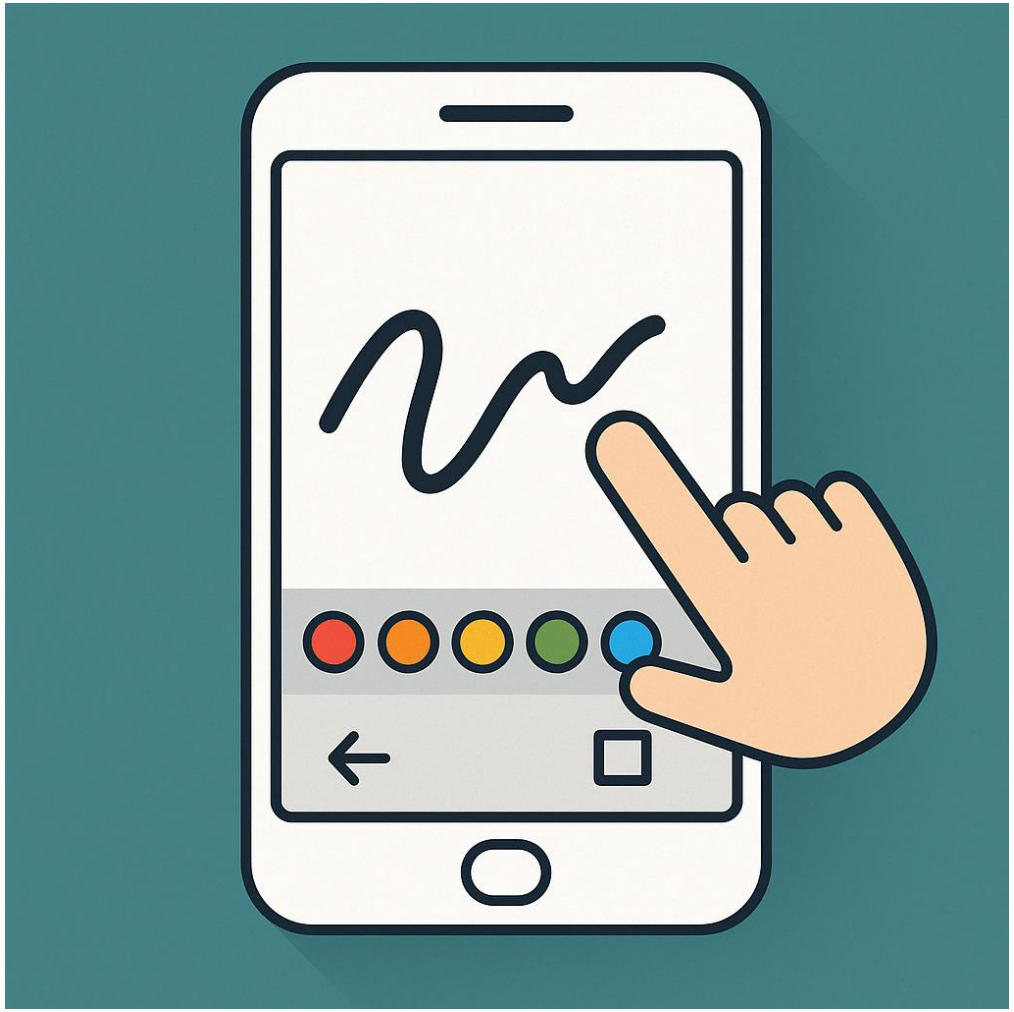
Mobile Applications for Sensing and Control

Sep Makhsous

Week 4

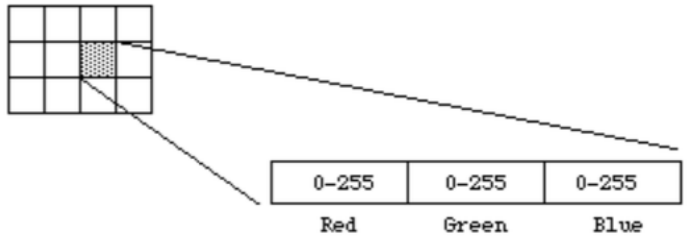
Custom View and User Touch Input



What is a Bitmap

- Bitmaps
  - Regular rectangular mesh of cells called pixels, each pixel containing a color value. They are characterized by only two parameters, the number of pixels and the information content (color depth) per pixel.

24 bit RGB
8 bits allocated to each red, green, and blue component. In each component the value of 0 refers to no contribution of that colour, 255 refers to fully saturated contribution of that colour. Since each component has 256 different states there are a total of 16777216 possible colours.



What is a Canvas

Definition: A Canvas is an Android class used for drawing shapes, text, images, or anything else onto a Bitmap.

How it works:

- Canvas doesn’t display anything on its own.
- It acts as a painter: you use it to draw things onto a Bitmap, which then gets shown on screen via a View's onDraw() method.
- You don’t draw directly onto the screen — instead, you draw onto the bitmap which is rendered on the screen by the system.

What is Paint?

Definition: Paint is a class in Android that holds style and color information about how to draw geometries, text, and bitmaps.

What it controls:

- The color of lines, fills, and text
- The stroke width (how thick your brush is)
- The style (stroke, fill, or both)
- Cap, join style, anti-aliasing, dither (for smoother rendering)

Capturing Finger Input

Goal: Convert finger movement into drawable paths on screen using onTouchEvent().

Three Main Actions:

- ACTION_DOWN → user touches screen (start new path)
- ACTION_MOVE → user drags finger (extend path)
- ACTION_UP → user lifts finger (commit the path)

Custom Views in Android

Goal: Understand how to build custom views — drawing your own content instead of using standard UI widgets.

Why use a Custom View?

- Standard Views like TextView, Button can’t be freely drawn on
- Custom Views give us full control over how and what is rendered on screen

Bitmaps on Android

- Using a BitmapFactory, you can create bitmaps in 3 common ways:
  - from a resource – you can create a bitmap from an image (such as .png or .jpg) located in your drawable folder.

    ```
    val pict = BitmapFactory.decodeResource(resources,R.drawable.bored)
    ```
  - File
  - InputStream (such as an image from the camera)

Supporting Bitmaps

- Bitmaps can very easily exhaust an app's memory budget.
  - For example, the camera on the Pixel phone takes photos of up to 4048x3036 pixels (12 megapixels).
  - If the bitmap configuration used is ARGB_8888, the default for Android 2.3 (API level 9) and higher, loading a single photo into memory takes about 48MB of memory (4048*3036*4 bytes).
  - Loading bitmaps on the UI thread can degrade your app's performance, causing slow responsiveness. It is therefore important to manage threading appropriately when working with bitmaps.

2D Graphics

- To draw our own custom 2D graphics on screen, we'll make a custom View subclass with the drawing code.
- If the app is animated (such as a game), we'll also use a thread to periodically update and redraw the graphics.

Class Exercise

- Let’s open Android Studio and build the app.