	38
3.1)	Trình gỡ lỗi.....	38
3.2)	Kiểm thử đơn vị.....	38
3.3)	Thư viện hỗ trợ.....	38
CHƯƠNG 2.	Trải nghiệm người dùng.....	39
Bài 1)	Tương tác người dùng.....	39
1.1)	Hình ảnh có thể chọn	39
1.2)	Các điều khiển nhập liệu	39
1.3)	Menu và bộ chọn	39
1.4)	Điều hướng người dùng.....	39
1.5)	RecyclerView.....	39
Bài 2)	Trải nghiệm người dùng thú vị	39
2.1)	Hình vẽ, định kiểu và chủ đề	39
2.2)	Thẻ và màu sắc.....	39

2.3)	Bố cục thích ứng.....	39
Bài 3)	Kiểm thử giao diện người dùng	39
3.1)	Espresso cho việc kiểm tra UI.....	39
CHƯƠNG 3.	Làm việc trong nền	39
Bài 1)	Các tác vụ nền	39
1.1)	AsyncTask	39
1.2)	AsyncTask và AsyncTaskLoader	39
1.3)	Broadcast receivers	39
Bài 2)	Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền.....	39
2.1)	Thông báo.....	39
2.2)	Trình quản lý cảnh báo	39
2.3)	JobScheduler	39
CHƯƠNG 4.	Lưu dữ liệu người dùng	40
Bài 1)	Tùy chọn và cài đặt.....	40
1.1)	Shared preferences	40
1.2)	Cài đặt ứng dụng	40
Bài 2)	Lưu trữ dữ liệu với Room	40
2.1)	Room, LiveData và ViewModel	40
2.2)	Room, LiveData và ViewModel	40
3.1)	Trình gowx loi	

CHƯƠNG 1. LÀM QUEN

Bài 1) Tạo ứng dụng đầu tiên

1.1) Android Studio và Hello World

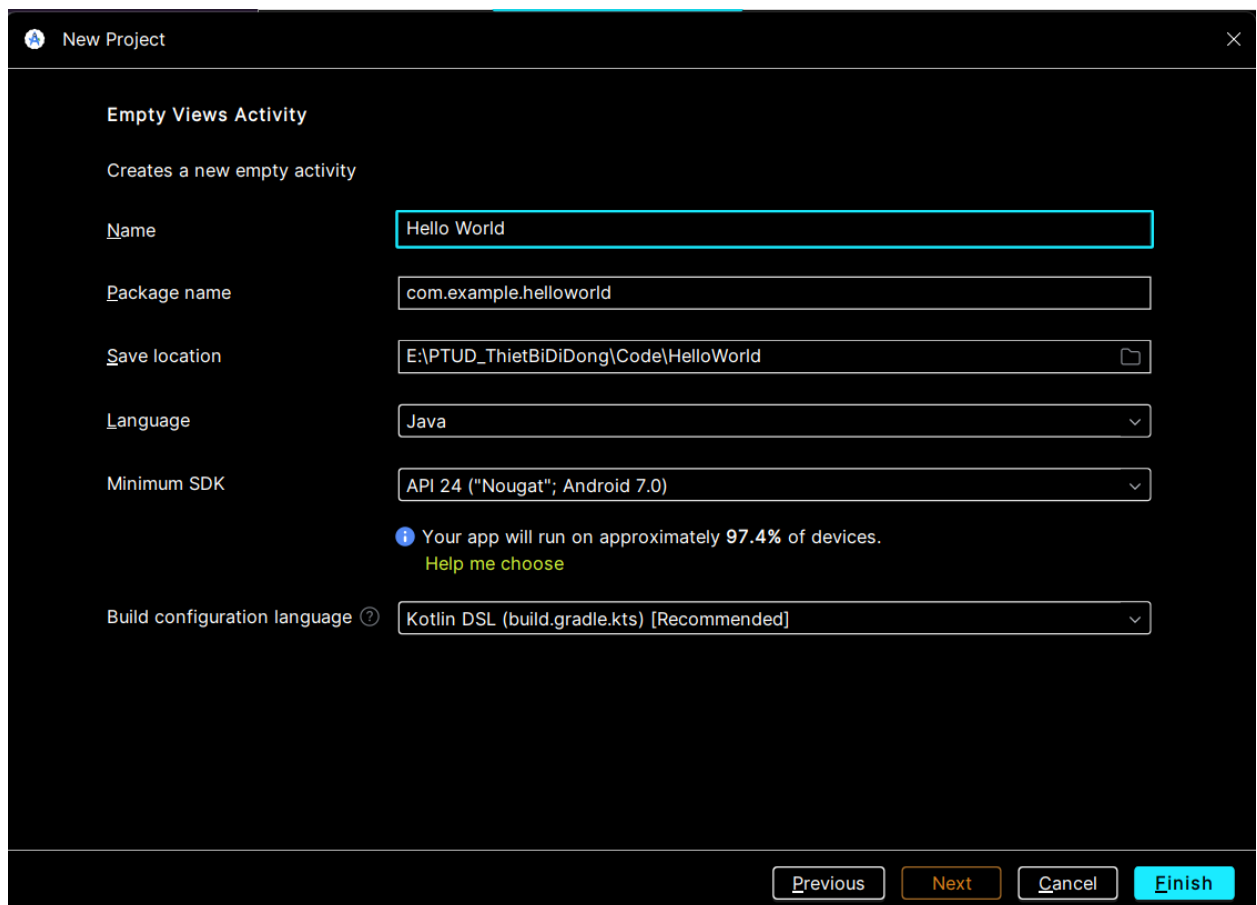
Giới thiệu

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.

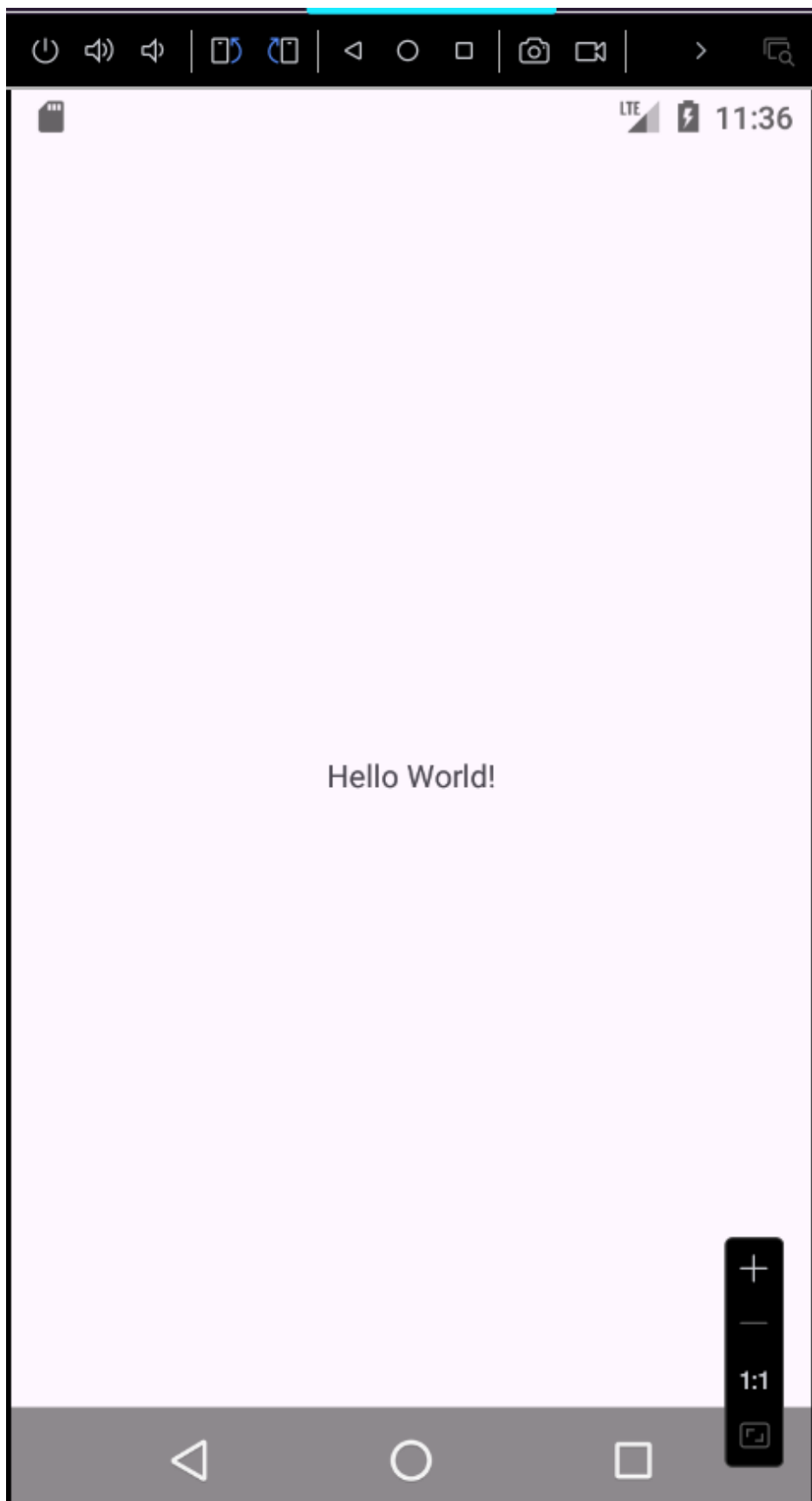
Những gì bạn sẽ làm

- 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**

Giới thiệu về ứng dụng

Sau khi cài đặt thành công Android Studio, bạn sẽ tạo một dự án mới từ mẫu có sẵn cho ứng dụng Hello World. Ứng dụng đơn giản này sẽ hiển thị dòng chữ “Hello World” trên màn hình của thiết bị Android (trên máy ảo và máy thật).

Hình ảnh của ứng dụng hoàn chỉnh sẽ trông như sau:



Bài 1: Cài đặt Android Studio

Android Studio cung cấp một môi trường phát triển thích hợp (IDE) hoàn chỉnh, bao gồm một trình chỉnh sửa 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 hiệu suất, giúp việc phát triển ứng dụng trở nên nhanh chóng và dễ dàng hơn. Bạn có thể kiểm thử ứng dụng trên nhiều trình giả lập được cấu hình sẵn hoặc trên thiết bị di động của chính mình, xây dựng ứng dụng hoàn chỉnh và xuất bản lên cửa hàng Google Play.

Lưu ý: Android Studio liên tục được cải tiến. Để biết thông tin mới nhất về yêu cầu hệ thống và hướng dẫn cài đặt, hãy xem tài liệu chính thức của Android Studio.

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

Để cài đặt và thiết lập Android Studio, trước tiên hãy kiểm tra yêu cầu hệ thống để đảm bảo thiết bị của bạn đáp ứng đủ điều kiện. Quy trình cài đặt tương tự trên tất cả các hệ điều hành, bất kỳ khác biệt nào sẽ được ghi chú riêng.

Các bước cài đặt:

1. Truy cập trang web chính thức của Android Developers và làm theo hướng dẫn để tải xuống và cài đặt Android Studio.
2. Chấp nhận các thiết lập mặc định trong suốt quá trình cài đặt, đồng thời đảm bảo rằng tất cả các thành phần cần thiết đều được chọn để cài đặt.
3. Sau khi quá trình cài đặt hoàn tất, Setup Wizard sẽ tiếp tục tải xuống và cài đặt một số thành phần bổ sung, bao gồm cả Android SDK. Hãy kiên nhẫn, vì quá trình này có thể mất thời gian tùy thuộc vào tốc độ Internet của bạn. Một số bước có thể trông giống nhau nhưng vẫn cần thiết.
4. Quá trình tải xuống hoàn tất, Android Studio sẽ khởi động và bạn đã sẵn sàng để tạo dự án đầu tiên của mình.

Xử lý sự cố:

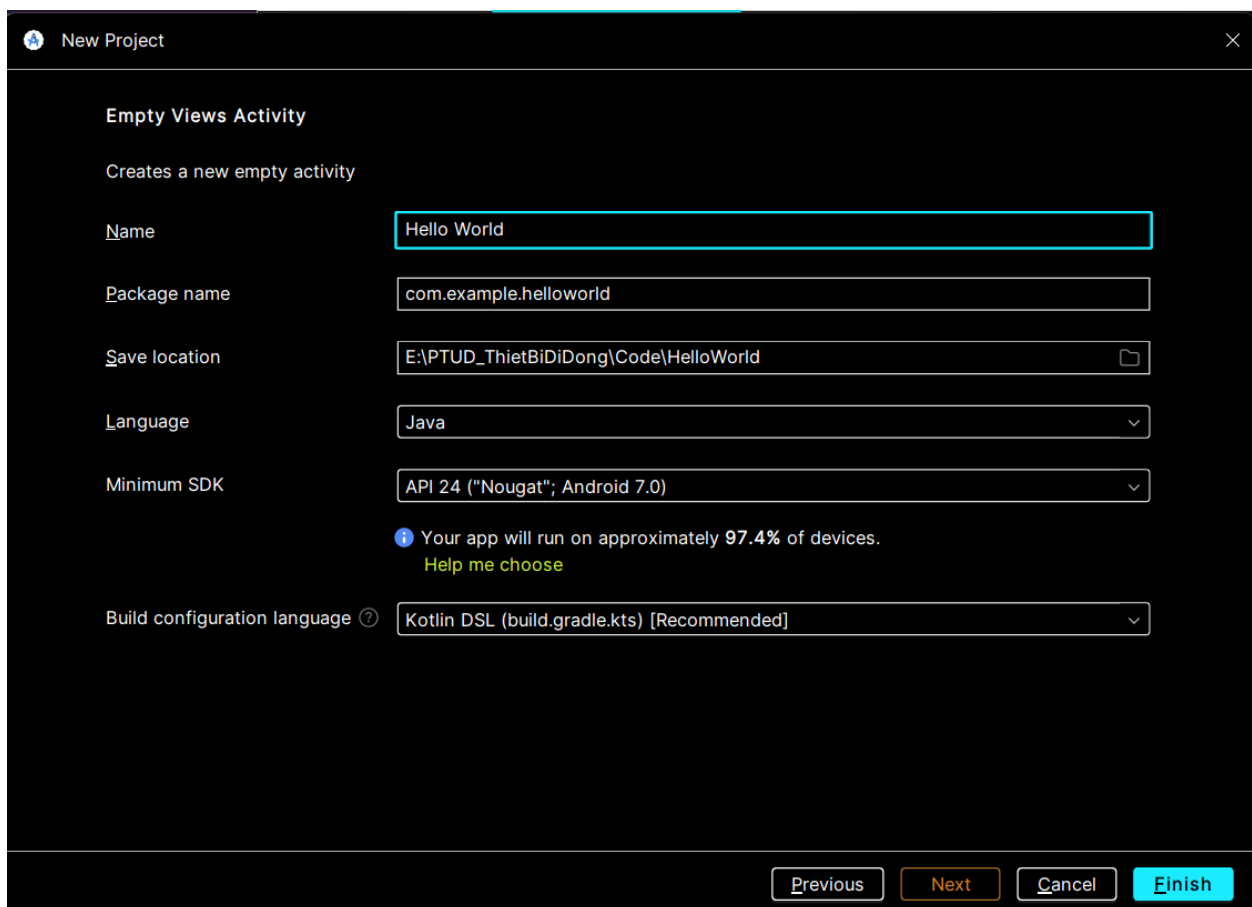
Nếu gặp sự cố khi cài đặt, hãy kiểm tra ghi chú phát hành của Android Studio hoặc tìm sự trợ giúp từ giảng viên của bạn.

Bài 2: Tạo ứng dụng "Hello World"

Trong bài tập này, bạn sẽ tạo một ứng dụng hiển thị dòng chữ "**Hello World**" để kiểm tra xem Android Studio đã được cài đặt đúng chưa, đồng thời làm quen với các bước phát triển ứng dụng trong Android Studio.

2.1 Tạo dự án ứng dụng

1. Mở **Android Studio** nếu nó chưa được mở.
2. Trong cửa sổ **Welcome to Android Studio**, nhấn vào **Start a new Android Studio project** (Bắt đầu một dự án Android Studio mới).
3. Trong cửa sổ **Create Android Project**, nhập **Hello World** vào ô **Application name** (Tên ứng dụng).

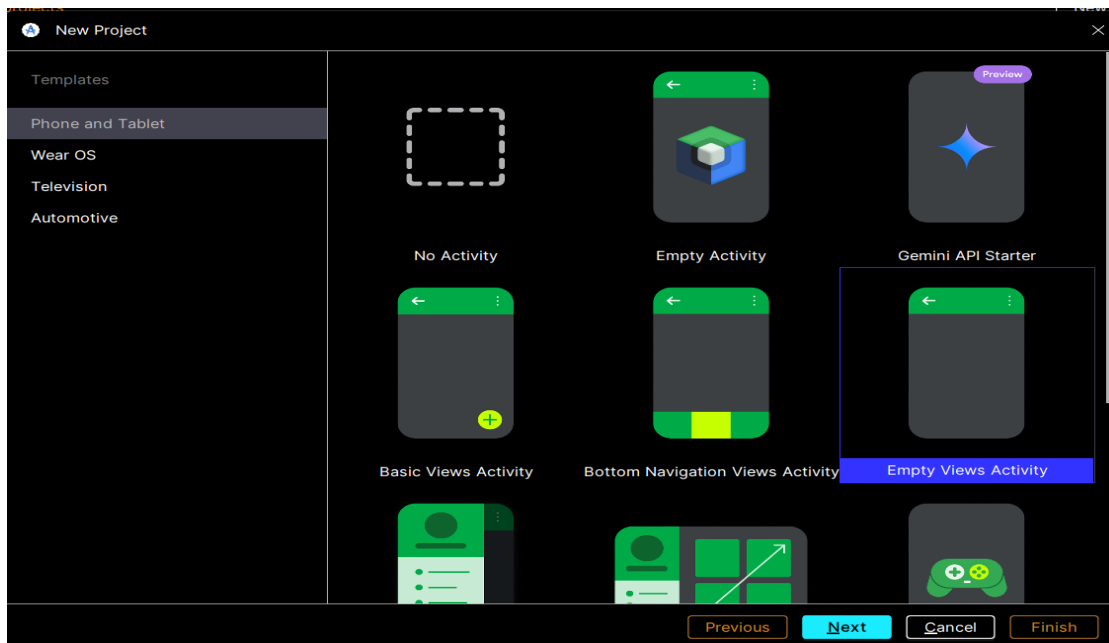


4. Kiểm tra xem Project location (vị trí lưu dự án) mặc định có đúng nơi bạn muốn lưu ứng dụng Hello World và các dự án Android Studio khác không. Nếu cần, hãy thay đổi sang thư mục bạn muốn.

- Giữ nguyên `android.example.com` làm Company Domain (tên miền công ty) hoặc tạo một tên miền riêng.

Nếu bạn không có ý định xuất bản ứng dụng, có thể giữ nguyên giá trị mặc định. Lưu ý rằng việc thay đổi package name (tên gói) sau này sẽ tốn thêm công sức.

- Bỏ chọn các tùy chọn Include C++ support (Hỗ trợ C++) và Include Kotlin support (Hỗ trợ Kotlin), sau đó nhấn Next.
- Ở màn hình Target Android Devices, đảm bảo rằng Phone and Tablet (Điện thoại và Máy tính bảng) được chọn. Kiểm tra xem API 15: Android 4.0.3 IceCreamSandwich có được đặt làm Minimum SDK (SDK tối thiểu) không; nếu không, hãy sử dụng menu thả xuống để đặt lại. Các bài học trong khóa học này sử dụng các cài đặt này, giúp ứng dụng Hello World tương thích với 97% thiết bị Android đang hoạt động trên Google Play Store.
- Bỏ chọn Include Instant App support (Hỗ trợ Instant App) và tất cả các tùy chọn khác, sau đó nhấn Next. Nếu dự án của bạn yêu cầu cài đặt thêm thành phần cho SDK mục tiêu, Android Studio sẽ tự động cài đặt chúng.
- Cửa sổ Add an Activity xuất hiện. Activity là một màn hình giao diện trong ứng dụng, nơi người dùng có thể thực hiện một hành động cụ thể. Mỗi Activity thường có một layout (bố cục giao diện) xác định cách hiển thị các thành phần UI trên màn hình. Android Studio cung cấp các mẫu Activity để giúp bạn bắt đầu nhanh hơn. Đối với dự án Hello World, hãy chọn Empty Activity (Hoạt động trống), sau đó nhấn Next.



10. Màn hình Configure Activity xuất hiện (có thể khác nhau tùy vào mẫu bạn chọn ở bước trước). Mặc định, Activity được đặt tên là MainActivity. Bạn có thể đổi nếu muốn, nhưng bài học này sử dụng MainActivity.
11. Đảm bảo rằng tùy chọn Generate Layout file (Tạo tệp bố cục) được chọn. Tên bố cục mặc định là activity_main. Bạn có thể đổi nếu muốn, nhưng bài học này sử dụng activity_main.
12. Đảm bảo rằng tùy chọn Backwards Compatibility (App Compat) (Hỗ trợ tương thích ngược) được chọn. Điều này giúp ứng dụng tương thích với các phiên bản Android cũ hơn.
13. Nhấn Finish để hoàn tất quá trình tạo dự án.

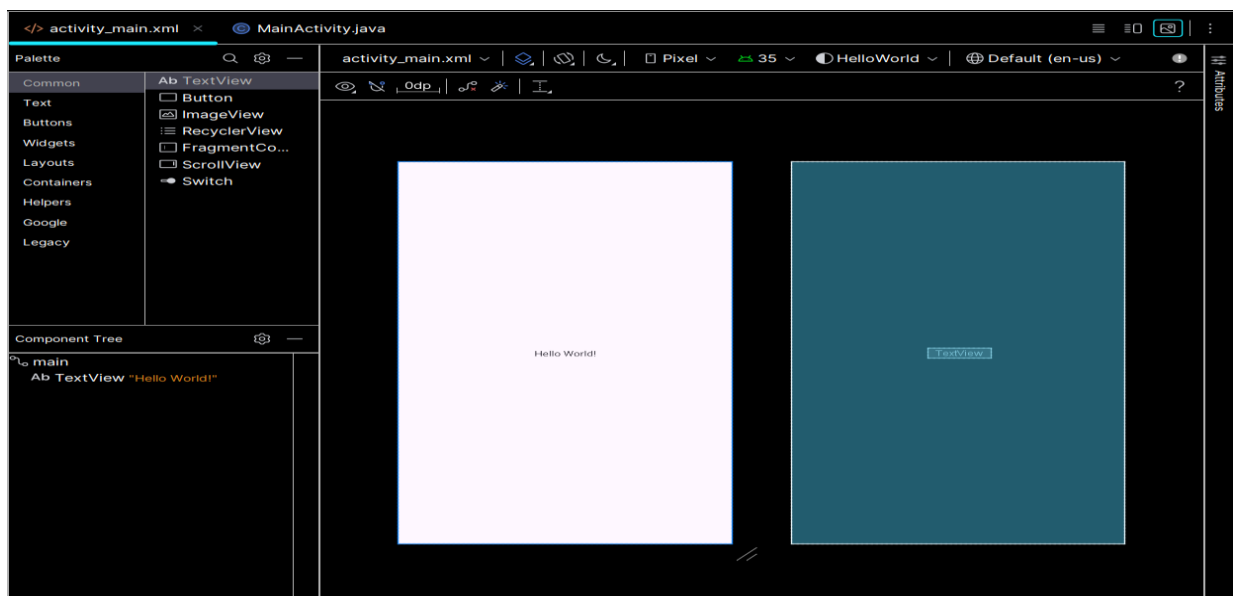
Android Studio tạo một thư mục cho các dự án của bạn và xây dựng dự án bằng Gradle (quá trình này có thể mất một vài phút).

Mẹo: Xem trang dành cho nhà phát triển **Cấu hình bản dựng (Configure your build)** để biết thông tin chi tiết.

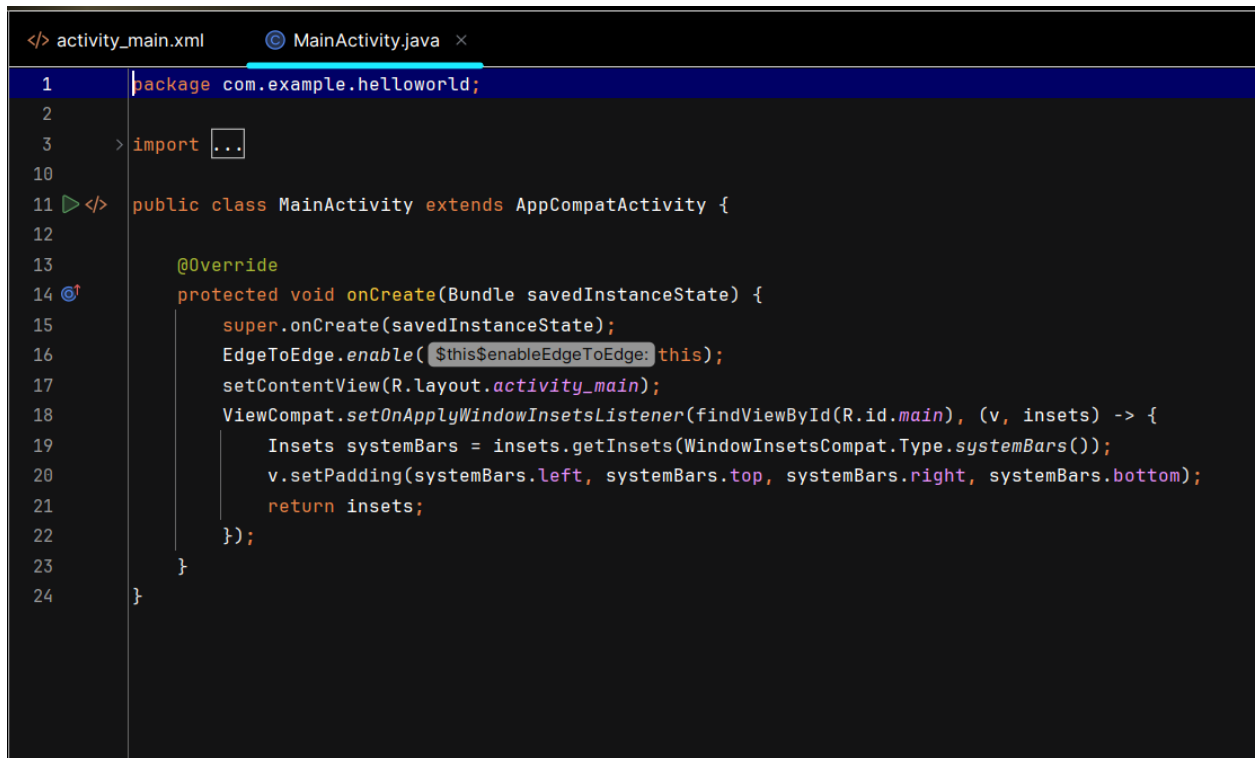
Bạn cũng có thể thấy một thông báo "Mẹo trong ngày" (Tip of the day) với các phím tắt và mẹo hữu ích khác. Nhấn **Close** để đóng thông báo.

Sau đó, trình soạn thảo của Android Studio sẽ xuất hiện. Hãy làm theo các bước sau:

1. Nhấp vào tab **activity_main.xml** để xem trình chỉnh sửa bố cục (layout editor).
2. Nhấp vào tab **Design** trong trình chỉnh sửa bố cục (nếu chưa được chọn) để hiển thị bản xem trước đồ họa của bố cục như hình bên dưới.



3. Nhấp vào tab **MainActivity.java** để xem trình chỉnh sửa mã nguồn như hình bên dưới.

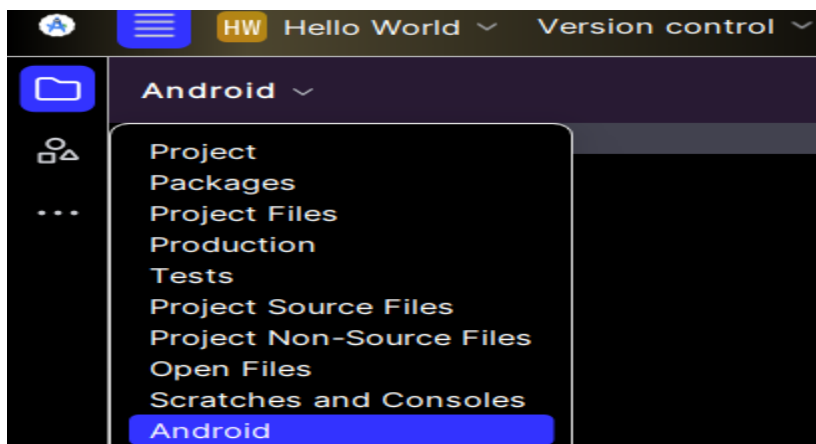
A screenshot of the MainActivity.java file in an Android Studio editor. The code is written in Java and defines the MainActivity class, which extends AppCompatActivity. The onCreate method is overridden to initialize the app, including enabling edge-to-edge display, setting the content view to R.layout.activity_main, and applying window insets to the main view.

```
1 package com.example.helloworld;
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);
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 ngăn Project > Android

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

1. Nếu chưa được chọn, hãy nhấp vào tab **Project** trong cột tab dọc ở bên trái cửa sổ Android Studio. Ngăn **Project** sẽ xuất hiện.
2. Để xem dự án theo cấu trúc tiêu chuẩn của Android, chọn **Android** từ menu thả xuống ở đầu ngăn **Project**, như hình bên dưới.

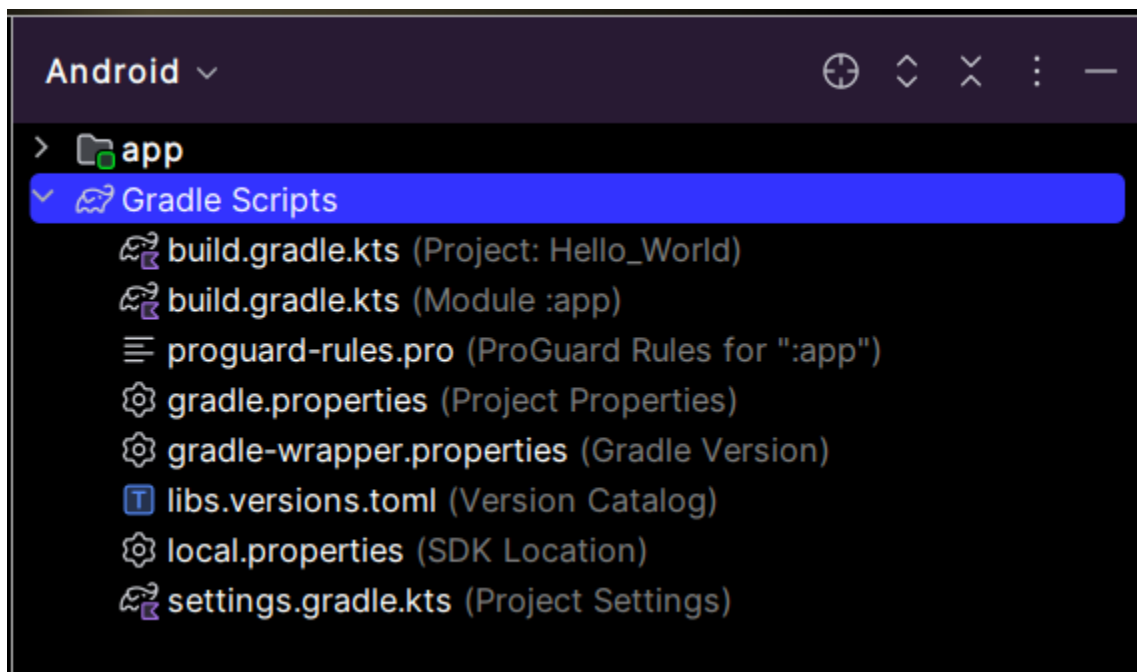


Lưu ý: Chương này và các chương khác sẽ gọi ngắn **Project** khi được đặt ở chế độ **Android** là **Project > Android**.

2.3 Khám phá thư mục Gradle Scripts

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

Khi bạn lần đầu tiên tạo một dự án ứng dụng, ngăn **Project > Android** xuất hiện với thư mục **Gradle Scripts** được mở rộng như hình dưới đây.



Hãy làm theo các bước sau để khám phá hệ thống Gradle:

1. Nếu thư mục **Gradle Scripts** chưa được mở rộng, hãy nhấp vào tam giác để mở rộng nó.
 - Thư mục này chứa tất cả các tệp cần thiết cho hệ thống xây dựng.
2. Tìm tệp **build.gradle (Project: HelloWorld)**.

Đây là nơi bạn sẽ tìm thấy các tùy chọn cấu hình chung cho tất cả các mô-đun tạo nên dự án của bạn.

Mỗi dự án trong Android Studio chứa một tệp Gradle build cấp cao nhất duy nhất. Hầu hết thời gian, bạn sẽ không cần thực hiện bất kỳ thay đổi nào đối với tệp này, nhưng việc hiểu nội dung của nó vẫn hữu ích.

Theo mặc định, tệp build cấp cao nhất sử dụng khối **buildscript** để xác định các kho lưu trữ Gradle và dependencies chung cho tất cả các mô-đun trong dự án. Khi dependency của bạn không phải là một thư viện cục bộ hoặc một thư mục tệp, Gradle sẽ tìm kiếm các tệp trong bất kỳ kho lưu trữ trực tuyến nào được chỉ định trong khối **repositories** của tệp này. Theo mặc định, các dự án Android Studio mới khai báo **JCenter** và **Google** (bao gồm **Google Maven repository**) làm vị trí kho lưu trữ.

```
1 // Top-level build file where you can add configuration options common to all sub-projects/modules.
2 plugins {
3     alias(libs.plugins.android.application) apply false
4 }
```

3. Tìm tệp **build.gradle (Module: app)**.

Ngoài tệp build.gradle cấp dự án, mỗi mô-đun có một tệp **build.gradle** riêng, cho phép bạn cấu hình các cài đặt bản dựng cho từng mô-đun cụ thể (ứng dụng HelloWorld chỉ có một mô-đun). Việc cấu hình các cài đặt bản dựng này cho phép bạn cung cấp các tùy chọn đóng gói tùy chỉnh, chẳng hạn như các loại bản dựng bổ sung và product flavors. Bạn cũng có thể ghi đè các cài đặt trong tệp **AndroidManifest.xml** hoặc tệp **build.gradle** cấp cao nhất.

Đây là tệp thường xuyên được chỉnh sửa nhất khi thay đổi các cấu hình cấp ứng dụng, chẳng hạn như khai báo dependencies trong phần **dependencies**. Bạn có thể khai báo một thư viện phụ thuộc bằng một trong số các cấu hình dependency khác nhau. Mỗi cấu hình dependency cung cấp cho Gradle các hướng dẫn khác nhau về cách sử dụng thư viện. Ví dụ, câu lệnh:
implementation fileTree(dir: 'libs', include: ['*.jar'])

thêm tất cả các tệp “.jar” bên trong thư mục **libs** làm dependency.

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

```

1  plugins {
2      alias(libs.plugins.android.application)
3  }
4
5  android {
6      namespace = "com.example.helloworld"
7      compileSdk = 35
8
9      defaultConfig {
10         applicationId = "com.example.helloworld"
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     compileOptions {
29         sourceCompatibility = JavaVersion.VERSION_11
30         targetCompatibility = JavaVersion.VERSION_11
31     }
32 }

```

```

33  dependencies {
34
35      implementation(libs.appcompat)
36      implementation(libs.material)
37      implementation(libs.activity)
38      implementation(libs.constraintlayout)
39      testImplementation(libs.junit)
40      androidTestImplementation(libs.ext.junit)
41      androidTestImplementation(libs.espresso.core)
42  }
43

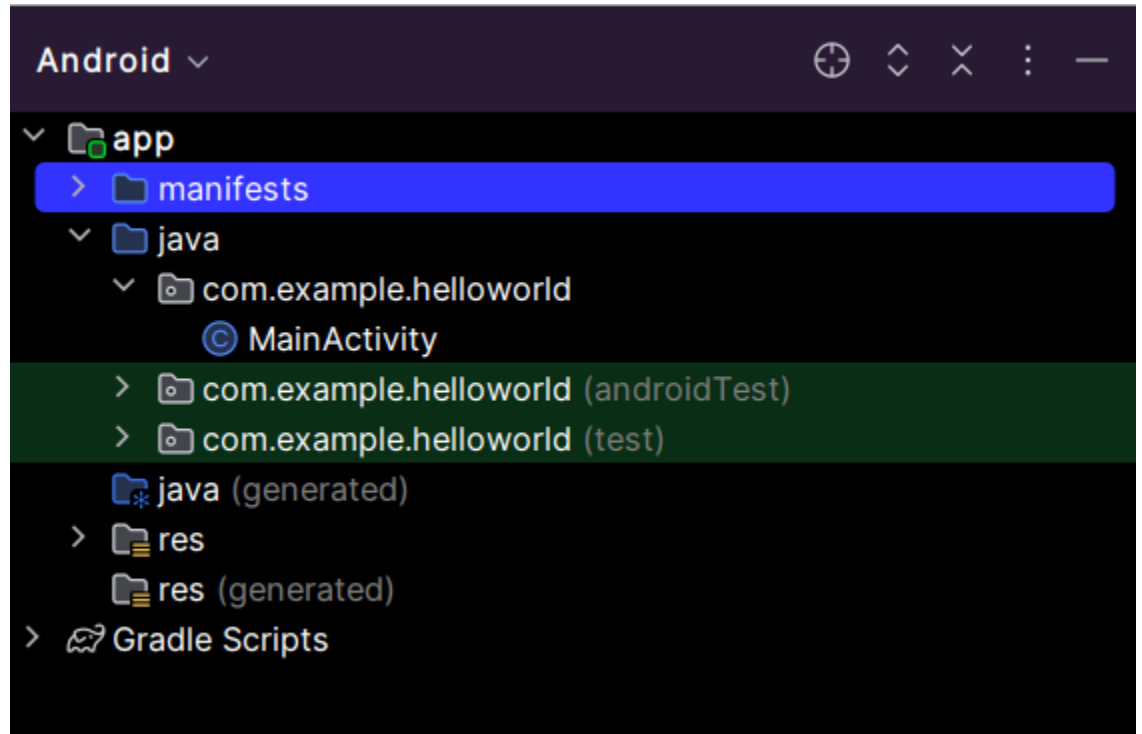
```

4. Nhấp vào tam giác để đóng **Gradle Scripts**.

2.4 Khám phá thư mục app và res

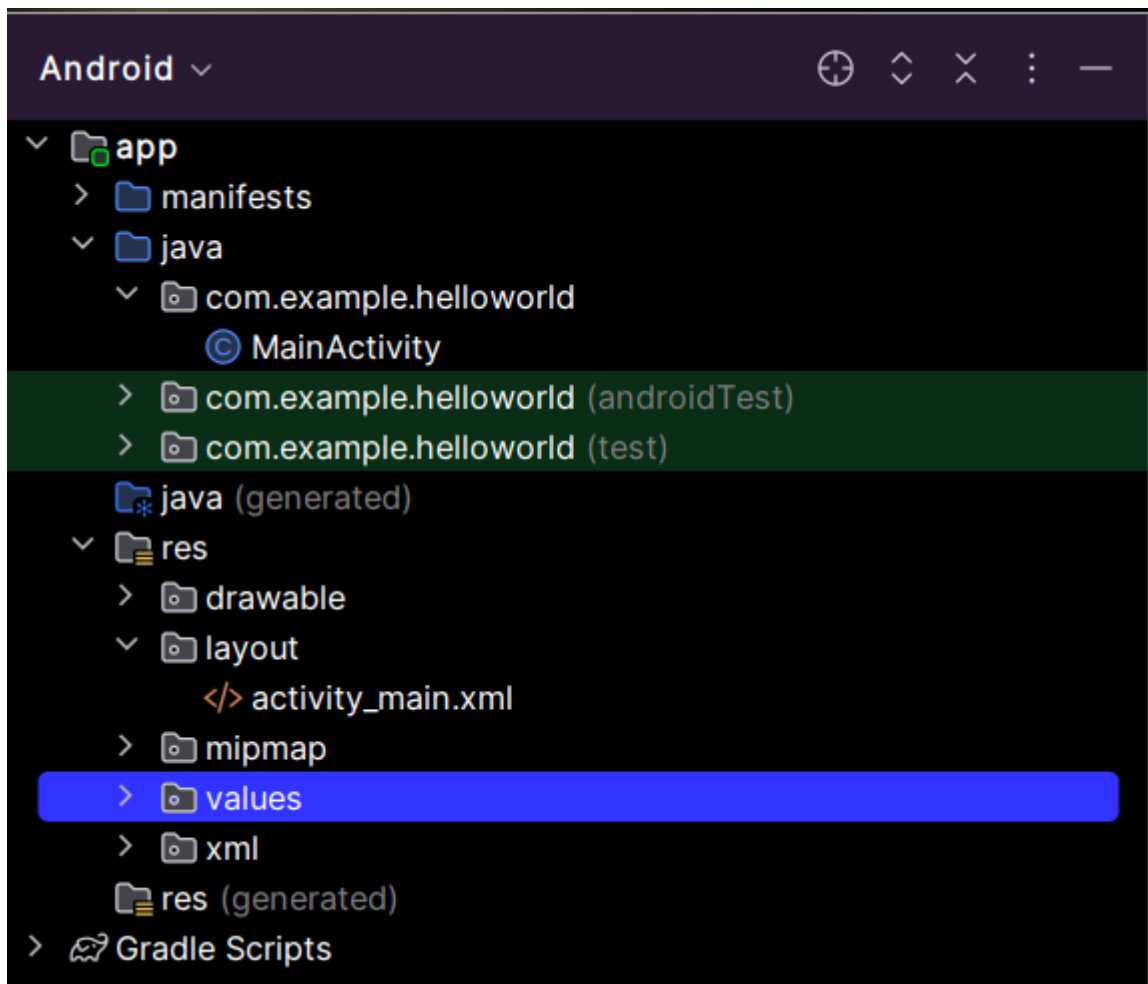
Tất cả mã nguồn và tài nguyên của ứng dụng đều nằm trong các thư mục **app** và **res**.

1. Mở rộng thư mục **app**, sau đó mở rộng thư mục **java** và thư mục **com.example.android.helloworld** để xem tệp **MainActivity.java**.
 - Nhấp đúp vào tệp này để mở nó trong trình chỉnh sửa mã nguồn.



- Thư mục **java** chứa các tệp lớp Java trong ba thư mục con, như hình minh họa bên trên.
- Thư mục **com.example.hello.helloworld** (hoặc tên miền 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** (màn hình) đầu tiên mà người dùng nhìn thấy, cũng là nơi khởi tạo các tài nguyên toàn cục của ứng dụng, thường được đặt là **MainActivity** (trong ngăn **Project > Android**, phần mở rộng tệp bị ẩn đi).

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 bố cục (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 **Activity** của nó.

2.5 Khám phá thư mục manifests

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

1. Mở rộng thư mục **manifests**.
2. Mở tệp **AndroidManifest.xml**.

Tệp **AndroidManifest.xml** mô tả tất cả các thành phần của ứng dụng Android của bạn.

- Mọi thành phần của ứng dụng, chẳng hạn như mỗi **Activity**, phải được khai báo trong tệp XML này.
- Trong các bài học khác của khóa học, bạn sẽ chỉnh sửa tệp này để thêm các tính năng và quyền (permissions) cho ứng dụng.
- Để tìm hiểu tổng quan, hãy xem **App Manifest Overview**.

Nhiệm vụ 3: Sử dụng thiết bị ảo (trình giả lập)

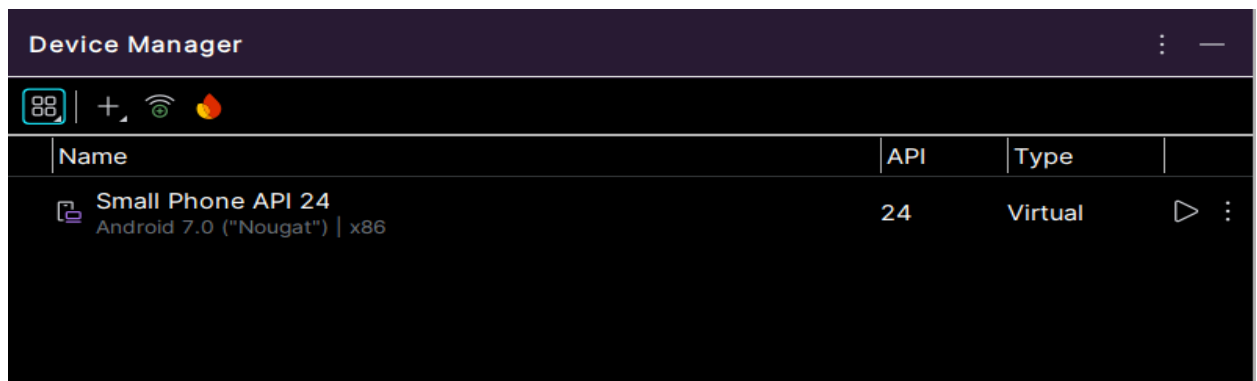
Trong nhiệm vụ này, bạn sẽ sử dụng **Android Virtual Device (AVD) Manager** để tạo một thiết bị ảo (còn gọi là trình giả lập) mô phỏng cấu hình của một loại thiết bị Android cụ thể và sử dụng thiết bị ảo đó để chạy ứng dụng. Lưu ý rằng **Android Emulator** có các yêu cầu bổ sung ngoài các yêu cầu hệ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

Sử dụng **AVD Manager**, bạn có thể xác định các đặc điểm phần cứng của thiết bị, cấp độ API, bộ nhớ, giao diện và các thuộc tính khác, sau đó lưu nó dưới dạng một thiết bị ảo. Với các thiết bị ảo, bạn có thể kiểm thử ứng dụng trên nhiều 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.

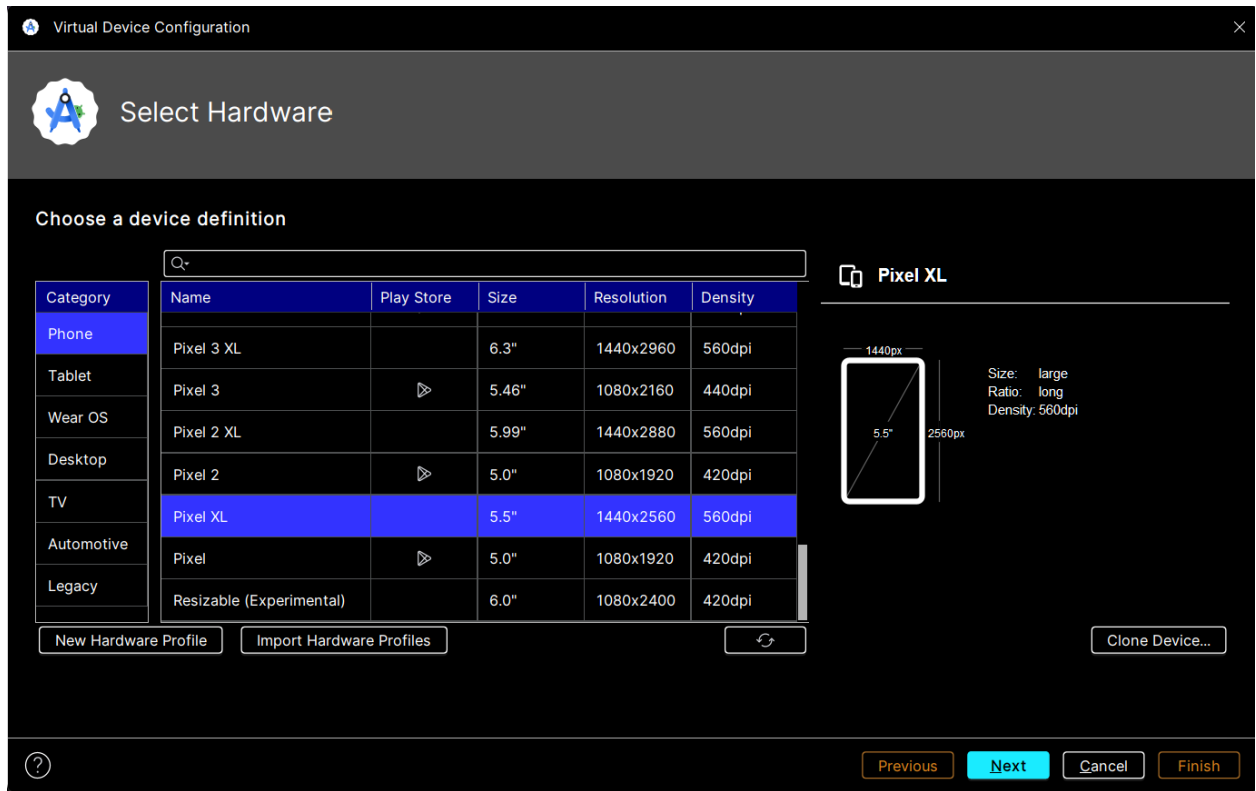
3.1 Tạo một thiết bị ảo Android (AVD)

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

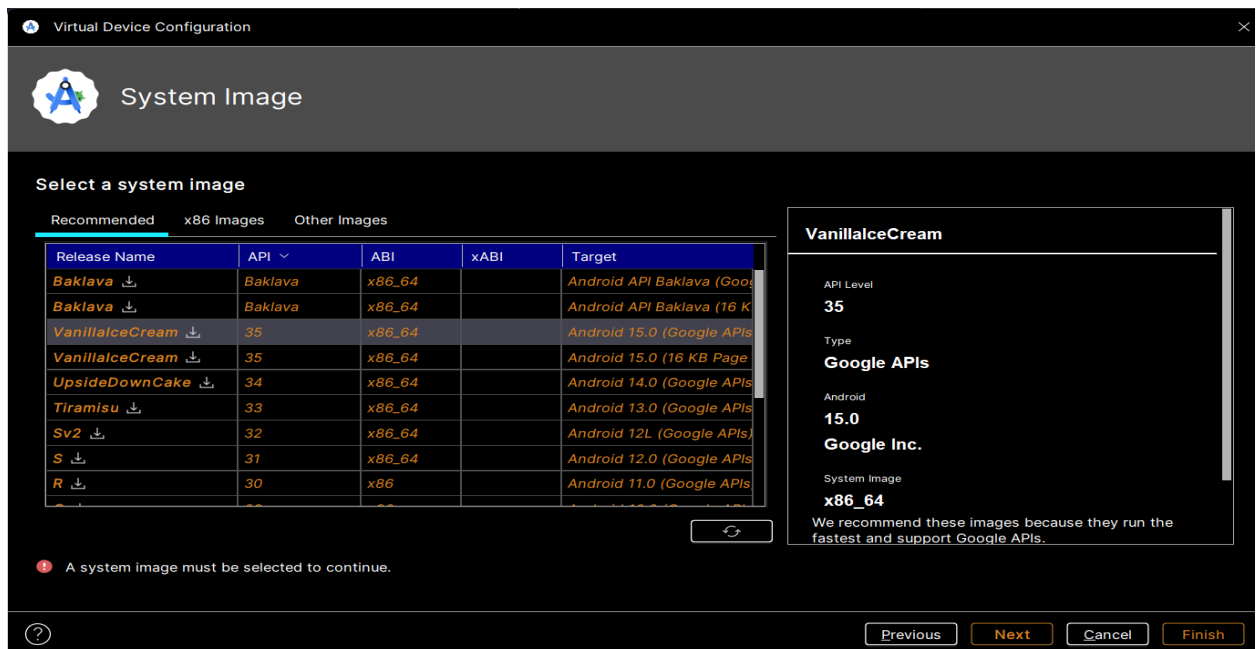
1. Trong **Android Studio**, chọn **Tools > Android > AVD Manager**, hoặc nhấp vào biểu tượng **AVD Manager** trên thanh công cụ.



2. Nhấp vào **+ Create Virtual Device**. Cửa sổ **Select Hardware** xuất hiện, hiển thị danh sách các thiết bị phần cứng được cấu hình sẵn. Mỗi thiết bị có các thông số như kích thước màn hình theo đường chéo (**Size**), độ phân giải màn hình tính theo pixel (**Resolution**), và mật độ điểm ảnh (**Density**).



3. Chọn một thiết bị, chẳng hạn như **Nexus 5X** hoặc **Pixel XL**, rồi nhấp vào **Next**. Màn hình **System Image** xuất hiện.
4. Nhấp vào tab **Recommended** (nếu chưa được chọn) và chọn phiên bản hệ điều hành Android mà bạn muốn chạy trên thiết bị ảo (chẳng hạn như **Oreo**).



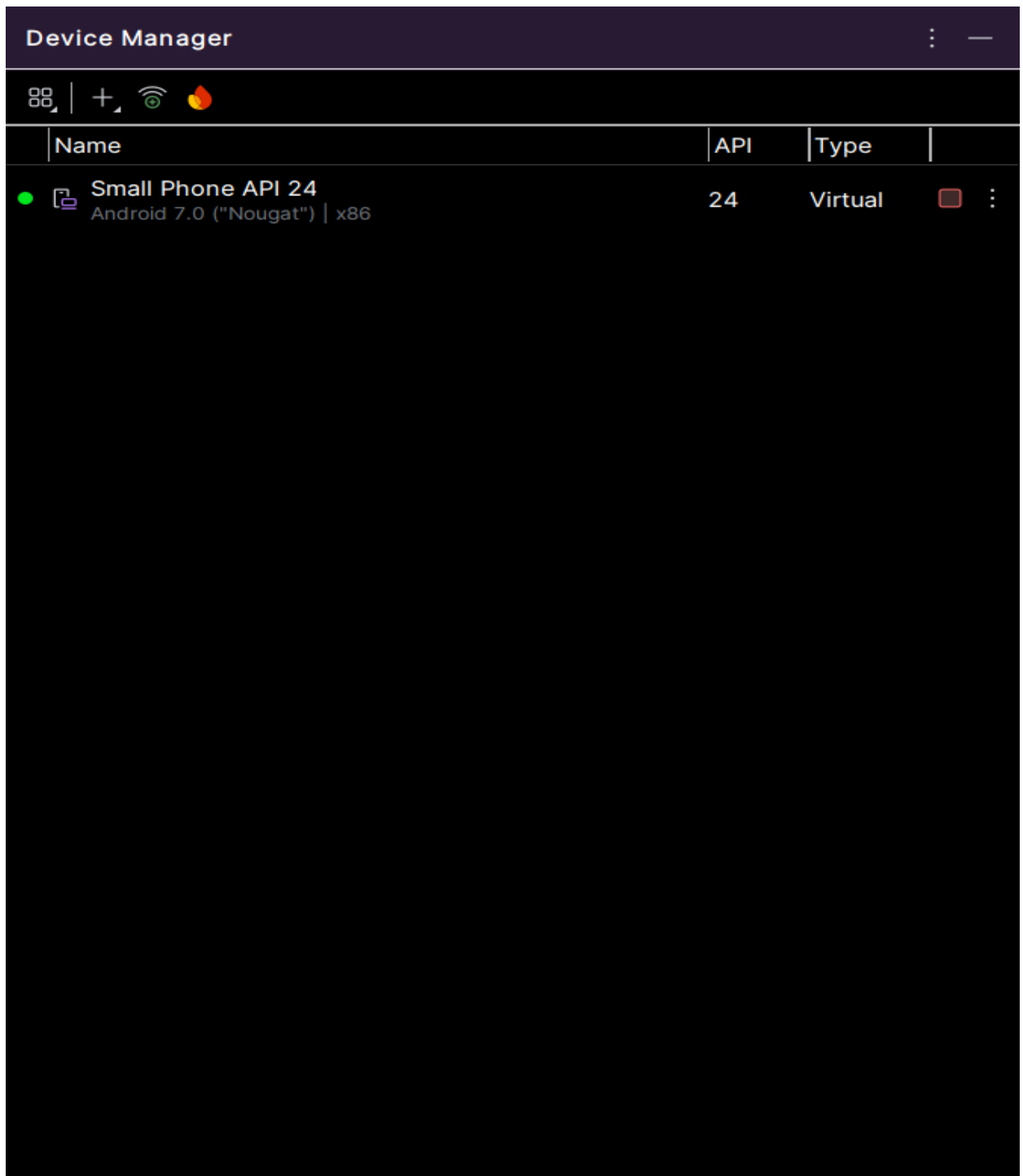
Có nhiều phiên bản hơn so với những gì được hiển thị trong tab **Recommended**. Bạn có thể xem thêm trong các tab **x86 Images** và **Other Images**. Nếu có liên kết **Download** bên cạnh hình ảnh hệ thống mà bạn muốn sử dụng, điều đó có nghĩa là nó chưa được cài đặt. Nhấp vào liên kết để bắt đầu tải xuống, sau đó nhấp **Finish** khi hoàn tất.

- Sau khi chọn xong hệ thống, nhấp vào **Next**. Cửa sổ **Android Virtual Device (AVD)** xuất hiện. Bạn có thể thay đổi tên của thiết bị ảo nếu muốn. Kiểm tra lại cấu hình và nhấp vào **Finish**.

3.2 Chạy ứng dụng trên thiết bị ảo

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



- Trong **Android Studio**, chọn **Run > Run app** hoặc nhấp vào biểu tượng **Run** trên thanh công cụ.
- Cửa sổ **Select Deployment Target** xuất hiện. Trong phần **Available Virtual Devices**, chọn thiết bị ảo mà bạn vừa tạo và nhấp vào **OK**.



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

Bạn sẽ thấy ứng dụng **Hello World** hiển thị như trong hình minh họa bên dưới.



LTE   10:35

Hello World!



Mẹo: Khi kiểm thử trên thiết bị ảo, tốt nhất bạn nên khởi động nó một lần ngay từ đầu phiên làm việc và không đóng nó cho đến khi bạn hoàn thành việc kiểm thử. Điều này giúp tránh việc ứng dụng phải trải qua quá trình khởi động thiết bị nhiều lần.

Để đóng thiết bị ảo, bạn có thể:

- Nhấp vào nút **X** ở góc trên của trình giả lập.
- Chọn **Quit** từ menu.
- Nhấn **Control + Q** trên Windows hoặc **Command + Q** trên macOS.

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

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

b) Bạn cần có:

- Một thiết bị Android, chẳng hạn như điện thoại hoặc máy tính bảng.
- Cáp dữ liệu để kết nối thiết bị Android với máy tính qua cổng USB.
- Nếu bạn đang sử dụng hệ thống **Linux** hoặc **Windows**, có thể cần thực hiện thêm một số bước để chạy ứng dụng trên thiết bị phần cứng.
 - Kiểm tra tài liệu **Using Hardware Devices**.
 - Bạn cũng có thể cần cài đặt **trình điều khiển USB** phù hợp với thiết bị của mình.
 - Đối với trình điều khiển USB trên Windows, hãy xem **OEM USB Drivers**.

4.1 Bật chế độ Gỡ lỗi USB (USB Debugging)

Để **Android Studio** có thể giao tiếp với thiết bị của bạn, bạn phải bật **Gỡ lỗi USB** trên thiết bị Android. Tùy chọn này nằm trong phần **Developer options** của thiết bị.

Trên **Android 4.2 trở lên**, màn hình **Developer options** bị ẩn theo mặc định. Để hiển thị **Developer options** và bật **USB Debugging**, thực hiện như sau:

1. Trên thiết bị, mở **Cài đặt (Settings)**, tìm **Giới thiệu về điện thoại (About phone)**, nhấn vào đó và chạm vào **Số bản dựng (Build number)** 7 lần liên tiếp.
2. Quay lại màn hình trước đó (**Cài đặt / Hệ thống (Settings / System)**). Mục **Developer options** sẽ xuất hiện trong danh sách. Nhấn vào **Developer options**.

3. Bật tùy chọn **USB Debugging**.

4.2 Chạy ứng dụng trên thiết bị

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

1. **Kết nối** thiết bị với máy tính bằng cáp USB.
 2. Nhấp vào **Run** trên thanh công cụ.
 - Cửa sổ **Select Deployment Target** sẽ mở ra, hiển thị danh sách trình giả lập và các thiết bị được kết nối.
 3. **Chọn thiết bị** của bạn và nhấp **OK**.
 - **Android Studio** sẽ cài đặt và chạy ứng dụng trên thiết bị của bạn.
-

Xử lý sự cố (Troubleshooting)

Nếu **Android Studio** không nhận diện được thiết bị, hãy thử các cách sau:

1. Rút cáp và cắm lại thiết bị.
2. Khởi động lại **Android Studio**.

Nếu máy tính vẫn không tìm thấy thiết bị hoặc hiển thị trạng thái "**unauthorized**", hãy làm như sau:

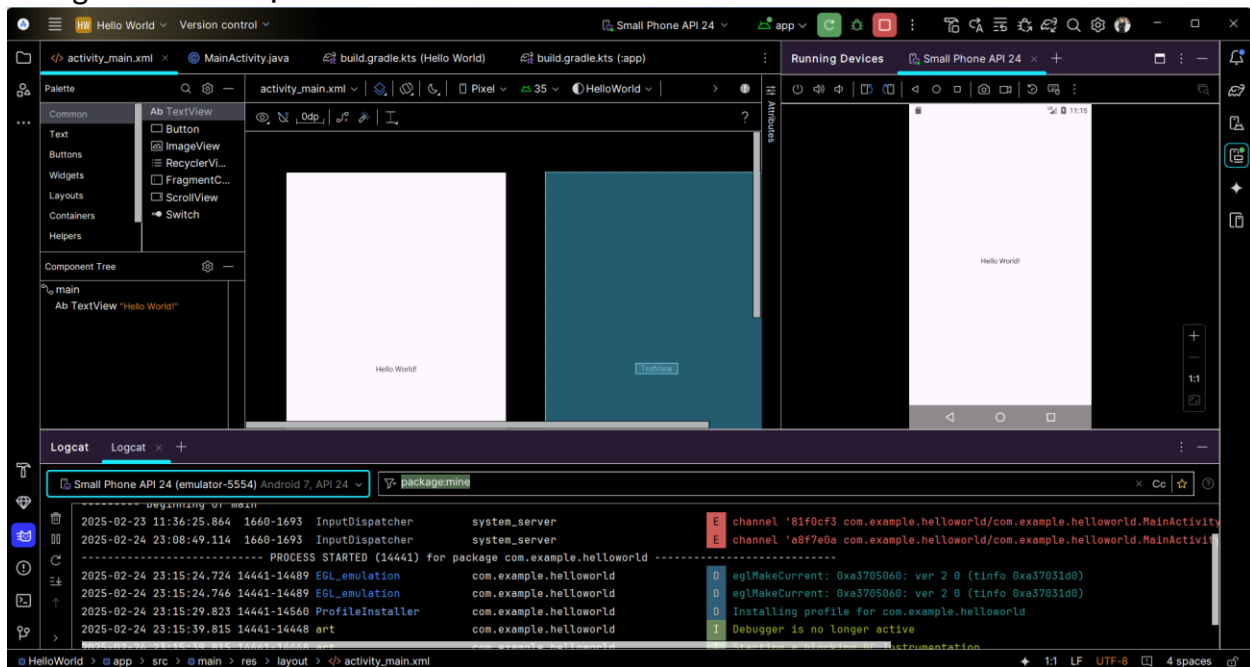
1. Rút kết nối thiết bị.
2. Trên thiết bị, mở **Developer Options** trong ứng dụng **Cài đặt (Settings)**.
3. Nhấn vào **Revoke USB Debugging authorizations**.
4. Kết nối lại thiết bị với máy tính.
5. Khi có thông báo yêu cầu cấp quyền, hãy chọn **Cho phép (Allow)**.

Bạn có thể cần cài đặt **trình điều khiển USB** phù hợp với thiết bị của mình. Xem tài liệu **Using Hardware Devices** để biết thêm chi tiết.

6.1 Xem Logcat pane

Để xem **Logcat pane**, hãy nhấp vào tab **Logcat** ở cuối cửa sổ **Android Studio**, như minh họa trong hình bên dưới.

Trong hình minh họa:



1. **Tab Logcat** dùng để mở và đóng **Logcat pane**, hiển thị thông tin về ứng dụng khi nó đang chạy.
 - o Nếu bạn thêm các câu lệnh **Log** vào ứng dụng, thông báo **Log** sẽ xuất hiện tại đây.
2. **Menu Log level** được đặt ở chế độ **Verbose** (mặc định), hiển thị tất cả các thông báo **Log**.
 - o Các tùy chọn khác bao gồm **Debug**, **Error**, **Info**, và **Warn**.

6.2 Thêm câu lệnh log vào ứng dụng của bạn

Các câu lệnh **Log** trong mã nguồn của ứng dụng sẽ hiển thị thông báo trong **Logcat pane**. Ví dụ:

```
Log.d("MainActivity", "Hello World");
```

Các thành phần của thông báo:

- **Log**: Lớp **Log** dùng để gửi thông báo log đến **Logcat pane**.
- **d**: Cấp độ **Debug** của **Log**, dùng để lọc thông báo trong **Logcat pane**.
 - o Các cấp độ log khác:
 - **e**: **Error** (Lỗi)

- **w: Warn** (Cảnh báo)
 - **i: Info** (Thông tin)
- **"MainActivity"**: Đối số đầu tiên là **tag**, giúp lọc thông báo trong **Logcat pane**.
 - Thông thường, đây là tên của **Activity** chứa thông báo.
 - Tuy nhiên, bạn có thể đặt bất kỳ giá trị nào có ích cho việc gỡ lỗi.
 - Theo quy ước, **log tag** thường được định nghĩa là **hằng số** trong **Activity**.

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

- **"Hello world"**: Đối số thứ hai là nội dung của thông báo.

Các bước thực hiện:

1. Mở ứng dụng **Hello World** trong **Android Studio** và mở tệp **MainActivity**.
2. Để tự động thêm các **import** cần thiết vào dự án (chẳng hạn như **android.util.Log** để sử dụng **Log**), thực hiện:
 - **Windows**: Chọn **File > Settings**
 - **macOS**: Chọn **Android Studio > Preferences**
3. Chọn **Editor > General > Auto Import**.
 - Tích chọn tất cả các hộp kiểm và đặt **Insert imports on paste** thành **All**.
4. Nhấp **Apply**, sau đó nhấp **OK**.
5. Trong phương thức **onCreate()** của **MainActivity**, thêm dòng lệnh sau:

```
Log.d("MainActivity", "Hello World");
```

Phương thức **onCreate()** bây giờ sẽ trông như đoạn mã sau:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    EdgeToEdge.enable(this);
    setContentView(R.layout.activity_main);
    Log.d(tag: "MainActivity", msg: "Hello World");
    ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
        Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
        v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
        return insets;
    });
}
```

6. Nếu **Logcat pane** chưa mở, nhấp vào tab **Logcat** ở cuối cửa sổ **Android Studio** để mở.
7. Kiểm tra xem **tên ứng dụng** và **tên gói (package name)** có chính xác không.
8. Thay đổi **Log level** trong **Logcat pane** thành **Debug** (hoặc giữ nguyên **Verbose** nếu có ít thông báo log).
9. Chạy ứng dụng của bạn.

Sau khi chạy, bạn sẽ thấy thông báo xuất hiện trong **Logcat pane**.

2025-02-25 00:00:04.497 15139-15139 MainActivity com.example.helloworld D Hello World

Thử thách lập trình

Lưu ý: Tất cả các thử thách lập trình đều là tùy chọn và không phải là điều kiện tiên quyết cho các bài học sau.

Thử thách:

Bây giờ bạn đã thiết lập xong môi trường và làm quen với quy trình phát triển cơ bản, hãy thực hiện các bước sau:

1. Tạo một dự án mới trong Android Studio.
 2. Thay đổi dòng chữ "Hello World" thành "Happy Birthday to " kèm theo tên của một người vừa có sinh nhật gần đây.
 3. (**Tùy chọn**) Chụp ảnh màn hình ứng dụng hoàn thành và gửi email cho ai đó mà bạn đã quên chúc mừng sinh nhật.
 4. Một cách sử dụng phổ biến của lớp `Log` là ghi lại các ngoại lệ (**exceptions**) trong chương trình Java khi chúng xảy ra. Có một số phương thức hữu ích như `Log.e()` mà bạn có thể dùng cho mục đích này. Hãy khám phá các phương thức cho phép bạn bao gồm một ngoại lệ trong một thông báo `Log`. Sau đó, viết mã trong ứng dụng của bạn để tạo ra và ghi lại một ngoại lệ.
-

Tóm tắt

- Để cài đặt Android Studio, truy cập [Android Studio](#) và làm theo hướng dẫn để tải xuống và cài đặt.
- Khi tạo một ứng dụng mới, đảm bảo rằng **API 15: Android 4.0.3 IceCreamSandwich** được đặt làm **Minimum SDK**.
- Để xem cấu trúc Android của ứng dụng trong **Project pane**, nhấp vào tab **Project** ở cột dọc và chọn **Android** trong menu thả xuống ở trên cùng.
- Chỉnh sửa tệp **build.gradle(Module:app)** khi bạn cần thêm thư viện mới hoặc thay đổi phiên bản thư viện trong dự án.
- Tất cả mã nguồn và tài nguyên của ứng dụng đều nằm trong thư mục **app** và **res**. Thư mục **java** chứa các activity, test, và các thành phần khác trong mã nguồn Java. Thư mục **res** chứa các tài nguyên như layout, chuỗi ký tự (**strings**) và hình ảnh (**images**).
- Chỉnh sửa tệp **AndroidManifest.xml** để thêm các thành phần (**components**) và quyền truy cập (**permissions**) cho ứng dụng Android của bạn. Mọi thành phần của ứng dụng, chẳng hạn như nhiều **activity**, phải được khai báo trong tệp XML này.
- Sử dụng **Android Virtual Device (AVD) manager** để tạo một thiết bị ảo (**emulator**) để chạy ứng dụng.
- Thêm các câu lệnh **Log** vào ứng dụng để hiển thị thông báo trong **Logcat pane**, giúp bạn gỡ lỗi (**debugging**) dễ dàng hơn.
- Để chạy ứng dụng trên thiết bị Android thực tế bằng Android Studio, bật **USB Debugging** trên thiết bị.
 - Mở **Settings > About phone** và nhấn vào **Build number** bảy lần.
 - Quay lại màn hình trước (**Settings**) và chọn **Developer options**.
 - Bật **USB Debugging**.

Các khái niệm liên quan

Tài liệu về các khái niệm liên quan có trong:

- **1.0: Giới thiệu về Android**
 - **1.1: Ứng dụng Android đầu tiên của bạn**
-

Tìm hiểu thêm

Tài liệu về Android Studio:

- [Trang tải xuống Android Studio](#)
- [Ghi chú phát hành Android Studio](#)
- [Giới thiệu về Android Studio](#)
- [Công cụ dòng lệnh Logcat](#)
- [Trình quản lý thiết bị ảo Android \(AVD Manager\)](#)
- [Tổng quan về Android Manifest](#)
- [Cấu hình Gradle](#)
- [Lớp Log trong Android](#)
- [Tạo và quản lý thiết bị ảo](#)

Khác:

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

Bài tập về nhà

Xây dựng và chạy một ứng dụng

1. Tạo một dự án Android mới từ **Empty Template**.
 2. Thêm các câu lệnh Log cho nhiều cấp độ nhật ký khác nhau (**log levels**) trong phương thức onCreate() của activity chính.
 3. Tạo một trình giả lập (**emulator**) cho một thiết bị, chọn phiên bản Android tùy ý, và chạy ứng dụng trên đó.
 4. Sử dụng bộ lọc trong **Logcat** để tìm các câu lệnh Log của bạn và điều chỉnh mức hiển thị chỉ để hiển thị **debug** hoặc **error**.
-

Trả lời các câu hỏi

Câu hỏi 1

Tên của tệp layout cho activity chính là gì?

- MainActivity.java
- AndroidManifest.xml
- activity_main.xml
- build.gradle

Câu hỏi 2

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

- app_name
- xmlns:app
- android:name
- applicationId

Câu hỏi 3

Công cụ nào được sử dụng để tạo trình giả lập mới?

- Android Device Monitor
- AVD Manager
- SDK Manager
- Theme Editor

Câu hỏi 4

Giả sử ứng dụng của bạn có câu lệnh log sau:

```
Log.i("MainActivity", "MainActivity layout is complete");
```

Bạn sẽ thấy câu lệnh "MainActivity layout is complete" trong **Logcat pane** nếu menu **Log level** được đặt ở mức nào? (Có thể chọn nhiều câu trả lời)

- Verbose
 - Debug
 - Info
 - Warn
 - Error
 - Assert
-

Nội bài kiểm tra ứng dụng

Kiểm tra để đảm bảo rằng ứng dụng có các yếu tố sau:

- Một **Activity** hiển thị "Hello World" trên màn hình.
- Các câu lệnh **Log** trong **onCreate()** của **MainActivity**.
- Mức **Log** trong **Logcat** chỉ hiển thị các thông báo **debug** hoặc **error**.

1.2) Giao diện người dùng tương tác đầu tiên

Giới thiệu

Giao diện người dùng (UI) hiển thị trên màn hình của một thiết bị Android bao gồm một hệ thống phân cấp các đối tượng được 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 như nút bấm, hộp kiểm và trường nhập văn bản.

Các lớp con phổ biến của **View** sẽ được mô tả trong nhiều bài học, bao gồm:

- **TextView** để hiển thị văn bản.
- **EditText** để 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 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 tệp XML (**eXtended Markup Language**).

Tệp XML này thường được đặt tên theo **Activity** của nó và xác định cách bố trí 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 bao gồm một **TextView** có nội dung "Hello World".

Trong các ứng dụng phức tạp hơn, một **Activity** có thể thực hiện các hành động để phản hồi thao tác chạm của người dùng, 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 một bài học khác.

Trong bài thực hành này, bạn sẽ học cách tạo ứng dụng tương tác đầu tiên của mình — một ứng dụng cho phép tương tác của người dùng. Bạn sẽ tạo ứng dụng bằng mẫu **Empty Activity**. Bạn cũng sẽ học cách sử dụng trình chỉnh sửa bố cục (**layout editor**) để thiết kế bố cục và chỉnh sửa bố cục bằng XML. Bạn cần phát triển các kỹ năng này để hoàn thành các bài thực hành khác trong khóa học này.

Những gì bạn cần biết trước

Bạn nên quen thuộc 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 trình chỉnh sửa bố cục để thiết kế giao diện.
- Cách chỉnh sửa bố cục bằng XML.
- Nhiều thuật ngữ mới. Hãy tham khảo **bảng thuật ngữ và khái niệm** để có các định nghĩa dễ hiểu.

Những gì bạn sẽ làm

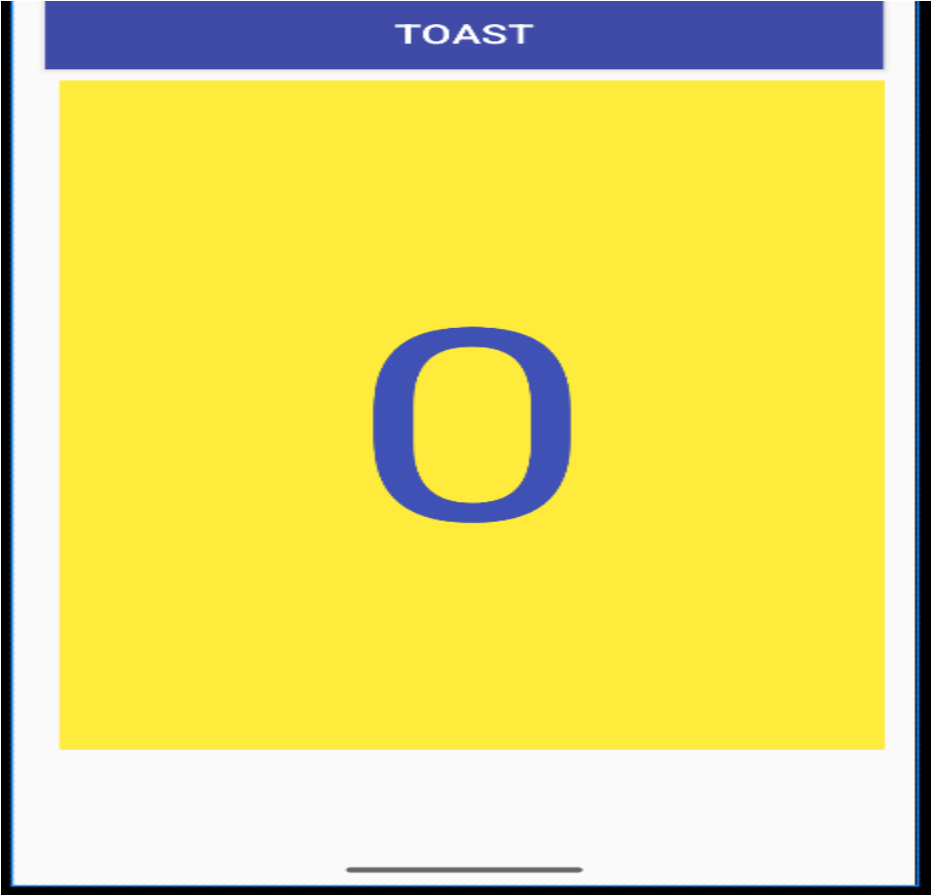
- Tạo một ứng dụng và thêm hai **Button** cùng một **TextView** vào bố cục.
 - Điều chỉnh từng phần tử trong **ConstraintLayout** để ràng buộc chúng với lề và các phần tử khác.
 - Thay đổi thuộc tính của các phần tử UI.
 - Chỉnh sửa bố cục của ứng dụng trong XML.
 - Trích xuất chuỗi văn bản cố định thành tài nguyên chuỗi (**string resources**).
 - Triển khai phương thức xử lý sự kiện nhấp chuột (**click-handler methods**) để hiển thị thông báo trên màn hình khi người dùng chạm vào từng **Button**.
-

Tổng quan về ứng dụng

Ứng dụng **HelloToast** bao gồm hai phần tử **Button** và một **TextView**. Khi người dùng chạm vào **Button** thứ nhất, một thông báo ngắn (**Toast**) sẽ xuất hiện trên màn hình. Khi chạm vào **Button** thứ hai, bộ đếm số lần nhấp (**click counter**) được hiển thị trong **TextView** sẽ tăng lên, bắt đầu từ số 0.

Đây là giao diện cuối cùng của ứng dụng:





TOAST

0

COUNT

Hello Toast

TOAST

0

Hello Toast

COUNT



11:13

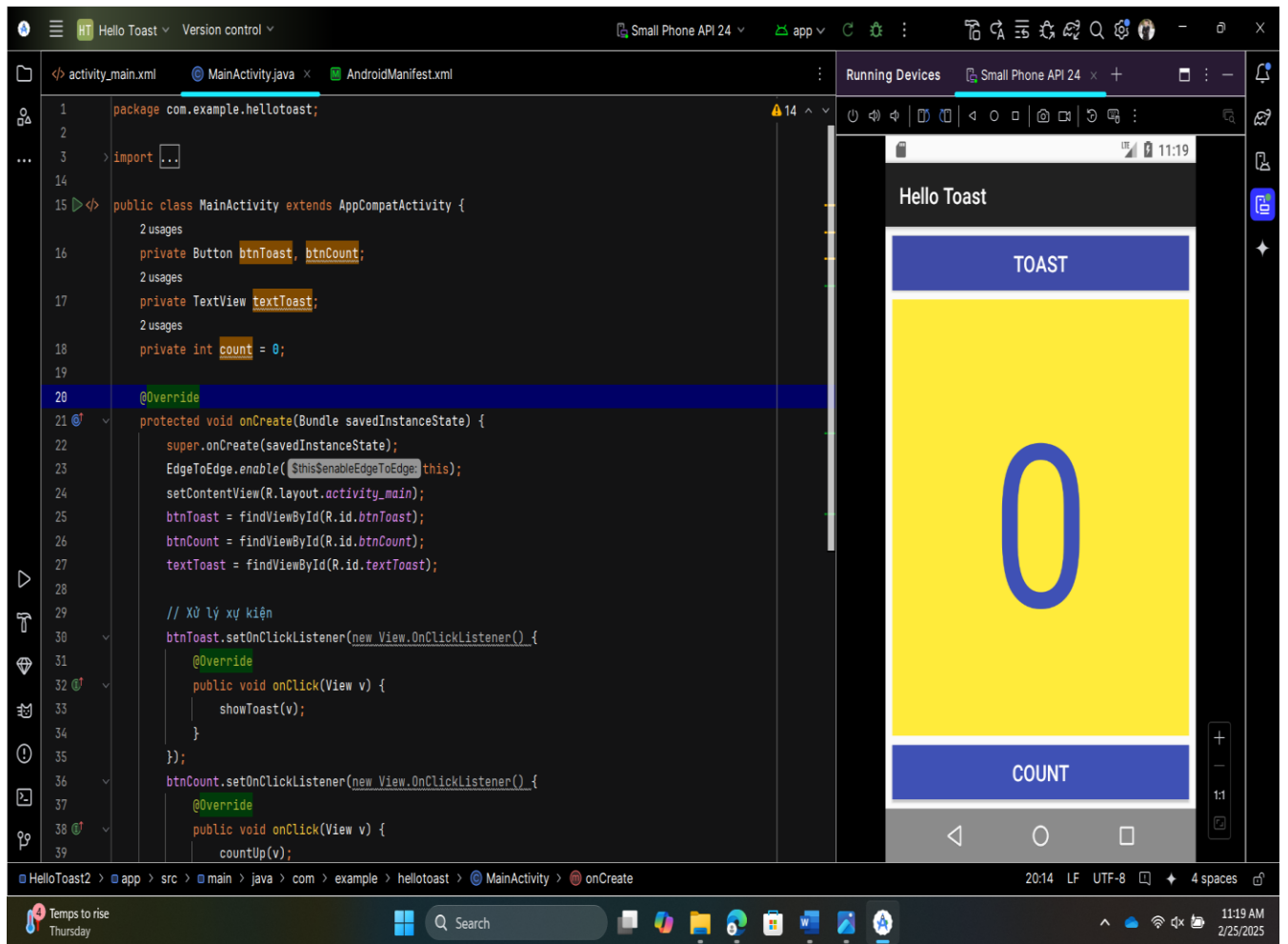
Hello Toast

TOAST

17

COUNT





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