ANDROID FUNDAMENTALS

Seven Advanced Academy

Debugging Your App

Lesson 7

Contents

- All code has bugs
- Android Studio logging
- Android Studio debugger
- Working with breakpoints
- Changing variables
- Stepping through code

Tasks

- Build the SimpleCalc app.
- Set and view breakpoints in the code for SimpleCalc.
- Step through your code as it runs.
- Examine variables and evaluate expressions.
- Identify and fix problems in the sample app.

All Code Has Bugs

Bugs

- Incorrect or unexpected result, wrong values
- Crashes, exceptions, freezes, memory leaks
- Causes
  - Human Design or Implementation Error > Fix your code
  - Software fault, but in libraries > Work around limitation
  - Hardware fault or limitation -> Make it work with what's available

Origin of the term "bug" (it's not what you think)

Debugging

- Find and fix errors
- Correct unexpected and undesirable behavior

- Unit tests help identify bugs and prevent regression
- User testing helps identify interaction bugs

Android Studio Debugging Tools

Android Studio has tools that help you:

- identify problems
- find where in the source code the problem is created
- so that you can fix it

Add log messages to your code

```
import android.util.Log;

// Use class variable with class name as tag
private static final String TAG =
    MainActivity.class.getSimpleName();

// Show message in Android Monitor, logcat pane
// Log.<log-level>(TAG, "Message");
Log.d(TAG, “Hello World”);
```

Open Android Monitor and logcat

Logcat pane

Android Monitor

com.example.android.whowroteit (andrc
com.example.android.whowroteit (test)
res
Gradle Scripts

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

/**
 * Created by fgreco on 8/8/16.
 */
public class BookLoader extends AsyncTaskLoader<String> {

    private String mQueryString;
    private static final String LOG_TAG = BookLoader.class.getSimpleName();
```

Android Monitor

LGE Nexus 5X 00c930a8acc71a94 [DISCONNECTED] com.example.android.whowroteit (6144) [DEAD]

logcat Monitors Verbose Regex Show only selected application

```
08-11 21:36:20.105 6144-6365/com.example.android.whowroteit I/OpenGLRenderer: Initialized EGL, version 1.4
08-11 21:54:14.535 6144-6144/com.example.android.whowroteit D/WhoWroteIt: button click - searchBooks()
08-11 21:54:14.538 6144-6144/com.example.android.whowroteit D/WhoWroteIt: Before test if networkInfo [moby dick]
08-11 21:54:14.542 6144-6144/com.example.android.whowroteit D/BookLoader: constructor - mQueryString [moby dick]
08-11 21:54:14.549 6144-18414/com.example.android.whowroteit D/loadInBackground(): URI [https://www.googleapis.com/books/v
08-11 21:54:16.493 6144-18414/com.example.android.whowroteit D/BookLoader: {
```

6: Android Monitor Terminal 0: Messages Event Log Gradle Console

Gradle build finished in 18s 701ms (yesterday 9:36 PM) 20:14 LF: UTF-8: Context: <no context>

Inspect Logging Messages

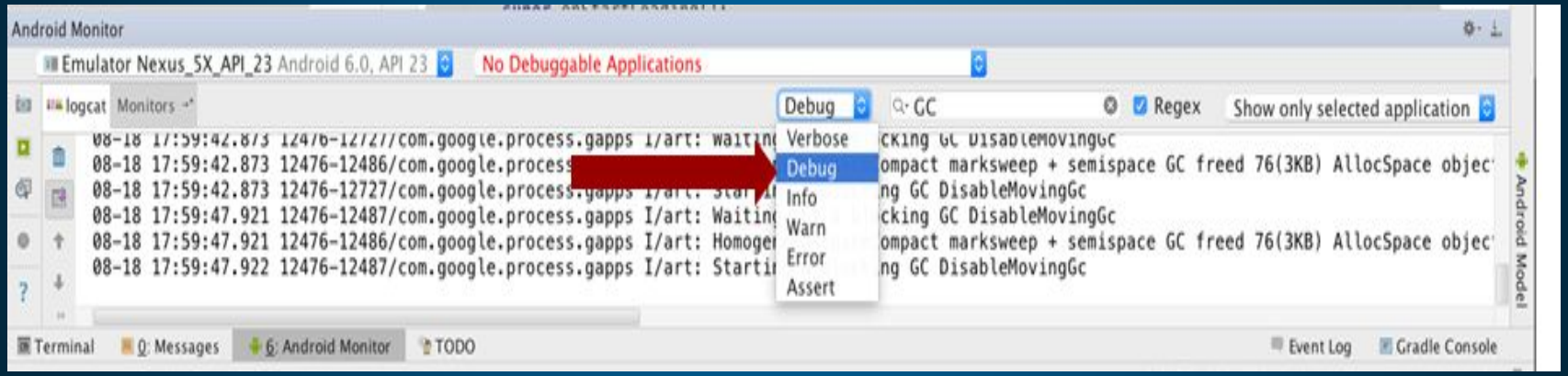
```
Log.d("MainActivity", "Hello World");
```

1

```
09-12 14:28:07.971 4304 /com.example.android.helloworld D/MainActivity: Hello World
```

2

Choose visible Logging Level



Displays logs with levels at this level or higher

Log Levels

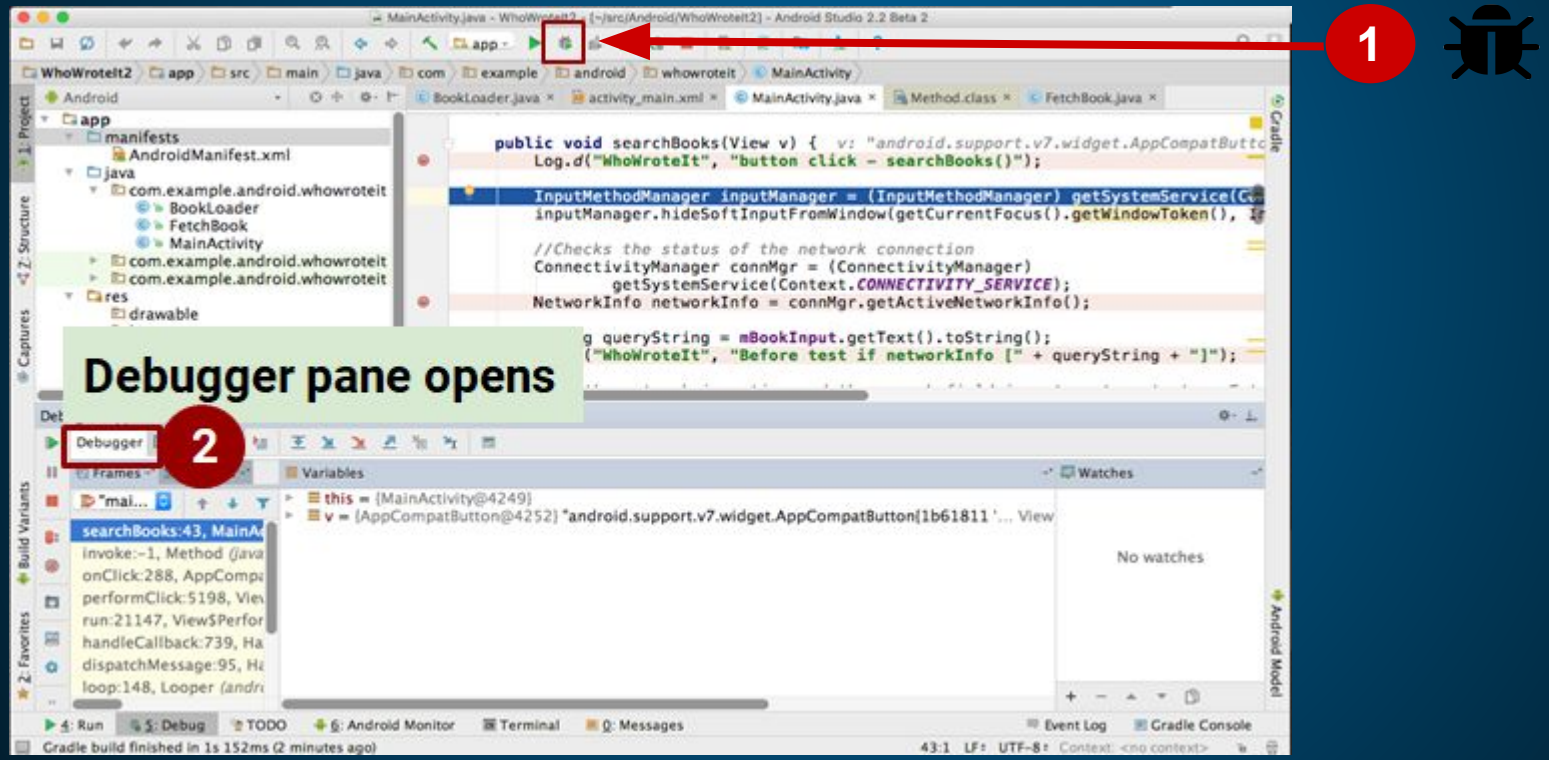
- Verbose - All verbose log statements and comprehensive system
- Debug - All debug logs, variable values, debugging notes
- Info - Status info, such as database connection
- Warning - Unexpected behavior, non-fatal issues
- Error - Serious error conditions, exceptions, crashes only

Debugging with Android Studio

What you can do

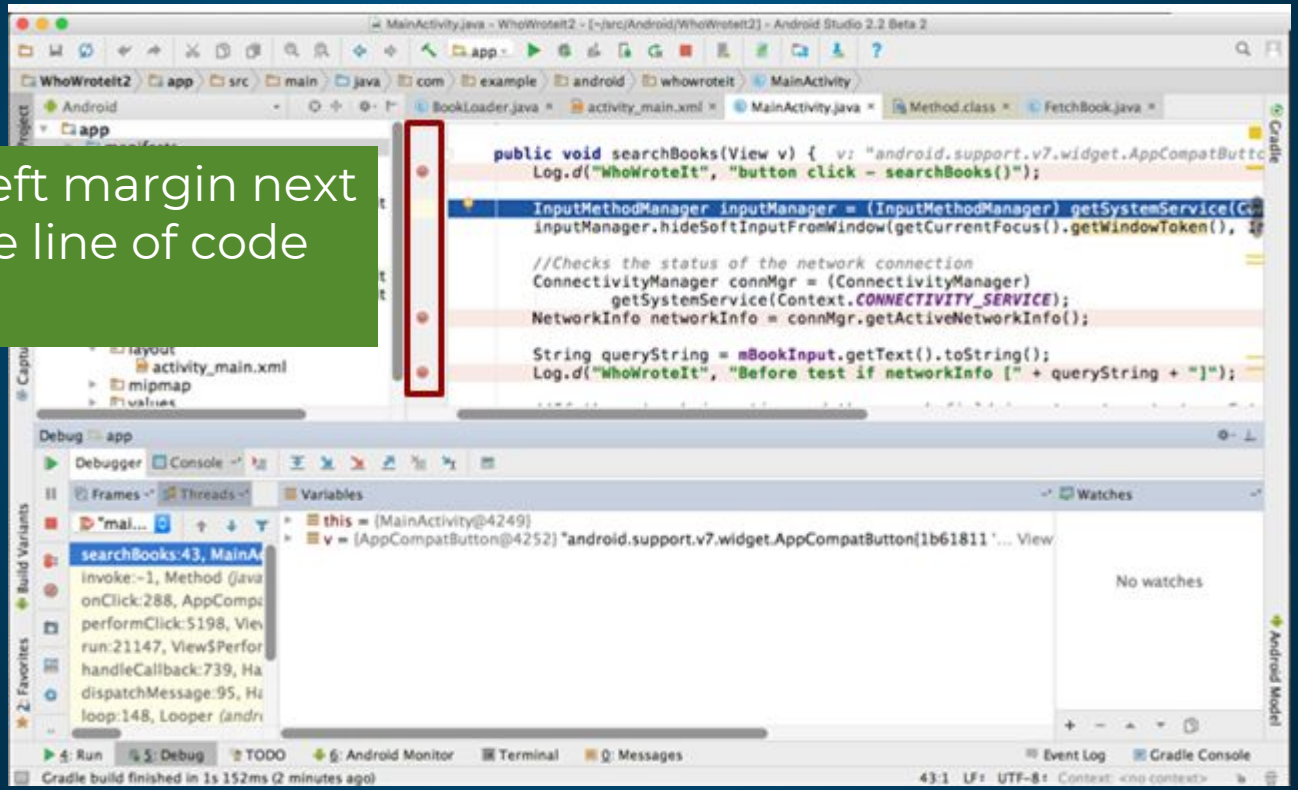
- Run in debug mode with attached debugger
- Set and configure breakpoints
- Halt execution at breakpoints
- Inspect execution stack frames and variable values
- Change variable values
- Step through code line by line
- Pause and resume a running program

Run in Debug Mode

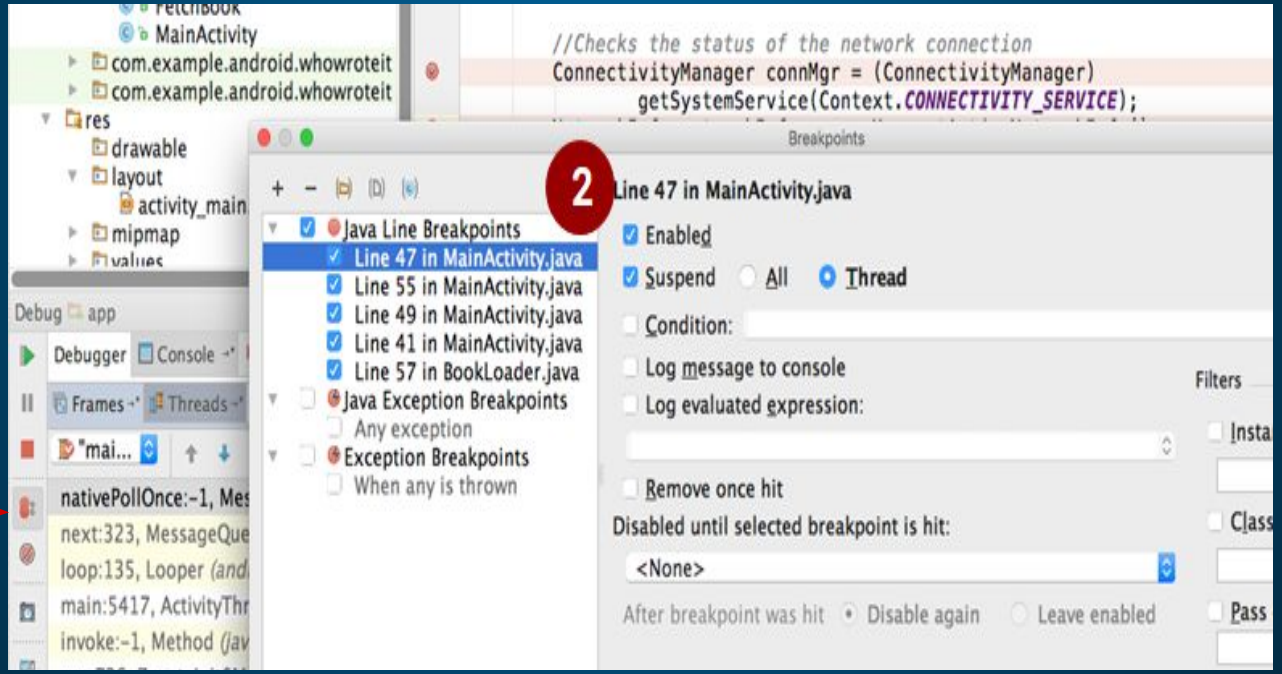


Set Breakpoints

Click in the left margin next to executable line of code

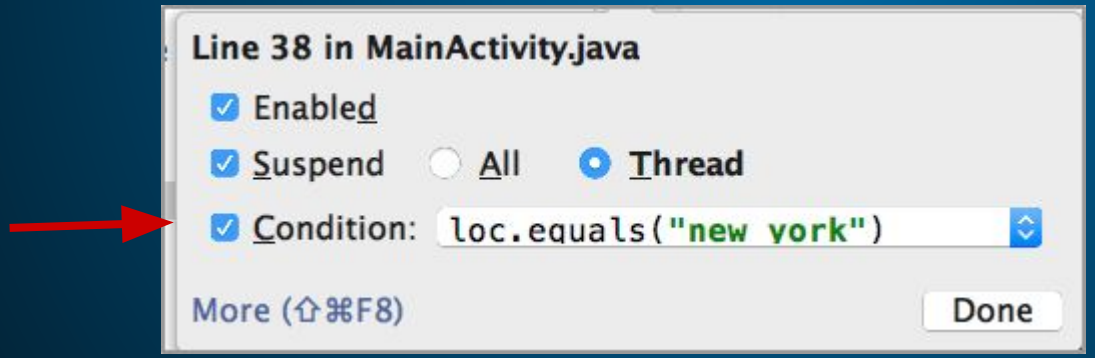


Edit Breakpoint Properties



Make Breakpoints Conditional

- In properties dialog or right -click existing breakpoint
- Any Java expression that returns a boolean
- Code completion helps you write conditions



Run until app stops at breakpoint

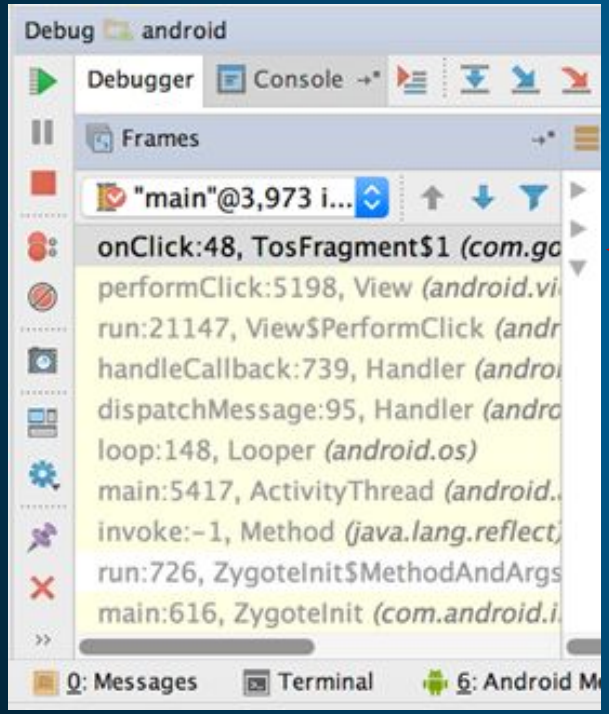
First Breakpoint

Frames

Variables in scope

Watches (C/C++)

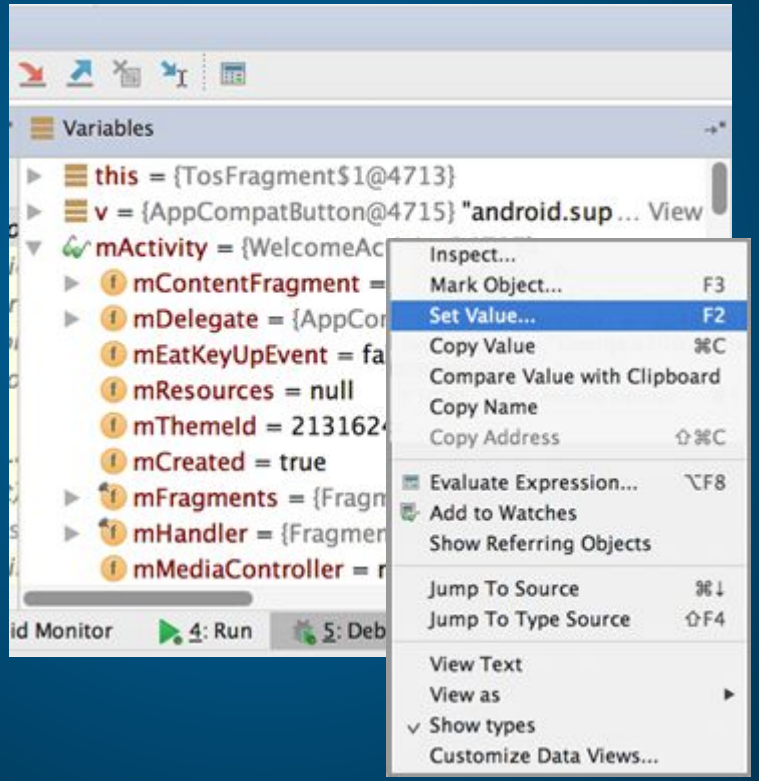
Inspect Frames



Top frame is where execution is halted in your code

Inspect and Edit Variables

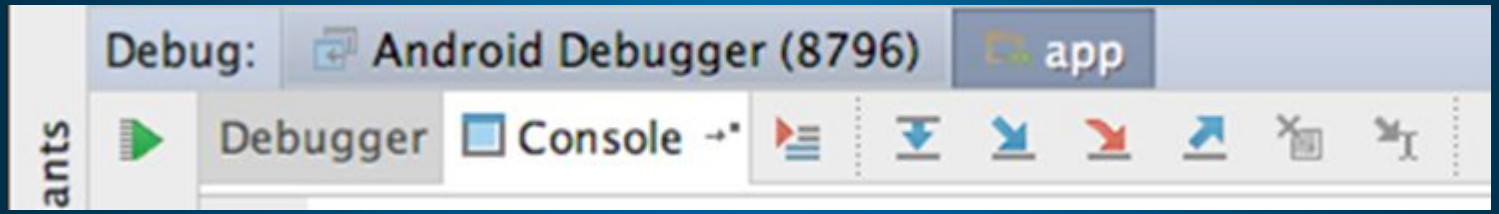
Right-click on variable for menu



Basic Stepping Commands

Step Over	F8	Step to the next line in current file
Step Into	F7	Step to the next executed line
Force Step Into	⇧F7	Step into a method in a class that you wouldn't normally step into, like a standard JDK class
Step Out	⇧F8	Step to first executed line after returning from current method
Run to Cursor	⌥F9	Run to the line where the cursor is in the file

Stepping through code

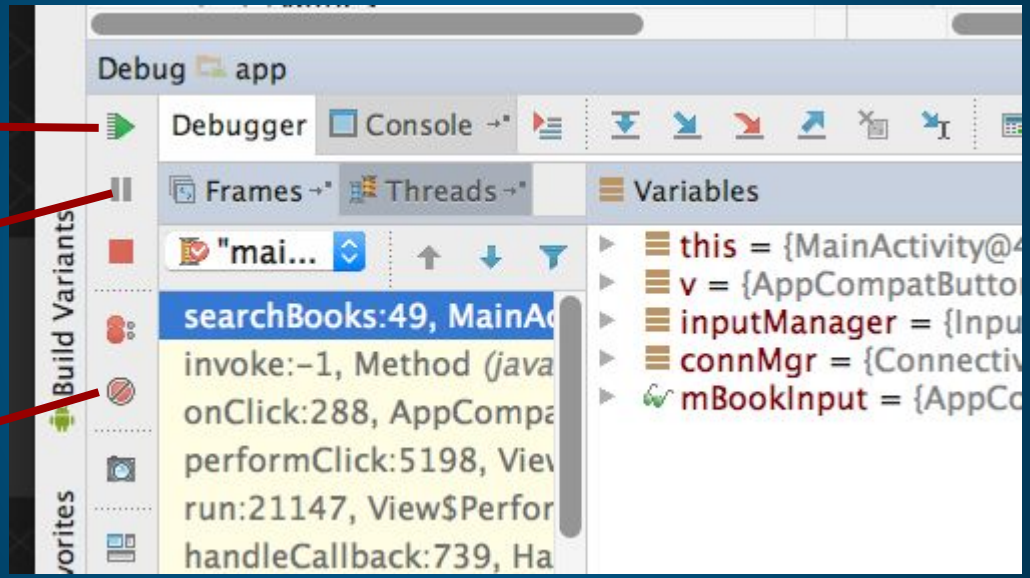


Resume and Pause

Resume

Pause

Mute all breakpoints



Menu:
Run->Pause Program...
Run->Resume Program...

Let’s code

App overview

The SimpleCalc app has two edit texts and four buttons. When you enter two numbers and click a button, the app performs the calculation for that button and displays the result.

Warning: This app contains errors that you will find and fix. If you run the app on a device or emulator you might run into unexpected behavior which may include crashes in the app.

Demo : Using the Debugger

Source code

- SimpleCalc

Learn More

- Debug Your App (Android Studio User Guide)
- Debugging and Testing in Android Studio (video)