Recording in Progress

This class is being recorded

Please turn off your video and/or video if you do not wish to be recorded
CMSC436: Programming Handheld Systems
Android Development Environment
The Android Platform

A multi-layered software stack for building and running mobile applications
The Android Development Environment

Starts with knowledge of the Android platform
Your workbench for writing Android applications
See:

https://developer.android.com/studio/intro/
Today’s Topics

- Downloading Android SDK
- Using the Android Studio IDE
- Using the Android emulator
- Debugging Android applications
- Other tools
Prerequisites

Supported Operating Systems:

- Microsoft Windows 8/10 (64-bit)
- Mac OS X 10.14 (Mojave) or higher
- Any 64-bit Linux that supports Gnome, KDE, Unity DE
General Prerequisites

8GB RAM min

8GB memory for Android SDK, emulator system images, and caches

1280 x 800 min screen resolution
Getting Started

Download & install Android Studio

See: https://developer.android.com/studio/
Android Studio

Android platform

Android Studio IDE

Key development tools

System image for emulator
The Android Emulator

Runs virtual devices

We’ll generally be using a Pixel 5 with API level 33
The Android Emulator

Pros

- Doesn’t require an actual phone
- Hardware is reconfigurable
- Changes are non-destructive
The Android Emulator

Cons

Slower than an actual device

Some features unavailable
  e.g., no support for NFC, SD card insert/eject, etc.

Performance / user experience can be misleading
Advanced Features

Can emulate many different device/user characteristics, such as:

- Network speed/latencies
- Battery power
- Location coordinates
- Sensor readings
Advanced Features

Ex: Change network speeds
Advanced Features

Ex: Emulate incoming phone calls & SMS messages
Advanced Features

Many more options

See:

Debugger

Tool for examining the internal state of a running application
The answer to life, the universe and everything is:

42
Development Tools

Android Studio provides numerous tools for monitoring application behaviors
Example Tools

Device File Explorer
Logcat
Layout Inspector
CPU Profiler
Device File Explorer

View, copy, and delete files on your device

Often used to examine and verify file creation and transfer
package course.examples.theanswer

class TheAnswer : AppCompatActivity {

    override fun onCreat() {
        private val answer = 42
        private val tag = "TheAnswer"
    }

    override fun onCreate(savedInstanceState: Bundle?) {
        // Required call through to AppCompatActivity.onCreate()
        // Restore any saved instance state
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        answerTextView.text = answer.toString()
    }

    private fun answerTextView() {int {Log.d(tag, "Ans: ${answer}")
        // Incorrect behavior
        return answers.firstOrNull { it == answer }
    }

    return answers.firstOrNull { it == answer }
}
```java
package courseexamples.theshower;

import android.util.Log;
import java.util.ArrayList;

public class Theshower {
    private static class TheshowerActivity {
        private static void onCreate(Bundle savedInstanceState) {
            // Set the title for the activity
            setCustomContentView(R.id.activity_theshower);
        }
    }

    public static void main(String[] args) {
        // Start the activity
        TheshowerActivity.onCreate(null);
    }
}
```
Logcat

Write and review log messages

Apps use Log class to write messages to log

Developer can search and filter log messages
The answer to life, the universe and everything is:

42
The answer to life, the universe and everything is: 42
Layout Inspector

Shows the runtime organization of the user interface
package course.examples.TheAnswer

import android.app.Application

class TheAnswer : Application() {

    companion object {

        private val answers = intArrayOf(42, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, 32768, 65536, 131072, 262144, 524288, 1048576, 2097152, 4194304, 8388608, 16777216, 33554432, 67108864, 134217728, 268435456, 536870912, 1073741824)

    }

    override fun onCreate(savedInstanceState: Bundle?) {

        // Required call through to Activity.onCreate()
        super.onCreate(savedInstanceState)

        // Set up the application's user interface (content view)
        val binding = AnswerViewBinding.inflate(layoutInflater)
        setContentView(binding.root)

        val output = findAnswer()
        val output2 = output.toString() + "... We may never know!"

        // Get a reference to a TextView in the content view
        binding.answerView.text = output

        private fun findAnswer(): Int {

            Log.d(tag, "Entering findAnswer()")

            // Incorrect behavior
            return answers.firstOrNull { it == answer } ?: answer

            Log.d(tag, "Correct behavior")
            return answer
        }
    }
}
package course.examples.theanswer

class TheAnswer : AppCompatActivity {

    companion object {

        private val answers = ArrayList<Int>(10, 8, 106, 1606)
        private const val answer = 42
        private const val TAG = "TheAnswer"

        override fun onCreate(savedInstanceState: Bundle?) {

            setContentView(R.layout.activity_theanswer)

            val textView: TextView = findViewById(R.id.textView)
            textView.text = "The answer to life, the universe and everything is:"

            val answerView: TextView = findViewById(R.id.answerView)
            answerView.text = String.valueOf(answer)
        }
    }
}
Next

Application Fundamentals
Example Applications

HelloWorld

TheAnswer