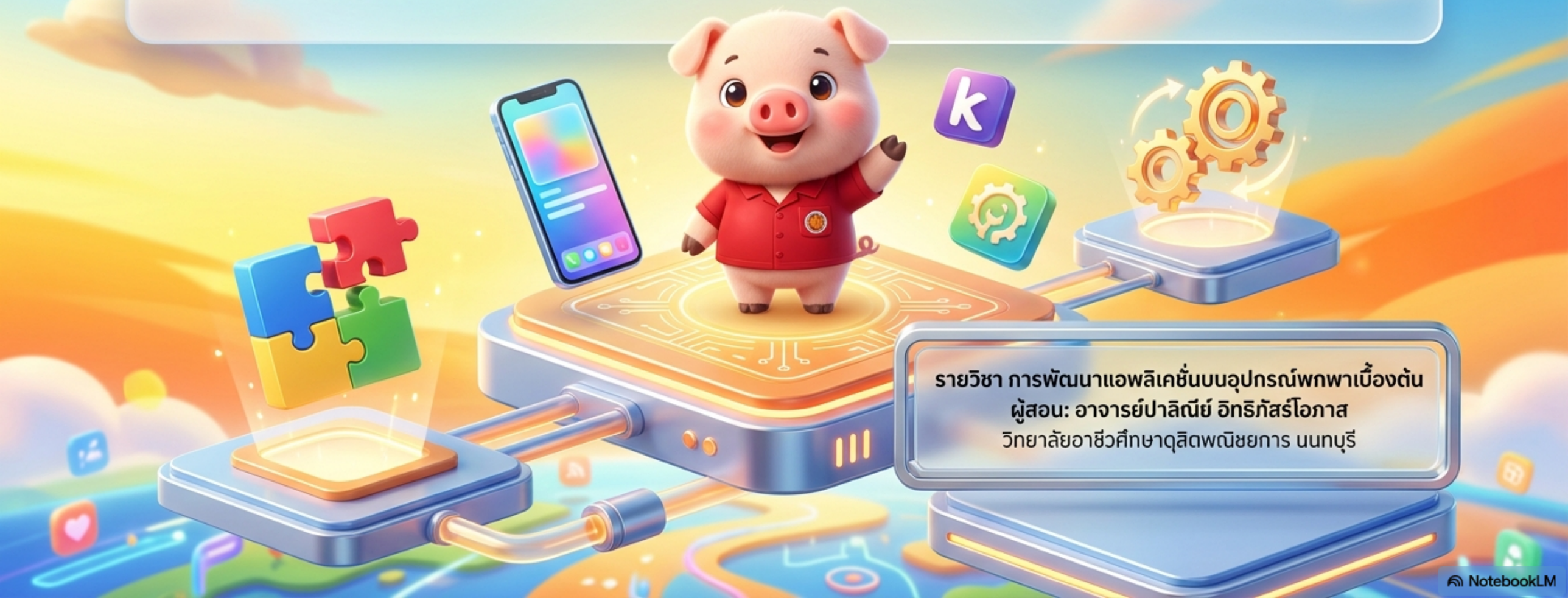
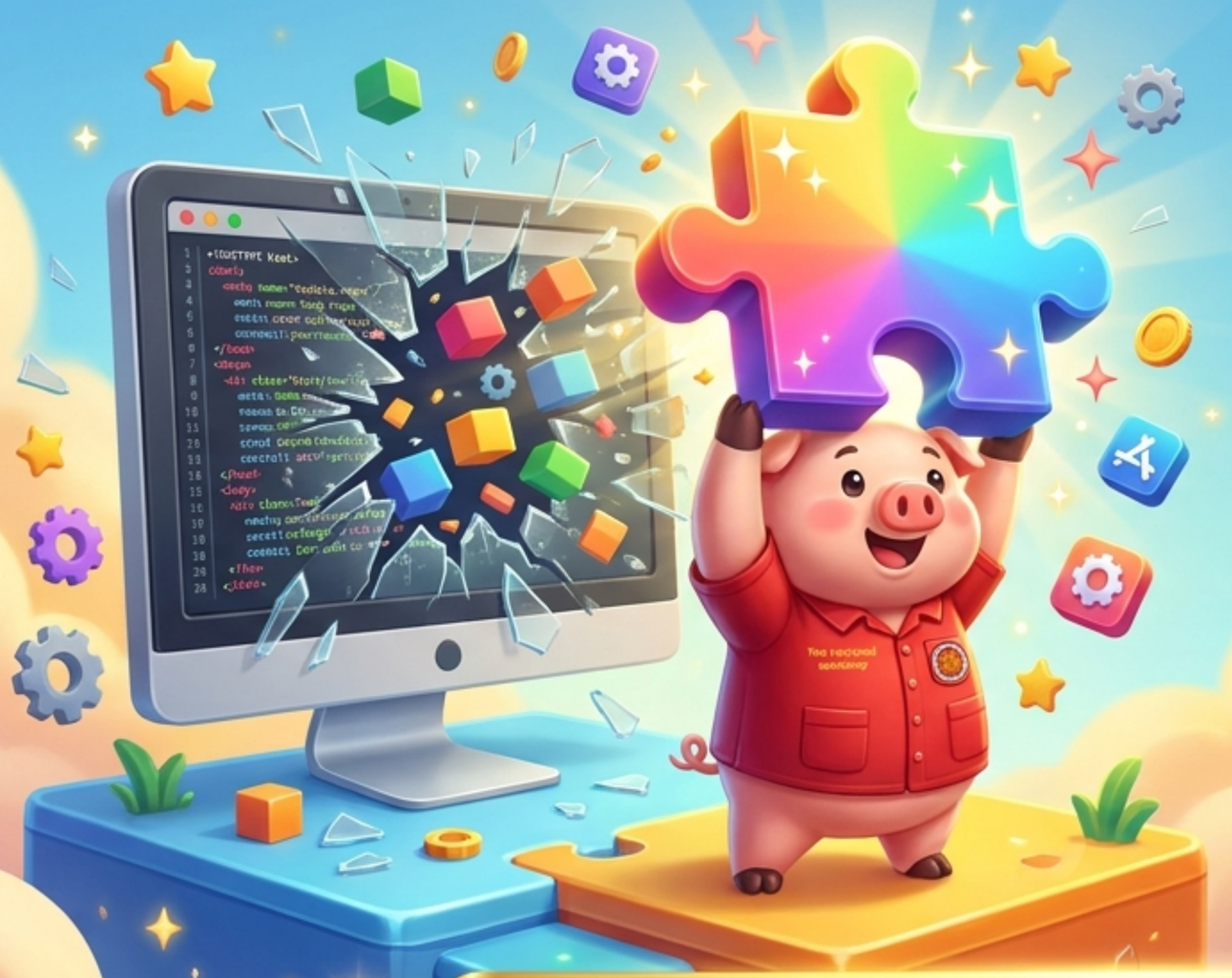
สร้างแอปพลิเคชันด้วยโปรแกรม Kodular

เนรมิตไอเดียให้กลายเป็นแอปมือถือ แบบไม่ต้องเขียนโค้ด



รายวิชา การพัฒนาแอพลิเคชั่นบนอุปกรณ์พกพาเบื้องต้น
ผู้สอน: อาจารย์ปาลิณีย์ อิทธิภัสร์โอภาส
วิทยาลัยอาชีวศึกษาดุสิตพณิชยการ นนทบุรี

Kodular คืออะไร? (เวทมนตร์แห่ง No-Code)



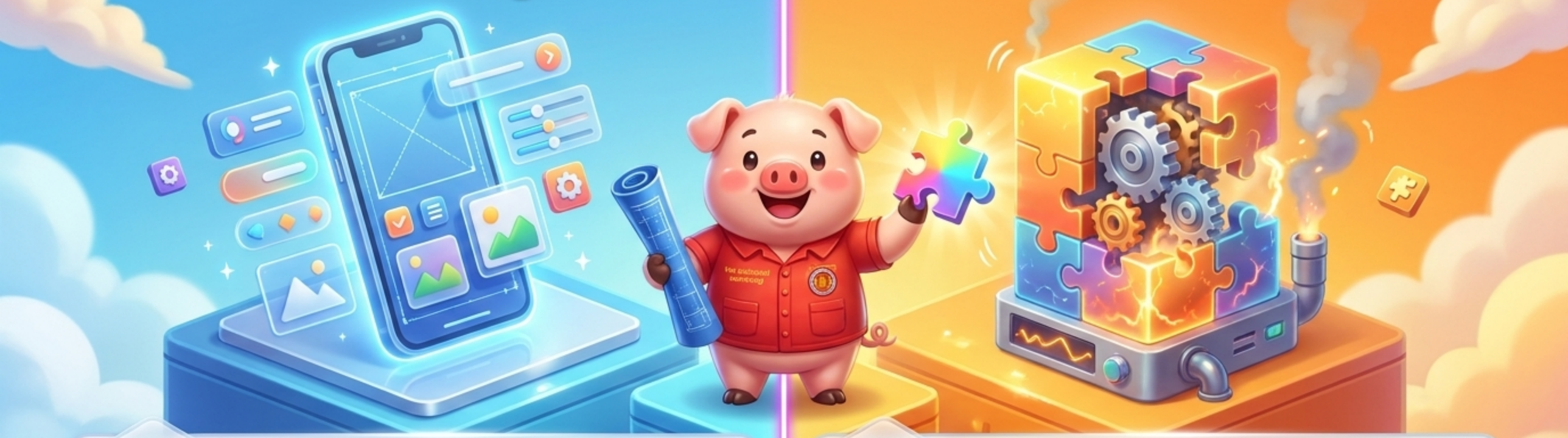
ลากแล้ววาง (Drag & Drop):
สร้างแอปพลิเคชัน Android ได้โดย
ไม่ต้องมีความรู้ด้านการเขียนโค้ดแบบเดิม

ทำงานบนคลาวด์ (Cloud-Based):
ทำงานผ่านเว็บเบราว์เซอร์ ทุกอย่าง
ถูกบันทึกอัตโนมัติ (Auto-save)

ต่อบล็อกจิ๊กซอว์:
เปลี่ยนตรรกะที่ซับซ้อนให้กลายเป็นการ
ต่อบล็อกสีสันสดใสที่เข้าใจง่าย

ทุกคนสามารถเป็นนักพัฒนาแอปได้!

2 โลกคู่ขนานของการสร้างแอป



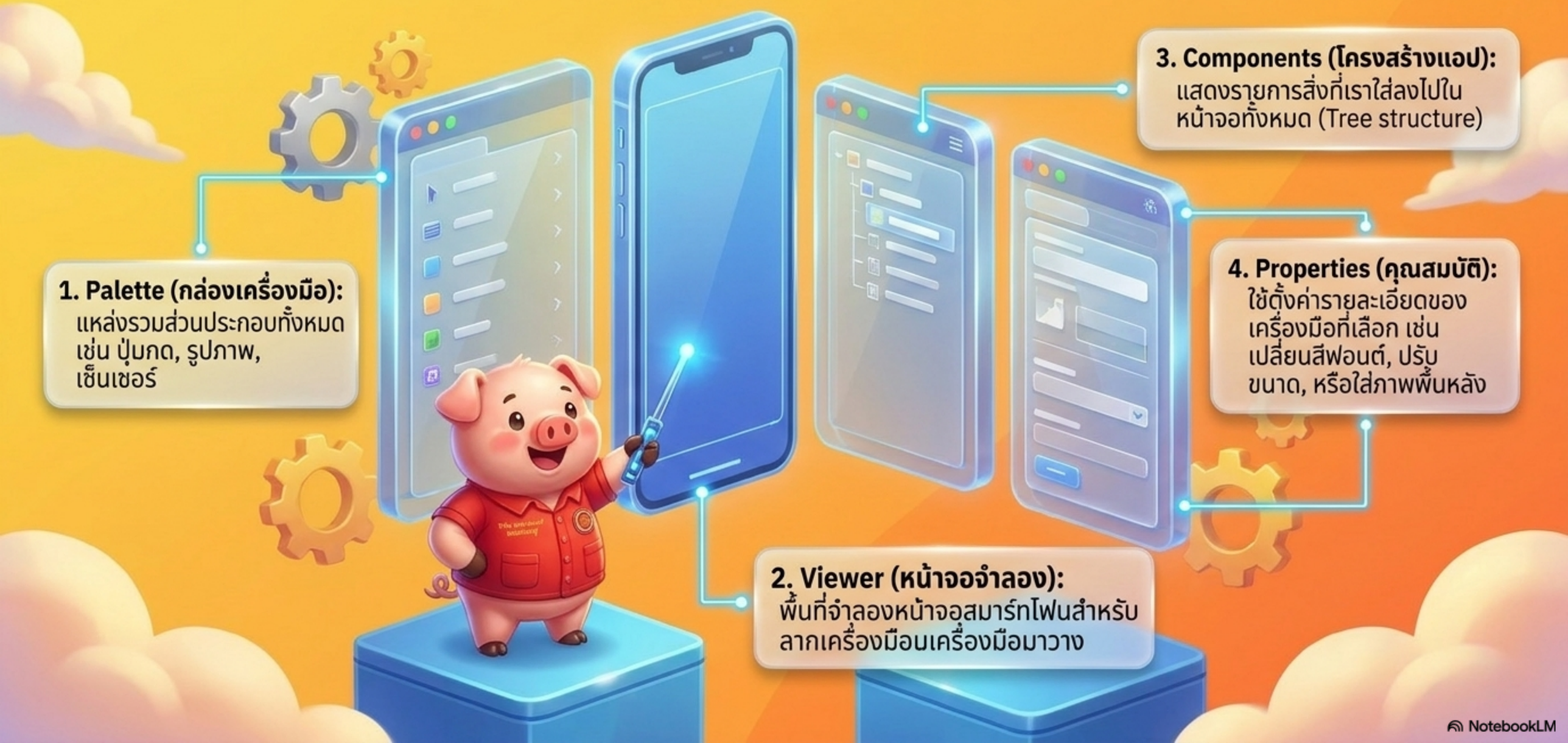
หน้าออกแบบ (Designer)

สิ่งที่ผู้ใช้มองเห็น (Frontend)
- การออกแบบหน้าตา (UI),
การจัดวางปุ่ม, สีสัน, และรูปภาพ

หน้าเขียนโปรแกรม (Blocks)

สิ่งที่แอปพลิเคชันคิด (Backend)
- การกำหนดเงื่อนไข, การคำนวณ,
และการสั่งงานให้แอปทำงานตามที่เราต้องการ

กายวิภาคของหน้าจอ Designer (Anatomy of the Workspace)



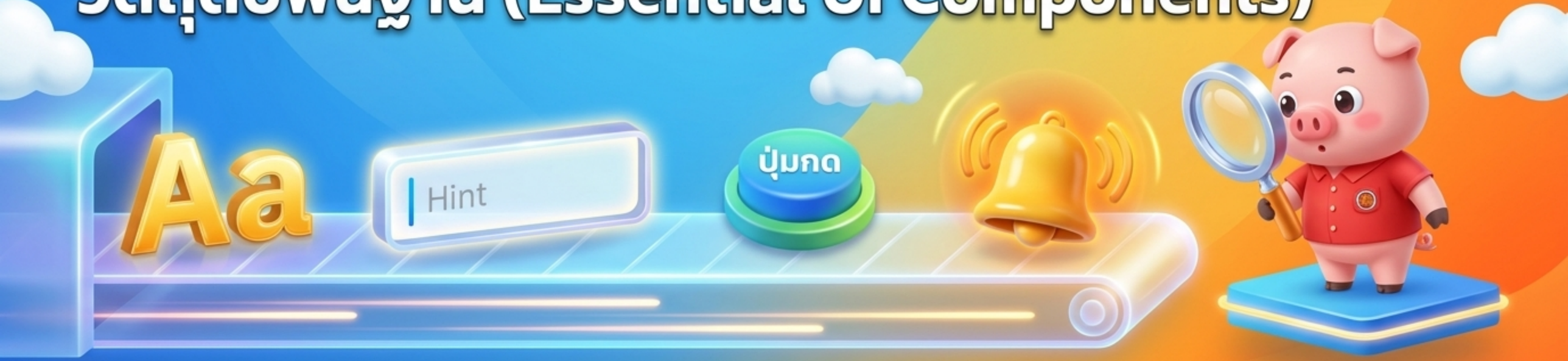
1. Palette (กล่องเครื่องมือ):
แหล่งรวมส่วนประกอบทั้งหมด เช่น ปุ่มกด, รูปภาพ, เซ็นเซอร์

2. Viewer (หน้าจอจำลอง):
พื้นที่จำลองหน้าจอสมาร์ทโฟนสำหรับลากเครื่องมือมาวาง

3. Components (โครงสร้างแอป):
แสดงรายการสิ่งที่เราใส่ลงไปในหน้าจอทั้งหมด (Tree structure)

4. Properties (คุณสมบัติ):
ใช้ตั้งค่ารายละเอียดของเครื่องมือที่เลือก เช่น เปลี่ยนสีฟอนต์, ปรับขนาด, หรือใส่ภาพพื้นหลัง

วัตถุดิบพื้นฐาน (Essential UI Components)



Label (ป้ายข้อความ)

ใช้สำหรับแสดงข้อความที่ผู้ใช้ไม่สามารถแก้ไขได้ เช่น หัวข้อแอป

TextBox (ช่องกรอกข้อมูล)

พื้นที่ให้ผู้ใช้พิมพ์ข้อความลงไป (เช่น ใส่ Username/Password)

Pro-tip: ใช้คุณสมบัติ Hint เพื่อสร้างคำใบ้

Button (ปุ่มกด)

พระเอกของการสั่งงาน เมื่อกดแล้วจะเกิดเหตุการณ์บางอย่าง

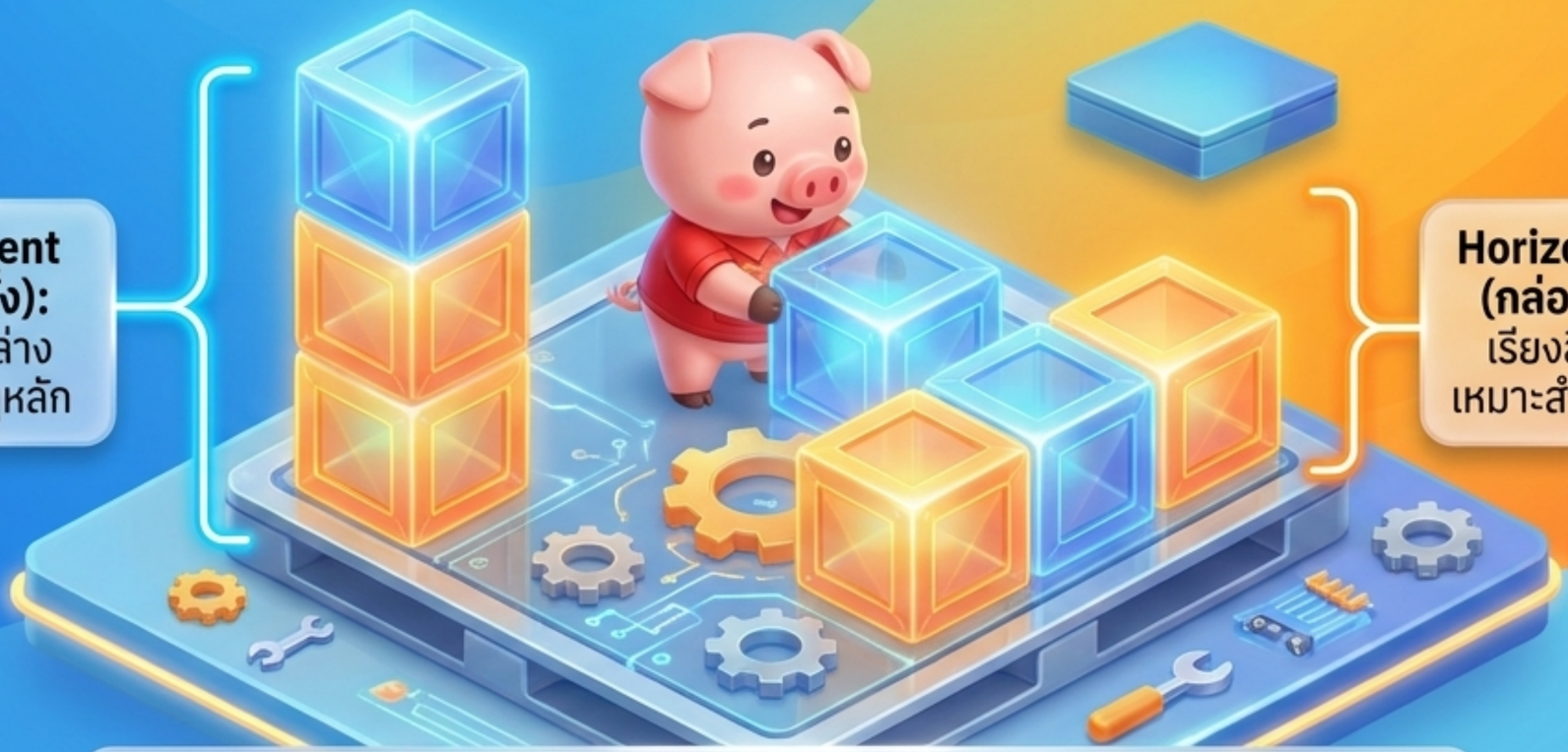
Notifier (ตัวแจ้งเตือน)

ข้อความ Pop-up เด้งเตือนผู้ใช้เมื่อทำรายการสำเร็จหรือผิดพลาด (Non-visible component)

การจัดระเบียบหน้าจอด้วย Layouts

ส่วนประกอบแอปไม่สามารถลอยเคว้งคว้างได้ ต้องมี ‘กล่อง’ ใส่เสมอ!

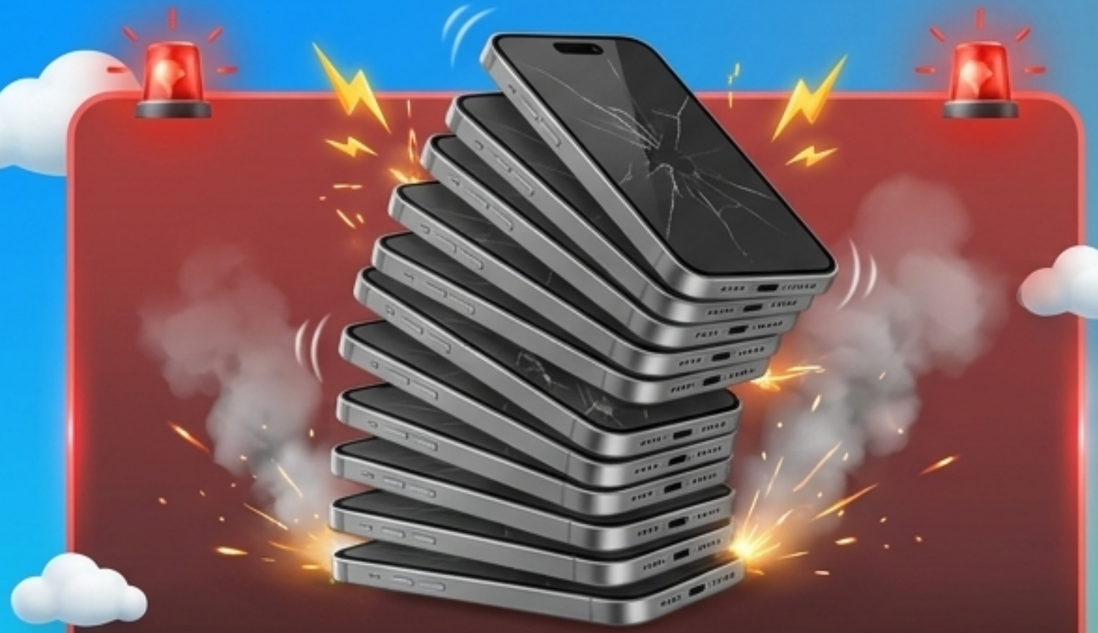
Vertical Arrangement (กล่องจัดเรียงแนวตั้ง):
เรียงสิ่งของจากบนลงล่าง
เหมาะสำหรับจัดหน้าเมนูหลัก



Horizontal Arrangement (กล่องจัดเรียงแนวนอน):
เรียงสิ่งของจากซ้ายไปขวา
เหมาะสำหรับวางปุ่มเรียงติดกัน

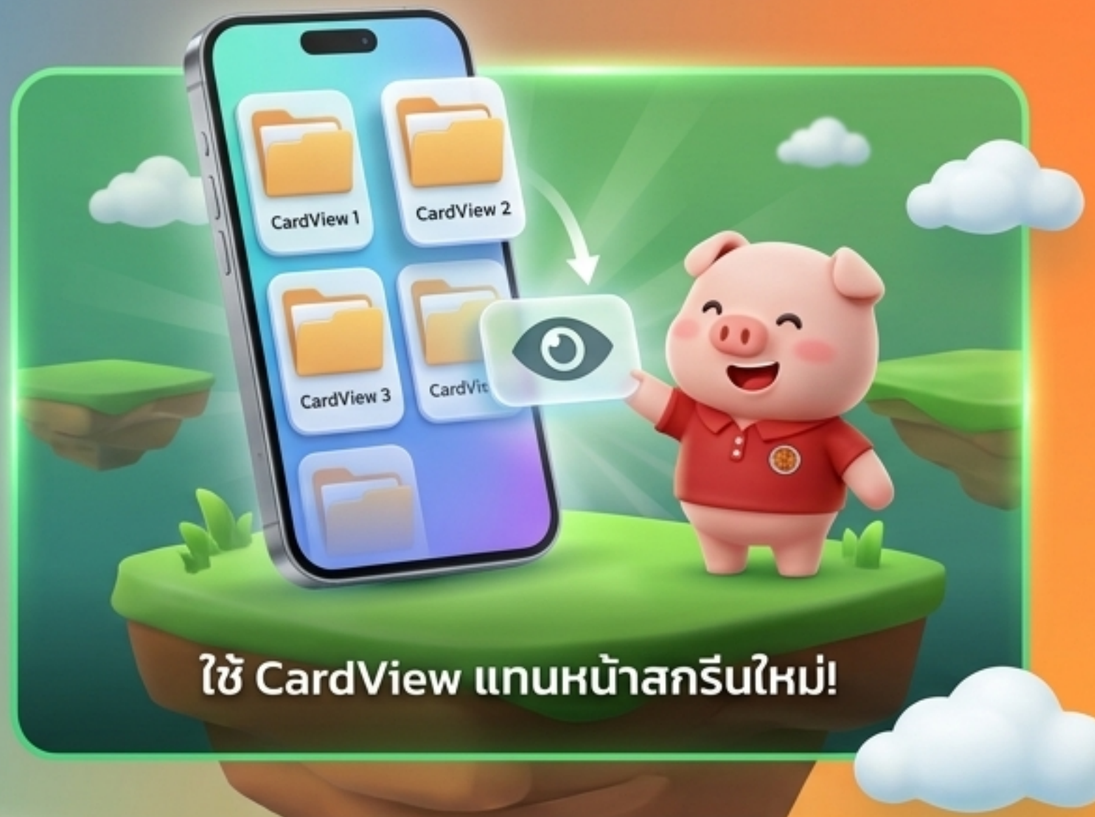
การใช้ Layout ช่วยให้แอปพลิเคชันของเราแสดงผลได้สวยงาม ไม่ผิดเพี้ยน ไม่ว่าจะเปิดบนมือถือหน้าจอเล็กหรือแท็บเล็ต

Pro-Tip: เทคนิค CardView สยบปัญหาแอปเด้ง



การสร้างหน้า Screen ใหม่มากเกินไป
จะทำให้แอปหนัก โหลดช้า และเกิด Error ตอน Export

VS



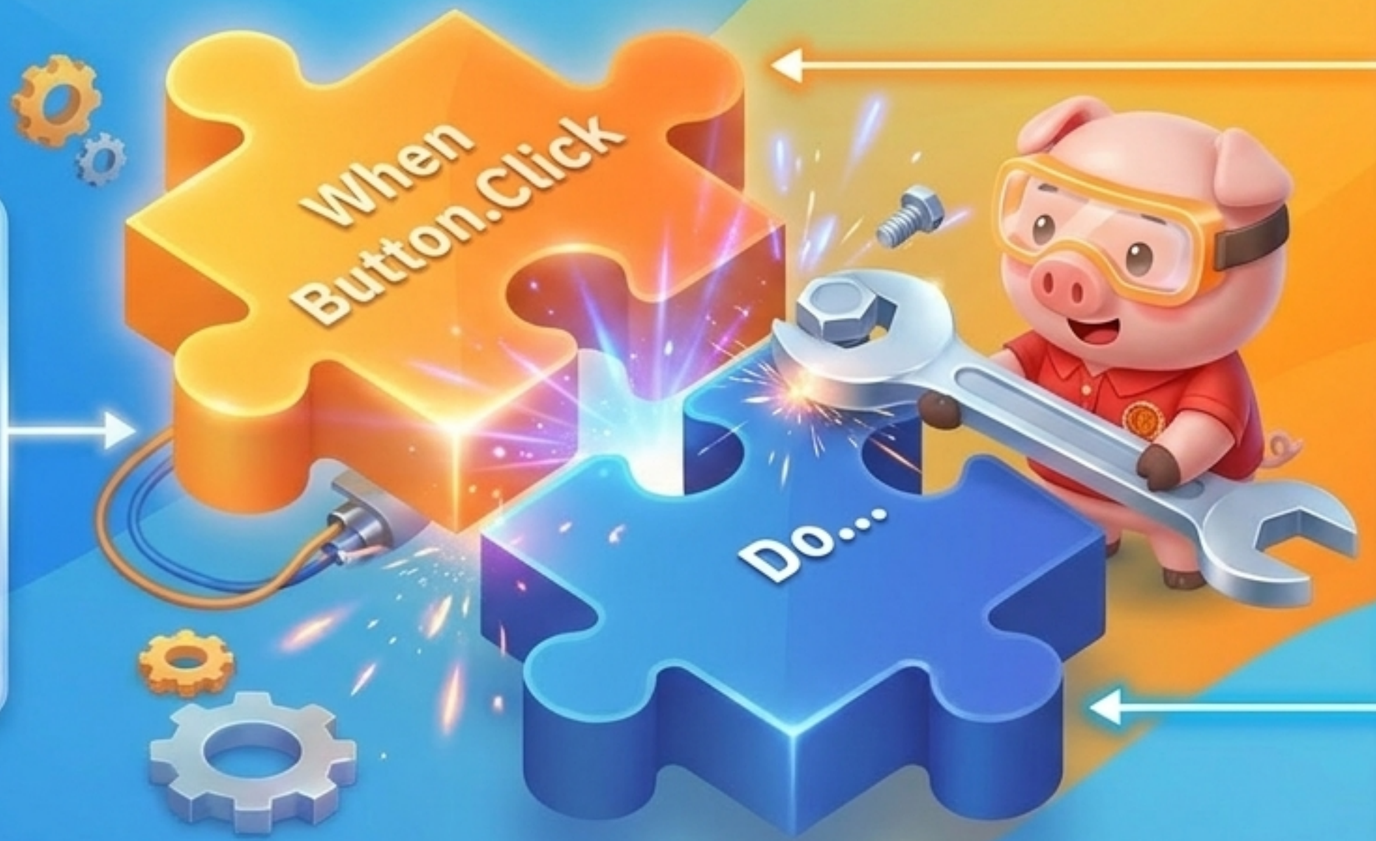
ใช้ CardView แทนหน้าสกรีนใหม่!

1. สร้าง CardView ซ้อนกันในหน้า Screen เดียว
2. ใช้คำสั่งตั้งค่า Visible (การมองเห็น) ให้เป็น True (เปิด) หรือ False (ปิด)
3. สลับหน้าไปมาได้ลื่นไหลเหมือนแอปมืออาชีพ โดยที่แอปไม่พัง!

สมองกลของแอป: การเขียนบล็อกคำสั่ง (Block Programming)

Kodular ใช้ตรรกะแบบ Event-Driven
(ขับเคลื่อนด้วยเหตุการณ์)

หลักการท่องจำ:
“เมื่อเกิดสิ่งนี้... ให้ทำสิ่งนั้น”
(When X happens -> Do Y)



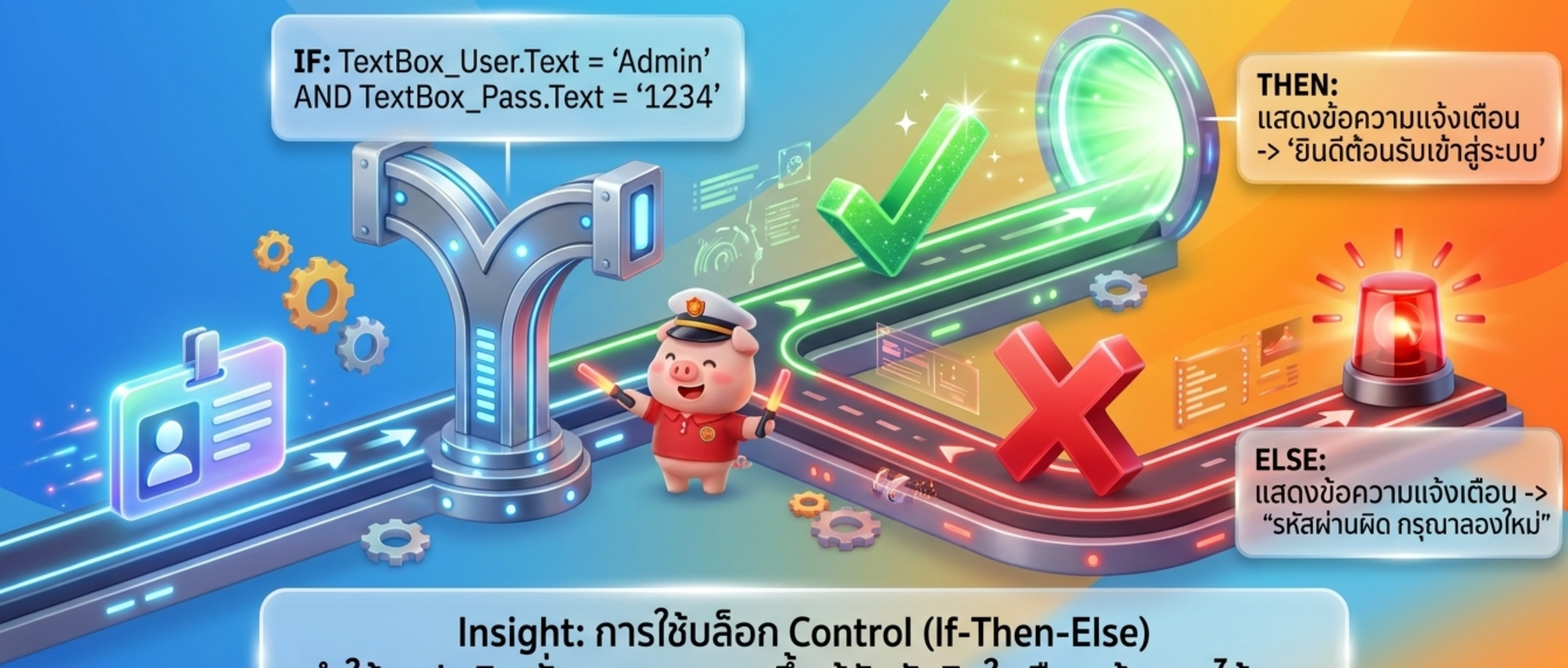
When (เมื่อ):
ผู้ใช้กดปุ่ม (Click),
หรือแอปเริ่มทำงาน
(Initialize)

Do (ให้ทำ):
เปลี่ยนสีข้อความ,
คำนวณเลข,
หรือเปิดหน้าต่าง
แจ้งเตือน

(Notifier)

ตัวอย่างการใช้ตรรกะ ระบบ Login เบื้องต้น

IF: TextBox_User.Text = ‘Admin’
AND TextBox_Pass.Text = ‘1234’



THEN:
แสดงข้อความแจ้งเตือน
-> ‘ยินดีต้อนรับเข้าสู่ระบบ’

ELSE:
แสดงข้อความแจ้งเตือน ->
“รหัสผ่านผิด กรุณาลองใหม่”

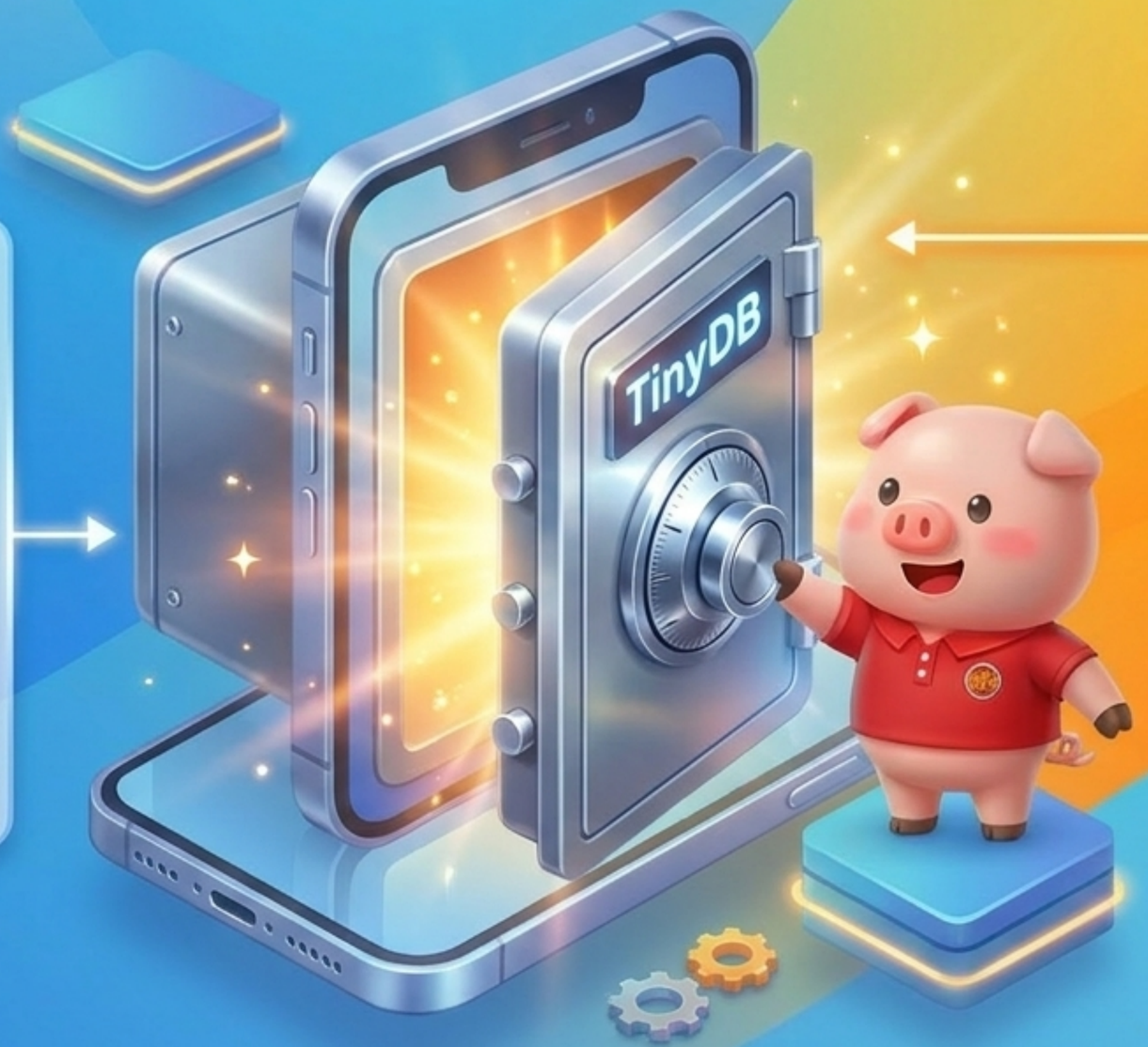
Insight: การใช้บล็อก Control (If-Then-Else)
ทำให้แอปพลิเคชั่นของเราฉลาดขึ้น รู้จักตัดสินใจเลือกเส้นทางได้เอง

ความทรงจำของแอป: แนะนำ TinyDB

Core Concept

โดยปกติแล้ว ตัวแปร (Variables) จะลบข้อมูลทิ้งเมื่อเราปิดแอป

TinyDB
(Database ขนาดจิ๋ว) คือคลังเก็บข้อมูลส่วนตัวของแอปที่ฝังอยู่ในเครื่อง

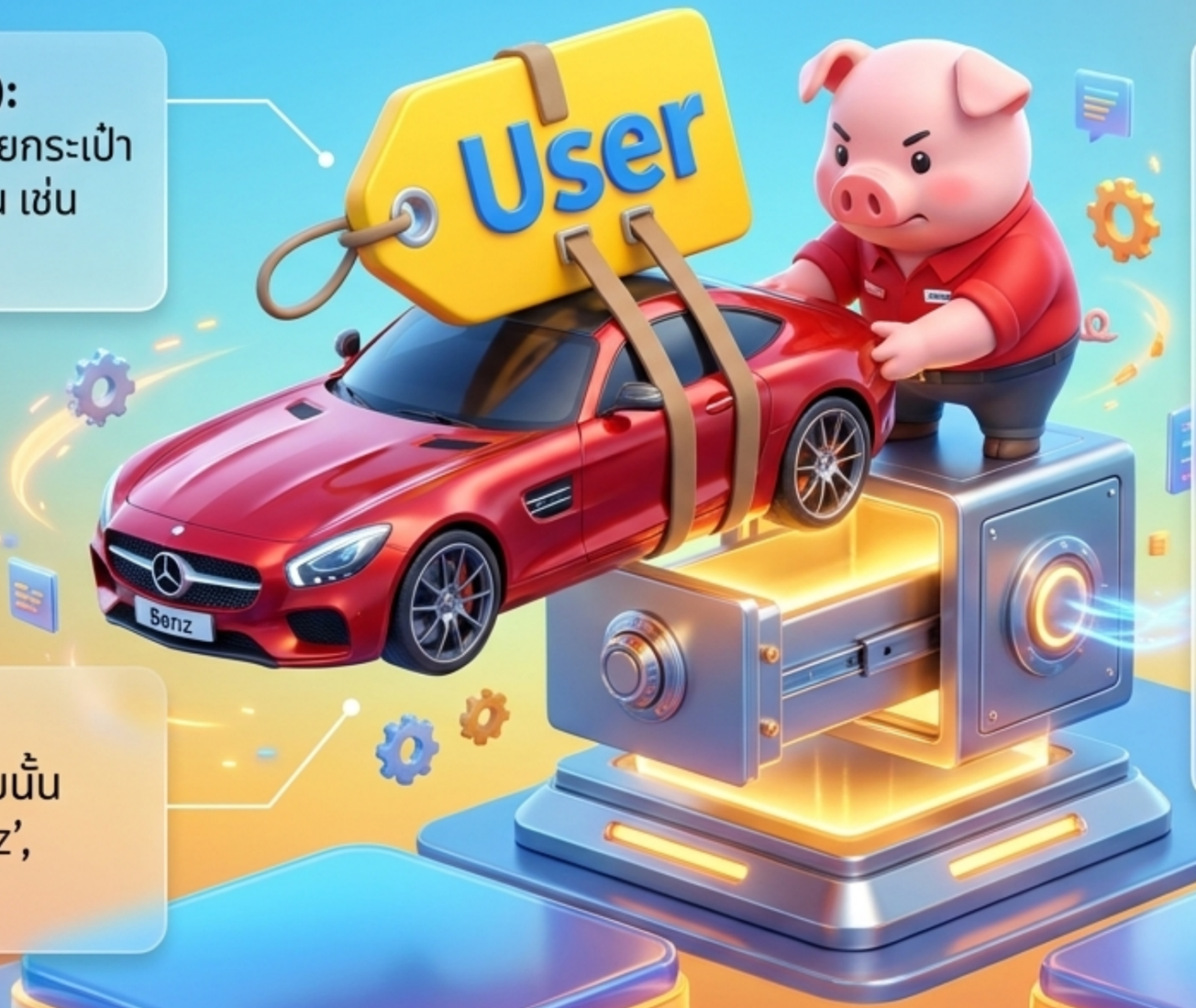


Key Benefits

- บันทึกข้อมูลแบบถาวร (Local Storage) บนมือถือของเราเอง
- ปิดแอปแล้วเปิดใหม่ ข้อมูลก็ยังอยู่ครบถ้วน!
- **เหมาะสำหรับบันทึก:** คะแนนสูงสุดในเกม, ข้อมูลโปรไฟล์ผู้ใช้, หรือการตั้งค่าแอป

กลไกของ TinyDB: ระบบ Tag และ Value

Tag (ป้ายชื่ออ้างอิง):
เปรียบเสมือนป้ายห้อยกระเป๋า (ต้องตั้งชื่อให้ไม่ซ้ำกัน เช่น 'User', 'Score')



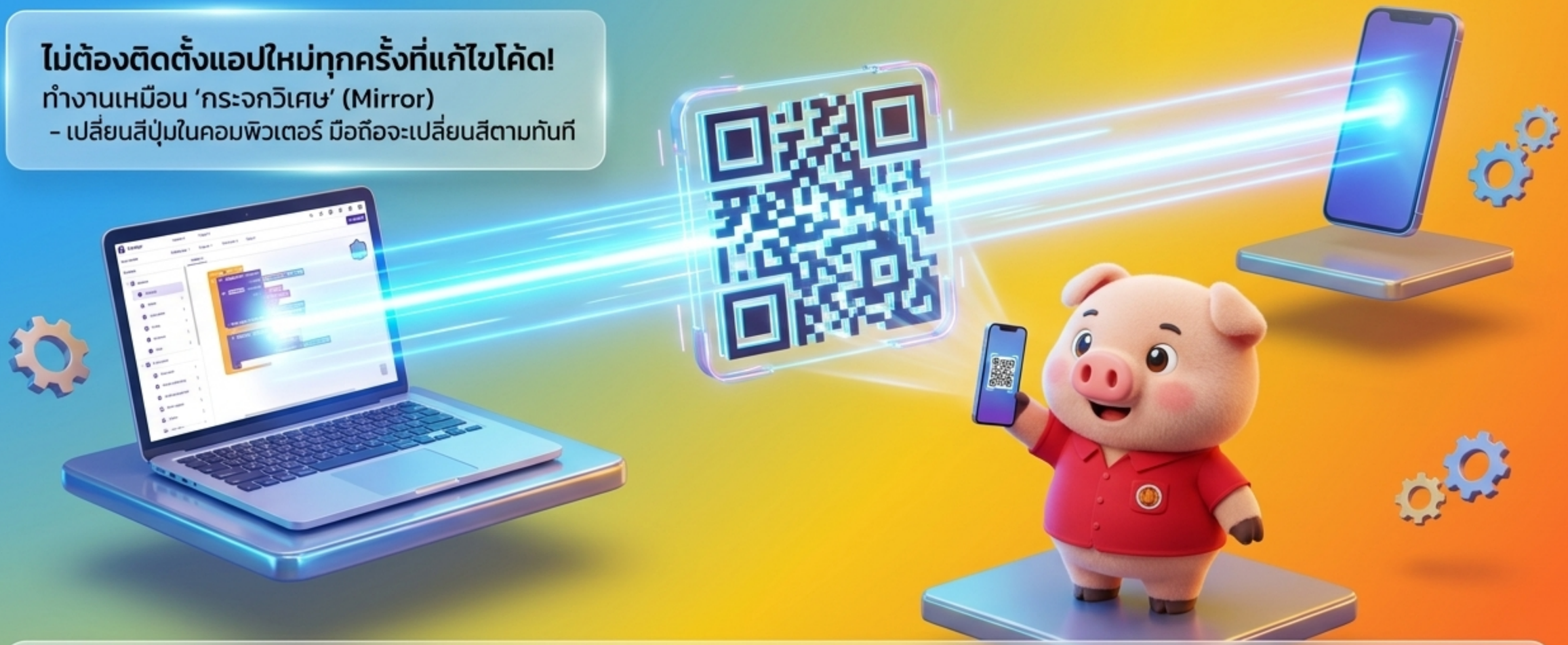
Value (ข้อมูล):
สิ่งของที่อยู่คู่กับป้ายนั้น (เช่น ข้อความ ‘Benz’, หรือตัวเลข ‘999’)

The 2 Essential Commands

1. **StoreValue (บันทึกข้อมูล):** เอาป้าย (Tag) ไปแปะกับข้อมูล (Value) แล้วเก็บเข้าตู้
2. **GetValue (เรียกดูข้อมูล):** บอกชื่อป้าย (Tag) ให้ตู้เซฟเพื่อหยิบข้อมูล (Value) กลับมาใช้งาน

ทดสอบแบบ Real-Time ด้วย Kodular Companion

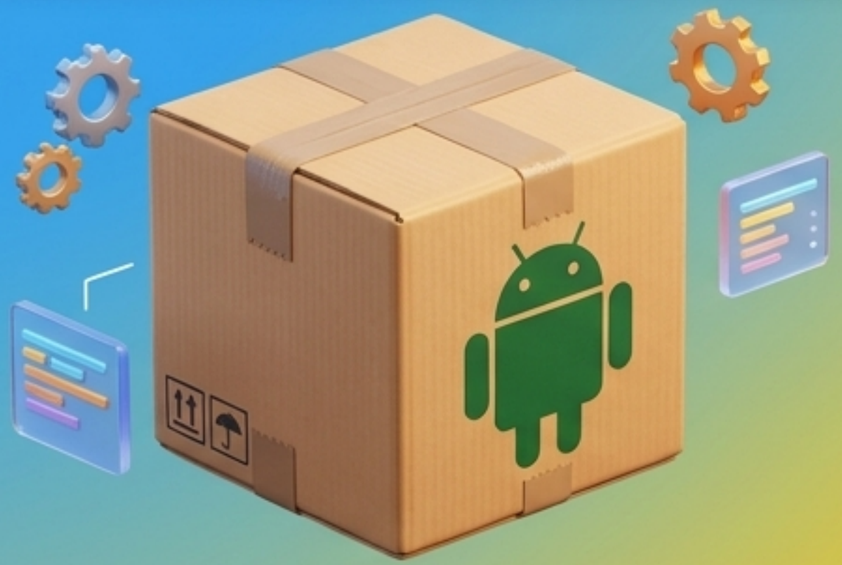
ไม่ต้องติดตั้งแอปใหม่ทุกครั้งที่แก้ไขโค้ด!
ทำงานเหมือน 'กระจกวิเศษ' (Mirror)
- เปลี่ยนสีปุ่มในคอมพิวเตอร์ มือถือจะเปลี่ยนสีตามทันที



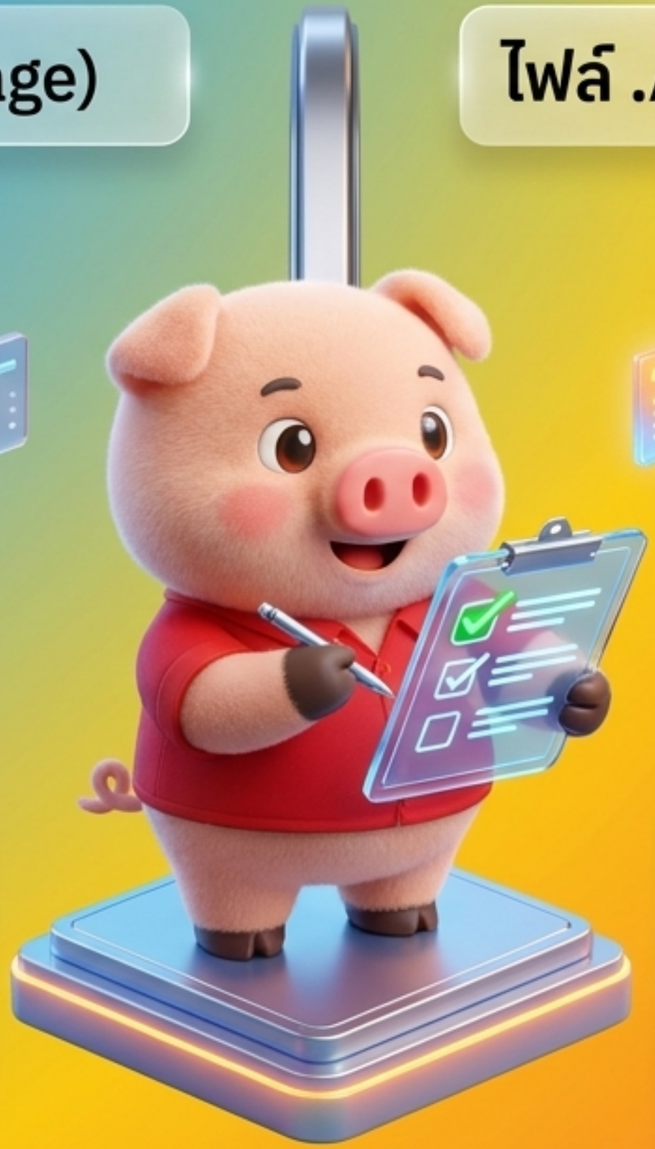
1. ดาวน์โหลดแอป Kodular Companion จาก Google Play Store ลงในมือถือ Android
2. คลิกเมนู Test -> Connect to companion บนหน้าเว็บ Kodular
3. ใช้แอปในมือถือ สแกน QR Code เพื่อดูผลลัพธ์การทำงานของแอปได้ทันที

สรุปงานและส่งออก: AAB vs. APK

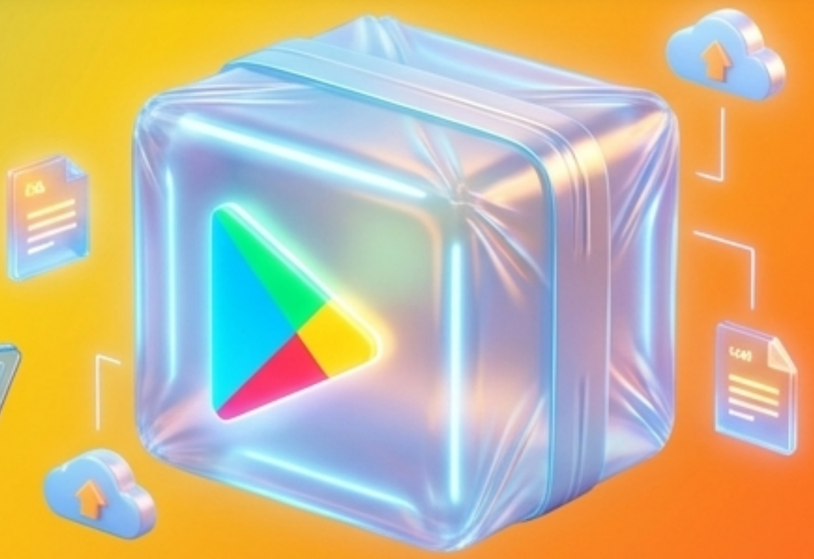
ไฟล์ .APK (Android Package)



- ไฟล์ขนาดใหญ่ มีทุกอย่างครบชุด
- **เหมาะสำหรับ:** ทดสอบในเครื่อง, ส่งให้เพื่อนติดตั้งโดยตรง (Sideloading), ส่งการบ้านอาจารย์

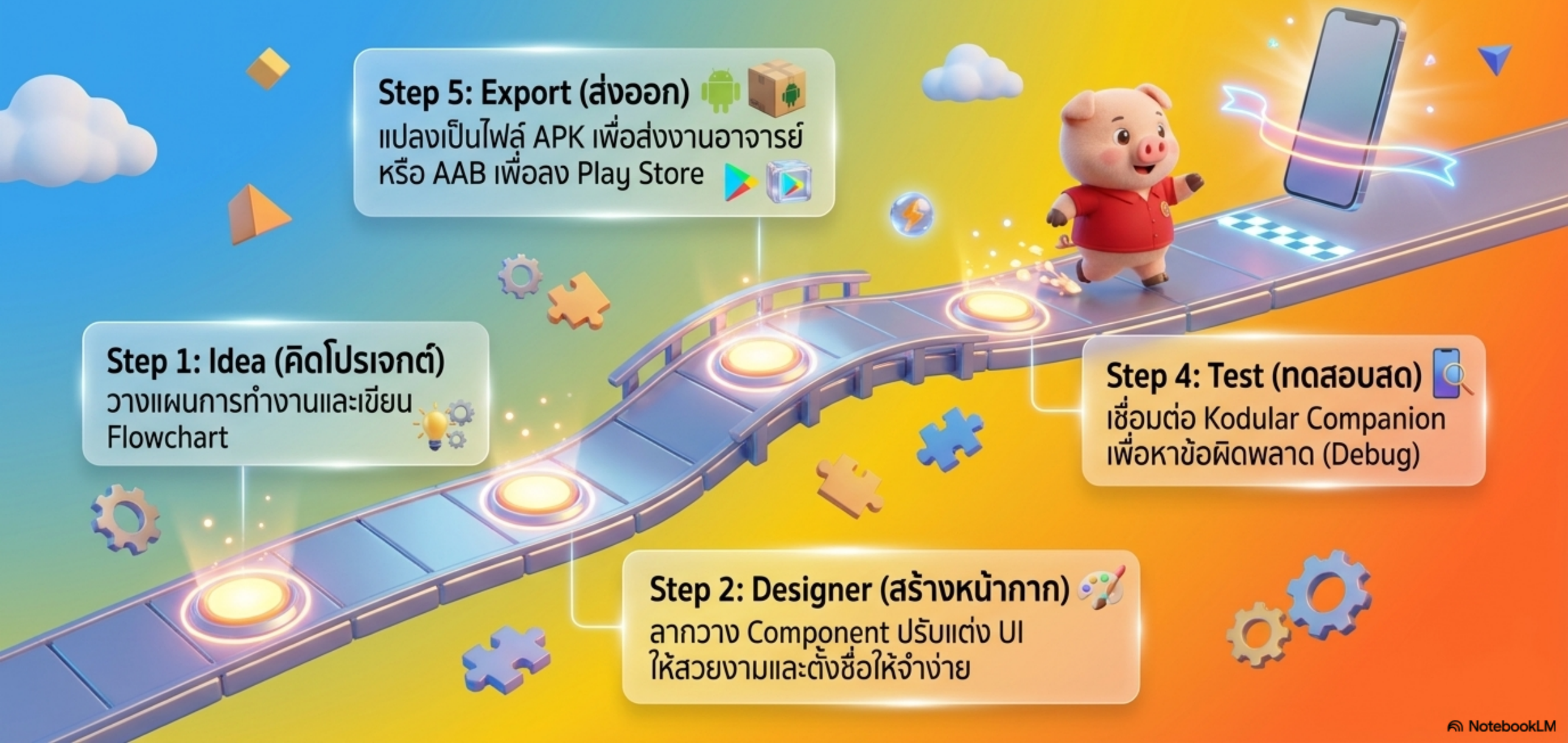


ไฟล์ .AAB (Android App Bundle)



- ไฟล์ถูกบีบอัดและปรับแต่งให้มีขนาดเล็กที่สุด (Optimized)
- **เหมาะสำหรับ:** การนำแอปพลิเคชันขึ้นเผยแพร่บน Google Play Store (เป็นข้อบังคับของ Google)

สรุป Workflow สำหรับนักพัฒนาแอป (The Builder's Journey)



Step 1: Idea (คิดโปรเจกต์)
วางแผนการทำงานและเขียน Flowchart

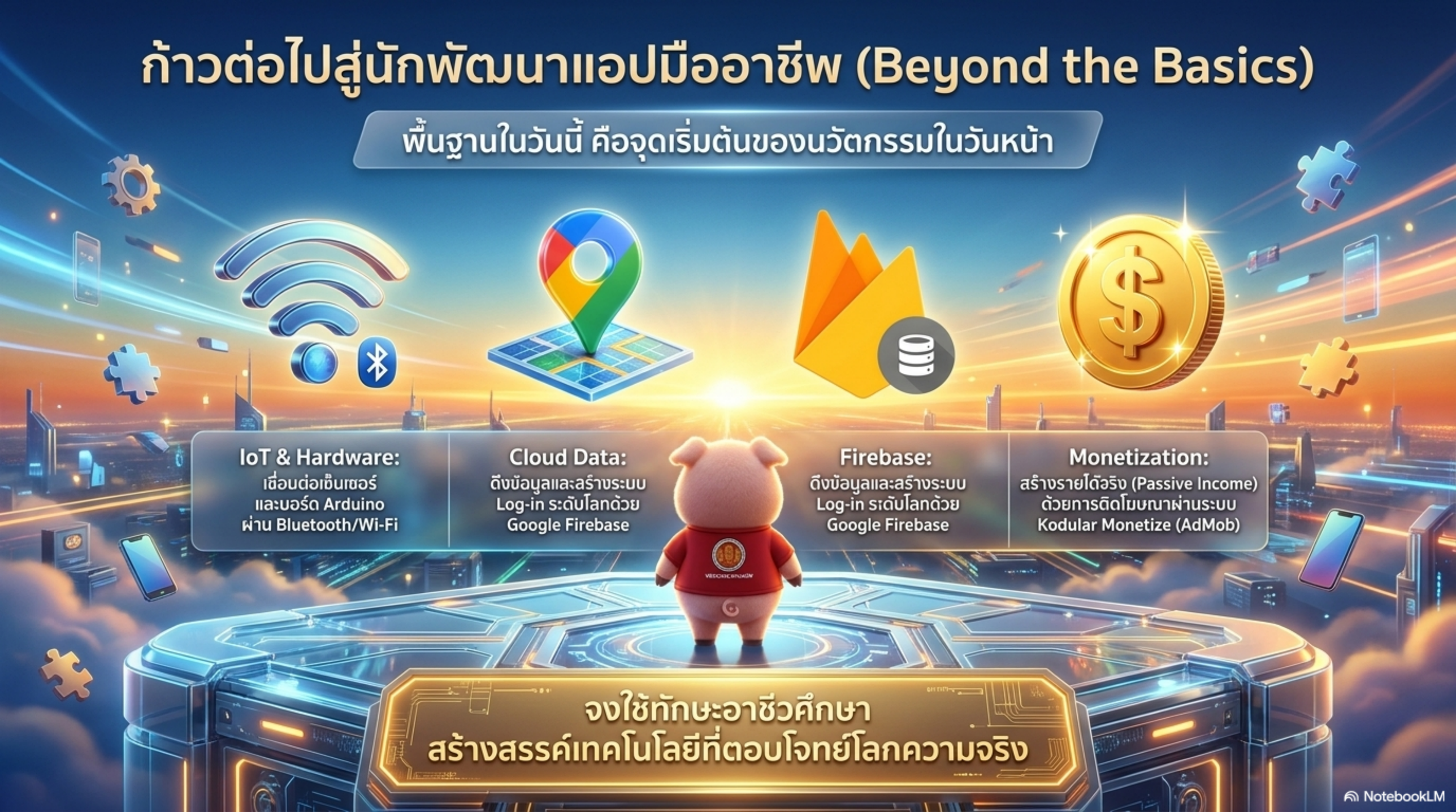
Step 2: Designer (สร้างหน้ากาก)
ลากวาง Component ปรับแต่ง UI ให้สวยงามและตั้งชื่อให้จำง่าย

Step 4: Test (ทดสอบสด)
เชื่อมต่อ Kodular Companion เพื่อหาข้อผิดพลาด (Debug)

Step 5: Export (ส่งออก)
แปลงเป็นไฟล์ APK เพื่อส่งงานอาจารย์ หรือ AAB เพื่อลง Play Store

ก้าวต่อไปสู่นักพัฒนาแอปมืออาชีพ (Beyond the Basics)

พื้นฐานในวันนี้ คือจุดเริ่มต้นของนวัตกรรมในวันหน้า



IoT & Hardware:
เชื่อมต่อเซ็นเซอร์และบอร์ด Arduino ผ่าน Bluetooth/Wi-Fi

Cloud Data:
ดึงข้อมูลและสร้างระบบ Log-in ระดับโลกด้วย Google Firebase

Firebase:
ดึงข้อมูลและสร้างระบบ Log-in ระดับโลกด้วย Google Firebase

Monetization:
สร้างรายได้จริง (Passive Income) ด้วยการติดโฆษณาผ่านระบบ Kodular Monetize (AdMob)

จงใช้ทักษะอาชีวศึกษา
สร้างสรรค์เทคโนโลยีที่ตอบโจทย์โลกความจริง