Introduction to Android App Development
Android Activities and Developing Basic Android App

Hui Chen [a]

[a]CUNY Brooklyn College, Brooklyn, NY, USA

February 16, 2023

Outline

Outline

Technical Readiness Preparation

Provide a technical readiness for

- Version Control Systems
- Develop simple Android Apps with Android Activities and Android Intent
- Test Android Apps

Outline

Lecture Module Outline

- An introduction to event-driven programming
- An introduction to Android
- The Android OS
- The IDE and build system
- Basic App Development
    - Understand Android Studio projects
    - Create basic Android app

Note: this is not an Android class!

Outline

Android Basics

Recall what we discussed,

- What is Android – OS, libraries, utility programs
- What goes into and Android App – Java/Kotlin + XML definitions + some other stuff
- Create an Android Studio project – (gradle; manifest, resources, src)

What to do today?

Concept of event-driven programming (Android apps are event-driven ...)

Learn the following by creating a basic app

- Putting together a basic Android App
  - Activities
  - Widgets
- Getting a closer look at SDK
- Creating an AVD and running app in the AVD

Outline

Event-Driven Programming

The platform puts events in an event queue and runs an event loop (in pseudo code)

```
do {
    e = getNextEvent()
    processEvent(e)
} while (e != EXIT_EVENT)


processEvent(e) {
  for handler in e.handler_list {
    handler.invoke()
  }
}

```

- This event loop often implemented by the platform.
- Users write event handler routines, register them with the platform, and use the handler routines to process events

Event-Driven versus Algorithm-Driven

Application-driven or algorithm-driven programs

- A program expects inputs in a pre-determined order and timing

Event-driven programming – a type of reactive programming

- Program waits for input events when it loads
- The programs runs particular code to response to an event
- The overall flow of the execution is determined by the events that occur
- The overall flow of what code is executed is determined by events in non-deterministic order and timing

GUI Event-Driven Programming

GUI programming are typically event-driven – applies to Android

- GUI Event – An object that represents a user's interaction with a GUI components (e.g., a button, a menu item)
- Event Listener – An object that waits for events and responds to them.
- Event Handler – An object that calls by the Event Listener to handle an event as a part of the response

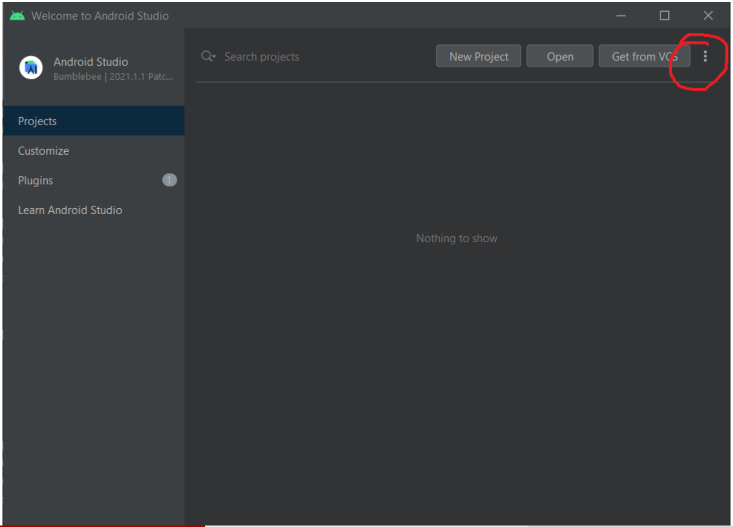
GUI Event Handling

- Programmer attaches a listener to a component for an event (e.g., a button, a menu)
- Platform notifies the listener when the event occur (e.g., a button click)
- The listener calls the Event Handler's methods as a part of reponse

Outline

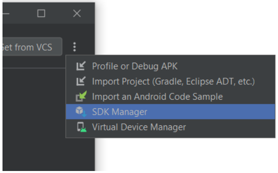
Preparing Development Environment

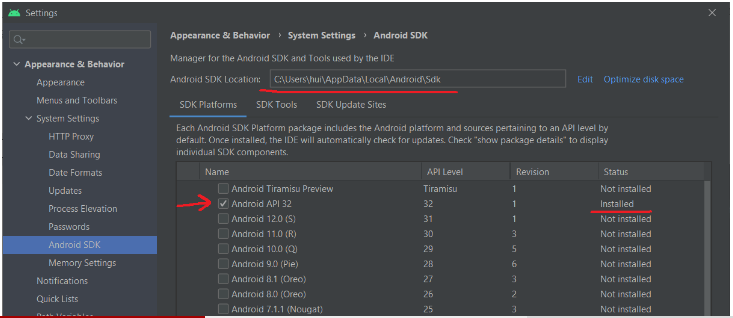
Take a closer look at Android Studio Start Screen,



Android SDK

Ensure that the desired release/version of Android SDK is installed


Get from VCS
Profile or Debug APK
Import Project (Gradle, Eclipse ADT, etc.)
Import an Android Code Sample
SDK Manager
Virtual Device Manager



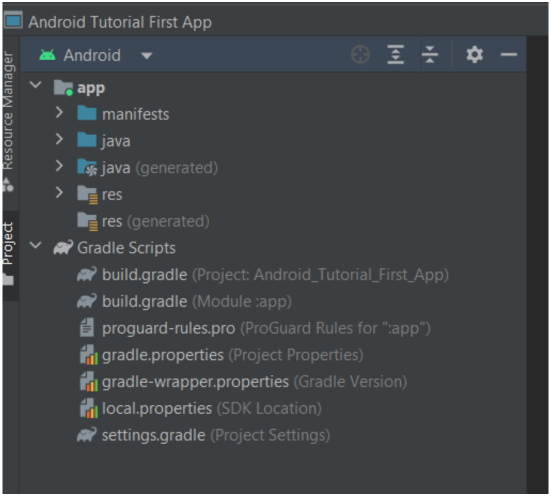
Manager for the Android SDK and Tools used by the IDE

Android SDK Location: C:\Users\hui\AppData\Local\Android\Sdk

Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, the IDE will automatically check for updates. Check "show package details" to display individual SDK components.

Name	API Level	Revision	Status
☐ Android Tiramisu Preview	Tiramisu	1	Not installed
✓ Android API 32	32	1	Installed
☐ Android 12.0 (S)	31	1	Not installed
☐ Android 11.0 (R)	30	3	Not installed
☐ Android 10.0 (Q)	29	5	Not installed
☐ Android 9.0 (Pie)	28	6	Not installed
☐ Android 8.1 (Oreo)	27	3	Not installed
☐ Android 8.0 (Oreo)	26	2	Not installed
☐ Android 7.1.1 (Nougat)	25	3	Not installed

Android Studio Project Layout


Android Tutorial First App

```
app
    manifests
    java
    java (generated)
    res
    res (generated)
Gradle Scripts
    build.gradle (Project: Android_Tutorial_First_App)
    build.gradle (Module :app)
    proguard-rules.pro (ProGuard Rules for ":app")
    gradle.properties (Project Properties)
    gradle-wrapper.properties (Gradle Version)
    local.properties (SDK Location)
    settings.gradle (Project Settings)
```

build.gradle Files

Notice there are two types of Gradle settings file

- Project-level/top-level build.gradle file
- Module-level/app-level build.gradle file

Using an Android SDK feature, we often need to edit these files, but don't confuse these two

Example build.gradle Files Setup: Using Navigation Component

Take a look at

https://developer.android.com/jetpack/androidx/releases/navigation

What should go to what build.gradle file?

Example build.gradle Files Setup: Using View Binding

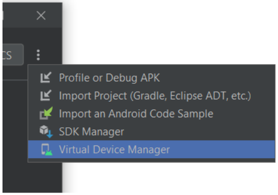
Take a look at

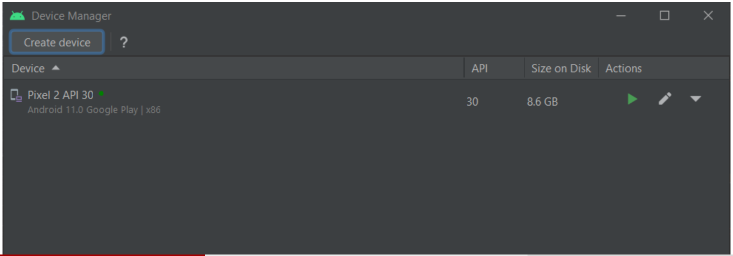
https://developer.android.com/topic/libraries/view-binding

What should go to what build.gradle file?

Android AVD

Ensure that the desired Android Virtual Device (AVD) supporting the the release/version of SDK is set up.


Profile or Debug APK
Import Project (Gradle, Eclipse ADT, etc.)
Import an Android Code Sample
SDK Manager
Virtual Device Manager



Device	API	Size on Disk	Actions
Pixel 2 API 30 Android 11.0 Google Play | x86	30	8.6 GB	

Tips ...

- Don't shut off or restart AVD unless you have to – AVD runs slowly.
- Android Studio, Android SDK, and AVDs are in different directors, it can useful to know where they are, such as, using `adb` to list devices.

Outline

Android Apps Lifecycle

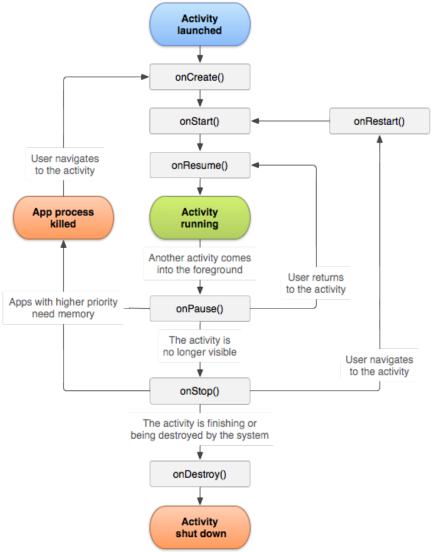
- Run in a separate OS process
- Apps don't typically close
- Can be killed by the Android OS if resources are needed
- Applications that aren't in use are killed
- Should design your Android apps with this knowledge in mind

Activities and Intents

Creating basic apps using Activities and Intents

- Activity roughly corresponds to a screen in an app
  - A typical app has many of these
  - Reusable screen parts are encapsulated in a Fragment
- Intents allow activities to communicate with each other
  - Including passing data to one another

Android Activity Lifecycle


Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

Creating Activity Class

To develop a basic Android apps, we begin with creating our own `Activity` class by extending an Android Activity class.

```
public class MainActivity extends Activity {
  // ...
}
```

or

```
public class MainActivity extends AppCompatActivity {
  // ...
}
```

and then implementing some of the lifecycle methods, e.g.,

```
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    //add some initialization code here
    // ...
  }
```

Example `onCreate()` Method

Assuming we do not use View Binding

Example `onCreate()` Method

```
@Override // what is this?
protected void onCreate(Bundle savedInstanceState)
  // 1. Reinitialize the Activity with the saved data if
    available
  super.onCreate(savedInstanceState);
  // 2. inflate the activity's UI,
  setContentView(R.layout.activity_main);

  // 3. Getting UI object
  mDateTimeTextView = findViewById(R.id.dateTimeTextView);
  final Button addTaskBtn = findViewById(R.id.addTaskBtn);
  final ListView listview = findViewById(R.id.taskListview);
  mList = new ArrayList<String>();

  // 4. Setting up listener for clicking on ListView item
  listview.setOnItemClickListener(new AdapterView.
    OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View
    view, int i, long l) {
      // 5. Not implemented for now
    }})
```

Example onCreate() Method

How to revise it if we do want to use View Binding

Elements in `onCreate()` Example

- what is `savedInstanceState`?
- what does `findViewByID` do?
- What is this `new AdapterView.OnItemClickListener() ...`?
- What to do with the UI widgets obtained by calling `findViewByID`?
- What are `R.id....`?

Connecting UI and Activity

MainActivity.java → activity_main.xml

```
<!-- ....-->
<Button
  android:id = "@+id/addTaskBtn",
  android:layout_height="wrap_content",
  android:layout_width="wrap_content",
  android:layout_below="@+id/dateTimeTextView",
  android:layout_centerHorizontal="true",
  android:padding="20dp",
  android:text="@string/add_task",
  android:onClick="addTaskClicked" />
<ListView
  android:id="@+id/taskListview",
  android:layout_width="wrap_content",
  android:layout_height="wrap_content",
  android:layout_below="@+id/addTaskBtn"/>
<!-- ....-->
```

Let's play with several UI widgets and related methods

- Layouts of UI widgets, e.g., `ConstraintLayout`
- UI controls, e.g., `Button`, `EditText`, `TextView`
- UI widget callback methods, e.g., `onClick`
- Activity methods, e.g., `findViewById`

Questions?

- An introduction to Android
- The Android OS
- The IDE and build system
- Basic App Development
  - Understand Android Studio projects
  - Create basic Android app
  - **Model-View-Controller**
  - **Event-driven Programming**