4

Lab

LẬP TRÌNH ỨNG DỤNG ANDROID CƠ BẢN

Basic Android Programming

Thực hành Bảo mật web và ứng dụng

Lưu hành nội bộ

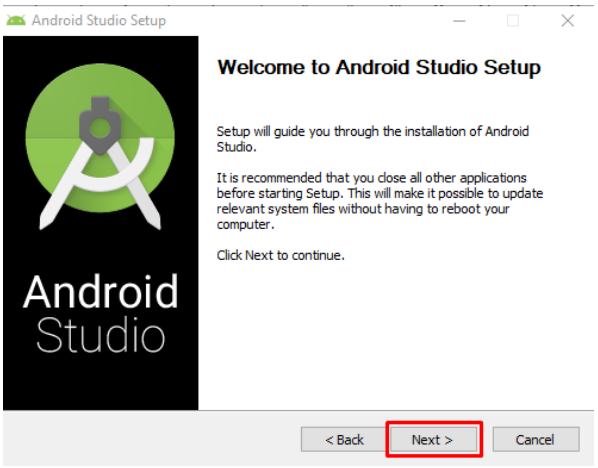
A. TỔNG QUAN

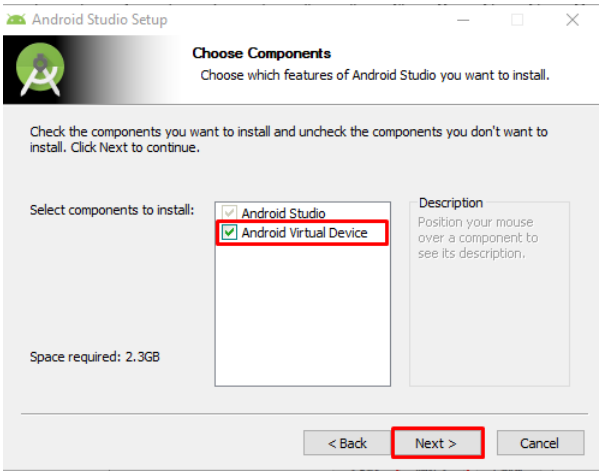
Giúp sinh viên có kiến thức và kỹ năng cơ bản trong việc lập trình các ứng dụng Android đơn giản sử dụng **Android Studio** và bước đầu sử dụng công cụ ProGuard để tối ưu mã nguồn của mình.

B. CHUẨN BỊ MÔI TRƯỜNG

B.1 Cài đặt công cụ Android Studio

Sinh viên tải và cài đặt Android Studio tương ứng với hệ điều hành của máy tính, tham khảo tại đường dẫn Hướng dẫn cài đặt Android Studio.





B.2 Thiết bị Android giả lập

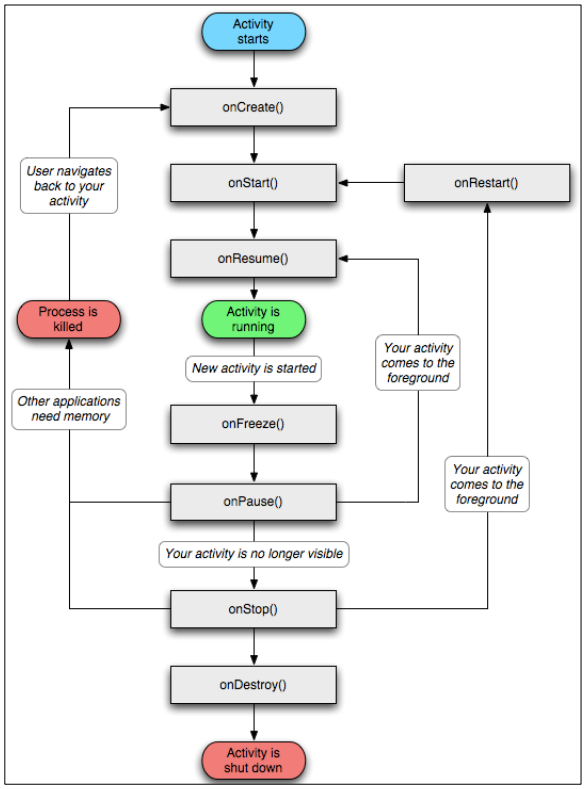
Sinh viên có thể sử dụng Android Emulator mặc định của Android Studio hoặc cài đặt Genymotion (Tải Genymotion) để giả lập thiết bị Android trong quá trình lập trình ứng dụng.

Sinh viên cài đặt 1 thiết bị Android giả lập với phiên bản Android tuỳ ý. Tuy nhiên, cần ghi nhớ giá trị API của thiết bị đã cài.

C. KIẾN THỨC LẬP TRÌNH ỨNG DỤNG ANDROID

C.1 Activity

- Về cơ bản, một **Activity** hiển thị một giao diện người dùng.
- Một ứng dụng có thể có 1 hoặc nhiều Activity, mỗi Activity kế thừa từ ***android.app.Activity*** và cần được định nghĩa trong ***AndroidManifest.xml.***
- Mỗi activity được cung cấp 1 window để hiển thị, window này có thể toàn màn hình hoặc nhỏ hơn và nằm bên trên window khác.
- Giao diện của window được dựng bằng một hệ phân cấp các object từ View class.
- Phương thức **Activity.setContentView()** thiết lập giao diện cho Activity.
- Vòng đời của 1 Activity:



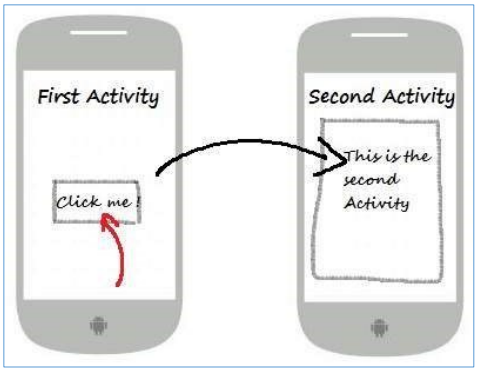
Tìm hiểu thêm về Activity: Activity trong Android

C.2 Intent & Intent Filter

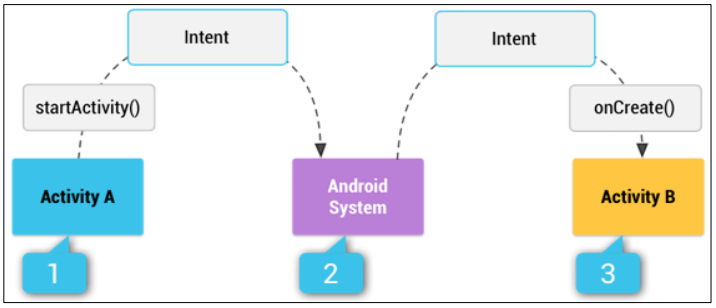
Intent trong Android có thể hiểu là lời nhắn để các Activity gọi nhau. Có 2 loại Intent:

- **Explicit Intent:** dùng để gọi nội bộ ứng dụng, thường là activity A gọi activity B.

```
Intent i = new Intent(FirstActivity.class, SeacondActivity.class);
startActivity(i);
```



- **Implicit Intent:** thường không cần tên của target. Implicit được sử dụng để gọi các thành phần của app khác.



- 1) Activty A tạo 1 intent với 1 lời mời "mời gọi đối tượng" cụ thể rồi gọi hàm startActivity()
- 2) Android System tự tìm ra các app có các đối tượng được định nghĩa trong "intent filter" của app khác.
- 3) Khi tìm ra được rồi => Hệ thống gọi actvity (Activity B) bằng cách gọi onCreate() method của nó và đem vào Intent.

Để giới hạn các Intent nào có thể gọi activity, Android sử dụng intent filter trong định nghĩa của activity đó với *<intent-filter>* trong tập tin *AndroidManifest.xml*:

```
<activity android:name=".MainActivity">
   <intent-filter android:scheme="http">
      <action android:name="android.intent.action.VIEW"></action>
      <action android:name="android.intent.action.MAIN" />
      <category android:name="android.intent.category.DEFAULT"></category>
      <category android:name="android.intent.category.LAUNCHER" />
   </intent-filter>
</activity>
```

⇨ Activity của app khác có thể được gọi thông qua intent android.intent.action.VIEW hoặc android.intent.category.LAUNCHER, android.intent.category.DEFAULT.

⇨ android.intent.category.LAUNCHER còn cho biết đây sẽ là activity chính, được chạy tự động khi ứng dụng khởi chạy.

Tìm hiểu thêm về Intent: Intent & Intent filter

C.3 Permission

Tìm hiểu về permission: Permission trong Android

D. THỰC HÀNH

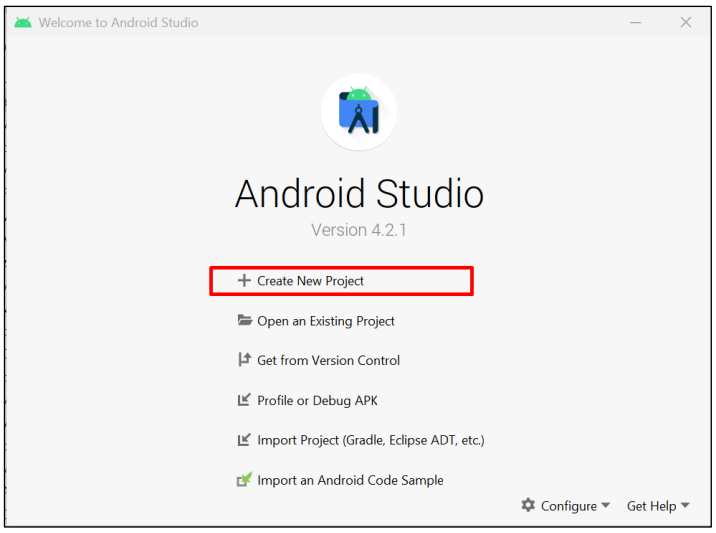
D.1 Làm quen với lập trình ứng dụng Android

D.1.1 Tạo ứng dụng Android đơn giản

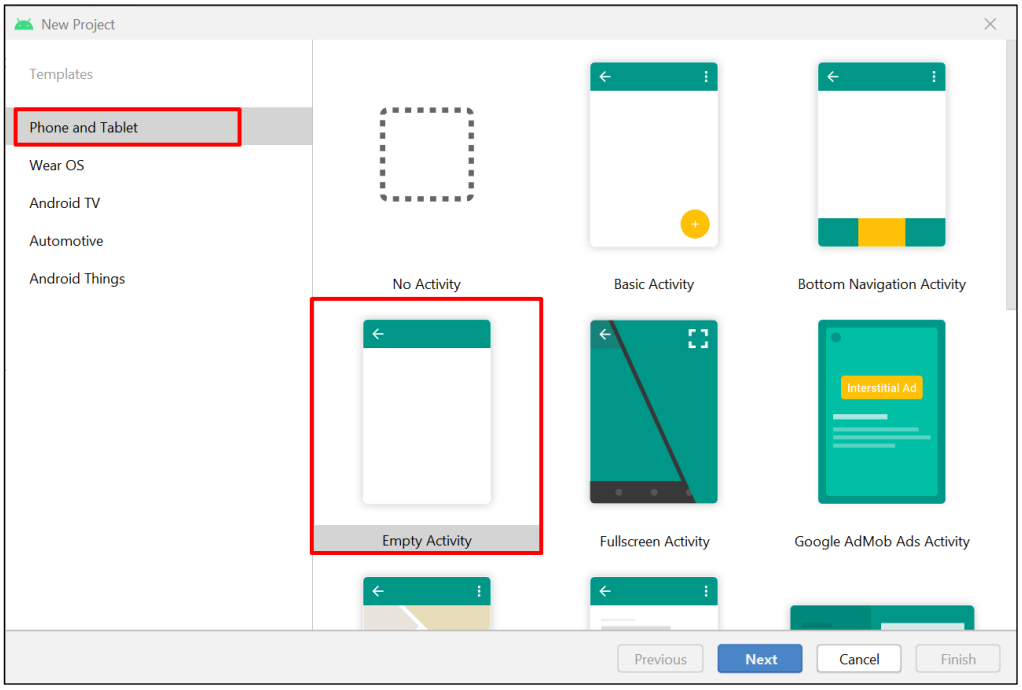
> **Yêu cầu 1.** Sinh viên thực hiện theo các bước hướng dẫn bên dưới để tạo một ứng dụng "Hello world" đơn giản trong Android Studio. ***Báo cáo bằng cách chụp màn hình ở một số bước <u>được yêu cầu</u>.***

Hướng dẫn:

- **<u>Bước 1:</u> Mở Android Studio:** Giao diện khi mở Android Studio



- **<u>Bước 2</u>: Tạo project mới** bằng cách chọn *Create New Project* > *Empty Activity* trong tab **Phone and Tablet** như bên dưới. Nhấn **Next** để tiếp tục.



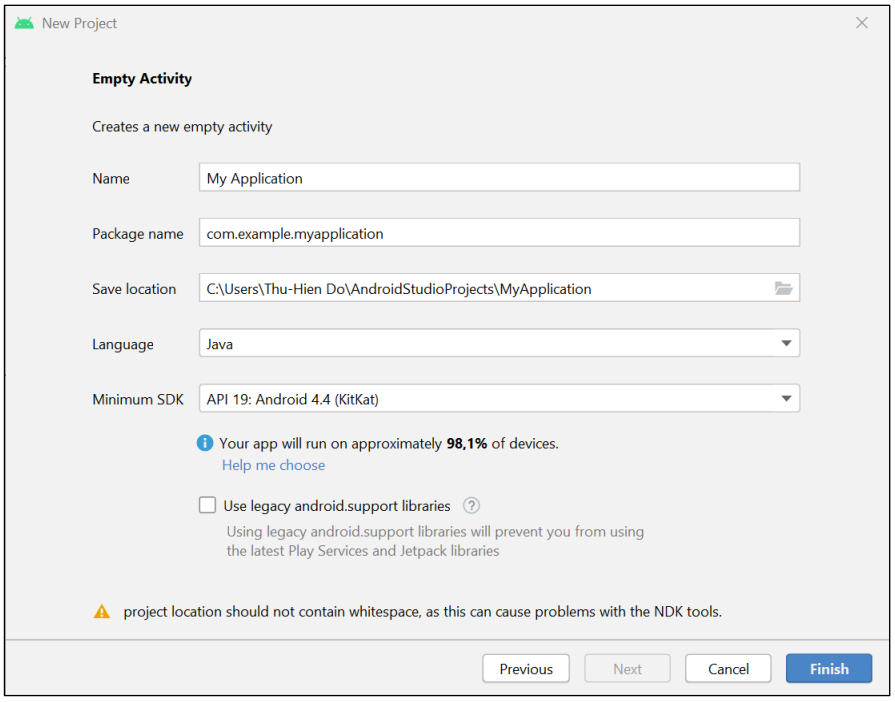
- **<u>Bước 3</u>: Cấu hình cho project**

Điền các thông tin cần thiết cho ứng dụng Android: tên ứng dụng, tên package chứa nó, vị trí lưu mã nguồn, ngôn ngữ (thường là Java) và phiên bản API tối thiểu của thiết bị để chạy ứng dụng này.

Lưu ý: API tối thiểu của ứng dụng phải đảm bảo phù hợp với API của thiết bị Android giả lập. API càng thấp thì càng có nhiều thiết bị hỗ trợ được ứng dụng, như ở đây **API 19 của Android 4.4** thì gần như **98.1%** thiết bị sẽ chạy được. Sau đó chọn **Finish**.

Đặt tên ứng dụng là: **NhomX Application**.

(Chụp màn hình minh chứng cấu hình: tên ứng dụng, API tối thiểu,…)

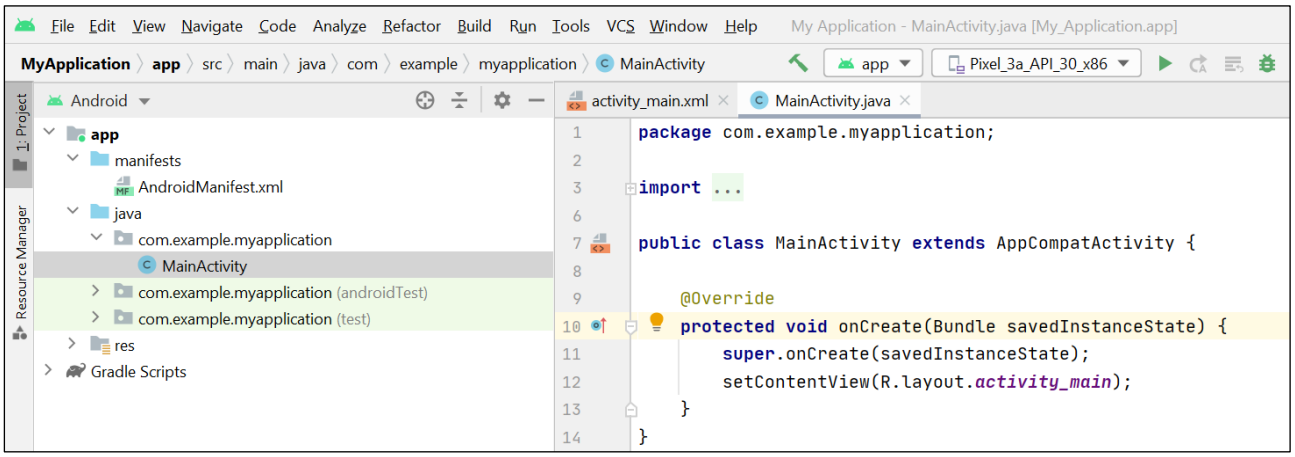


Android Studio sẽ tiến hành set-up project cho ứng dụng Android cần tạo. *Lưu ý bước này nên đảm bảo có kết nối mạng để Android Studio tải các gói cần thiết.*

- **<u>Bước 4</u>: Xem các file mã nguồn của ứng dụng**:

- **manifests/AndroidManifest.xml**: file cấu hình ứng dụng Android, khai báo cấu trúc thành phần ứng dụng, định nghĩa khai báo các quyền hạn của ứng dụng…
- **Các file java của các activity**: định nghĩa function thực hiện chức năng của activity.
- **Các file layout xml của các activity**: định nghĩa giao diện của các activity. Trong Android Studio có tùy chọn **Design** cho phép kéo thả các thành phần hoặc **Text** để thêm các thành phần bằng mã nguồn XML.

Code chính của project vừa tạo sẽ nằm ở tập tin ***MainActivity.java***, với một hàm ***onCreate()*** được tự động tạo để chạy khi activity khởi chạy. Mặc định khi mới tạo, hàm này chỉ hiển thị một cửa sổ trống ứng với giao diện activity MainActivity của ứng dụng.



- **<u>Bước 5</u>: Thêm chức năng in ra thông điệp**

Dòng code bên dưới thêm ở hàm **onCreate()** để in ra một dòng thông điệp "Hello world" ở log của Android (log cat).

Sinh viên điều chỉnh nội dung để chuỗi in ra có dạng **"Hello MSSV1 – MSSV2"**.

*(Chụp màn hình đoạn code đã điều chỉnh trong **onCreate()**)*

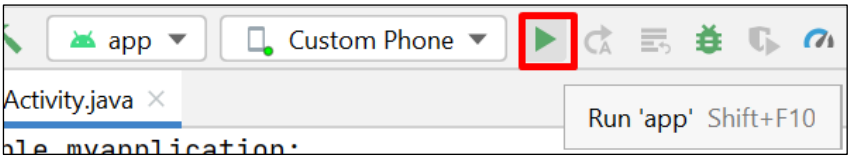
```
Log.d("Android Mobile", "Hello World");
```

```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Log.d( tag: "Android Mobile ", msg: "Hello World");
    }
}
```

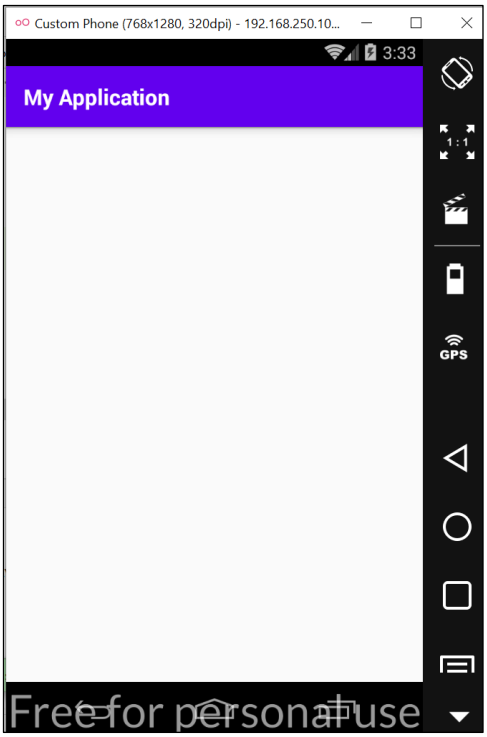
- **<u>Bước 6</u>: Chạy code và quan sát kết quả**

<u>Lưu ý:</u> Nếu sử dụng Genymotion thì cần chạy thiết bị giả lập Android trước.

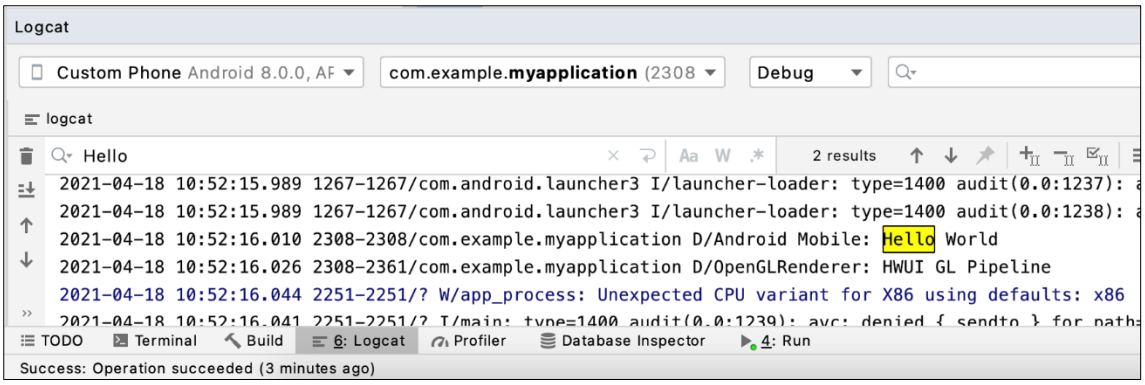
Lựa chọn thiết bị Android sẽ chạy ứng dụng, sau đó chọn ***Run 'app'*** trên thanh công cụ để chạy ứng dụng.



Ứng dụng mới tạo sẽ được cài đặt và tự động khởi chạy trên một thiết bị Android đã chọn nếu như có cấu hình phù hợp.

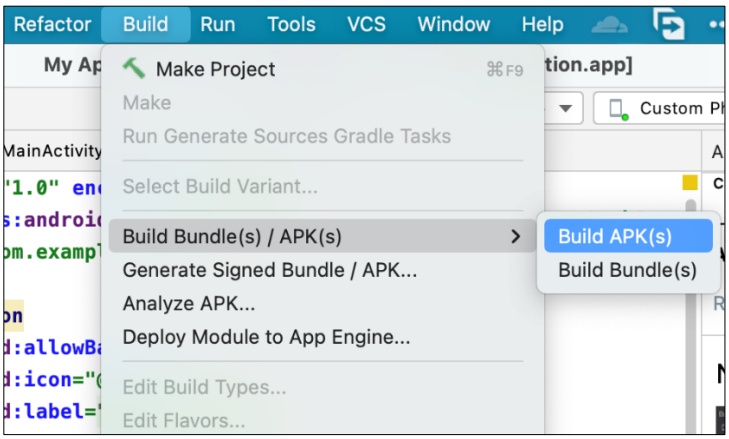


Nhìn vào Logcat (Debug) thì in được thông điệp, ở đây là thông điệp **"Hello world"**. Quan sát và *Chụp màn hình kết quả in chuỗi trong log tương ứng với mỗi nhóm sinh viên*.

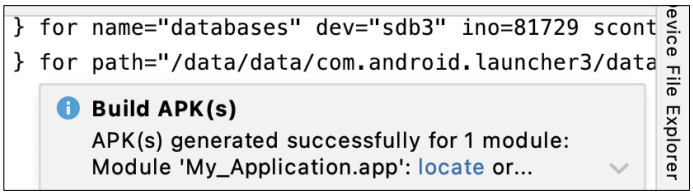


- **<u>Bước 7:</u> Build ứng dụng thành file APK**

Khi muốn build ứng dụng vừa code thành tập tin APK để có thể cài đặt trên nhiều thiết bị Android, sử dụng ***Build > Build Bundle(s)/APK > Build APK(s)***



Kết quả build thành công:



Có thể click chọn ***locate*** trong khung thông báo trên để đến được thư mục chứa tập tin APK vừa build. *Chụp màn hình thư mục chứa file APK đã build.*

- **Bước 8: Thực nghiệm để hiểu về vòng đời của 1 activity**

Bên cạnh hàm **onCreate()**, một Activity còn hỗ trợ nhiều hàm callback khác như **onStart(), onResume(),...** trong vòng đời của nó:

onCreate()	Được gọi khi activity được khởi tạo
onStart()	Được gọi khi activity bắt đầu hiện lên cho user thấy
onResume()	Được gọi khi activity được user sử dụng
onPause()	Được gọi khi user "focus" qua 1 activity khác
onStop()	Được gọi khi activity không còn được nhìn thấy bởi user
onDestroy()	Được gọi trước khi activity bị hệ thống xoá
onRestart()	Được gọi khi acitivity được bật lên sau khi stop

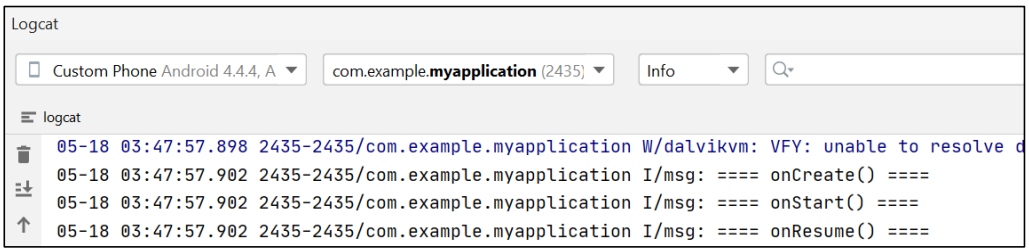
Tạo tất cả *callback* trong *MainActivity*, trong đó ghi 1 dòng log khi *callback* được gọi. Có thể click chuột phải vào tên MainActivity, chọn **Generate... →Override methods** và lựa chọn các callback muốn tạo.

```
//    Called when the activity is about to become visible.
    @Override
    protected void onStart() {
        super.onStart();
        Log.i(msg, "=== onStart() ===");
    }
```

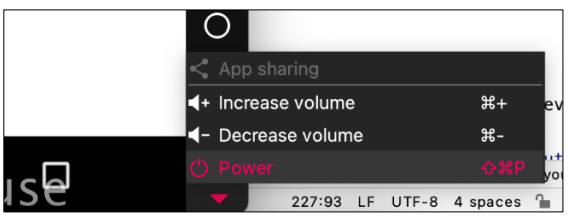
Chỉnh sửa và chạy ứng dụng, quan sát màn hình log (Info) khi thực hiện các tác vụ*:*

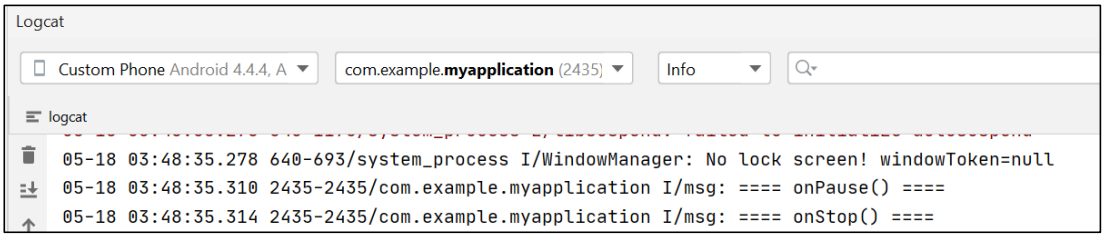
(Chụp màn hình khung log và thử lý giải thích thứ tự của các callback?)

- **Tác vụ 1:** Chạy ứng dụng với **Run 'app'**



- **Tác vụ 2:** Thử khoá thiết bị Android bằng cách chọn button **Power**





Lưu ý: khoá thiết bị cũng là 1 *activity* khác.

- **Tác vụ 3:** Thử nhấn button **Power** để mở phone

```
05-18 03:48:58.742 640-693/system_process I/WindowManager: No lock screen! windowToken=null
05-18 03:48:58.774 2435-2435/com.example.myapplication I/msg: ==== onRestart() ====
05-18 03:48:58.774 2435-2435/com.example.myapplication I/msg: ==== onStart() ====
05-18 03:48:58.774 2435-2435/com.example.myapplication I/msg: ==== onResume() ====
```

- **Tác vụ 4:** Tắt app

```
05-18 03:54:45.133 2673-2673/com.example.myapplication I/msg: ==== onPause() ====
05-18 03:54:45.181 1205-1205/com.android.launcher E/EGL_emulation: tid 1205: eglSurfaceAttrib(1210)
05-18 03:54:45.181 1205-1205/com.android.launcher W/HardwareRenderer: Backbuffer cannot be preserve
05-18 03:54:45.665 2673-2673/com.example.myapplication I/msg: ==== onStop() ====
05-18 03:54:45.665 2673-2673/com.example.myapplication I/msg: ==== onDestroy() ====
```

D.1.2 Mở rộng chức năng của MainActivity

> **Yêu cầu 2.** Sinh viên thực hiện theo các bước hướng dẫn bên dưới, mở rộng chức năng của ứng dụng "Hello world" để kết nối đến 1 URL và tải dữ liệu về.

***Hướng dẫn:* Mục tiêu tạo HTTP Request đến 1 URL để tải dữ liệu.**

- **<u>Bước 1</u>: Thêm permission cho ứng dụng**

Mặc định Android sẽ không cho ứng dụng tương tác tới các thành phần hệ thống (Camera, Internet, Disk…). Nếu muốn sử dụng các thành phần này, ứng dụng cần yêu cầu một số permission để Android cấp quyền tương ứng. Để có thể gửi *HTTP Request* từ ứng dụng, cần khai báo việc sử dụng 2 permission bên dưới cho ứng dụng trong *AndroidManifest.xml*:

```
<manifest …>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    …
</manifest>
```

- **<u>Bước 2</u>: Điều chỉnh file *MainActivity.java* để gửi HTTP Request**
  - Thêm các thư viện sau ở đầu file *MainActivity.java*

```
import androidx.appcompat.app.AppCompatActivity;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.URL;
import java.net.URLConnection;
import android.os.StrictMode.ThreadPolicy.Builder;
import android.os.StrictMode;
```

```
import android.os.Bundle;
import android.util.Log;
```

- Trong hàm **onCreate()** của *MainActivity.java*, thêm dòng sau:

```
StrictMode.setThreadPolicy(new Builder().permitAll().build());
```

StrictMode là class được load mặc định trong Android, chặn các tương tác Disk I/O, Network Access từ UI thread. Dòng code trên nhằm mục đích ghi đè policy để cho phép tương tác ra ngoài mạng.

- Xác định giá trị URL muốn truy cập bằng đoạn code như bên dưới:

```
String url = "https://inseclab.uit.edu.vn/robots.txt";
StringBuilder url_holder = new StringBuilder();
url_holder.append(url);
```

- Tạo một connection:

```
URLConnection conn = new URL(url_holder.toString()).openConnection();
```

- Thiết lập các trường Header cho connection:

```
conn.setRequestProperty("Content-Type", "application/x-www-form-urlencoded");
conn.setRequestProperty("charset", "utf-8");
conn.setUseCaches(false);
```

- Tạo buffer để lấy dữ liệu về:

```
BufferedReader buffer = new BufferedReader(new InputStreamReader(conn.getInputStream()));
```

- Sau đó đọc response trả về:

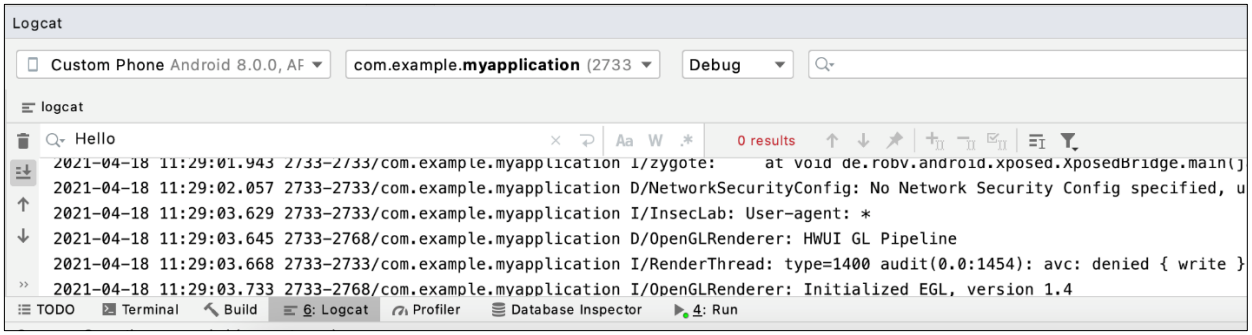
```
String response;
String data_from_stream;
for (response = new String(); true; response += data_from_stream) {
    String stream = buffer.readLine();
    data_from_stream = stream;
    if (stream == null) {
        break;
    }
}
```

- Sau đó log ra để xem

```
Log.i("InsecLab ", response);
```

- **<u>Bước 3:</u> Chạy thử và quan sát kết quả**

(Chụp màn hình log kết quả dữ liệu tải về từ URL cung cấp)



D.2 Viết ứng dụng Android đơn giản

D.2.1 Xây dựng giao diện cho ứng dụng

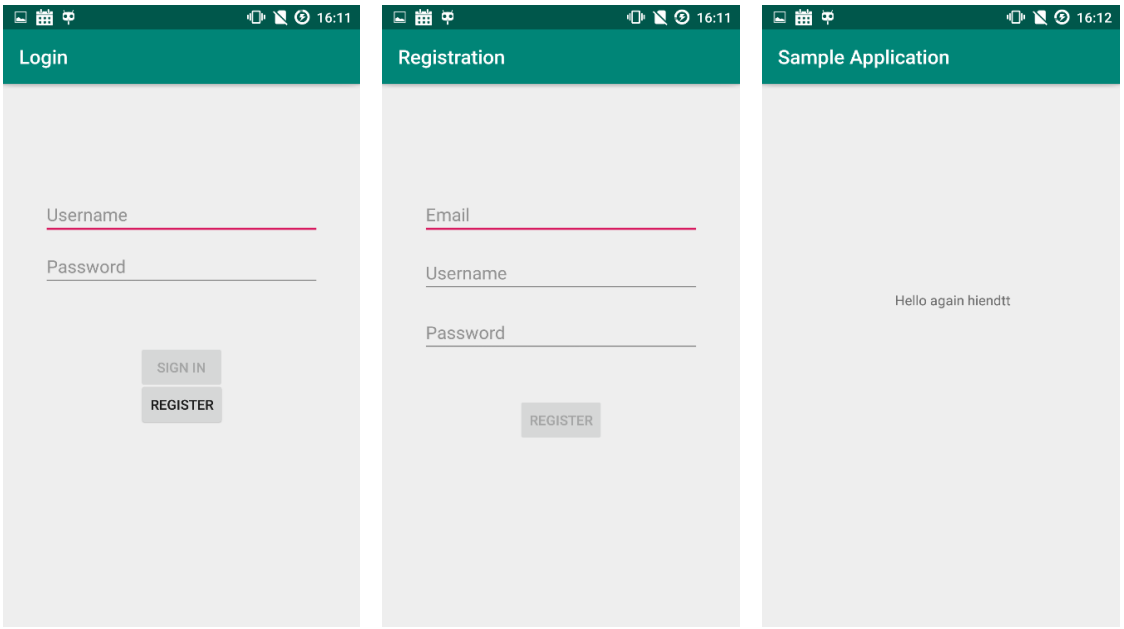
Yêu cầu 2.1. Sinh viên xây dựng ứng dụng Android gồm 3 giao diện chức năng:

1) **Register** - Đăng ký thông tin với ứng dụng (email, username, password).
2) **Login** - Đăng nhập vào ứng dụng (username, password).
3) **Hiển thị thông tin người dùng** (một lời chào có tên người dùng).

Yêu cầu của ứng dụng:

- **Login Activity** là **main activity** (activity khởi chạy ứng dụng), cho phép người dùng nhập thông tin (username và password) đăng nhập hoặc có thể nhấp chọn nút **Đăng ký** nếu chưa có tài khoản.
- **Register Activity** cho phép nhập 3 thông tin như trên để lưu lại trên ứng dụng.
- **Display Activity** hiển thị lời chào cho người dùng khi đã đăng nhập thành công.

Giao diện tham khảo:



Sinh viên thực hiện lại các bước ở **Phần D.1** để tạo 1 project cho ứng dụng mới.

Các file giao diện của các activity nằm ở thư mục **res/layout**. Android Studio cho phép chỉnh sửa giao diện dưới dạng code hoặc kéo/thả các nút chức năng.

D.2.2 Gọi 1 activity từ 1 activity khác

Yêu cầu 2.2. Sinh viên chỉnh sửa các file Java xử lý của các activity và hiện thực các luồng gọi sau:

- Từ Login Activity → Register Activity nếu người dùng click chọn Đăng ký.
- Từ Login Activity → Display Activity nếu người dùng đăng nhập thành công.

Sử dụng kiến thức về Intent và Intent filter (**Phần C.2**) để hiện thực luồng gọi.

Đoạn code minh hoạ việc gọi Activity:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_main);
   Intent activity_to_call=new Intent();
   // call activity dựa trên intent-filter
   activity_to_call.setAction(android.content.Intent.ACTION_VIEW);
   // call activity dựa trên class name
   activity_to_call.setClassname("com.example.myapplication","com.example.myapplication.myActivity")
   // truyền data cho activity khác?
   activity_to_call.putExtra(<data key>, <data value>);
   startActivity(read_contact);
}
```

D.2.3 Kết nối CSDL cho chức năng đăng nhập/đăng ký cho ứng dụng

Yêu cầu 2.3. Sinh viên viết mã nguồn Java cho chức năng đăng nhập và đăng ký, sử dụng tập tin **SQLiteConnector** được cung cấp để thực hiện kết nối đến cơ sở dữ liệu SQLite với các yêu cầu bên dưới.

Sinh viên trình bày các bước đã thực hiện và chụp màn hình các kết quả đã thực hiện. Giải thích những gì quan sát được.

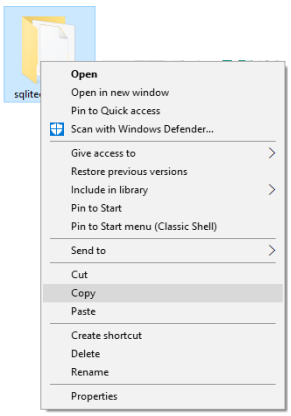
2.1a. Thêm thông tin gồm email, username và password vào cơ sở dữ liệu khi người dùng sử dụng chức năng Đăng ký.

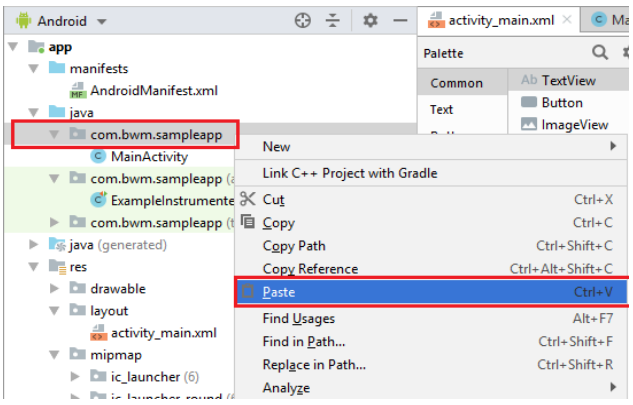
Lưu ý: điều chỉnh mã nguồn để lưu password dưới dạng hash thay vì plaintext.

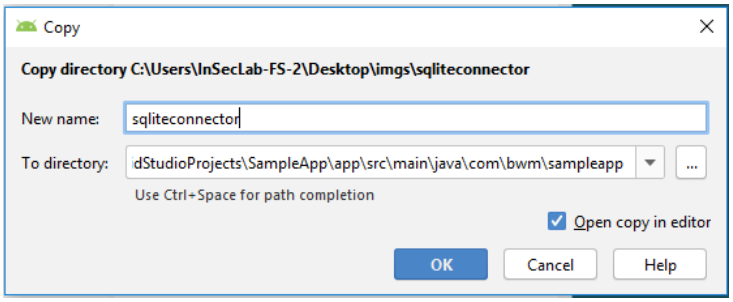
2.1b. Truy vấn thông tin username và password cho chức năng đăng nhập.

- ***Hướng dẫn import file SQLiteConnector vào project đã tạo:***

Sao chép thư mục **SQLiteConnector** được cung cấp và ở mục **java/*<tên package của ứng dụng>***, nhấp chuột phải và chọn **Paste** để thêm vào project ứng dụng.







Lưu ý trong file **SQLiteConnector** có sử dụng một model có tên **User**, sinh viên tự tạo một file **.java** định nghĩa một class **User** gồm các trường tối thiểu id, username, password, email cùng các phương thức get và set giá trị của các trường này.

```
public class User {
    private int id;
    private String name;
    private String email;
    private String password;

    public int getId() { return id; }

    public void setId(int id) { this.id = id; }

    public String getName() { return name; }

    public void setName(String name) { this.name = name; }

    public String getEmail() { return email; }

    public void setEmail(String email) { this.email = email; }

    public String getPassword() { return password; }

    public void setPassword(String password) { this.password = password; }
}
```

Sau đó sinh viên cần điều chỉnh đường dẫn import cho phù hợp trong file SQLiteConnector. Ví dụ:

```
import com.bwm.sampleapplication.data.model.User;
```

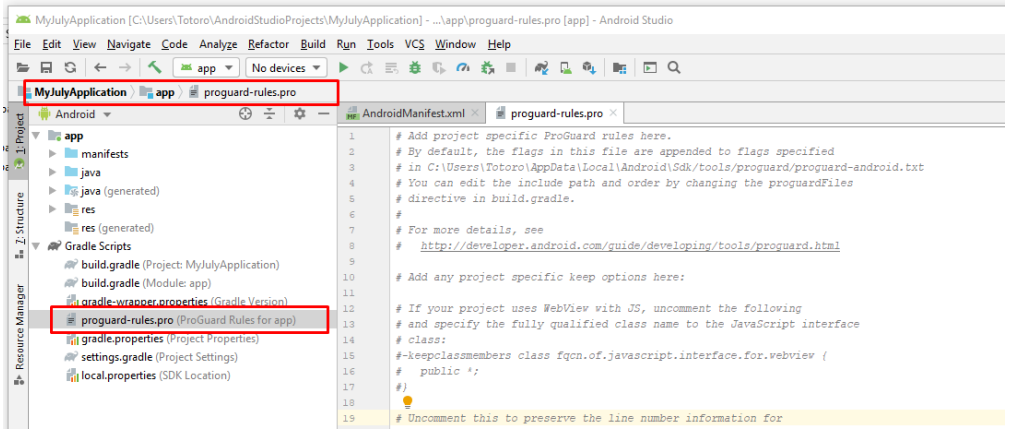
> **Yêu cầu 2.4.** ***(Optional)*** Tạo một cơ sở dữ liệu bên ngoài thiết bị, viết mã nguồn thực hiện kết nối đến CSDL này thay vì sử dụng SQLite (nằm trong thiết bị).

Gợi ý: Sinh viên có thể tận dụng CSQL MySQL và các tập tin xử lý PHP đã thực hiện ở bài thực hành trước và kết nối sử dụng Web REST API. Lưu ý, có cần điều chỉnh quyền hạn gì của ứng dụng hay không?

D.2.4 Tối ưu mã nguồn với ProGuard

> **Yêu cầu 2.5.** Với ứng dụng đã xây dựng, tìm hiểu và sử dụng công cụ ProGuard để tối ưu hóa mã nguồn. Trình bày khác biệt trước và sau khi sử dụng ProGuard?

Gợi ý 1: Mở file **proguard-rules.pro** để cấu hình ProGuard cho ứng dụng:



Gợi ý 2: Sinh viên thử build APK và so sánh khác biệt của 2 file APK trong trường hợp có và không sử dụng ProGuard, như kích thước file

E. YÊU CẦU

- Sinh viên thực hành theo nhóm đã đăng ký.
- Nộp báo cáo cho cả 2 phần D.1 và D.2, trong đó phần D.2 cần nộp kèm mã nguồn (**Code**), báo cáo có kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).

Báo cáo:

- File **.PDF**, trong file báo cáo yêu cầu **ghi rõ** nhóm sinh viên thực hiện.
- Đặt tên theo định dạng: [Mã lớp]-Lab4_NhomX-MSSV1-MSSV2.pdf

  Ví dụ: [NT213.L11.ATCL.1]-Lab4-NhomX-1852xxxx-1852yyyy.pdf
- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Bài sao chép, trễ,... sẽ được xử lý tùy mức độ vi phạm.

HẾT