

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.....	18
1.3)	Trình chỉnh sửa bố cục	18
1.4)	Văn bản và các chế độ cuộn.....	18
1.5)	Tài nguyên có sẵn	18
Bài 2)	Activities	18
2.1)	Activity và Intent	18
2.2)	Vòng đời của Activity và trạng thái	18
2.3)	Intent ngầm định	18
Bài 3)	Kiểm thử, gỡ lỗi và sử dụng thư viện hỗ trợ.....	18
3.1)	Trình gỡ lỗi.....	18
3.2)	Kiểm thử đơn vị	18
3.3)	Thư viện hỗ trợ.....	18
CHƯƠNG 2.	Trải nghiệm người dùng	19
Bài 1)	Tương tác người dùng.....	19
1.1)	Hình ảnh có thể chọn	19
1.2)	Các điều khiển nhập liệu.....	19
1.3)	Menu và bộ chọn.....	19
1.4)	Điều hướng người dùng	19
1.5)	RecyclerView	19
Bài 2)	Trải nghiệm người dùng thú vị	19
2.1)	Hình vẽ, định kiểu và chủ đề.....	19
2.2)	Thẻ và màu sắc.....	19
2.3)	Bố cục thích ứng	19
Bài 3)	Kiểm thử giao diện người dùng	19

3.1)	Espresso cho việc kiểm tra UI.....	19
CHƯƠNG 3. Làm việc trong nền		19
Bài 1)	Các tác vụ nền	19
1.1)	AsyncTask	19
1.2)	AsyncTask và AsyncTaskLoader	19
1.3)	Broadcast receivers	19
Bài 2)	Kích hoạt, lập lịch và tối ưu hóa nhiệm vụ nền	19
2.1)	Thông báo.....	19
2.2)	Trình quản lý cảnh báo.....	19
2.3)	JobScheduler	19
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	20
Bài 2)	Lưu trữ dữ liệu với Room	20
2.1)	Room, LiveData và ViewModel	20
2.2)	Room, LiveData và ViewModel	20

CHƯƠNG 1. LÀM QUEN

Bài 1) Tạo ứng dụng đầu tiên

1.1) Android Studio và Hello World

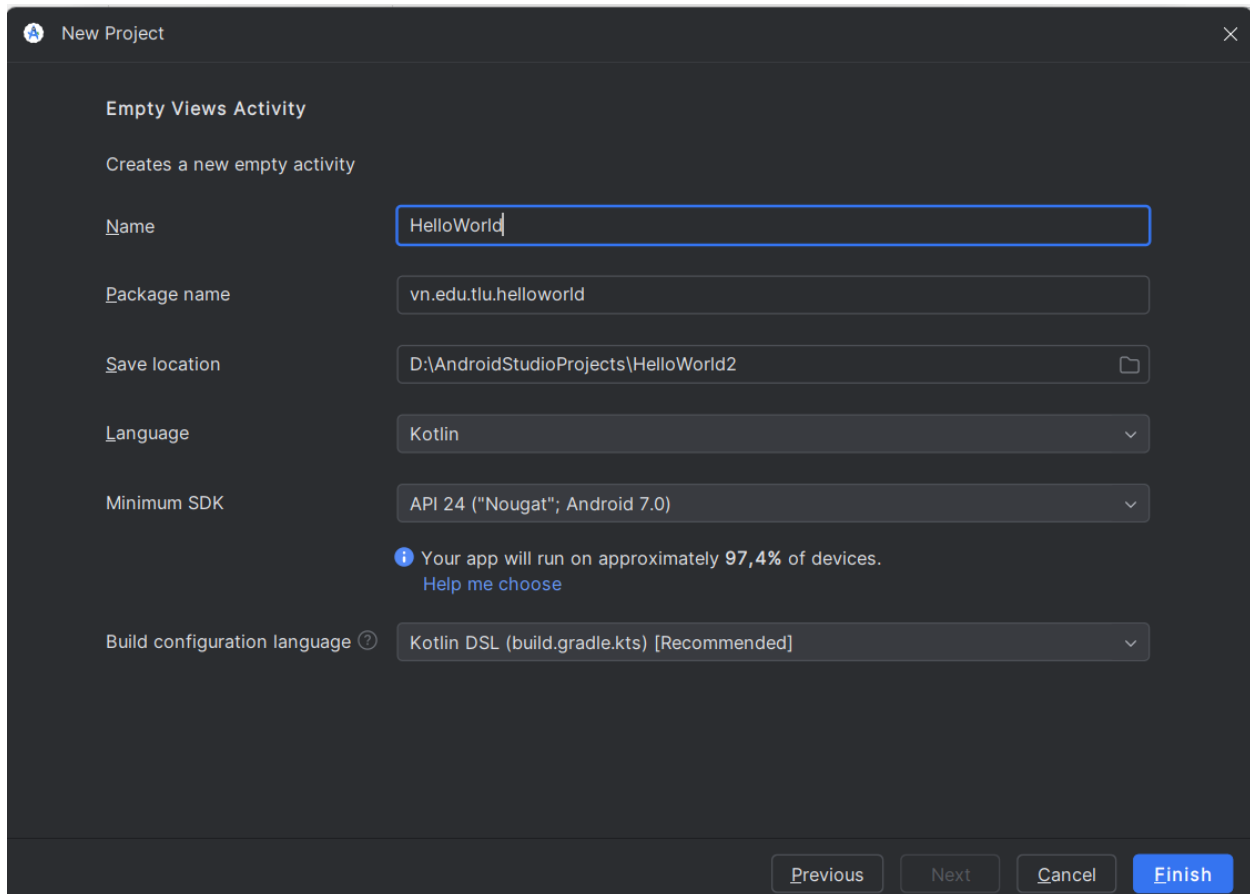
Giới thiệu

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.

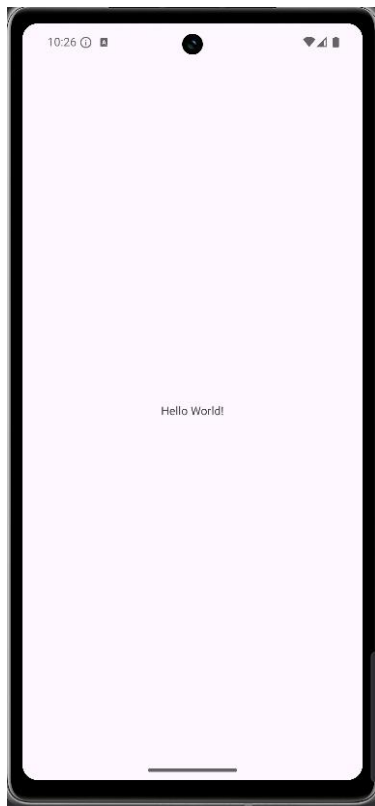
Những gì bạn sẽ làm

- 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, từ một mẫu có sẵn, một dự án mới cho ứng dụng Hello World. Ứng dụng đơn giản này hiển thị chuỗi “Hello World” trên màn hình của thiết bị Android ảo hoặc vật lý.

Dưới đây là giao diện của ứng dụng khi hoàn thiện:



Nhiệm vụ 1: Cài đặt Android Studio

Android Studio cung cấp một môi trường phát triển tích hợp (IDE) hoàn chỉnh, bao gồm một trình chỉnh sửa mã nâng cao và một tập hợp các mẫu ứng dụng. Ngoài ra, nó còn chứa các công cụ để phát triển, gỡ lỗi, kiểm thử và tối ưu hiệu suất, giúp việc phát triển ứng dụng nhanh hơn và dễ dàng hơn. Bạn có thể kiểm thử ứng dụng của mình với một loạt các trình giả lập được cấu hình sẵn hoặc trên thiết bị di động của riêng bạn, xây dựng các ứng dụng phiên bản sản xuất và xuất bản lên cửa hàng Google Play.

Lưu ý: Android Studio liên tục được cải thiện. Để biết thông tin mới nhất về yêu cầu hệ thống và hướng dẫn cài đặt, hãy tham khảo [Android Studio](#).

Android Studio hỗ trợ cho các máy tính chạy Windows hoặc Linux, và cho các máy Mac chạy macOS. Phiên bản mới nhất của OpenJDK (Java Development Kit) được tích hợp cùng Android Studio.

Để bắt đầu sử dụng Android Studio, trước tiên hãy kiểm tra yêu cầu hệ thống để đảm bảo rằng hệ thống của bạn đáp ứng được chúng. Quá trình cài đặt tương tự trên tất cả các nền tảng. Bất kỳ sự khác biệt nào sẽ được ghi chú dưới đây.

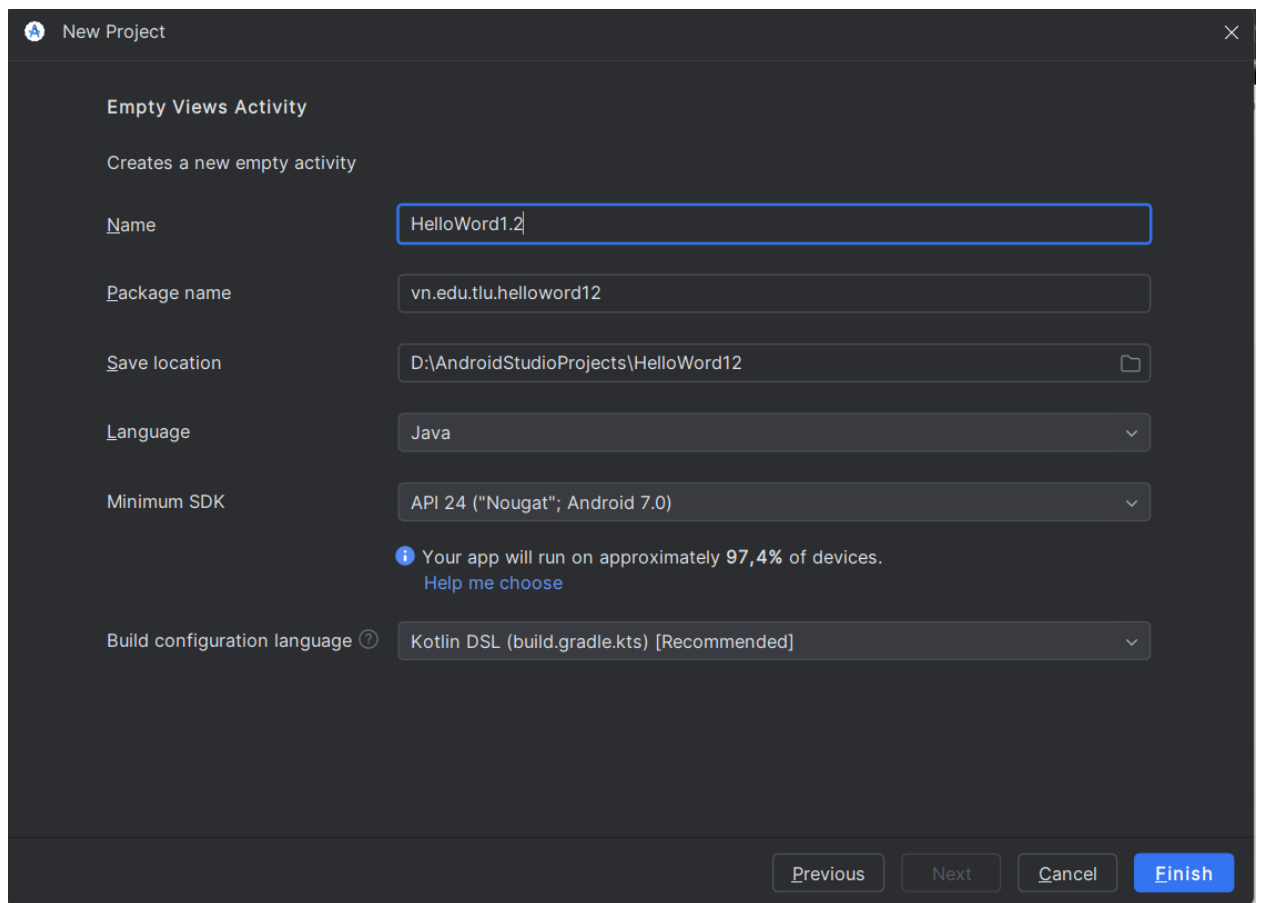
1. Truy cập trang web dành cho nhà phát triển Android và làm theo hướng dẫn để tải xuống và cài đặt Android Studio.
2. Chấp nhận các cấu hình mặc định cho tất cả các bước và đảm bảo rằng tất cả các thành phần đều được chọn để cài đặt.
3. Sau khi hoàn tất cài đặt, Trình hướng dẫn thiết lập (Setup Wizard) sẽ tải xuống và cài đặt một số thành phần bổ sung, bao gồm Android SDK. Hãy kiên nhẫn, quá trình này có thể mất một chút thời gian tùy thuộc vào tốc độ Internet của bạn, và một số bước có thể dường như lặp lại.
4. Khi quá trình tải xuống hoàn tất, Android Studio sẽ khởi động, và bạn đã sẵn sàng để tạo dự án đầu tiên của mình.

Nhiệm vụ 2: Tạo ứng dụng Hello World

Trong nhiệm vụ này, bạn sẽ tạo một ứng dụng hiển thị dòng chữ "Hello World" để kiểm tra xem Android Studio đã được cài đặt đúng hay chưa, đồng thời làm quen với những kiến thức cơ bản về phát triển ứng dụng bằng Android Studio.

2.1 Tạo dự án ứng dụng

1. Mở Android Studio nếu nó chưa được mở.
2. Trong cửa sổ chính "Welcome to Android Studio", nhấp vào "Start a new Android Studio project".
3. Trong cửa sổ "Create Android Project", nhập "Hello World" vào phần "Application name".

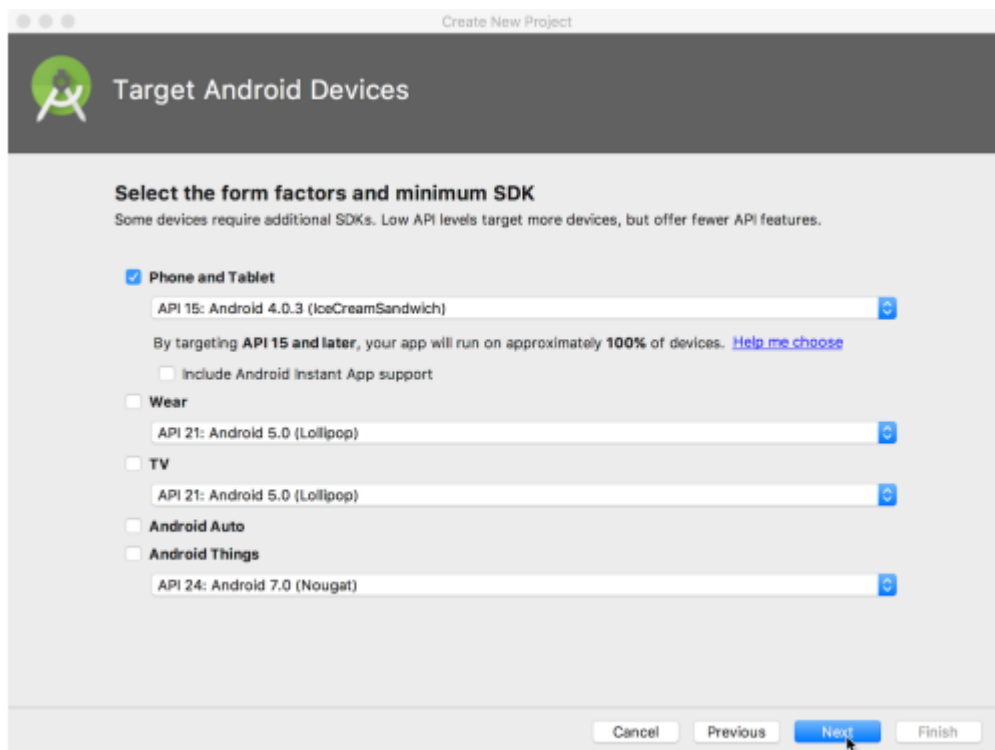


4. Xác nhận **Project location** mặc định là nơi bạn muốn lưu ứng dụng Hello World và các dự án Android Studio khác, hoặc thay đổi thành thư mục bạn muốn.

5. Chấp nhận **android.example.com** mặc định cho **Company Domain**, hoặc tạo một tên miền công ty độc nhất. Nếu không định xuất bản ứng dụng, bạn có thể giữ nguyên mặc định. Lưu ý thay đổi package name sau này sẽ tốn thêm công sức.

6. Bỏ chọn **Include C++ support** và **Include Kotlin support**, rồi nhấp **Next**.

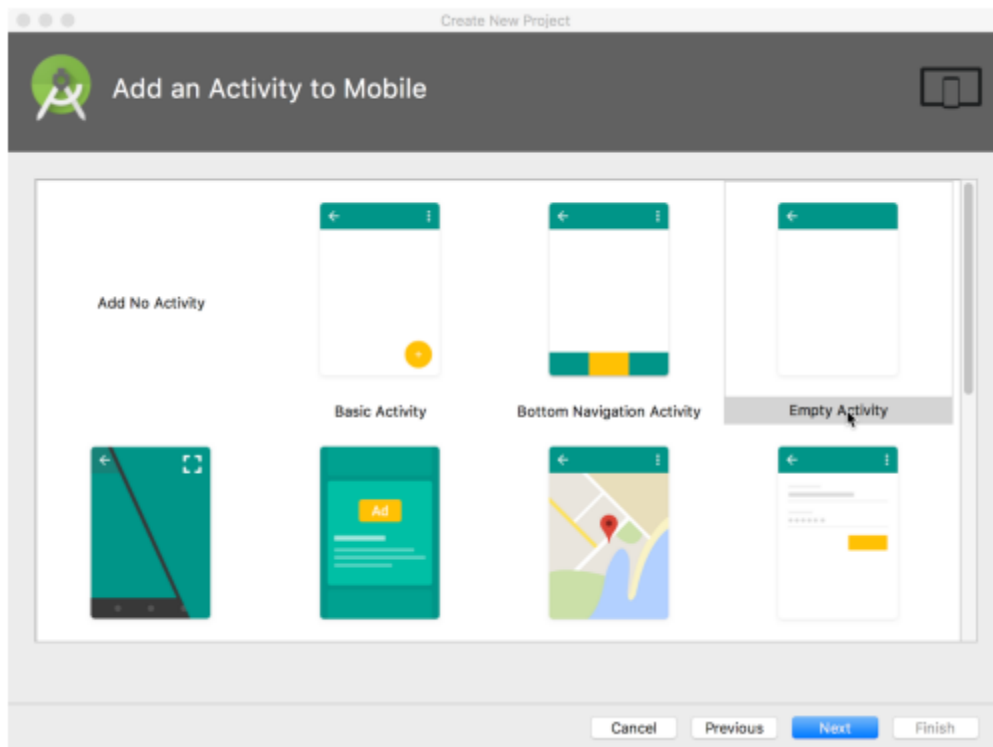
7. Trên màn hình **Target Android Devices, Phone and Tablet** nên được chọn. Đảm bảo **API 15: Android 4.0.3 IceCreamSandwich** được đặt làm Minimum SDK; nếu chưa, dùng menu thả xuống để chọn.



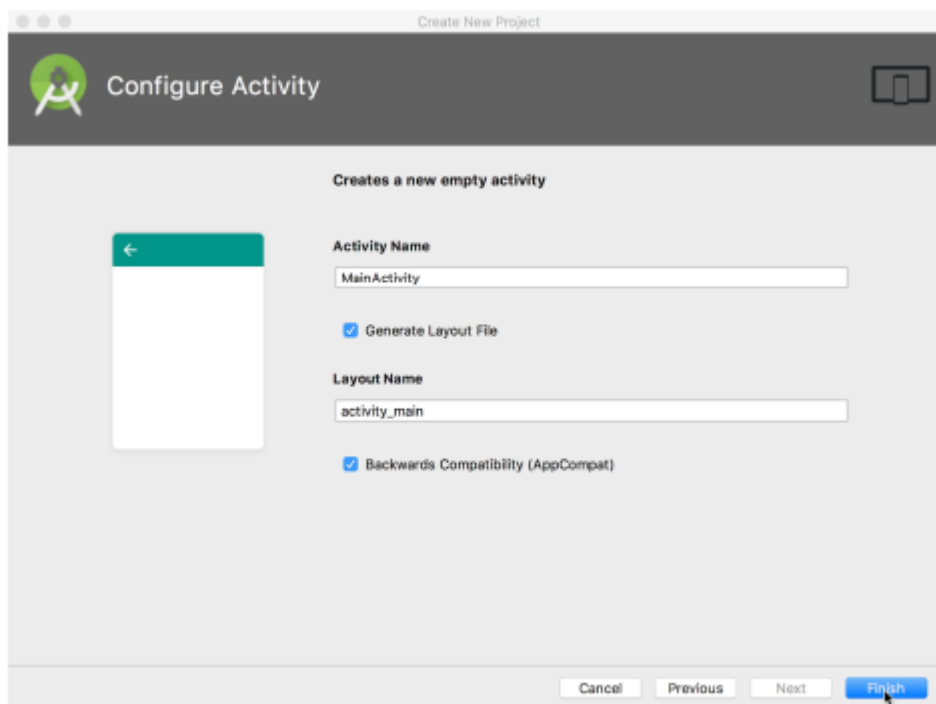
Đây là các cài đặt được sử dụng trong ví dụ của bài học này. Tính đến thời điểm hiện tại, những cài đặt này giúp ứng dụng Hello World của bạn tương thích với 97% thiết bị Android đang hoạt động trên Google Play Store.

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 target SDK đã chọn, Android Studio sẽ tự động cài đặt chúng.

9. Cửa sổ **Add an Activity** xuất hiện. Activity là một tác vụ đơn lẻ, tập trung mà người dùng có thể thực hiện. Nó là thành phần cốt lõi của bất kỳ ứng dụng Android nào. Một Activity thường có layout liên kết, định nghĩa cách các yếu tố giao diện người dùng hiển thị trên màn hình. Android Studio cung cấp các mẫu Activity để giúp bạn bắt đầu. Đối với dự án Hello World, chọn **Empty Activity** như hình dưới đây, rồi nhấp **Next**.



10. Màn hình **Configure Activity** xuất hiện (nội dung khác nhau tùy thuộc vào mẫu bạn chọn ở bước trước). Theo mặc định, Activity trống do mẫu cung cấp được đặt tên là MainActivity. Bạn có thể thay đổi tên này nếu muốn, nhưng bài học này sử dụng MainActivity.



11. Đảm bảo rằng tùy chọn **Generate Layout file** được chọn. Mặc định, tên của tệp giao diện là **activity_main**. Bạn có thể thay đổi nếu muốn, nhưng bài học này sử dụng **activity_main**.
12. Đảm bảo rằng tùy chọn **Backwards Compatibility (App Compat)** được chọn. Điều này giúp ứng dụng của bạn tương thích với các phiên bản Android cũ hơn.
13. Nhấn **Finish**.

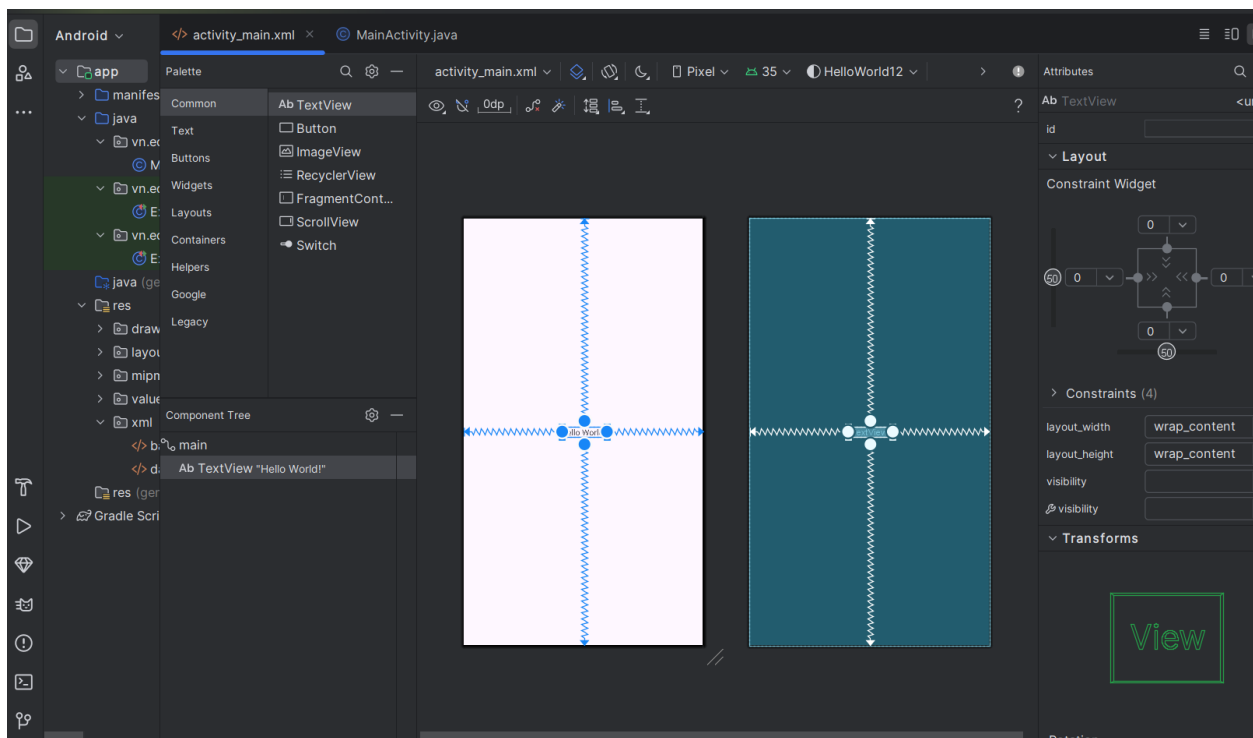
Android Studio sẽ tạo thư mục cho dự án và tiến hành build bằng Gradle (quá trình này có thể mất một chút thời gian).

Mẹo: Xem trang **Configure your build** để biết thông tin chi tiết.

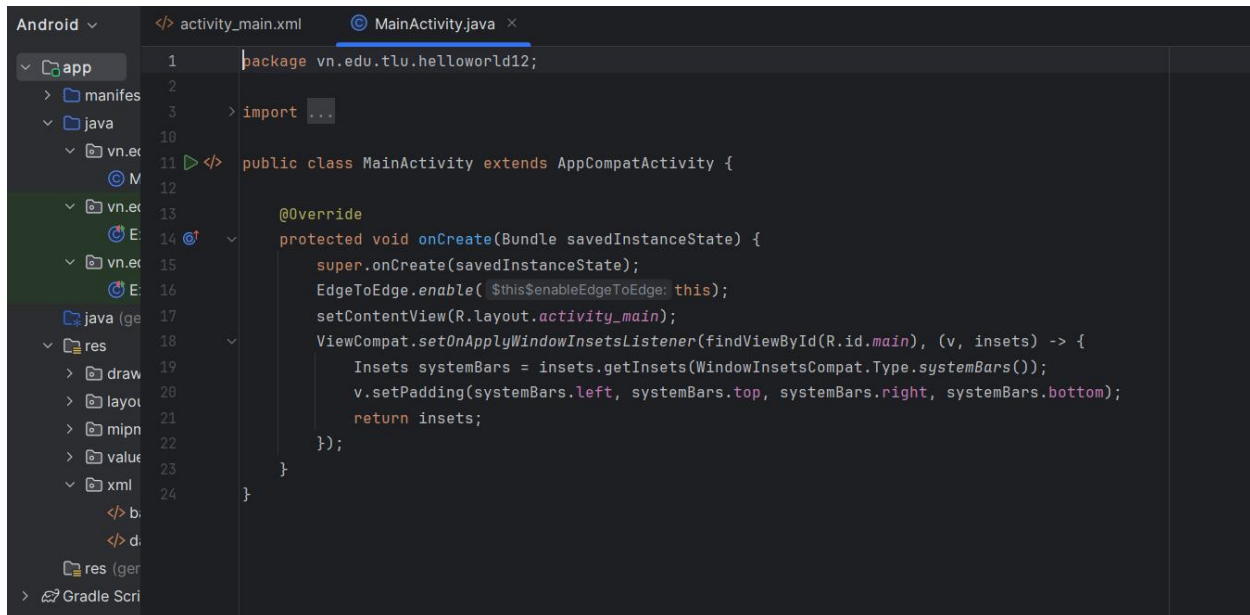
Bạn cũng có thể thấy thông báo "**Tip of the day**" với các phím tắt và mẹo hữu ích khác. Nhấn **Close** để đóng thông báo.

Giao diện Android Studio sẽ xuất hiện. Thực hiện các bước sau:

1. Nhấn vào tab **activity_main.xml** để mở trình chỉnh sửa giao diện.
2. Nhấn vào tab **Design** trong trình chỉnh sửa giao diện (nếu chưa được chọn) để hiển thị bản xem trước đồ họa của giao diện như bên dưới.



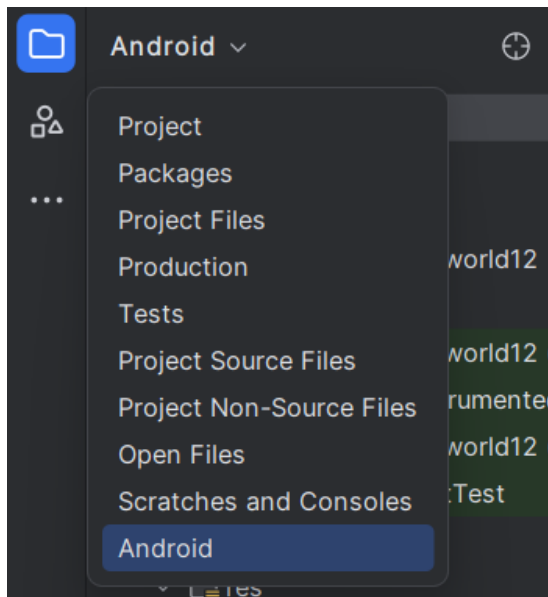
3. Nhấn vào tab **MainActivity.java** để mở trình chỉnh sửa mã nguồn như hình dưới.



2.2 Khám phá ngăn Project > Android

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

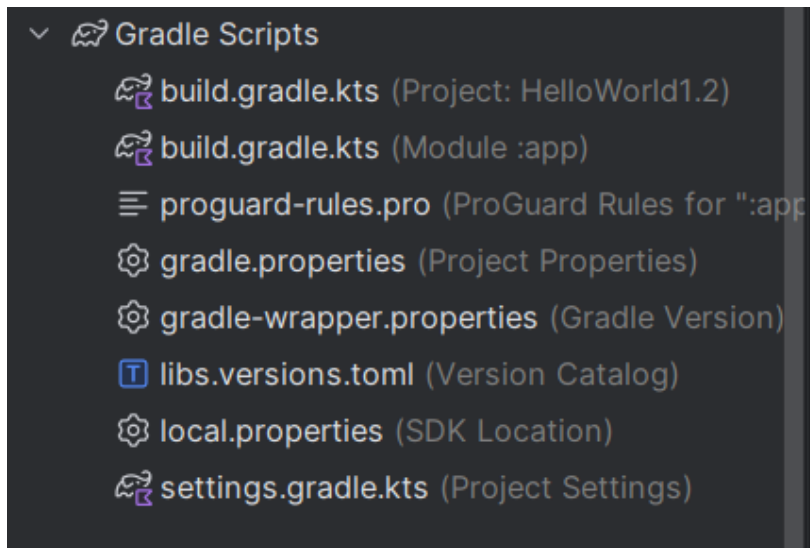
1. Nếu chưa được chọn, nhấn vào tab **Project** trong cột tab dọc bên trái của sổ Android Studio. Ngăn **Project** sẽ xuất hiện.
2. Để xem dự án theo cấu trúc chuẩn của Android, chọn **Android** từ menu thả xuống ở đầu ngăn **Project**, như hình dưới đây.



2.3 Khám phá thư mục Gradle Scripts

Hệ thống build Gradle trong Android Studio cho phép bạn dễ dàng tích hợp các tệp nhị phân bên ngoài hoặc các mô-đun thư viện vào quá trình build dưới dạng thành phần phụ thuộc.

Khi bạn tạo một dự án ứng dụng mới, ngăn **Project > Android** sẽ hiển thị với thư mục **Gradle Scripts** được mở rộng như hình dưới đây.



Thực hiện các bước sau để khám phá hệ thống Gradle:

1. Nếu thư mục **Gradle Scripts** chưa được mở rộng, nhấn vào biểu tượng tam giác để mở.
Thư mục này chứa tất cả các tệp cần thiết cho hệ thống build.
2. Tìm tệp **build.gradle (Project: HelloWorld)**.
Đây là nơi chứa các tùy chọn cấu hình chung cho tất cả các mô-đun trong dự án. Mỗi dự án trong Android Studio đều có một tệp build Gradle cấp cao nhất. Thông thường, bạn sẽ không cần chỉnh sửa tệp này, nhưng hiểu nội dung của nó vẫn rất hữu ích.

Theo mặc định, tệp build Gradle cấp cao nhất sử dụng khối **buildscript** để xác định các kho lưu trữ (**repositories**) và các **thư viện phụ thuộc** chung cho toàn bộ dự án. Khi một **thành phần phụ thuộc** không phải là thư viện cục bộ hoặc cây tệp, Gradle sẽ tìm kiếm tệp trong các kho lưu trữ trực tuyến được chỉ định trong khối **repositories** của tệp này. Mặc định, các dự án Android Studio mới khai báo **JCenter** và **Google** (bao gồm **Google Maven repository**) làm kho lưu trữ.

```

14  dependencyResolutionManagement {
15      repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
16      repositories {
17          google()
18          mavenCentral()
19      }
20  }

```

3. Tìm tệp build.gradle (Module: app)

Ngoài tệp **build.gradle** cấp dự án, mỗi mô-đun cũng có một tệp **build.gradle** riêng, cho phép bạn cấu hình các thiết lập build cho từng mô-đun cụ thể (ứng dụng **HelloWorld** chỉ có một mô-đun). Việc cấu hình các thiết lập build này giúp bạn tùy chỉnh các tùy chọn đóng gói, chẳng hạn như thêm kiểu build và phiên bản sản phẩm (**product flavors**). Bạn cũng có thể ghi đè các thiết lập trong tệp **AndroidManifest.xml** hoặc **build.gradle** cấp cao nhất.

Tệp này thường được chỉnh sửa nhiều nhất khi bạn muốn thay đổi các cấu hình cấp ứng dụng, chẳng hạn như khai báo **thư viện phụ thuộc** trong phần **dependencies**. Bạn có thể khai báo một thư viện phụ thuộc bằng nhiều cách khác nhau, mỗi cách sẽ hướng dẫn Gradle cách sử dụng thư viện theo những cách khác nhau.

```

You can use the Project Structure dialog to view and edit your project configuration
Open (Ctrl+Alt+Shift+S) Hide n

1  plugins {
2      alias(libs.plugins.android.application)
3  }
4
5  android {
6      namespace = "vn.edu.tlu.helloworld12"
7      compileSdk = 35
8
9      defaultConfig {
10         applicationId = "vn.edu.tlu.helloworld12"
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

```

```

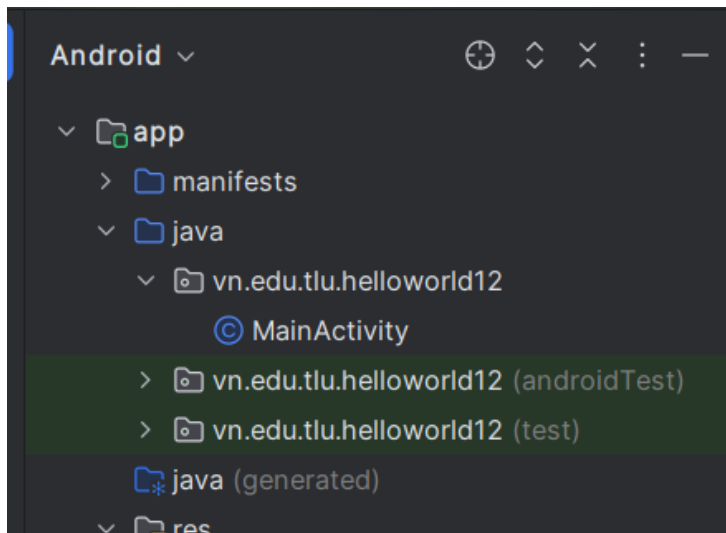
39     implementation(libs.constraintlayout)
40     testImplementation(libs.junit)
41     androidTestImplementation(libs.ext.junit)
42     androidTestImplementation(libs.espresso.core)
43 }

```

2.4 Khám phá thư mục app và res

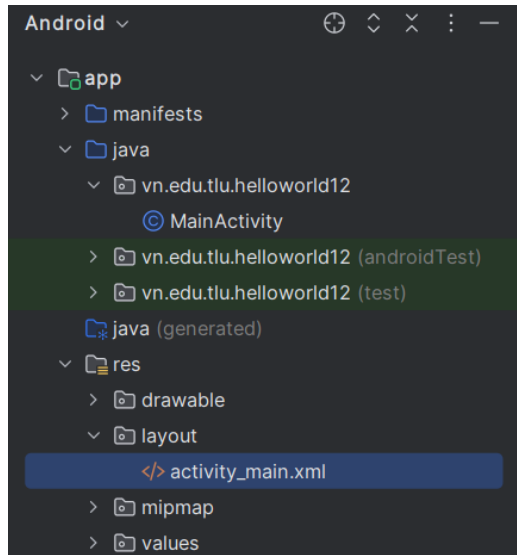
Tất cả mã nguồn và tài nguyên của ứng dụng được lưu trữ trong thư mục **app** và **res**.

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



Thư mục **java** chứa các tệp lớp Java trong ba thư mục con, như trong hình minh họa. 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 cho kiểm thử và sẽ được đề cập trong bài học khác. Đối với ứng dụng **Hello World**, chỉ có một gói duy nhất và nó chứa tệp **MainActivity.java**. Màn hình đầu tiên mà người dùng nhìn thấy, đồng thời khởi tạo các tài nguyên toàn cục của ứng dụng, thường được đặt tên là **MainActivity** (phần mở rộng tệp bị ẩn trong ngăn **Project > Android**).

2. Mở rộng thư mục **res**, sau đó mở thư mục **layout**, và nhấn đúp vào tệp **activity_main.xml** để mở nó trong trình chỉnh sửa giao diện.



Thư mục **res** chứa các tài nguyên như bố cục (**layouts**), chuỗi ký tự (**strings**) và hình ảnh (**images**). Mỗi **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 **Activity** tương ứng.

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 những thông tin này trước khi có thể chạy bất kỳ đoạn 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. Mọi thành phần của ứng dụng, chẳng hạn như mỗi **Activity**, đều phải được khai báo trong tệp XML này.

Trong các bài học sau, bạn sẽ chỉnh sửa tệp này để thêm tính năng và quyền truy cập cho ứng dụng. Để tìm hiểu tổng quan, hãy xem [**App Manifest Overview**](#).

Task 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 thiết bị Android cụ thể, và chạy ứng dụng trên thiết bị ảo đó. Lưu ý rằng **Android Emulator** có các yêu cầu bổ sung so với 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, phiên bản API, dung lượng lưu trữ, giao diện (**skin**) và các thuộc tính khác, sau đó lưu nó dưới dạng một thiết bị ảo. Với thiết bị ảo, bạn có thể kiểm thử ứng dụng trên nhiều cấu hình thiết bị khác nhau (chẳng hạn như điện thoại và máy tính bảng) với các mức API khác nhau, mà không cần sử dụng thiết bị thật.

3.1 Tạo một thiết bị ảo Android (AVD)

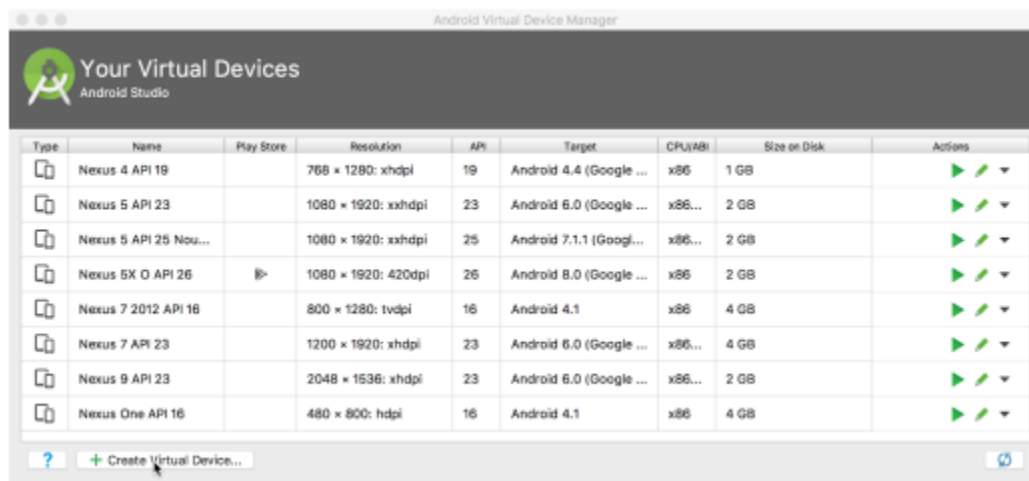
Để chạy trình giả lập trên máy tính, bạn cần tạo một cấu hình mô tả thiết bị ảo.

1. Trong **Android Studio**, chọn **Tools > Android > AVD Manager**, hoặc




nhấn vào biểu tượng **AVD Manager** trên thanh công cụ.

2. Màn hình **Your Virtual Devices** sẽ xuất hiện. Nếu bạn đã tạo thiết bị ảo trước đó, danh sách sẽ hiển thị chúng (như trong hình minh họa bên dưới). Nếu chưa, danh sách sẽ trống.



2. Nhấn vào **+Create Virtual Device**. Cửa sổ **Select Hardware** sẽ xuất hiện, hiển thị danh sách các thiết bị phần cứng được cấu hình sẵn. Đối với mỗi thiết bị, bảng danh sách cung cấp một cột cho kích thước màn hình theo đường chéo (**Size**), độ phân giải màn hình tính theo pixel (**Resolution**), và mật độ điểm ảnh (**Density**).








Virtual Device Configuration



Select Hardware

Choose a device definition

Q-

Category	Name	Play Store	Size	Resolution	Density
Phone	Pixel 8 Pro		6,7"	1344x2992	xxhdpi
Tablet	Pixel 8		6,17"	1080x2400	420dpi
Wear OS	Pixel Fold		7,58"	1840x2208	420dpi
Desktop	Pixel 7a		6,1"	1080x2400	420dpi
TV	Pixel 7 Pro		6,71"	1440x3120	560dpi
Automotive	Pixel 7		6,31"	1080x2400	420dpi
Legacy	Pixel 6a		6,13"	1080x2400	420dpi

Pixel 7a

1080px

6,1"

2400px

Size: large


Ratio: long

Density: 420dpi

Clone Device...

New Hardware Profile

Import Hardware Profiles



?

Previous

Next

Cancel

Finish

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