LE/EECS 4443: Mobile User Interfaces (LAB)

Week 2: Architecture, Components, Prerequisites

S.Toyonaga[1]

[1]Lassonde School of Engineering
York University

January 19, 2024

Table of Contents

Motivation & Problem Statement

By the end of this tutorial, you will be able to...

1. Apply the Model-View-Controller (MVC) software design pattern in your Android Studio Projects
2. Understand the fundamental Android Components
3. Understand how Interactive Components Communicate with Views

LASSONDE SCHOOL OF ENGINEERING

Android Architecture

- Android follows a version of the layered architecture (Model-View-Controller)
  1. **Model:** Holds and manages the application's data (Application & Business Logic)
  2. **View:** If you can see it on screen, then it is a view.
  3. **Controller:** Facilitates the interaction and logic between a View and Model. Responds to event listeners and manages the flow of data between the model and view.

Android Architecture

Example

Billy creates an Android Application that simulates a scantron quiz about himself. Given the following files, identify the model, view, and controller.

1. Question.java
2. MainActivity.java
3. activity_main.xml

Android Components

This is **not** an exhaustive list. However, these are pretty imortant to talk about at the moment.

1. **AndroidManifest.xml**: An XML file containing metadata that describes your application to the Android OS. Every activity in an application must be declared in the manifest so that the OS can access it.
2. **XML Layout (Views)**
3. **Build Script:** Used to tell the compiler how to assemble the application (compileSdk, minSdk, targetSkd)

Interactive Components

Some example components that users may interact with include...

1. Buttons
2. EditText
3. Slider
4. Checkbox
5. RadioButton

Note: You should use components *judiciously*. Design with clear intention and care!

Facilitating Interactions to Controller

Remark

An event listener is an interface in the View class that contains a single callback method. These methods will be called by the Android framework when the View to which the listener has been registered is triggered by user interaction with the item in the UI.

Source:
https://developer.android.com/develop/ui/views/touch-and-input/input-events

Facilitation: Buttons

1. **Option 1:** Anonymous Inner Classes
   - **Example:** Reminders Demo
2. **Option 2:** Implement Listener Class & Override Methods
   - **Example:** Styles Demo

Download Here: https://github.com/stoyonaga/EECS4443_W24_Assets/tree/main/TA%20Demos

Environment Configuration

1. Select *New Project*
2. Select *Empty Views Activity*
3. Configure your application settings and finish the setup. Wait for the Gradle to complete building the model.
4. Navigate to *app/res/layout/activity_main.xml*. Build your application using the code or GUI editor, setting attributes (i.e., identifiers) as necessary.
5. Navigate back to the Activity class:
   - Declare your attributes below the class declaration
   - In *onCreate* and under *setContentView(R · · ·)*, instantiate the attributes, initialize event listeners, and continue building your application.
6. Test, Test, Debug, Test (JUnit / Espresso)

Fundamental Implementation of Your Activity Class

```
package com.example.democomponents;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
// Attributes go here
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // Controller and View Configuration Goes Here
    }
}
```