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

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 cho ứng dụng Hello World từ một mẫu có sẵn.

Ứng dụng đơn giản này sẽ hiển thị chuỗi "Hello World" trên màn hình của thiết bị Android, có thể là thiết bị ảo hoặc thiết bị vật lý.

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

ảnh

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 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 trực tiếp trên thiết bị di động của mình, đồng thời có thể xây dựng ứng dụng hoàn chỉnh và xuất bản 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 truy cập Android Studio.

Android Studio có sẵn cho các hệ điều hành Windows và Linux trên máy tính, macOS trên máy Mac.

Phiên bản mới nhất của OpenJDK (Java Development Kit) được tích hợp sẵn trong Android Studio.

Cài đặt Android Studio

Trước khi bắt đầu, hãy kiểm tra yêu cầu hệ thống để đảm bảo thiết bị của bạn đáp ứng đủ điều kiện. Quá trình cài đặt trên các nền tảng là tương tự nhau, nếu có khác biệt sẽ được ghi chú riêng.

Các bước cài đặt:

Truy cập trang web của nhà phát triển Android và làm theo hướng dẫn để tải xuống và cài đặt Android Studio.

Chấp nhận cấu hình mặc định trong tất cả các bước và đảm bảo rằng tất cả các thành phần cần thiết đều được chọn để cài đặt.

Sau khi cài đặt xong, Setup Wizard sẽ tải xuống và cài đặt thêm một số thành phần bổ sung, bao gồm Android SDK.

Lưu ý: Quá trình này có thể mất một khoảng thời gian tùy vào tốc độ internet của bạn. Một số bước có thể trông giống nhau, nhưng hãy kiên nhẫn.

Khi quá trình tải xuống hoàn tất, Android Studio sẽ khởi động và bạn đã sẵn sàng để tạo dự án đầu tiên.

Xử lý sự cố:

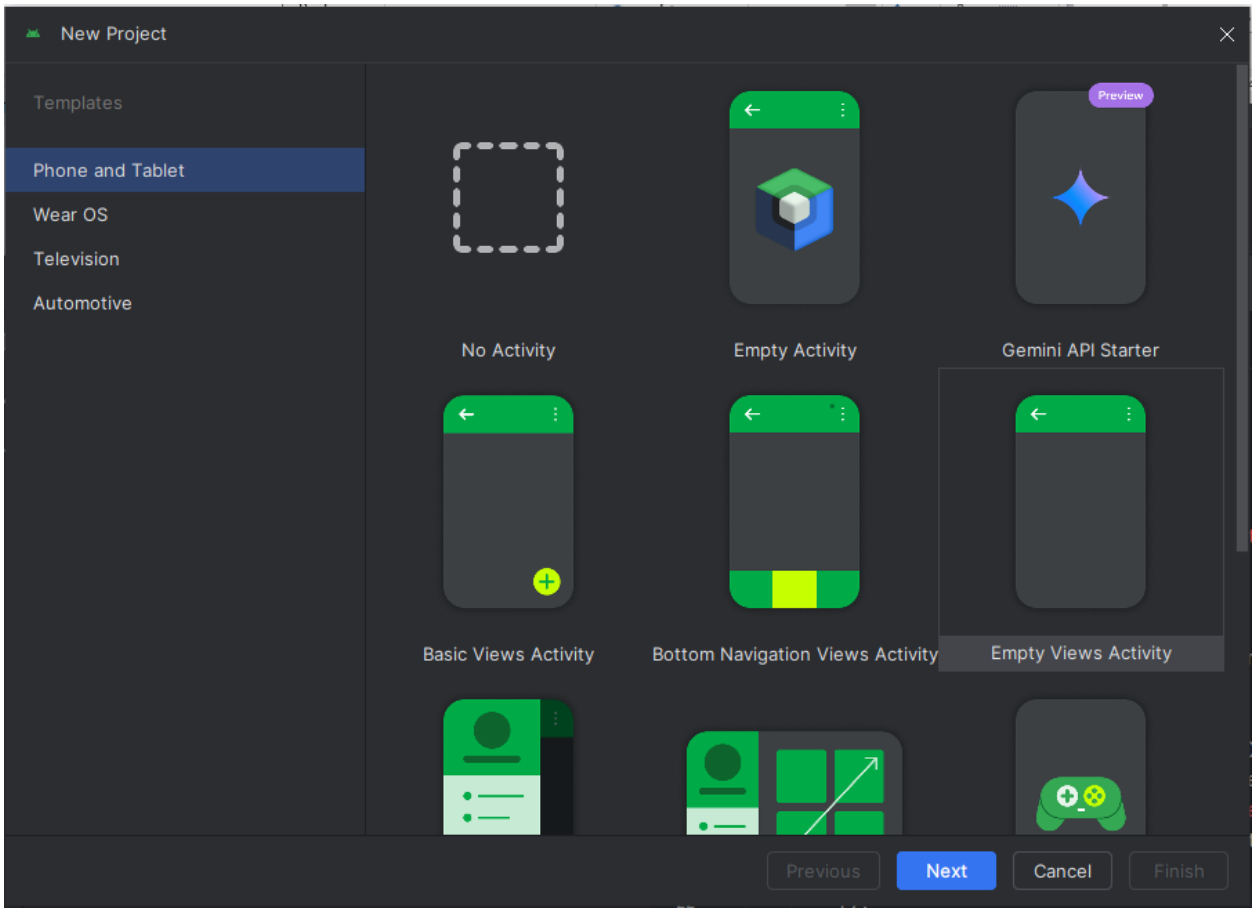
Nếu gặp vấn đề trong quá trình cài đặt, hãy kiểm tra ghi chú phát hành của Android Studio hoặc nhờ sự hỗ trợ từ giảng viên/diễn đàn hỗ trợ.

Nhiệm vụ 2: Tạo ứng dụng Hello World

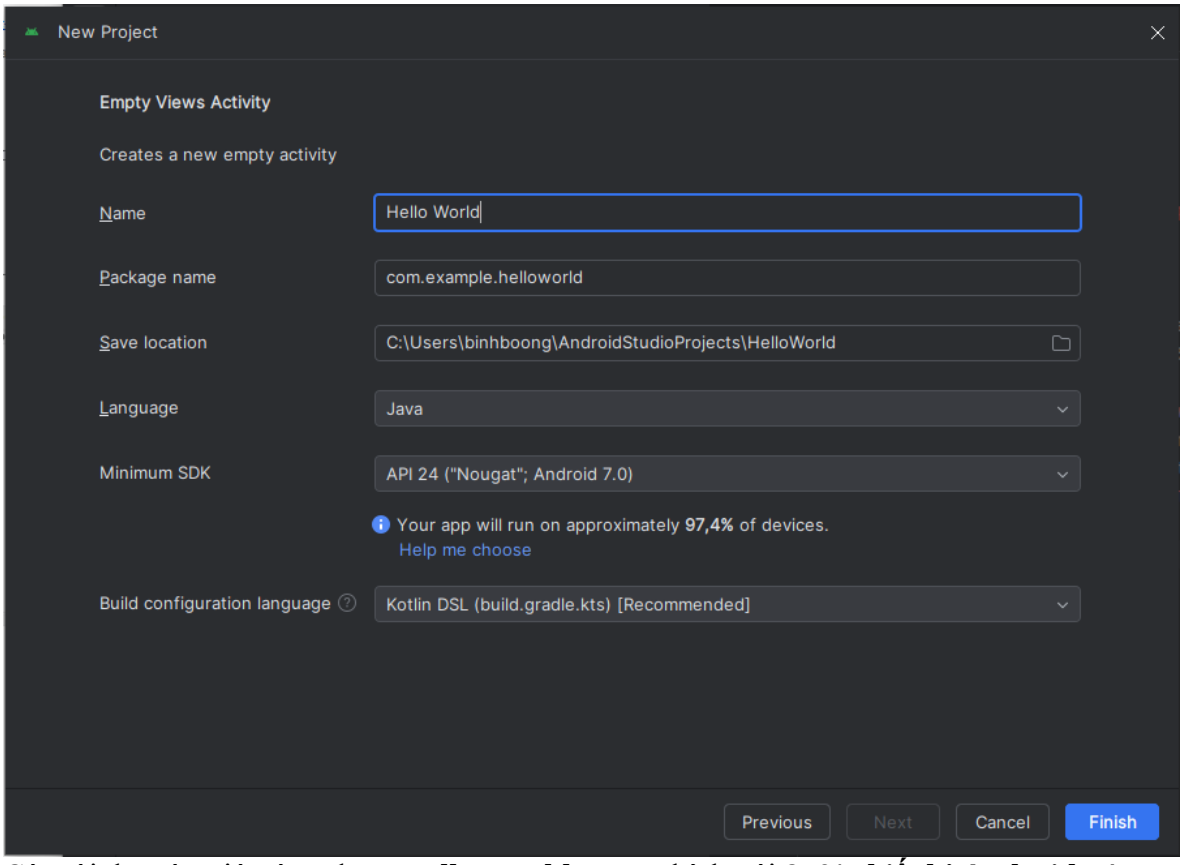
Trong nhiệm vụ này, bạn sẽ tạo một ứng dụng hiển thị dòng chữ "Hello World" để kiểm tra xem Android Studio đã được cài đặt đúng cách và học các bước cơ bản để phát triển ứng dụng Android.

2.1 Tạo dự án ứng dụng

1. Mở Android Studio (nếu chưa mở).
2. Tại màn hình Welcome to Android Studio, nhấp vào Start a new Android Studio project.
3. Trong cửa sổ Create Android Project, nhập Hello World vào ô Application name.



4. • Xác nhận rằng vị trí mặc định của **Project location** là nơi bạn muốn lưu ứng dụng **Hello World** và các dự án Android Studio khác hoặc thay đổi thành thư mục mong muốn.
5. • Chấp nhận mặc định **android.example.com** cho **Company Domain**, hoặc tạo một tên miền công ty riêng. Nếu bạn không có kế hoạch xuất bản ứng dụng, hãy giữ nguyên mặc định. Lưu ý rằng việc thay đổi **package name** của ứng dụng sau này sẽ tốn công.
6. • Bỏ chọn **Include C++ support** và **Include Kotlin support**, sau đó nhấp **Next**.
7. • Trên màn hình **Target Android Devices**, chọn **Phone and Tablet**. Đảm bảo **API 15: Android 4.0.3 (Ice Cream Sandwich)** được đặt làm **Minimum SDK**. Nếu không, hãy sử dụng menu thả xuống để chọn.



Các cài đặt này giúp ứng dụng **Hello World** tương thích với **97% thiết bị Android** 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 yêu cầu thêm thành phần cho SDK đã chọn, Android Studio sẽ tự động cài đặt.
9. • Cửa sổ **Add an Activity** xuất hiện. **Activity** là một tác vụ mà người dùng có thể thực hiện. Đây là một thành phần quan trọng của mọi ứng dụng Android. Activity thường đi kèm với một **layout**, xác định cách các phần tử giao diện hiển thị trên màn hình. Chọn **Empty Activity**, sau đó nhấp **Next**.
10. Màn hình **Configure Activity** xuất hiện (giao diện này có thể khác nhau tùy theo mẫu bạn đã chọn ở bước trước). Mặc định, Empty Activity do mẫu cung cấp sẽ được đặt tên là MainActivity. Bạn có thể thay đổi nếu muốn, nhưng trong bài học này, chúng ta sẽ sử dụng MainActivity.

11. Đảm bảo rằng tùy chọn **Generate Layout file** được chọn. Mặc định, tên layout sẽ là **activity_main**. Bạn có thể thay đổi nếu muốn, nhưng trong bài học này, chúng ta sẽ 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 ngược với các phiên bản Android trước đó.
13. Nhấp vào **Finish**.

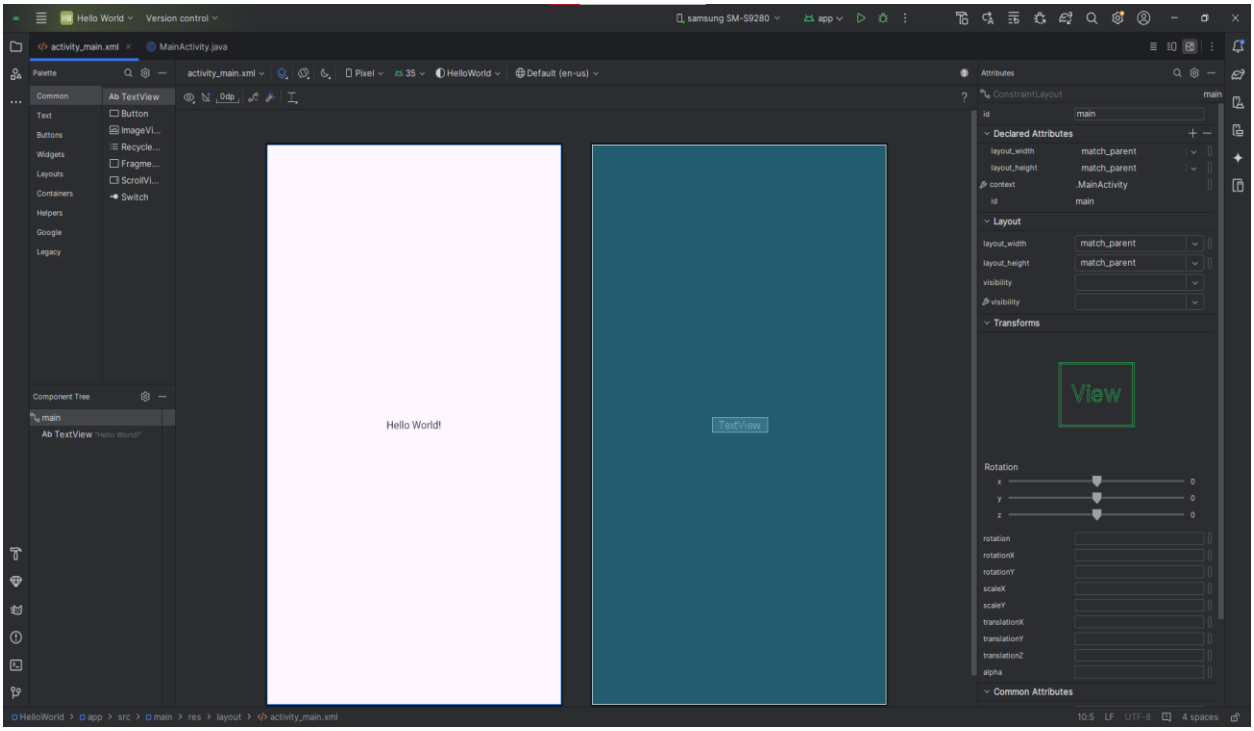
Android Studio sẽ tạo một thư mục cho dự án của bạn và tiến hành xây dựng dự án bằng **Gradle** (quá trình này có thể mất vài phút).

Mẹo: Truy cập trang **Configure your build** dành cho nhà phát triển để biết thông tin chi tiết.

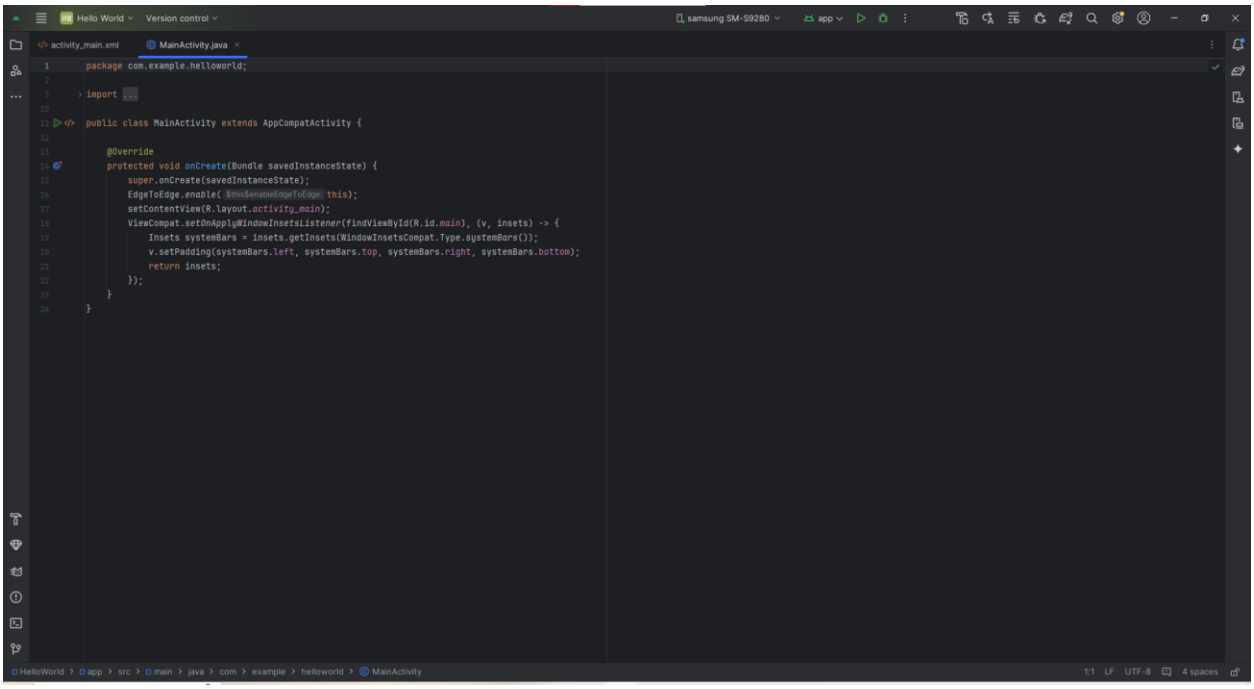
Bạn cũng có thể thấy một thông báo **"Tip of the day"** hiển thị các phím tắt và mẹo hữu ích khác. Nhấp vào **Close** để đóng thông báo.

Giao diện chỉnh sửa của Android Studio xuất hiện. Thực hiện các bước sau:

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



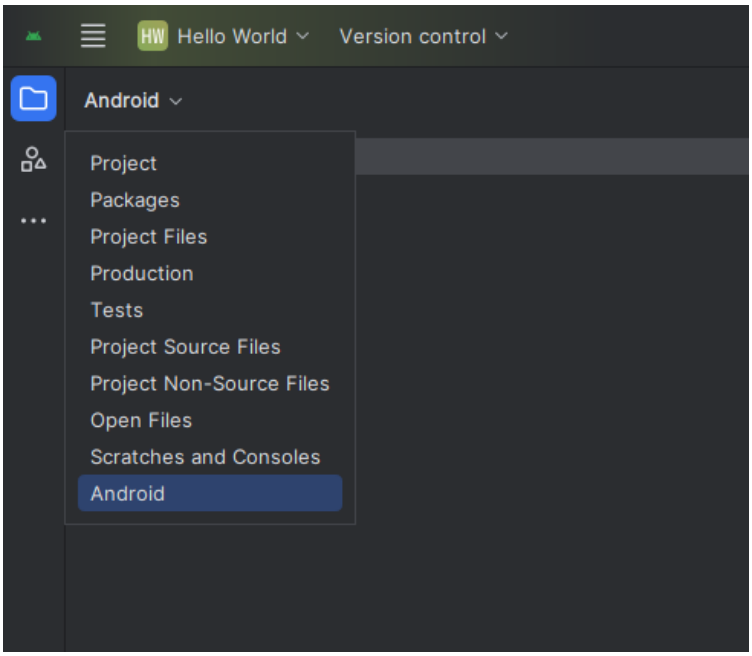
3. Nhấp vào tab MainActivity.java để mở trình chỉnh sửa mã như hình dưới đây.



2.2 Khám phá Project > Android pane

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

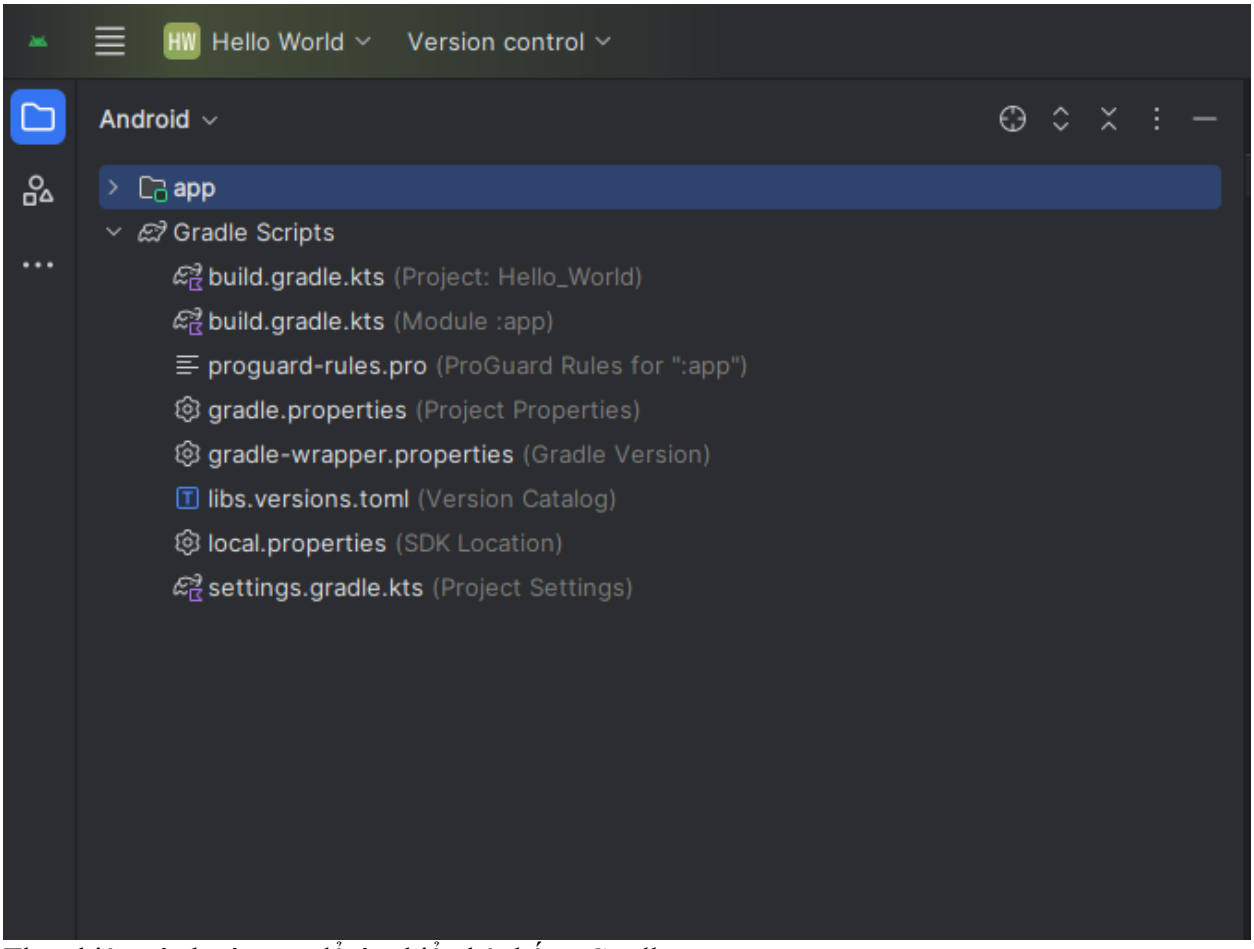
1. Nếu chưa được chọn, nhấp vào tab Project trong cột tab dọc bên trái cửa sổ Android Studio. Project pane sẽ xuất hiện.
2. Để xem dự án theo cấu trúc chuẩn của Android, chọn Android từ menu thả xuống ở đầu Project pane, như hình dưới đây.



2.3 Khám phá thư mục Gradle Scripts

Hệ thống Gradle build trong Android Studio giúp bạn dễ dàng thêm các tệp nhị phân bên ngoài hoặc các mô-đun thư viện khác vào dự án dưới dạng dependencies.

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



Thực hiện các bước sau để tìm hiểu hệ thống Gradle:

1. Nếu thư mục **Gradle Scripts** chưa được mở rộng, nhấp vào biểu tượng **tam giác** để mở rộng.
   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 Android Studio đều có một tệp Gradle build cấp cao nhất. Hầu hết thời gian, bạn không cần chỉnh sửa tệp này, nhưng hiểu nội dung của nó vẫn rất hữu ích.

   Mặc định, tệp build cấp cao nhất sử dụng **khối buildscript** để xác định **kho lưu trữ Gradle** và **dependencies** chung cho tất cả các mô-đun trong dự án. Khi một dependency

không phải là thư viện cục bộ hoặc tệp trong cây thư mục, Gradle sẽ tìm kiếm các tệp trong **các kho lưu trữ trực tuyến** được chỉ định trong khối **repositories** của tệp này.

Theo mặc định, các dự án mới trong Android Studio khai báo **JCenter** và **Google** (bao gồm cả Google Maven repository) làm vị trí kho lưu trữ:

3. Tìm tệp **build.gradle (Module: app)**.
   Ngoài tệp **build.gradle** cấp dự án, mỗi mô-đun cũng có 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 giúp bạn tùy chỉnh **tùy chọn đóng gói**, chẳng hạn như **các kiểu build bổ sung** và **các 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.

   Đây là tệp thường 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 **thư viện phụ thuộc** bằng nhiều cách khác nhau, mỗi cách sẽ cung cấp hướng dẫn khác nhau cho Gradle về cách sử dụng thư viện.

   Ví dụ, câu lệnh sau:

   ```
   implementation fileTree(dir: 'libs', include: ['*.jar'])
   ```

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

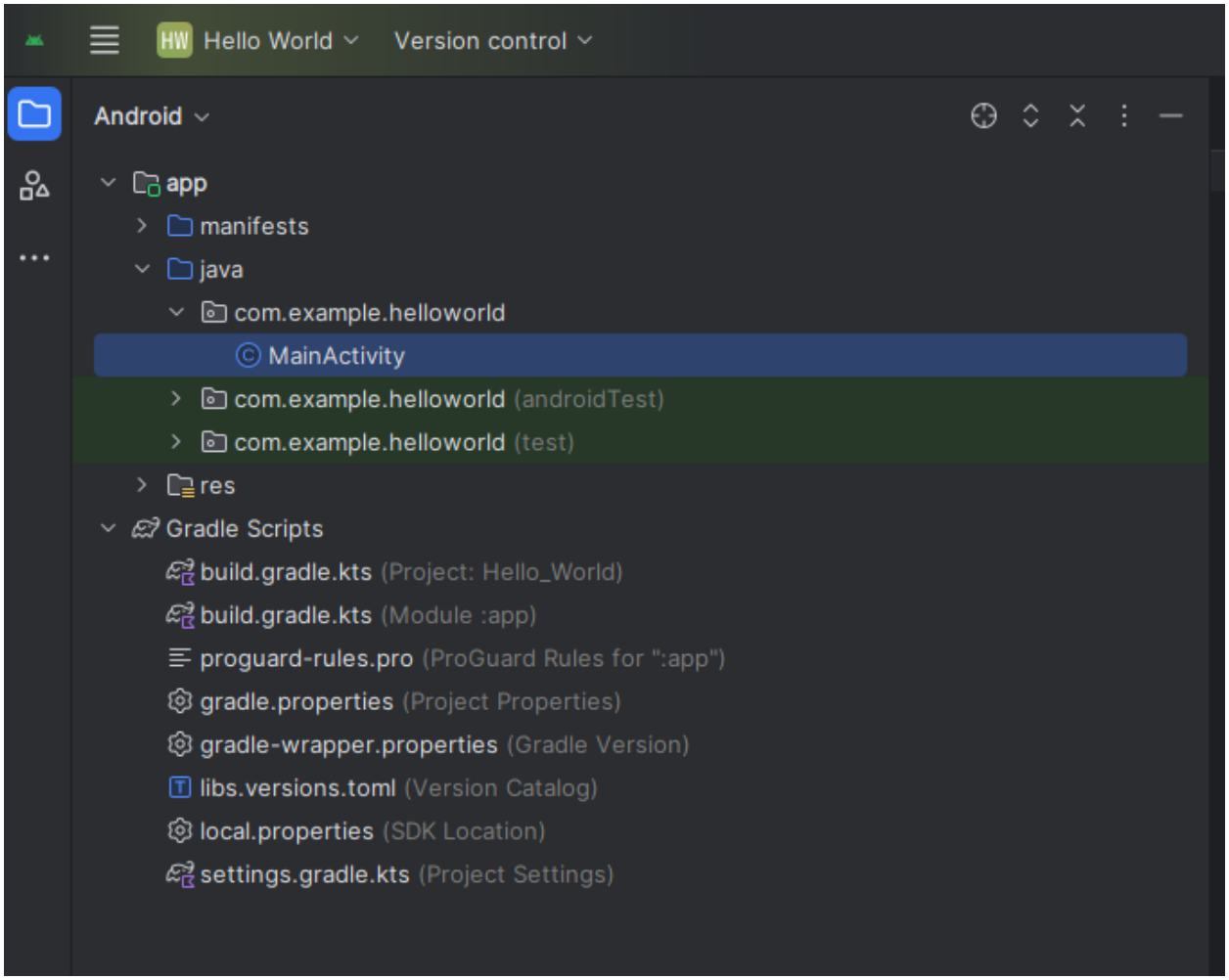
   Dưới đây là nội dung tệp **build.gradle (Module: app)** của ứng dụng **HelloWorld**:

4. Nhấp vào biểu tượng tam giác để đóng Gradle Scripts.

2.4 Tìm hiểu thư mục app và res

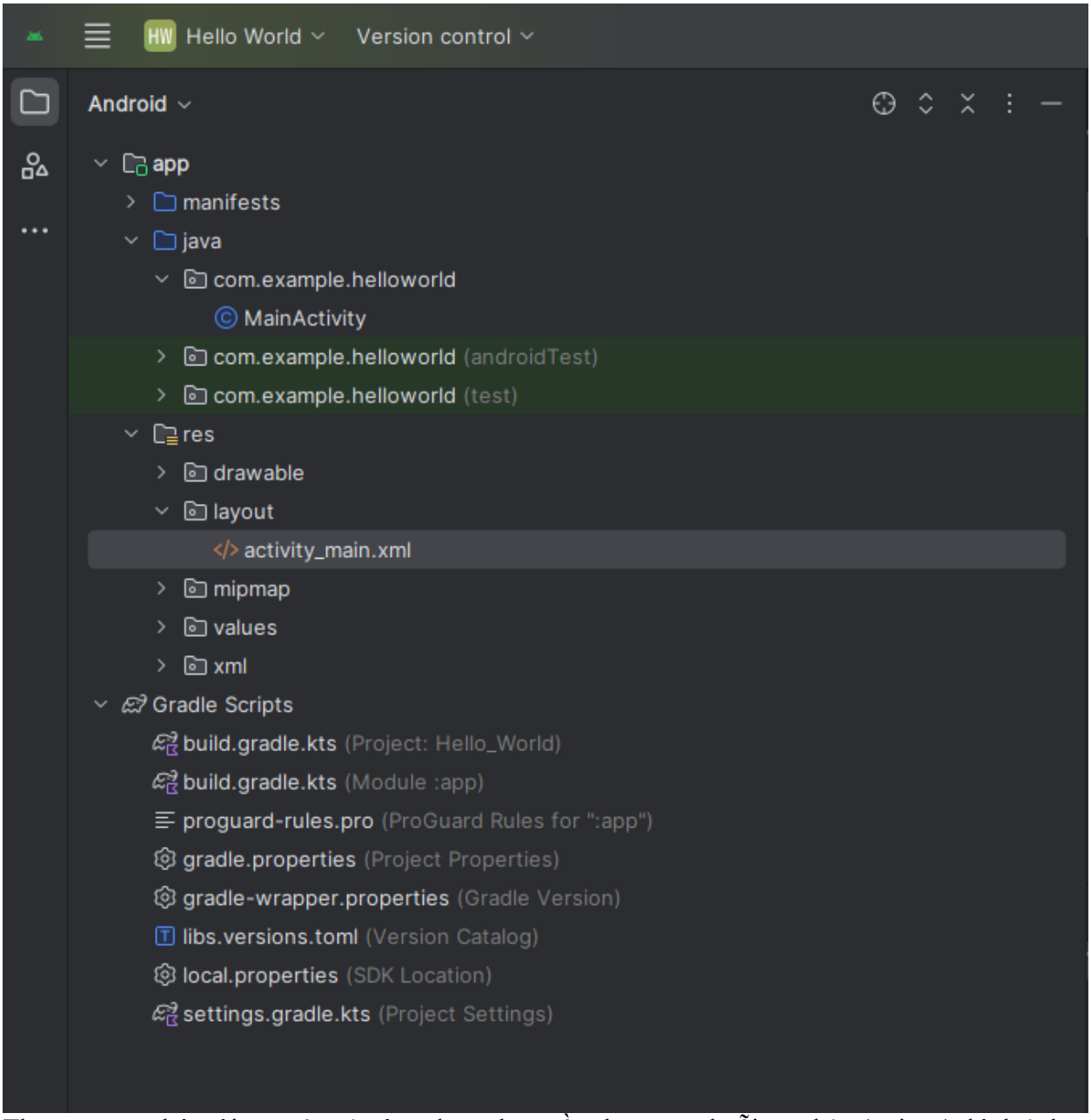
1. Tất cả mã nguồn và tài nguyên của ứng dụng được lưu trữ trong thư mục app và res.

Mở rộng thư mục app, sau đó mở rộng thư mục java và thư mụccom.example.android.helloworld để tìm tệp MainActivity.java. Nhấp đúp vào tệp này để mở trong trình chỉnh sửa mã.



Thư mục java chứa các tệp Java class trong ba thư mục con, như hình minh họa bên trên. Thư mục com.example.android.helloworld (hoặc tên miền bạn đã đặt) chứa tất cả các tệp của một gói ứng dụng. Hai thư mục còn lại được sử dụng cho mục đích kiểm thử (testing) và sẽ được giải thích trong một bài học khác. Trong ứng dụng Hello World, chỉ có một gói và nó chứa MainActivity.java. MainActivity là tên thông thường của Activity đầu tiên mà người dùng nhìn thấy khi mở ứng dụng, đồng thời khởi tạo các tài nguyên dùng chung của ứng dụng. Trong Project > Android pane, phần mở rộng .java của tệp này sẽ bị ẩn.

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



Thư mục res chứa tài nguyên của ứng dụng, bao gồm layouts, chuỗi văn bản (strings), hình ảnh (images), v.v. Một Activity thường liên kết với một layout được định nghĩa trong một tệp XML. Thông thường, tệp này sẽ được đặt tên theo tên của Activity tương ứng.

2.5 Tìm hiểu thư mục manifests

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

3. Mở rộng thư mục manifests.
4. Mở tệp AndroidManifest.xml.

Tệp AndroidManifest.xml mô tả tất cả các thành phần của ứng dụng Android. Mọi thành phần trong ứng dụng, chẳng hạn như 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 (permissions) cho ứng dụng. Để tìm hiểu tổng quan, hãy xem App Manifest Overview.

Nhiệm vụ 3: Sử dụng thiết bị ảo (trình mô phỏng)

Trong tác vụ này, bạn sẽ sử dụng trình quản lý Thiết bị ảo Android (AVD) để tạo một thiết bị ảo (còn được gọi là trình mô phỏng) mô phỏng cấu hình cho một loại thiết bị Android cụ thể và sử dụng thiết bị ảo đó để chạy ứng dụng. Xin lưu ý rằng Trình mô phỏng Android có các yêu cầu bổ sung ngoài các yêu cầu hệ thống cơ bản đối với Android Studio.

Khi sử dụng Trình quản lý AVD, bạn xác định các đặc điểm phần cứng của thiết bị, cấp độ API, bộ nhớ, giao diện và các thuộc tính khác và lưu thiết bị đó dưới dạng thiết bị ảo. Với 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 phải sử dụng thiết bị thực.

3.1 Tạo thiết bị ảo Android (AVD)

Để chạy **trình giả lập (emulator)** trên máy tính, bạn cần 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.
    - Màn hình **Your Virtual Devices** sẽ xuất hiện.
    - Nếu bạn đã tạo thiết bị ảo trước đó, danh sách sẽ hiển thị chúng (như hình minh họa bên dưới).
    - Nếu chưa có thiết bị ảo nào, danh sách sẽ trống.

ảnh

2. Nhấp vào +Tạo thiết bị ảo. Cửa sổ Chọn phần cứng xuất hiện hiển thị danh sách các thiết bị phần cứng được định cấu hình sẵn. Đối với mỗi thiết bị, bảng cung cấp một cột cho kích thước hiển thị đường chéo (Kích thước), độ phân giải màn hình tính bằng pixel (Độ phân giải) và mật độ điểm ảnh (Mật độ).

ảnh

3. Chọn một thiết bị như Nexus 5x hoặc Pixel XL và nhấp vào Tiếp theo . Màn hình Hình ảnh hệ thống xuất hiện.
4. Nhấp vào tab Đề xuất nếu nó chưa được chọn và chọn phiên bản hệ thống Android để chạy trên thiết bị ảo (chẳng hạn như Oreo).

ảnh

Có nhiều phiên bản hơn được hiển thị trong tab Đề xuất. Nhìn vào tab Hình ảnh x86 và Hình ảnh khác để xem chúng.

Nếu liên kết Tải xuống hiển thị bên cạnh hình ảnh hệ thống bạn muốn sử dụng, thì liên kết đó chưa được cài đặt. Nhấp vào liên kết để bắt đầu tải xuống và nhấp vào Kết thúc khi hoàn tất.

5. Sau khi chọn hình ảnh hệ thống, hãy nhấp vào Tiếp theo . Cửa sổ Thiết bị ảo Android (AVD) sẽ xuất hiện. Bạn cũng có thể thay đổi tên của AVD. Kiểm tra cấu hình của bạn và nhấp vào Finish .

3.2 Chạy ứng dụng trên thiết bị ảo Trong tác vụ này, cuối cùng bạn sẽ chạy ứng dụng Hello World của mình.

1. Trong Android Studio, chọn Ứng dụng Run > Run hoặc nhấp vào biểu tượng Run trên thanh công cụ.

2. Cửa sổ Chọn mục tiêu triển khai, trong Thiết bị ảo có sẵn , chọn thiết bị ảo mà bạn vừa tạo và nhấp vào OK .

ảnh

Trình giả lập khởi động và khởi động 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 xây dựng và sau khi trình mô phỏng đã sẵn sàng, Android Studio sẽ tải ứng dụng lên trình mô phỏng và chạy ứng dụng đó. Bạn sẽ thấy ứng dụng Hello World như trong hình sau.

ảnh

Mẹo: Khi thử nghiệm trên một thiết bị ảo, bạn nên khởi động nó một lần, ngay từ đầu phiên của bạn. Bạn không nên đóng ứng dụng cho đến khi hoàn tất việc kiểm tra ứng dụng của mình để ứng dụng của bạn không phải trải qua quá trình khởi động thiết bị một lần nữa. Để đóng thiết bị ảo, hãy nhấp vào nút X ở đầu trình giả lập, chọn Thoát từ menu hoặc nhấn Control-Q trong Windows hoặc Command-Q trong 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 thiết bị di động vật lý như điện thoại hoặc máy tính bảng. Bạn phải luôn kiểm tra ứng dụng của mình trên cả thiết bị ảo và thiết bị vật lý. Những gì bạn cần:

- Thiết bị Android 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, bạn có thể cần thực hiện các bước bổ sung để chạy trên thiết bị phần cứng. Kiểm tra tài liệu Using Hardware Devices. Bạn cũng có thể cần cài đặt trình điều khiển USB thích hợp cho thiết bị của mình. Đối với trình điều khiển USB dựa trên Windows, hãy xem Trình OEM USB Drivers.

4.1 Bật chế độ gỡ lỗi USB

Để Android Studio có thể giao tiếp với thiết bị của bạn, bạn cần bật USB Debugging trên thiết bị Android. Tùy chọn này nằm trong Developer options (Tùy chọn nhà phát triển).

Trên Android 4.2 trở lên, màn hình Developer options bị ẩn theo mặc định. Để hiển thị Developer options và bật USB Debugging, làm theo các bước sau:

1. Trên thiết bị của bạn, mở Settings (Cài đặt), tìm và chọn About phone (Giới thiệu về điện thoại), sau đó nhấn vào Build number (Số phiên bản) bảy lần.

2. Quay lại màn hình trước (Settings / System). Mục Developer options (Tùy chọn nhà phát triển) sẽ xuất hiện trong danh sách. Nhấn vào đó.

3. Tìm và bật USB Debugging.

4.2 Chạy ứng dụng trên thiết bị

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

1. Kết nối thiết bị với máy tính bằng cáp USB.
2. Nhấp vào nút Run trên thanh công cụ. Cửa sổ Select Deployment Target sẽ mở ra, hiển thị danh sách các trình giả lập và thiết bị đã kết nối.
3. Chọn thiết bị của bạn và nhấp vào OK.

Android Studio sẽ cài đặt và chạy ứng dụng trên thiết bị của bạn.

Khắc phục sự cố

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

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

Nếu máy tính vẫn không tìm thấy thiết bị hoặc báo lỗi "unauthorized", hãy làm theo các bước sau:

1. Rút thiết bị ra khỏi máy tính.
2. Trên thiết bị, mở Developer Options trong ứng dụng 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 chấp nhận.

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

5.1 Thay đổi phiên bản SDK tối thiểu cho ứng dụng

Làm theo các bước sau:

1. Mở rộng thư mục Gradle Scripts nếu nó chưa được mở, sau đó nhấp đúp vào tệp build.gradle(Module:app). Nội dung của tệp sẽ hiển thị trong trình chỉnh sửa mã.
2. Trong khối defaultConfig, thay đổi giá trị minSdkVersion thành 17, như ví dụ bên dưới (giá trị ban đầu là 15).

ảnh

Trình chỉnh sửa mã sẽ hiển thị một thanh thông báo ở trên cùng với liên kết Sync Now.

5.2 Đồng bộ cấu hình Gradle mới

Khi bạn thay đổi các tệp cấu hình build.gradle trong dự án, Android Studio yêu cầu bạn đồng bộ (sync) lại dự án. Việc này giúp nhập các thay đổi và kiểm tra xem có lỗi biên dịch nào không.

Để đồng bộ dự án, thực hiện một trong hai cách sau:

Nhấp vào Sync Now trên thanh thông báo xuất hiện sau khi thay đổi (như trong hình trước đó).

Nhấp vào biểu tượng Sync Project with Gradle Files trên thanh công cụ.

Sau khi quá trình đồng bộ Gradle hoàn tất, một thông báo "Gradle build finished" sẽ xuất hiện ở góc dưới bên trái của cửa sổ Android Studio.

Để tìm hiểu chi tiết hơn về Gradle, hãy xem tài liệu Build System Overview và Configuring Gradle Builds.

6.1 Xem bảng Logcat

Để mở bảng Logcat, nhấp vào tab Logcat ở phía dưới cửa sổ Android Studio, như hình minh họa bên dưới.

ảnh

Trong hình minh họa bên trên:

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

6.2 Thêm câu lệnh log vào ứng dụng

Các câu lệnh Log trong mã ứng dụng sẽ hiển thị thông báo trong bảng Logcat. Ví dụ:

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

Các phần của thông điệp log bao gồm:

- **Log**: Lớp **Log** được sử dụng để gửi thông báo log đến bảng **Logcat**.
- **d**: Cấp độ log **Debug**, giúp lọc các thông báo log trong bảng **Logcat**. Các cấp độ khác bao gồm:
    - **e**: **Error** (Lỗi)
    - **w**: **Warn** (Cảnh báo)
    - **i**: **Info** (Thông tin)
- **"MainActivity"**: Đối số đầu tiên là **tag** giúp lọc thông báo trong bảng **Logcat**. Thông thường, **tag** là tên của **Activity** chứa thông báo, nhưng bạn có thể đặt bất kỳ giá trị nào hữu ích cho việc gỡ lỗi.
    - Theo quy ước, **tag** log được định nghĩa dưới dạng hằng số trong **Activity** như sau:

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

- **"Hello world"**: Đối số thứ hai là nội dung thông báo thực sự.
- **Thực hiện các bước sau:**

1. Mở ứng dụng **Hello World** trong **Android Studio**, sau đó mở **MainActivity**.
2. Để tự động thêm các **import** cần thiết vào dự án (ví dụ: **android.util.Log** để sử dụng **Log**), vào:
    - **Windows**: Chọn **File > Settings**
    - **macOS**: Chọn **Android Studio > Preferences**
3. Chọn **Editor > General > Auto Import**.
    - **Chọn tất cả các hộp kiểm**
    - Đặt **Insert imports on paste** thành **All**
4. Nhấn **Apply**, sau đó nhấn **OK**.
5. Trong phương thức **onCreate()** của **MainActivity**, thêm câu lệnh sau:

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

Sau khi thêm, phương thức **onCreate()** sẽ trông như sau:

```
@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

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

}
```

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

Thông báo sau sẽ xuất hiện trong bảng **Logcat**:
11-24 14:06:59.001 4696-4696/? D/MainActivity: 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 và 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 chào "Hello World" thành **"Happy Birthday to "** và thêm tên của một người vừa có sinh nhật gần đây.
3. *(Tùy chọn)* Chụp ảnh màn hình ứng dụng đã hoàn thành và gửi email cho người có sinh nhật mà bạn đã quên.
4. Một cách sử dụng phổ biến của lớp **Log** là ghi lại các ngoại lệ (**exception**) trong **Java** khi chúng xảy ra trong chương trình. Có một số phương thức hữu ích như **Log.e()** mà bạn có thể dùng cho mục đích này.
    - Tìm hiểu các phương thức cho phép bạn ghi lại ngoại lệ trong **Log**.
    - Sau đó, viết mã trong ứng dụng của bạn để kích hoạt và ghi lại một ngoại lệ

Tóm tắt

- Để cài đặt **Android Studio**, truy cập trang **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 phân cấp Android của ứng dụng trong **Project pane**, nhấn vào tab **Project** ở cột dọc bên trái, sau đó chọn **Android** trong menu thả xuống ở phía trên.
- Chỉnh sửa tệp **build.gradle(Module:app)** khi 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 nằm trong thư mục **app** và **res**.
    - **Thư mục java** chứa các activity, bài kiểm tra và các thành phần khác bằng mã nguồn **Java**.
    - **Thư mục res** chứa các tài nguyên như giao diện bố cục, chuỗi ký tự và hình ảnh.
- Chỉnh sửa tệp **AndroidManifest.xml** để thêm các thành phần và quyền truy cập vào ứng dụng Android.
    - 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**) và chạy ứng dụng của bạn.
- Thêm **Log** vào ứng dụng để hiển thị thông báo trong **Logcat**, giúp theo dõi và gỡ lỗi dễ dàng hơn.
- Để chạy ứng dụng trên thiết bị Android thật bằng **Android Studio**, hãy bật **USB Debugging**:

1. Mở **Settings > About phone** và nhấn **Build number** 7 lần.
2. Quay lại **Settings**, nhấn vào **Developer options** và bật **USB Debugging**.

- **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**

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 **views** – 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 của **View** thường được sử dụng bao gồm:
● **TextView** – Hiển thị văn bản.
● **EditText** – Cho phép người dùng nhập và chỉnh sửa văn bản.
● **Button** và các phần tử có thể nhấp (**RadioButton**, **CheckBox**, **Spinner**) – Cung cấp hành vi tương tác.
● **ScrollView** và **RecyclerView** – Hiển thị danh sách có thể cuộn.
● **ImageView** – Hiển thị hình ảnh.
● **ConstraintLayout** và **LinearLayout** – Chứa các phần tử **View** khác và định vị chúng trên giao diện.

Mã Java điều khiển và hiển thị UI nằm trong một lớp mở rộng từ **Activity**. Một **Activity** thường được liên kết với một bố cục giao diện UI được định nghĩa trong 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 bố cục được định nghĩa trong tệp **activity_main.xml**, trong đó có một **TextView** với nội dung "Hello World".

Trong các ứng dụng phức tạp hơn, một **Activity** có thể thực hiện các chức năng sau:
● Xử lý phản hồi khi người dùng chạm hoặc nhấn vào các phần tử UI.
● Vẽ nội dung đồ họa.
● Gửi và nhận dữ liệu từ cơ sở dữ liệu hoặc internet.

Bạn sẽ tìm hiểu chi tiết hơn về **Activity** trong các bài học tiếp theo.

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 ứng dụng bằng mẫu **Empty Activity**, đồng thời học cách sử dụng **layout editor** để thiết kế giao diện và chỉnh sửa bố cục trong XML. Bạn cần phát triển những kỹ năng này để hoàn thành các bài thực hành khác trong khóa học.

Những gì bạn cần biết trước

Bạn nên quen thuộc với:

● Cách cài đặt và mở Android Studio.

● Cách tạo ứng dụng HelloWorld.

● Cách chạy ứng dụng HelloWorld.

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 **layout** bằng **XML**.
● Nhiều thuật ngữ mới. Hãy xem **bảng thuật ngữ và khái niệm** để có định nghĩa dễ hiểu.

Những gì bạn sẽ làm

● Tạo một ứng dụng và thêm hai Button cùng một TextView vào layout.

● 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 giao diện người dùng (UI elements).

● Chỉnh sửa layout của ứng dụng bằng XML.

● Trích xuất chuỗi mã cứng (hardcoded strings) thành tài nguyên chuỗi (string resources).

● Triển khai các phương thức xử lý sự kiện nhấn (click-handler methods) để hiển thị thông báo trên màn hình khi người dùng nhấn vào từng Button.

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

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

Nhiệm vụ 1: Tạo và khám phá một dự án mới
Trong bài thực hành này, bạn sẽ thiết kế và triển khai một dự án cho ứng dụng HelloToast. Một liên kết đến mã giải pháp sẽ được cung cấp ở cuối.

1.1 Tạo dự án Android Studio

1. Mở Android Studio và tạo một dự án mới với các tham số sau:
2. Chọn **Run > Run app** hoặc nhấp vào **biểu tượng Run** trên thanh công cụ để biên dịch và chạy ứng dụng trên trình giả lập hoặc thiết bị của bạn.

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) RecycleView

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