**เอกสารประกอบการปฏิบัติการ**

**รายวิชา SC362007 Mobile Device Programming**

**Lab 3:** **การจัดการ Layout**

**วัตถุประสงค์**

1. เพื่อให้นักศึกษาสามารถใช้การออกแบบหน้าจออุปกรณ์เคลื่อนที่และการใช้ Layout

2. เพื่อให้นักศึกษาสามารถใช้ Element ของ Text และ Image

**คำชี้แจง**

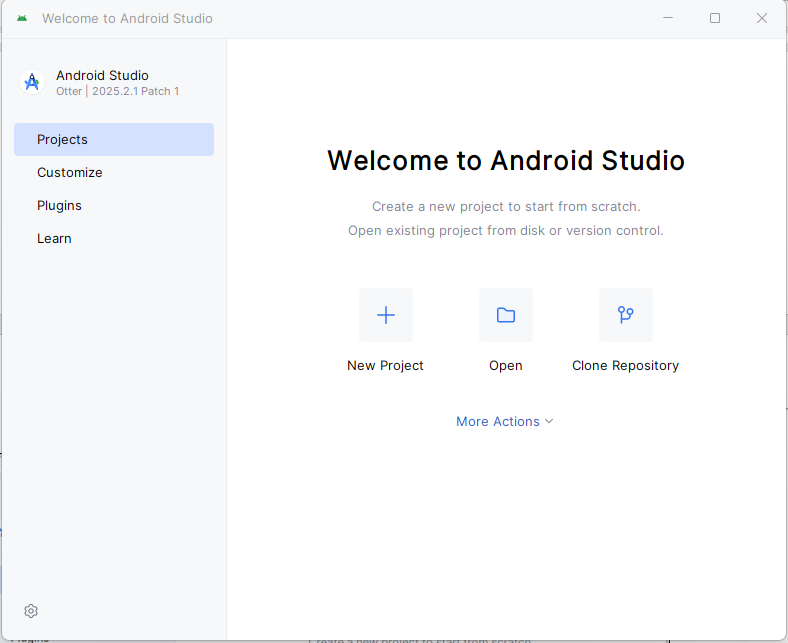
* 1. ให้นักศึกษาแคปคำสั่งและผลลัพธ์ลง Word โดยให้ตั้งชื่อไฟล์ว่า Lab03\_รหัสนักศึกษา(มีขีด).pdf
  2. เมื่อนักศึกษาทำเสร็จเรียบร้อยแล้วให้แจ้งผู้ช่วยสอน (TA) สำหรับตรวจ และตอบคำถาม เพื่อรับคะแนนปฏิบัติการ

(คะแนนจะมาจากการตรวจภายในห้องเรียนเท่านั้น)

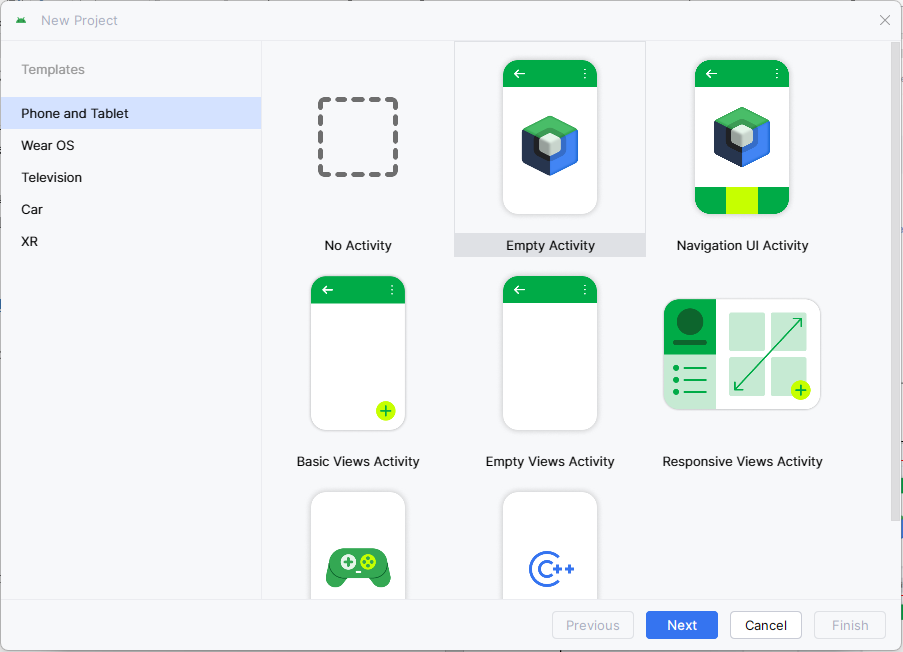
* 1. เมื่อตรวจกับ TA เรียบร้อยแล้วให้นักศึกษาส่งใน Classroom เพื่อสำรองข้อมูลงานของนักศึกษา
  2. ไม่อนุญาตให้ส่งงานย้อนหลังได้ ยกเว้นกรณีลาป่วย อุบัติเหตุ หรือเหตุจำเป็น (ต้องมีใบรับรองแพทย์ หรือหลักฐานอื่น ๆ)

**การสร้างโปรเจคใหม่**

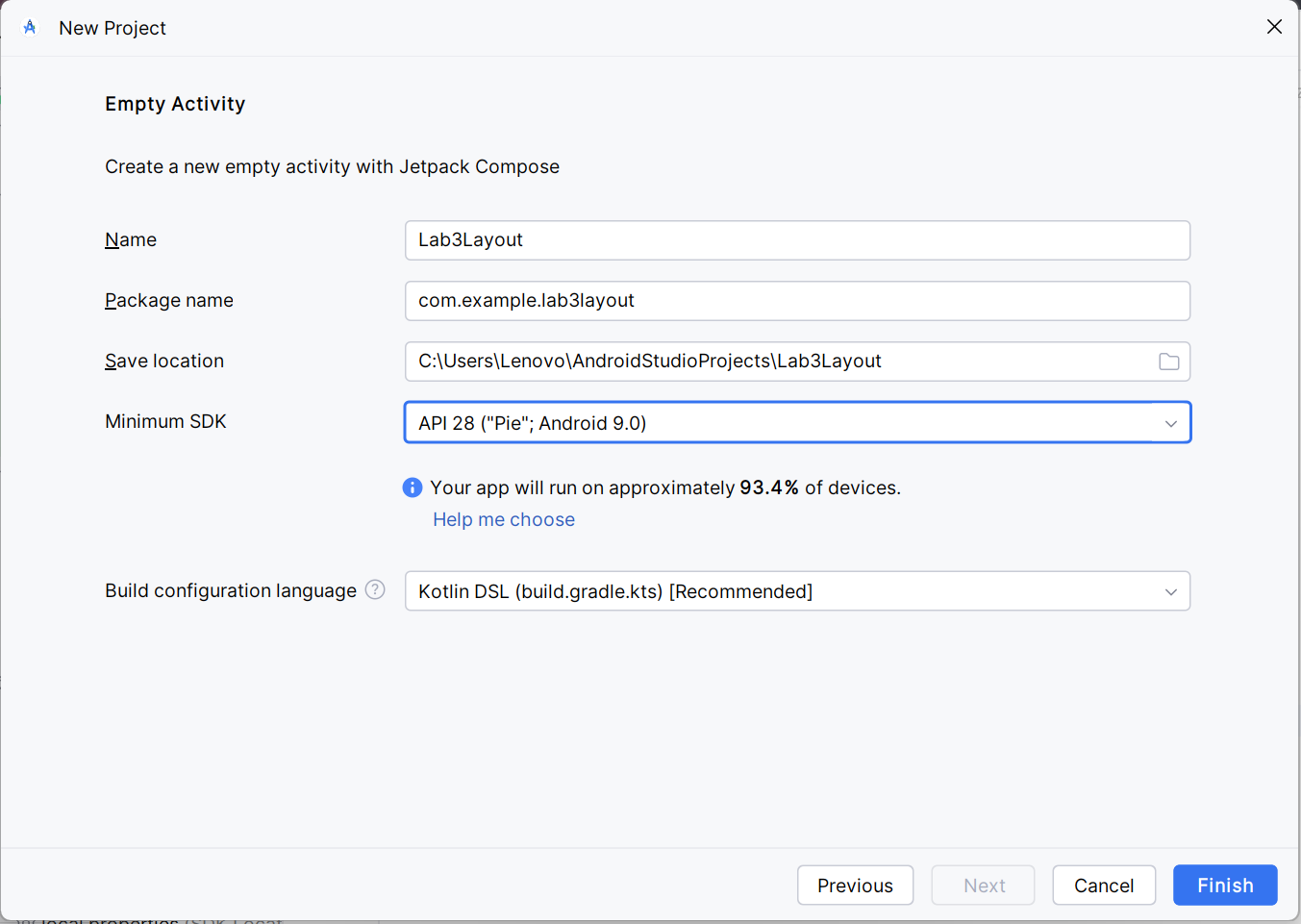
1.ให้เปิดโปรแกรม Android Studio จะแสดงหน้าจอดังนี้ ให้เลือก Create New Project



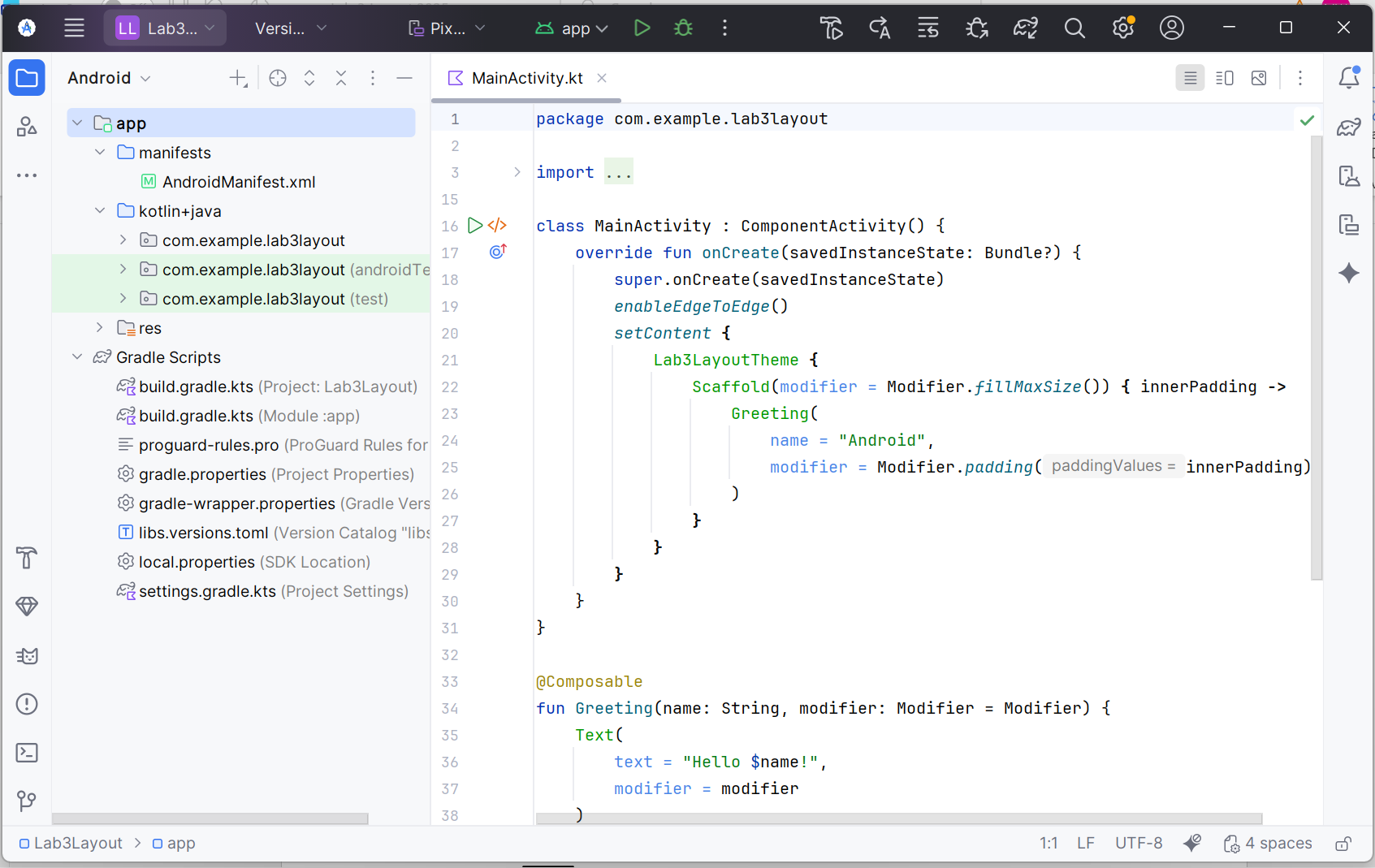
2. จากนั้นเลือก Empty Activity แล้วคลิกที่ปุ่ม Next



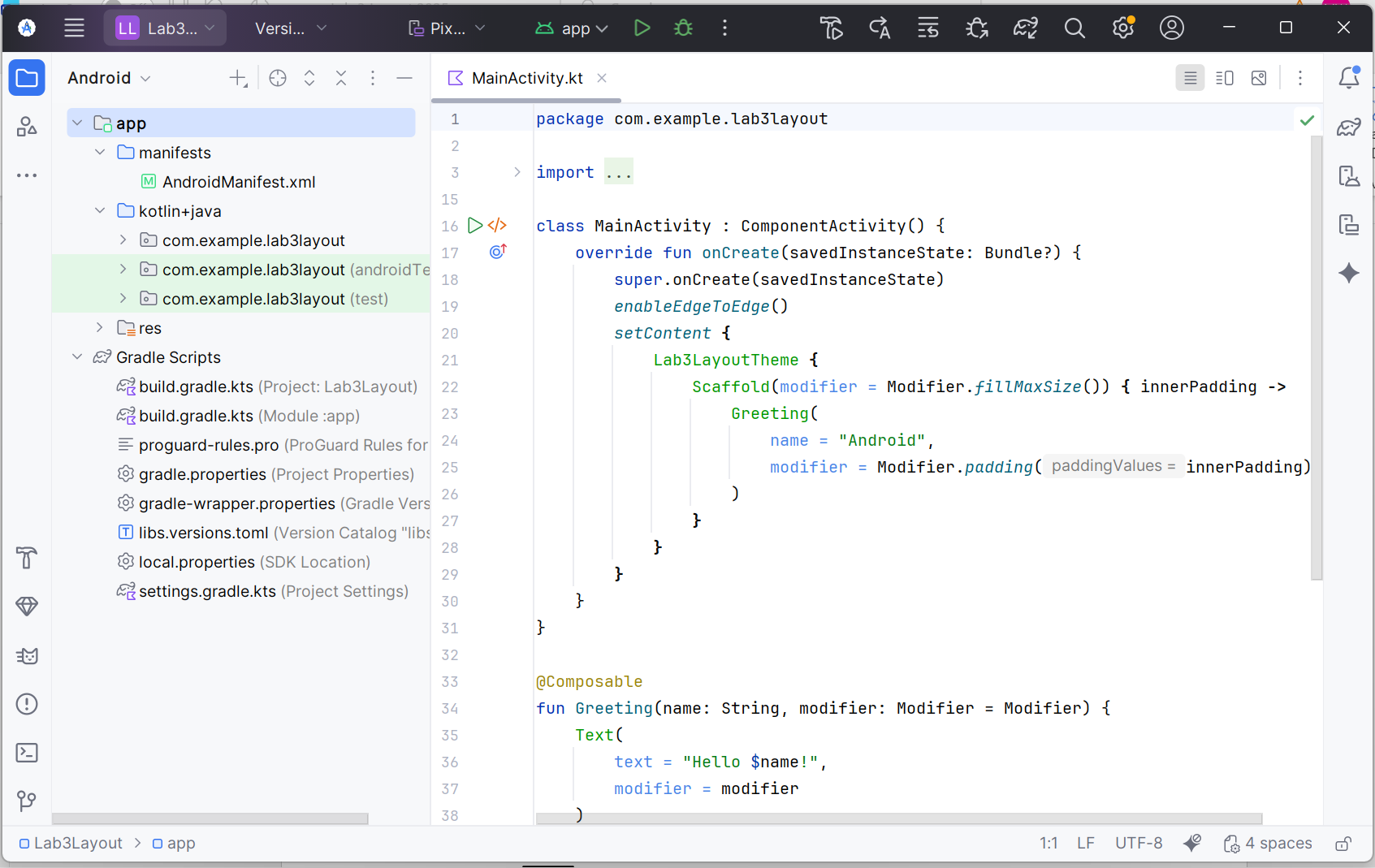
3. ถัดมาให้ตั้งชื่อ Application name คือ Lab3Layout จากนั้นกด Finish



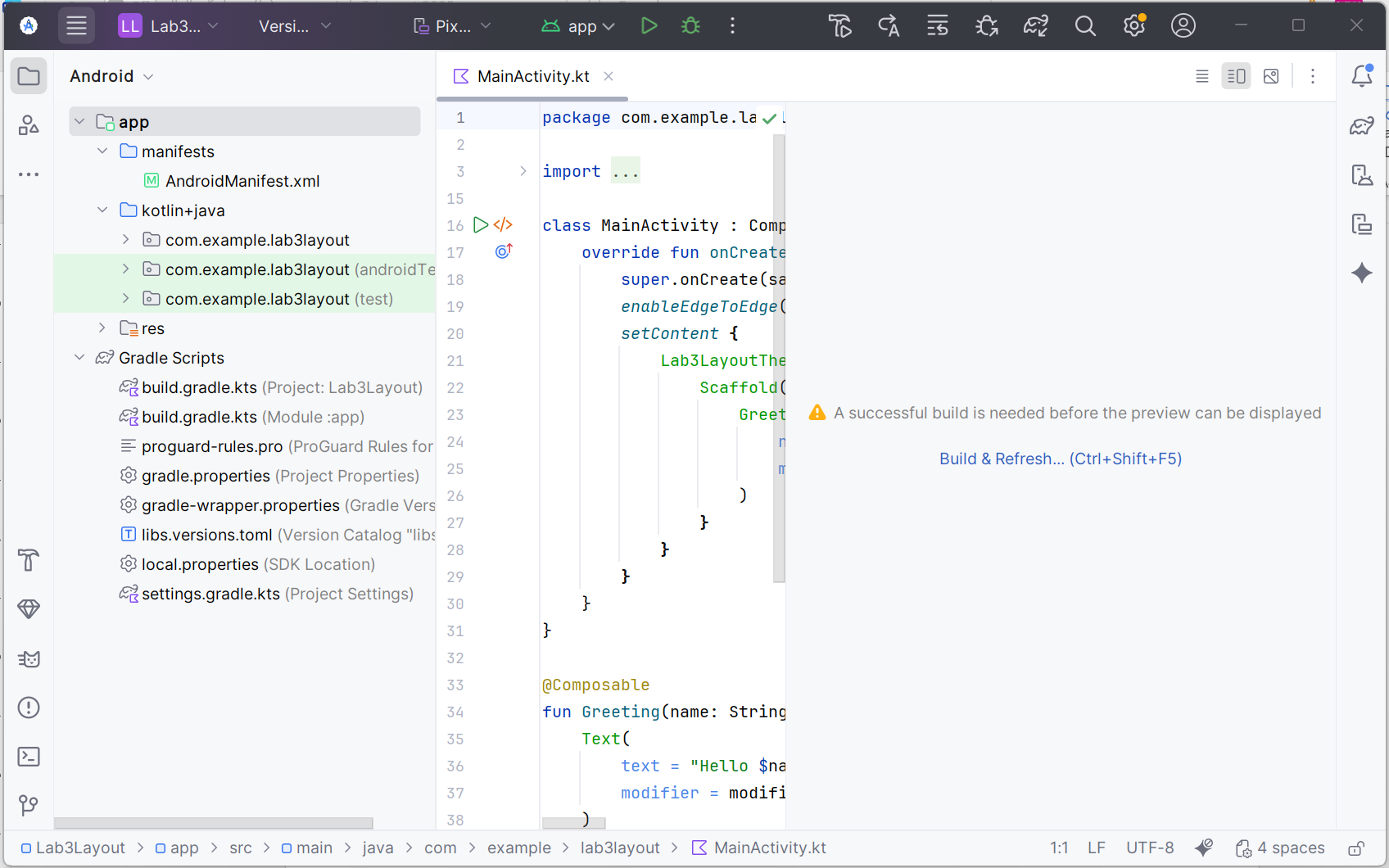
จะต้องรอสักครู่ เพื่อให้โปรแกรมสร้างโปรเจคเสร็จ



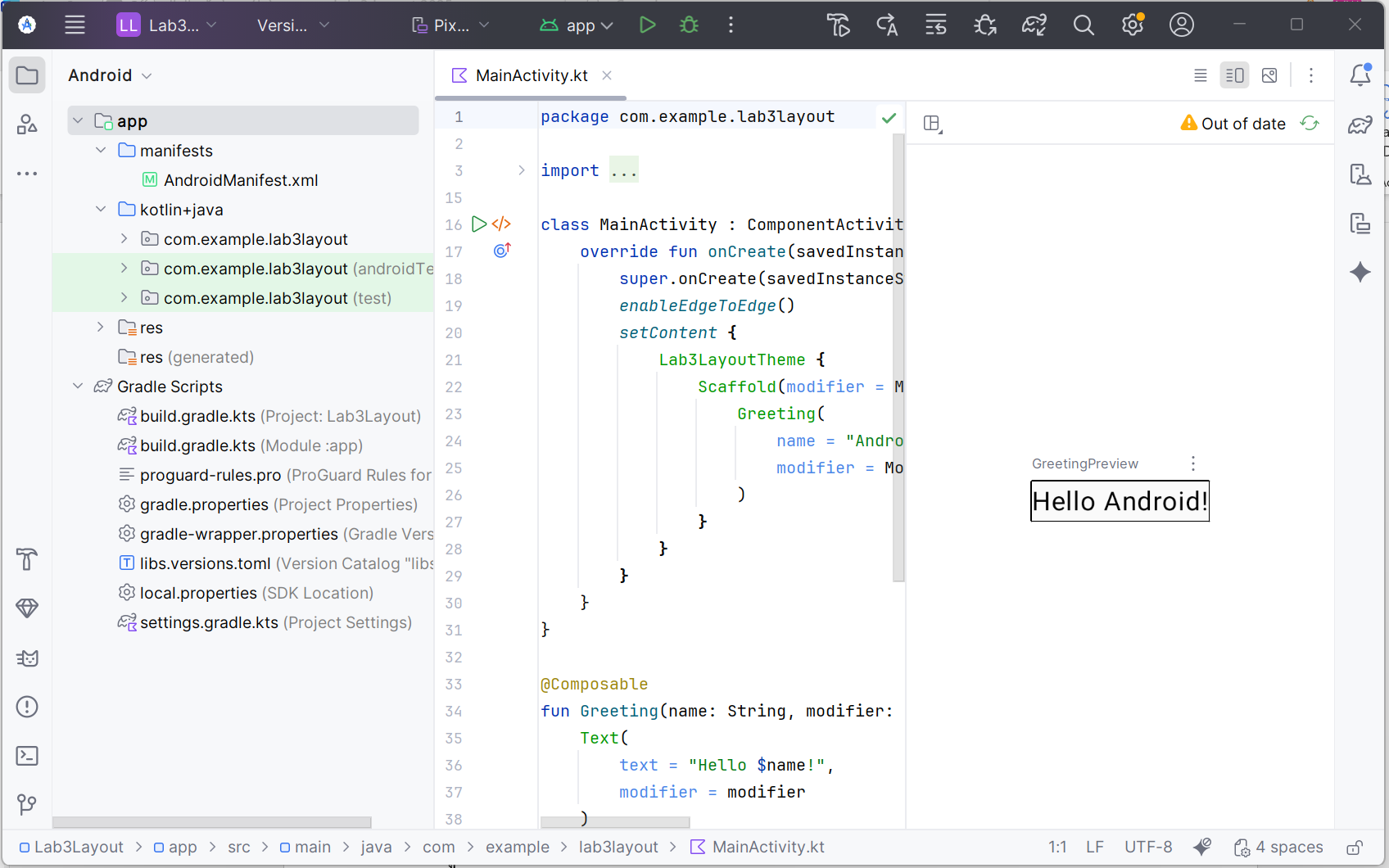
4. โปรแกรมสร้างไฟล์ MainActivity.kt ให้ ซึ่งใช้ในการเขียนคำสั่งออกแบบ Layout โดยสามารถ Preview หน้าจอที่ออกแบบโดยให้คลิกที่  (Split)



จากนั้นให้คลิกที่ Build & Refresh



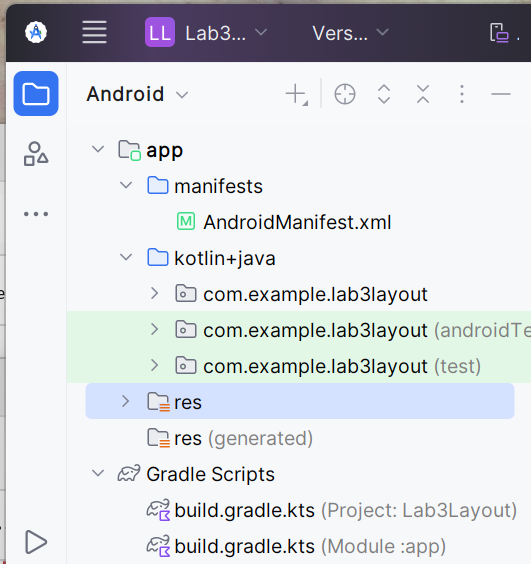
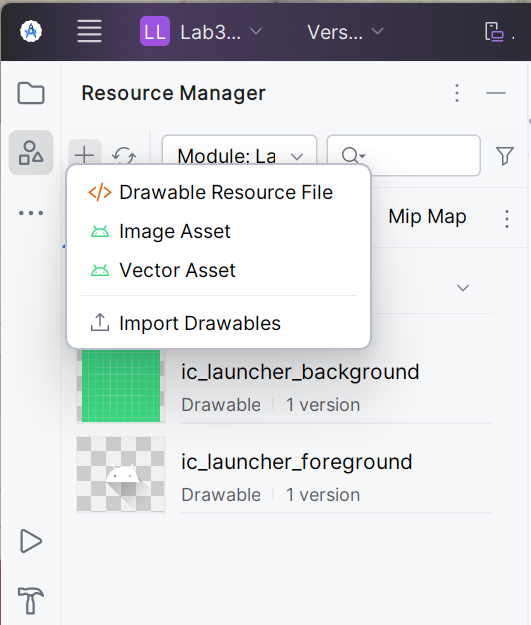
ในส่วนของ Preview จะแสดงข้อความ Hello Android แสดงว่าสามารถแสดงผลลัพธ์ของหน้าจอการทำงานได้แล้ว

****

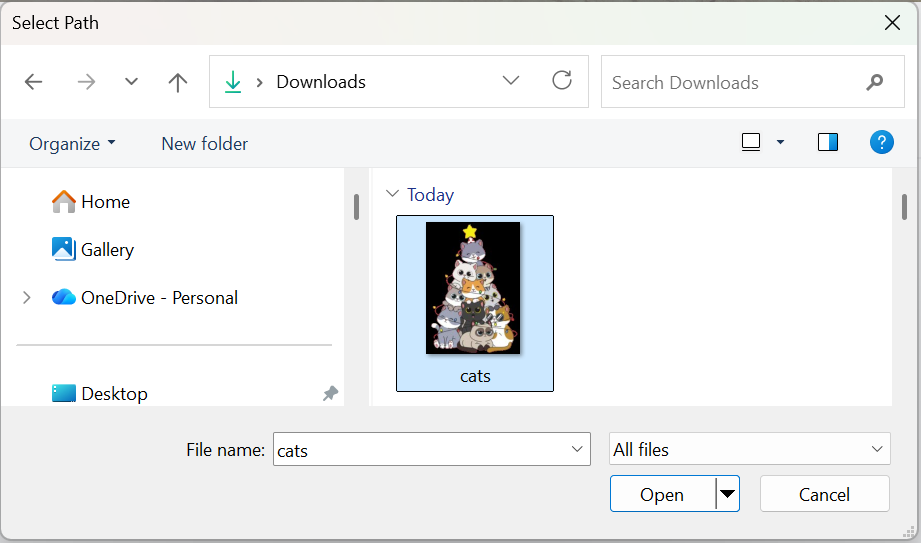
**วิธีการเพิ่มรูปภาพ**

การเพิ่มรูปภาพโดยนำรูปภาพไปเก็บที่ drawable โดยมีขั้นตอนดังนี้

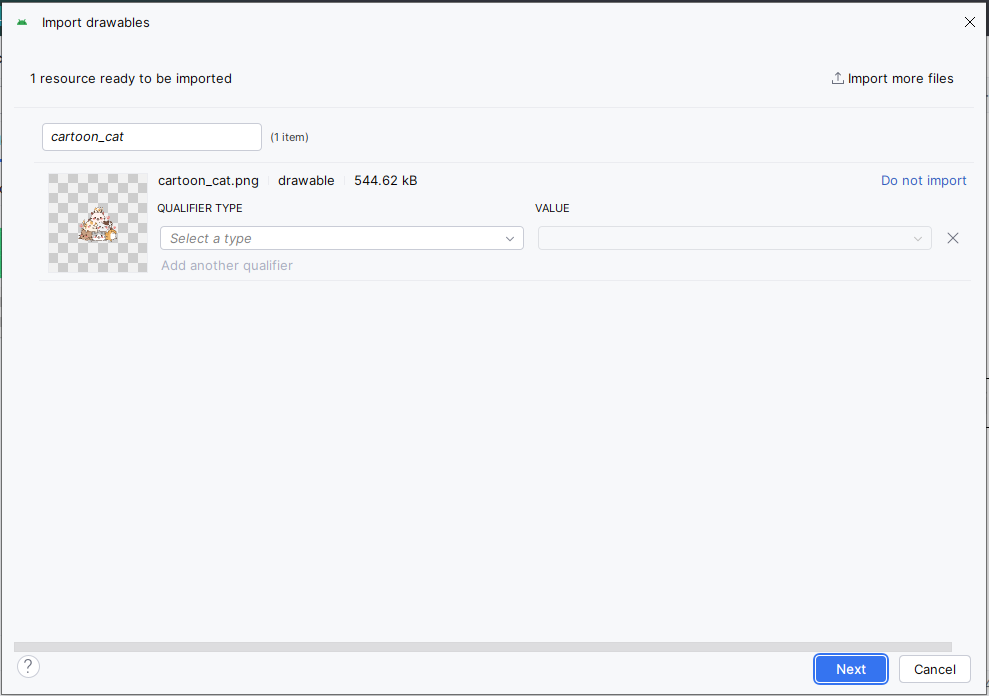
1. ให้คลิกที่  (**Resource Manager)** แล้วเลือกที่ **+ (Add resources to the module) -> Import Drawables**

** **

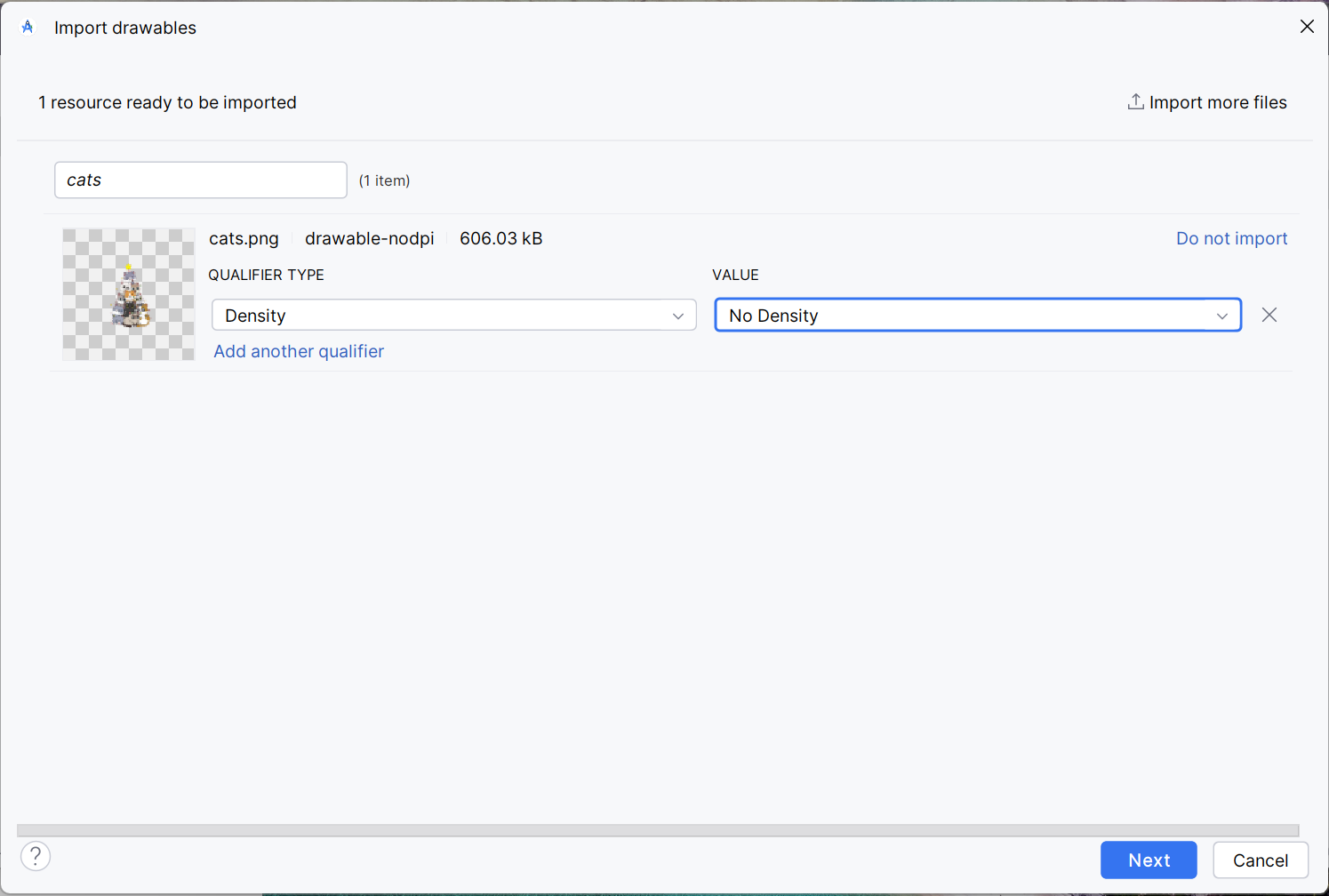
**2. จากนั้นจะแสดง** dialog ให้เลือกรูปภาพที่ต้องการเพิ่มตาม path ที่เก็บไฟล์รูปภาพ แล้วคลิก OK



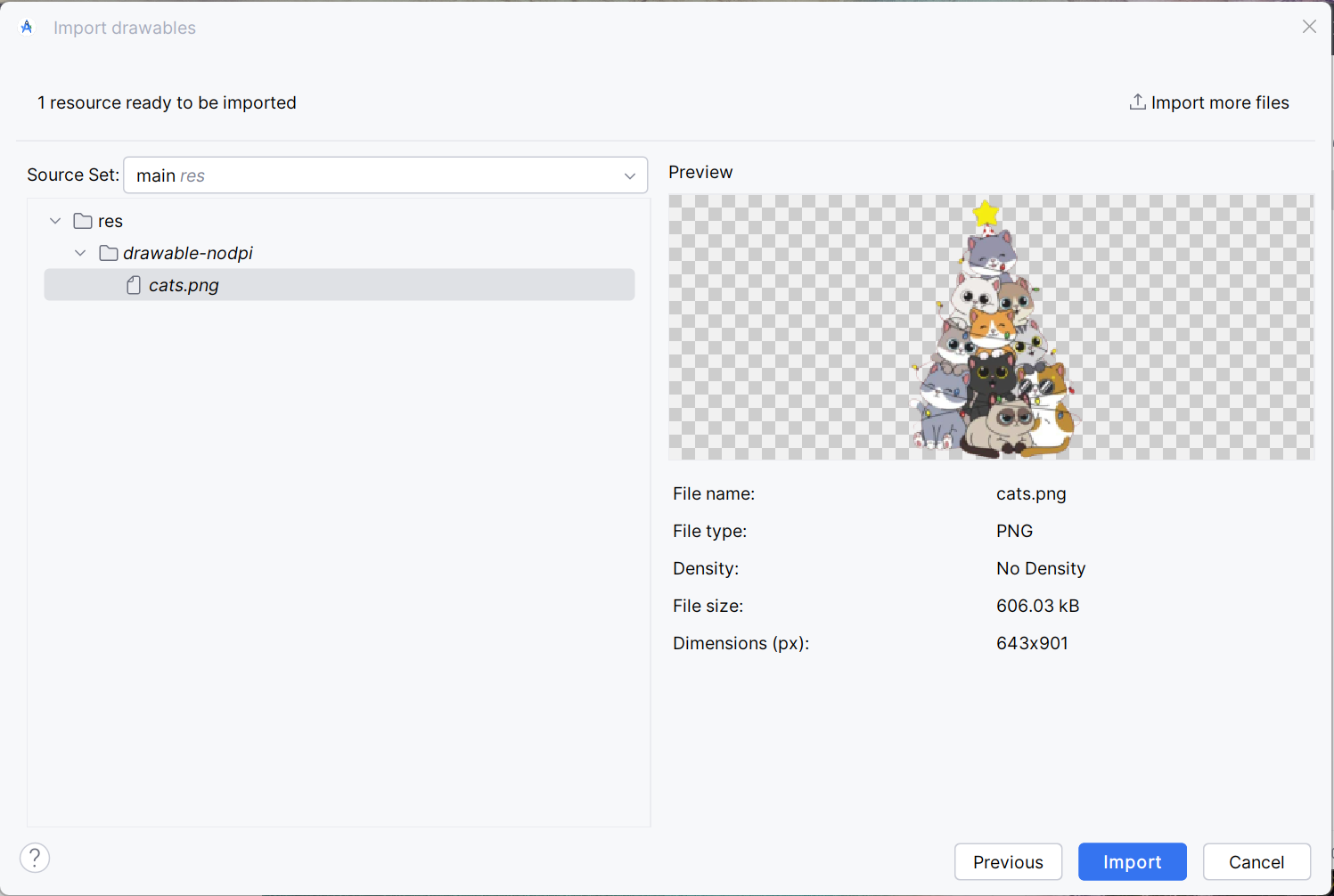
3. ถัดมาจะแสดงรูปภาพที่เราเลือก



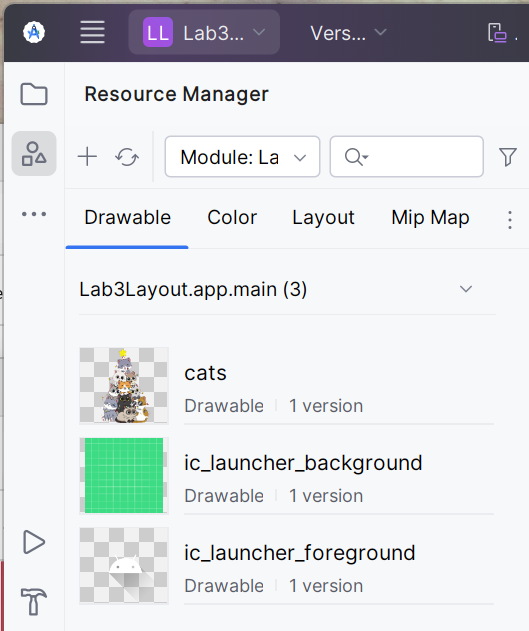
4. ให้กำหนด Qualifier Type และ Value ตามภาพด้านล่าง แล้วคลิกที่ Next



5. ถัดมาแสดงรายละเอียดของภาพและ path ที่เก็บไฟล์ จากนั้นคลิกที่ Import



6. รูปภาพที่ต้องการจะแสดงที่ **Resource Manager ถือว่าเพิ่มรูปภาพเสร็จแล้ว**



**การจัด Layout โดยใช้ Jetpack Compose**

ในปฏิบัติการนี้ทำการทดสอบการจัดรูปแบบหน้าจอ (UI Layout) ของ element ต่างๆ โดยควบคุมรูปแบบและการจัดวางด้วย Layout ใน Jetpack Compose ซึ่งประกอบด้วย 4 ประเภทดังนี้

1. Row Layout สำหรับการจัดวางองค์ประกอบในแนวนอน
2. Column Layout สำหรับการจัดวางองค์ประกอบในแนวตั้ง
3. Box Layout สำหรับการจัดวางองค์ประกอบแบบซ้อนทับกัน หรือกำหนดตำแหน่งอิสระ
4. ConstraintLayout สำหรับการจัดวางที่ซับซ้อน โดยกำหนดความสัมพันธ์ระหว่างองค์ประกอบต่าง ๆ

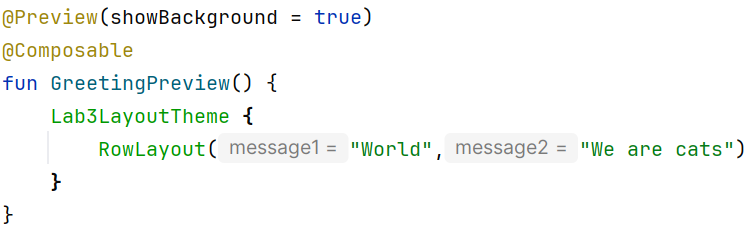
โดยรายละเอียดการใช้งาน Layout แต่ละประเภท ดังนี้

**1. การใช้ Row Layout**

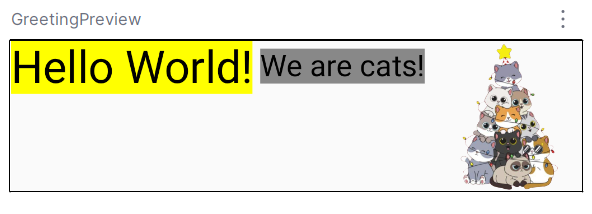
ให้พิมพ์คำสั่งต่อไปนี้ ที่ไฟล์ MainActivity



จากนั้นให้แก้ไขคำสั่งในส่วนของ @Preview เพื่อแสดงผลลัพธ์ของคำสั่งด้านบน



ผลลัพธ์เมื่อคลิกที่  ดังนี้



หากต้องการแสดงผลลัพธ์ที่ Emulator ให้แก้ไขคำสั่งในส่วนของ setContent โดยเรียกฟังก์ชัน RowLayout() ดังนี้



แล้วคลิกที่ปุ่ม  เพื่อ Run ผลลัพธ์แสดงที่ Emulator จะแสดงผลลัพธ์ดังนี้

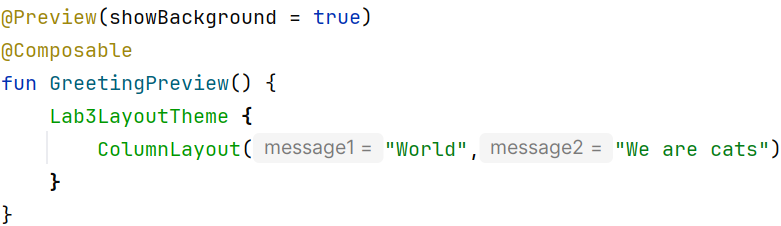


**2. การใช้ Column Layout**

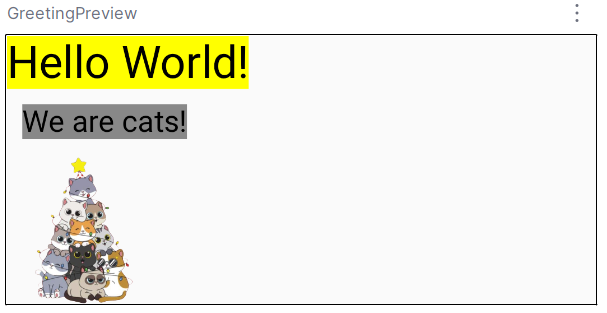
ให้คัดลอกคำสั่งจากตัวอย่างของ Row แล้วเปลี่ยนคำสั่งดังนี้



จากนั้นให้แก้ไขคำสั่งในส่วนของ @Preview เพื่อแสดงผลลัพธ์ของคำสั่งด้านบน



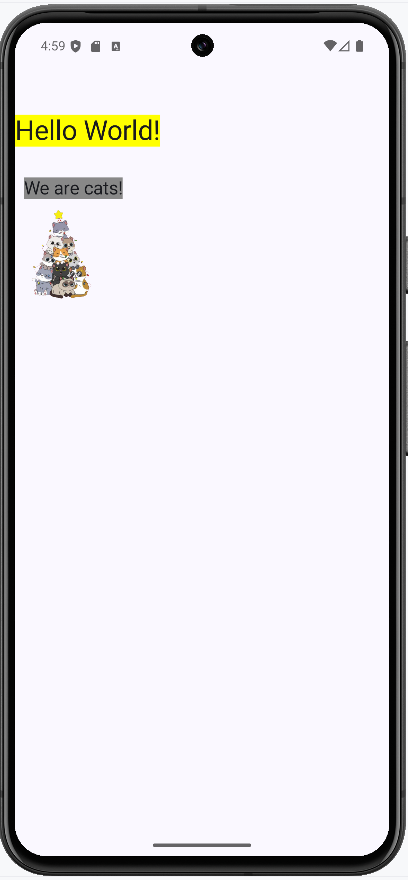
ผลลัพธ์ ดังนี้



หากต้องการแสดงผลลัพธ์ที่ Emulator ให้แก้ไขคำสั่งในส่วนของ setContent โดยเรียกฟังก์ชัน ColumnLayout() ดังนี้



แล้วคลิกที่ปุ่ม  เพื่อ Run ผลลัพธ์แสดงที่ Emulator จะแสดงผลลัพธ์ดังนี้



**3. การใช้ Box Layout**

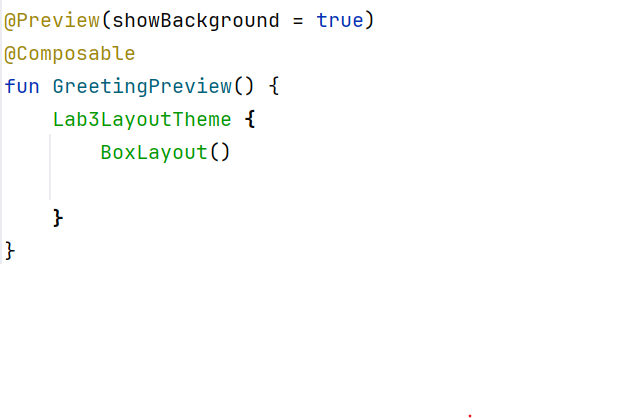
Box เป็นการอ้างอิงตำแหน่งของ View หรือ Parentโดยใช้ modifier.align(Alignment.ตำแหน่ง) ค่าของตำแหน่งที่สามารถกำหนดได้ คือTopStart, TopCenter, TopEnd, CenterStart, Center, CenterEnd, BottomStart, BottomCenter, และBottomEnd โดยให้พิมพ์คำสั่ง ดังนี้

\*\* ตอนแก้ Error ของ Alignment ต้อง import ตัว androidx.compose.ui.Alignment

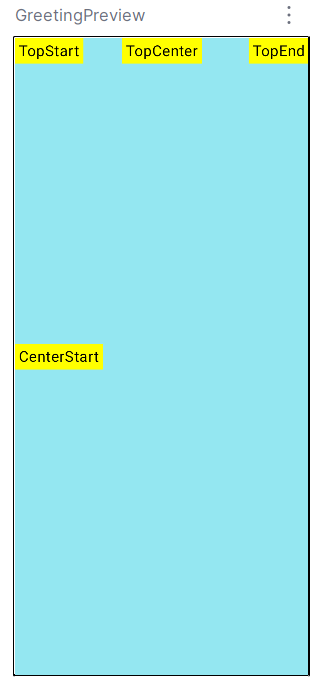
****

****

จากนั้นให้แก้ไขคำสั่งในส่วนของ @Preview เพื่อแสดงผลลัพธ์ของคำสั่งด้านบน

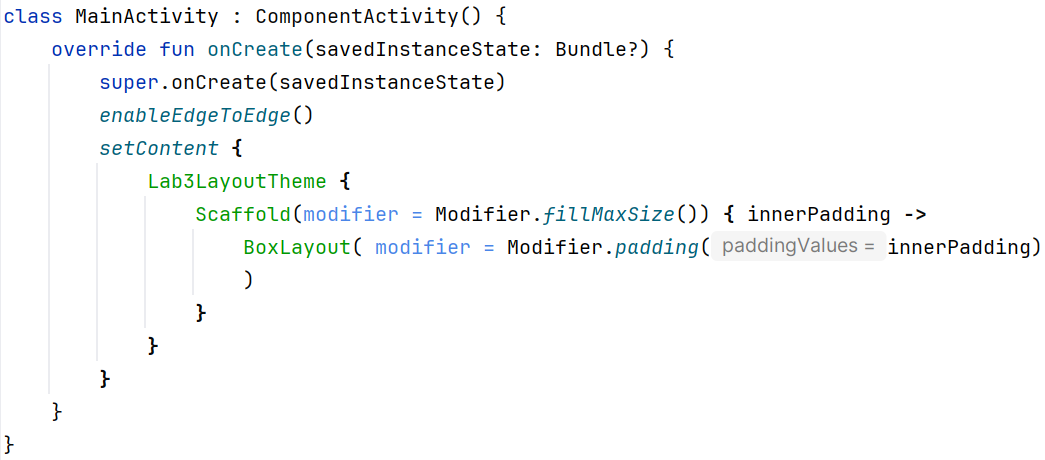
****

ผลลัพธ์ ดังนี้

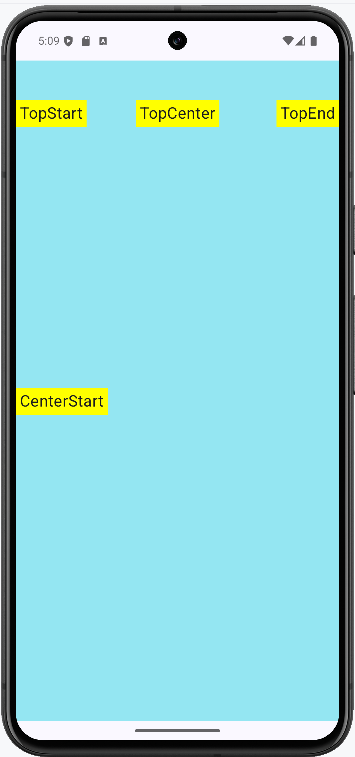
****

หากต้องการแสดงผลลัพธ์ที่ Emulator ให้แก้ไขคำสั่งในส่วนของ setContent โดยเรียกฟังก์ชัน BoxLayout()

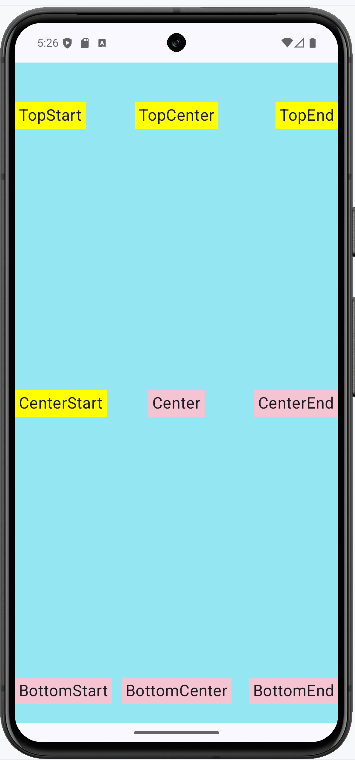
ดังนี้



แล้วคลิกที่ปุ่ม  เพื่อ Run ผลลัพธ์แสดงที่ Emulator จะแสดงผลลัพธ์ดังนี้

****

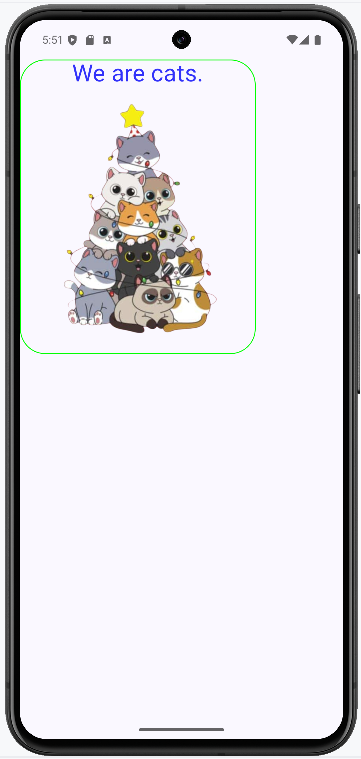
งาน Lab ข้อ 1: ให้แก้ไขคำสั่งโดยให้แสดงกล่องข้อความสีชมพูตามตำแหน่งของผลลัพธ์ที่แสดงดังภาพด้านล่าง

****

ตัวอย่างที่ 2 การใช้งาน Box Layout เป็นการกำหนด Element ใส่ไว้ใน Box แล้วจัดตำแหน่งของ Element ภายใน Box ดังคำสั่งต่อไปนี้

****

เมื่อ Run แล้วจะได้ผลลัพธ์ ดังนี้



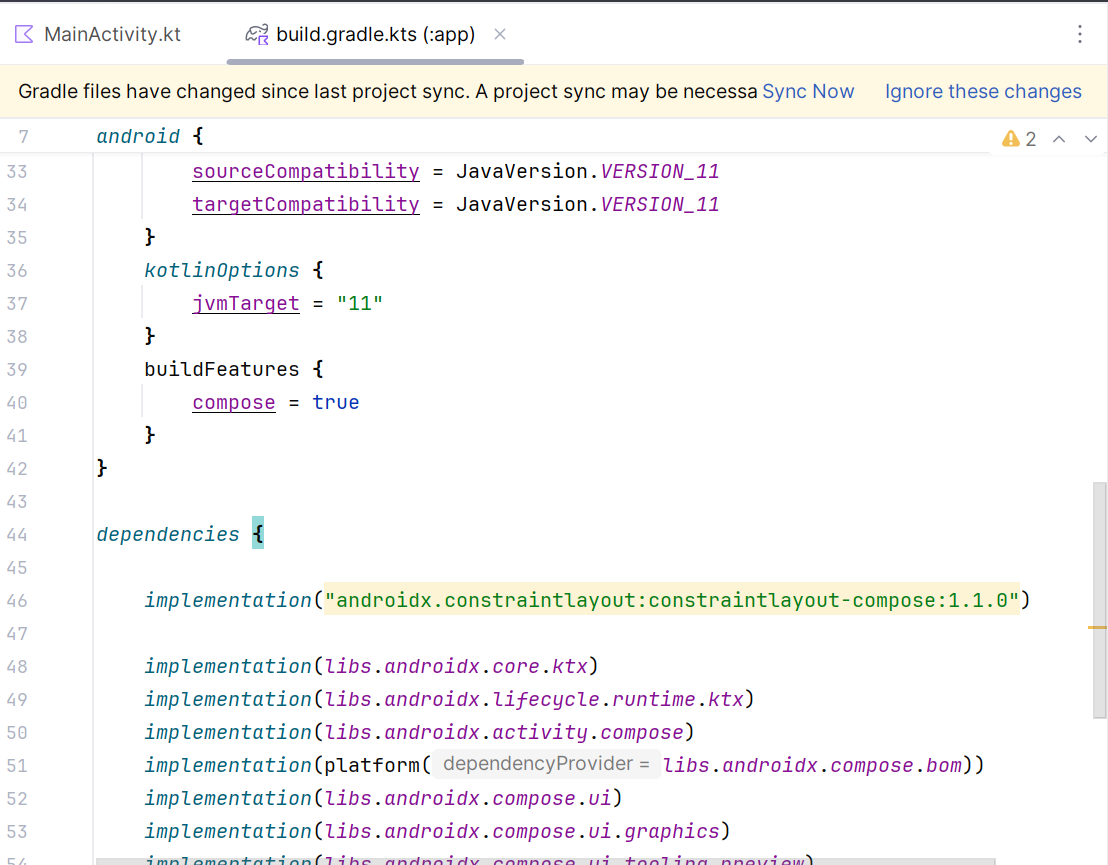
**4. ConstraintLayout**

ConstraintLayout เป็นการอ้างอิงตำแหน่งของ Element จากตำแหน่งของ Element อื่น การใช้ ConstraintLayout

ต้องเพิ่มคำสั่งที่ dependency ในไฟล์ build.gradle.kys(Modules:app) แล้วให้คลิกที่ Sync Now

คำสั่ง ดังนี้

implementation("androidx.constraintlayout:constraintlayout-compose:1.1.0")

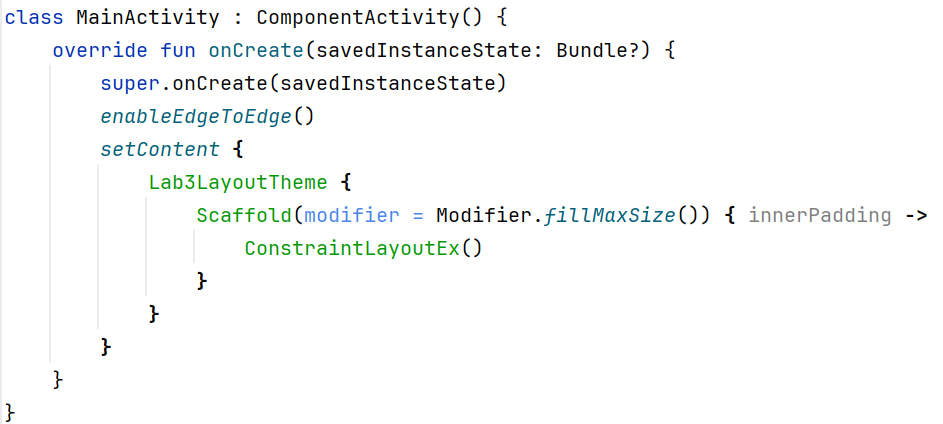
**

จากนั้นกลับมายัง MainActivity.kt แล้วพิมพ์คำสั่งดังนี้





จากนั้นให้แก้คำสั่งในส่วนของ setContent โดยให้เรียกฟังก์ชัน ConstraintLayoutEx() แล้วคลิกที่ปุ่ม  เพื่อ Run



ผลลัพธ์แสดงที่ Emulator จะแสดงผลลัพธ์ดังนี้

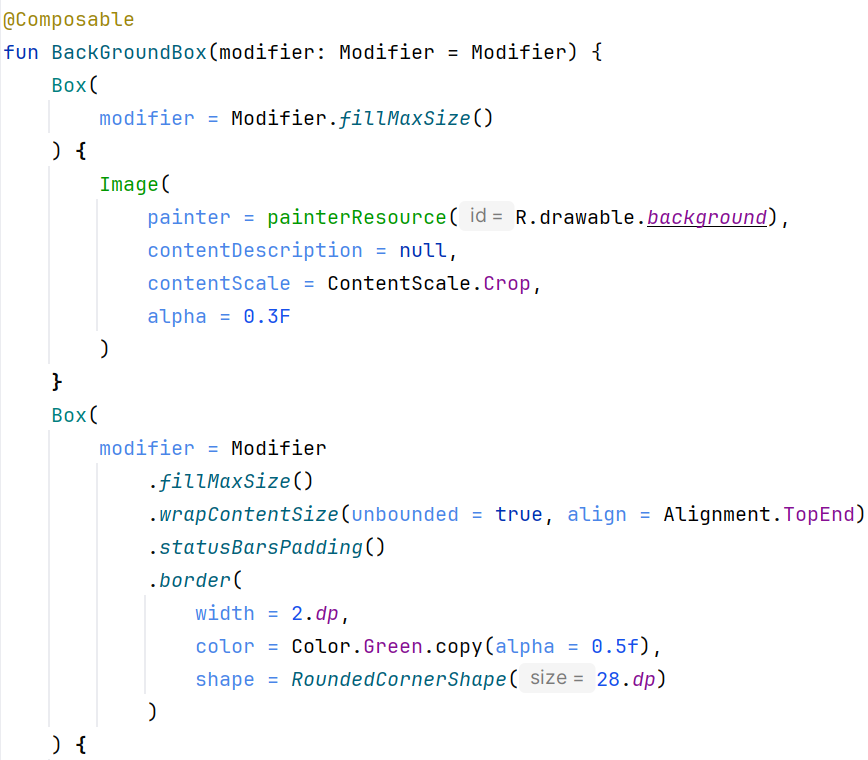
**

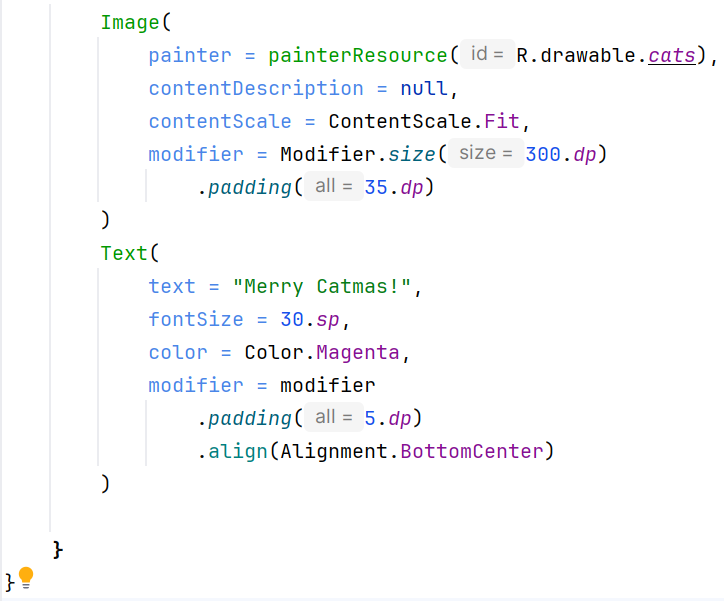
งาน Lab ข้อ 2: ให้เขียนคำสั่งเพื่อให้ข้อความและรูปภาพอยู่ตรงกลาง ดังแสดงผลลัพธ์ตามภาพด้านล่าง



**การใส่ภาพพื้นหลังให้แอปพลิเคชัน**

ก่อนการใส่ภาพพื้นหลังให้เพิ่มรูปภาพก่อน ดังที่ได้อธิบายไว้ในหัวข้อ “วิธีการเพิ่มรูปภาพ” โดยให้เพิ่มไฟล์ภาพ background.jpg จากนั้นให้เขียนคำสั่ง ดังนี้



****

จากนั้นให้แก้คำสั่งในส่วนของ setContent โดยให้เรียกฟังก์ชัน BackGroundBox() แล้วคลิกที่ปุ่ม  เพื่อ Run



ผลลัพธ์แสดงที่ Emulator จะแสดงผลลัพธ์ดังนี้



งาน Lab ข้อ 3: ให้เขียนคำสั่งเพื่อแสดงข้อความว่า “Happy Cat!” เป็นตัวอักษรสีแดง ซึ่งอยู่ด้านบนของรูปภาพ และตัวหนังสือและรูปภาพมีขนาดใหญ่ขึ้น และอยู่ตรงกลางของหน้าจอ โดยให้ผลลัพธ์แสดงตามภาพด้านล่าง



🙠🙡🙣🙠🙡🙣🙠🙡🙣🙠🙡🙣