Mobile Applications for Sensing and Control

Sep Makhsous

Week 3

Reminder

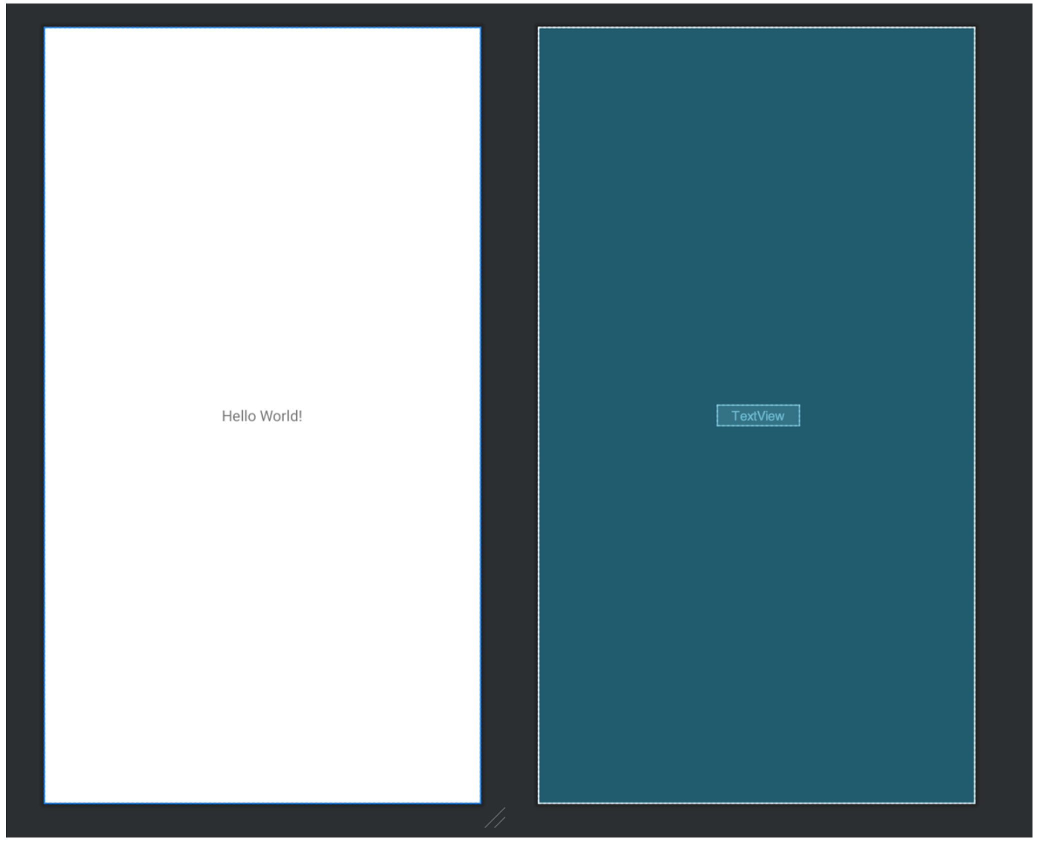
- HW1 – Due April 22nd with Automatic 24hr Extension

What we have covered so far

- First App - A Simple Calculator:
  - The Layout File (activity_main.xml)
    - LinearLayout
    - EditText
    - Button
    - TextView
- Second App - Simple Camera
  - The Layout File (activity_main.xml)
    - ImageView
    - Basic Permissions

The Layout file (activity_main.xml)

- Defines UI structure and layout
- Sets properties for UI elements (size, color, text, etc.)
- Ensures responsive design across different devices
- Links UI elements with code for event handling



LinearLayout

- A layout manager that aligns all its child views in a single direction, either vertically or horizontally.
- In the first app, LinearLayout uses android:orientation="vertical" to stack the EditText, Button, and TextView elements vertically.
- It supports attributes like android:layout_width, android:layout_height, and android:layout_margin to set size and spacing.

EditText

- A user interface element that allows users to enter text.
- In the app, two EditText fields are used for inputting numbers, identifiable by IDs @+id/et_a and @+id/et_b.
- Properties like android:hint provide users with guidance on what to input.

Button

- A UI element that users can click or tap to perform an action.
- The app features buttons for adding, subtracting, multiplying, and dividing numbers, with respective IDs like @+id/bt_add.
- They trigger event handling in Kotlin code, where their actions are defined.

TextView

- Used to display text to the user.
- In the app, a TextView with the ID @+id/result_tv is used to show the calculation result.
- It can be styled with attributes such as android:textSize and android:gravity to enhance readability and appearance.

ImageView

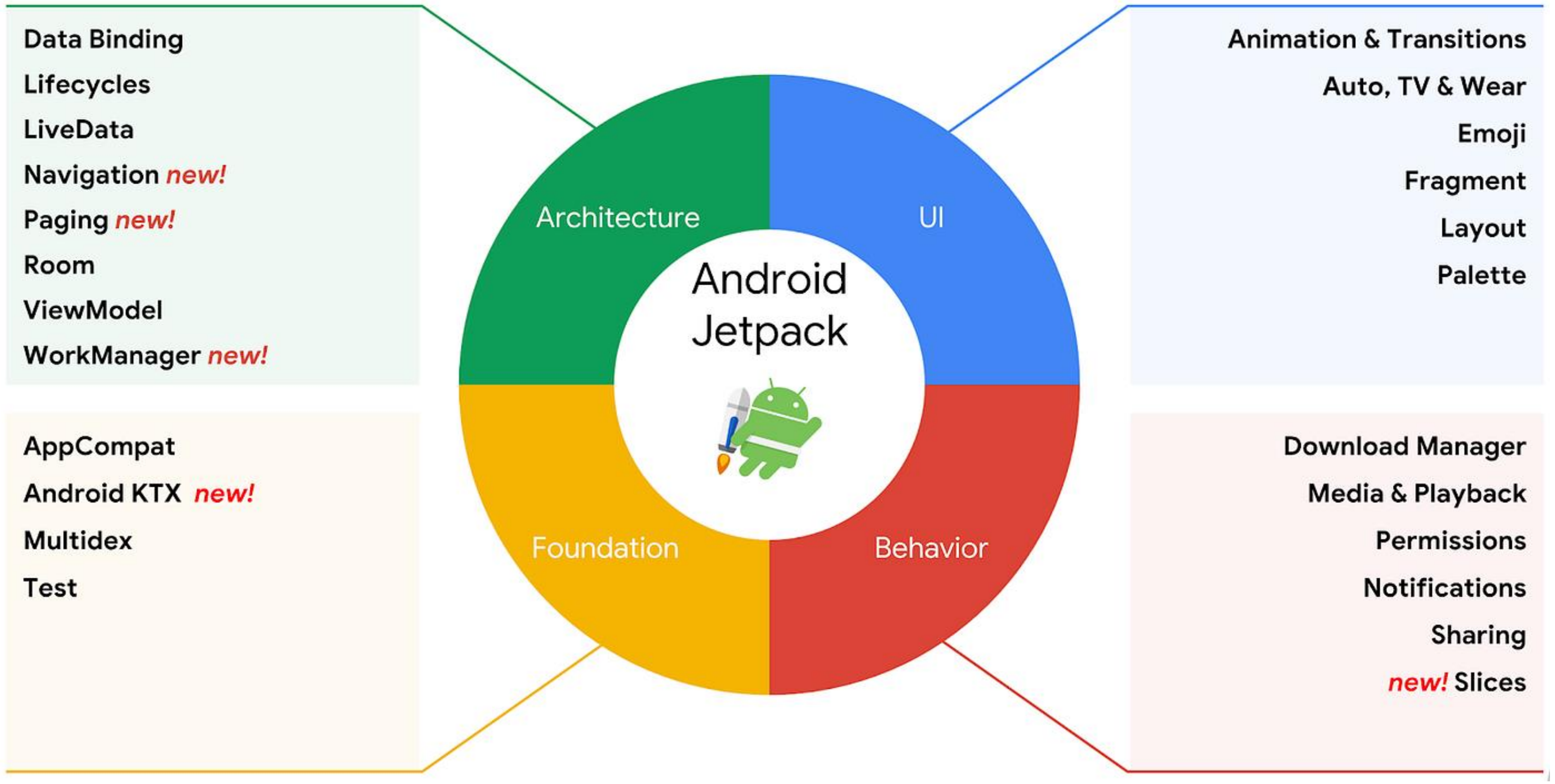
- A UI element in Android used to display images.
- In the second app, an ImageView with the ID @+id/selected_image_view is used to show the image captured from the camera.
- Initially hidden (android:visibility="gone"), it becomes visible after capturing an image.

Permission

- Mechanism to ensure the app has the necessary rights to access certain device functionalities.
- The second app requires camera and storage permissions to capture and save images.
- Permissions like android.permission.CAMERA
- The app checks and requests permissions at runtime, complying with Android's security framework.

Where can you find other features like these?

Kotlin Jetpack Library



What is Jetpack

- A suite of libraries, tools, and guidance to help developers write high-quality apps.
- Designed to simplify common development tasks.
- Ensures backward compatibility and forward functionality.

Core Components of Android Jetpack

- UI Components: Enhance the user interface.
- Architecture Components: Manage UI components, data, and application logic.
- Behavior Components: Handle app behaviors like notifications and permissions.
- Foundation Components: Provide backward compatibility and basic functionality.

UI Components in Android Jetpack

- View: Basic building block of UI.
- RecyclerView: Display large sets of data efficiently.
- ConstraintLayout: Complex and flexible layouts.
- Navigation: Manage app navigation.

Architecture Components in Android Jetpack

- ViewModel: Manage UI-related data in a lifecycle-conscious way.
- LiveData: Data objects that notify views when the underlying database changes.
- Room: Abstraction layer over SQLite.

Behavior Components in Android Jetpack

- Permissions: Simplify permission handling.
- Notifications: Enhance user engagement.
- CameraX: Camera functionality made simpler.

Foundation Components in Android Jetpack

- AppCompat: Ensure backward compatibility.
- Android KTX: Write more concise, idiomatic Kotlin code.
- Multidex: Support apps with multiple DEX files.

Where do I find Jetpack Samples and Notes

- Go to:
  - https://developer.android.com/jetpack/samples

Want to stay up to date with Android Dev best practices?

Install: https://github.com/android/nowinandroid

What about MainActivity.kt

Kotlin Basics in Android

- Class and Inheritance: Kotlin classes and inheritance using **AppCompatActivity**.
- Lifecycle Methods: **onCreate** and its role in app initialization.
- View Binding: Accessing UI elements like **Button**, **EditText**, **TextView**, and **ImageView** using **findViewById**.
- Late-Initialized Properties: Using **lateinit** to defer initialization of UI components.
- Implementing **View.OnClickListener**: Managing click events within a single onClick method.

Kotlin Basics in Android

- Class and Inheritance in Kotlin:

- Lifecycle Methods in Android:

- Understanding the onCreate method is fundamental to working with Android activities, as it sets the stage for how the app behaves when the user navigates through it.

Kotlin Basics in Android

- View Binding with findViewById
    - Connecting UI elements like Button, EditText, and TextView to Kotlin code.

- Late-Initialized Properties
    - Usage of lateinit to defer initialization of UI components until onCreate.

Kotlin Basics in Android

- Implementing Click Listeners
    - Setting up and handling user interactions with UI elements.
- View.OnClickListener Interface
    - Managing click events within the onClick method for multiple buttons.