Mobile App Development

Android Native App Development

Why Native?

- Performance
  - Native apps are compiled to machine code, which can run faster than interpreted code used in cross-platform frameworks.
- Access to platform-specific features
  - Better integration with the operating system
  - Access to native APIs and libraries

Languages

- Java
  - Official language for Android development until 2017
- Kotlin
  - Official language for Android development since 2017
  - Interoperable with Java
  - More concise and modern syntax

Android Studio

- Official IDE for Android development
- Provides tools for designing, coding, and debugging Android apps
- Supports both Java and Kotlin

Jetpack Compose vs XML Layouts

- Jetpack Compose
  - Modern toolkit for building native UI
  - Uses a declarative approach to UI development
  - More concise and easier to read than XML layouts
- XML Layouts
  - Traditional way to design Android UI
  - Uses an XML-based language to define UI components and their properties
  - More verbose and less intuitive than Jetpack Compose

Kotlin Primer

```kotlin
fun main() {
    println("Hello World")

    val lambda_fn_1: (Int) -> Unit = { a: Int -> println(a) }
    lambda_fn_1(5)

    val lambda_fn_2: () -> Unit = { println("Lambda") }
    lambda_fn_2()

    MyFun(a = 5, content = { println("Lambda") })
}

fun MyFun(a: Int, content: () -> Unit = {}) {
    println(a)
    content()
}
```

Counter App

- Original Code
- Hello World (Change)
- Button (Change)
- Structure (Change)
- State (Change)

Final Thought

- Feel like a mixture between Flutter and React.
- Need to experiment with sensor and camera to see how it performs.