

ใบงานการทดลองที่ 3

เรื่อง Flutter

1. จุดประสงค์

1. สามารถติดตั้ง Flutter SDK บนคอมพิวเตอร์ของตนได้
2. สามารถเปิดใช้งาน Flutter SDK บน VScode ได้
3. สามารถเชื่อมต่อ Android SDK กับ VScode ได้

2. ทฤษฎี

บทที่ 2.2 การใช้ Android Studio สำหรับ Flutter

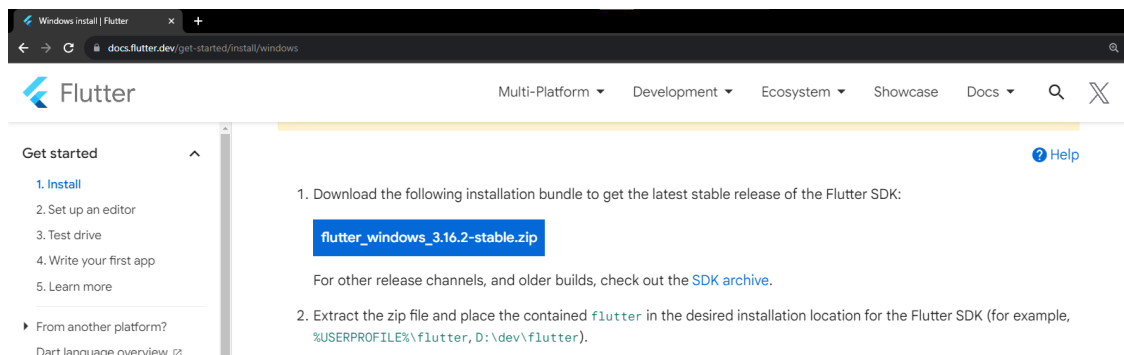
2.2.1 เกี่ยวกับ Android Studio

Android Studio เป็นเครื่องมือในการพัฒนาแอปสำหรับระบบ Android โดยตรง และสามารถติดตั้งปลั๊กอินเพื่อใช้ร่วมกับ Flutter ได้ ซึ่งเครื่องมือเหล่านี้ต่างก็เป็นของ Google ทั้งหมด ดังนั้น จึงสามารถใช้งานร่วมกันได้อย่างไม่มีปัญหา และถือว่าเป็นวิธีมาตรฐานที่นิยมใช้กันเป็นส่วนใหญ่ การใช้ Android Studio เราต้องดำเนินการเพิ่มเติมในหลายขั้นตอนกว่าจะใช้งานได้ ทั้งนี้ก็เพราะ ไม่ได้ถูกสร้างมาสำหรับ Flutter โดยตรงนั่นเอง ซึ่งการติดตั้งและเซตค่า Android Studio เพื่อใช้งานร่วมกับ Flutter มีรายละเอียดดังต่อไปนี้

2.2.2 การติดตั้ง Flutter SDK

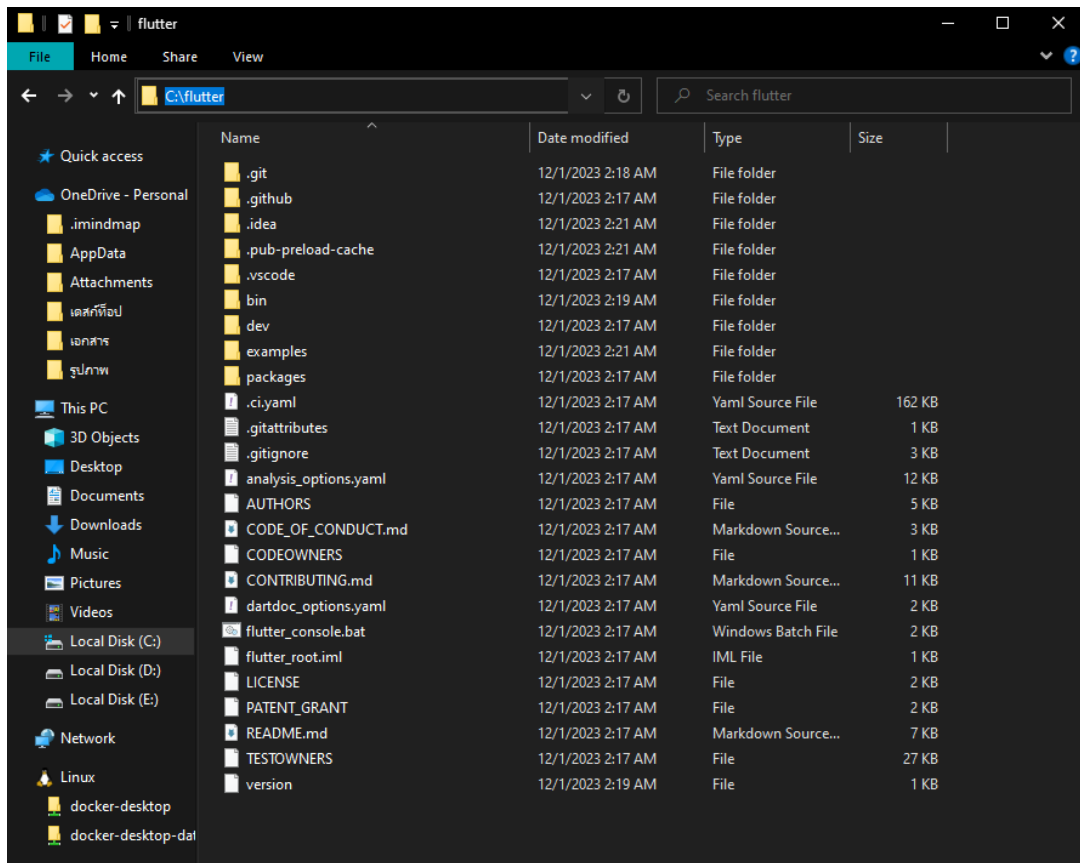
Flutter SDK คือ ชุดเครื่องมือสำหรับการพัฒนาแอปด้วย Flutter ซึ่งในขณะที่เขียนหนังสือเล่มนี้มันไม่ได้อยู่ใน Android Studio แต่เราต้องติดตั้งแยกต่างหาก ดังขั้นตอนต่อไปนี้

1. สามารถดาวน์โหลด Flutter SDK ได้ที่ <https://docs.flutter.dev/get-started/install/windows> ตามรูปที่ 2.1



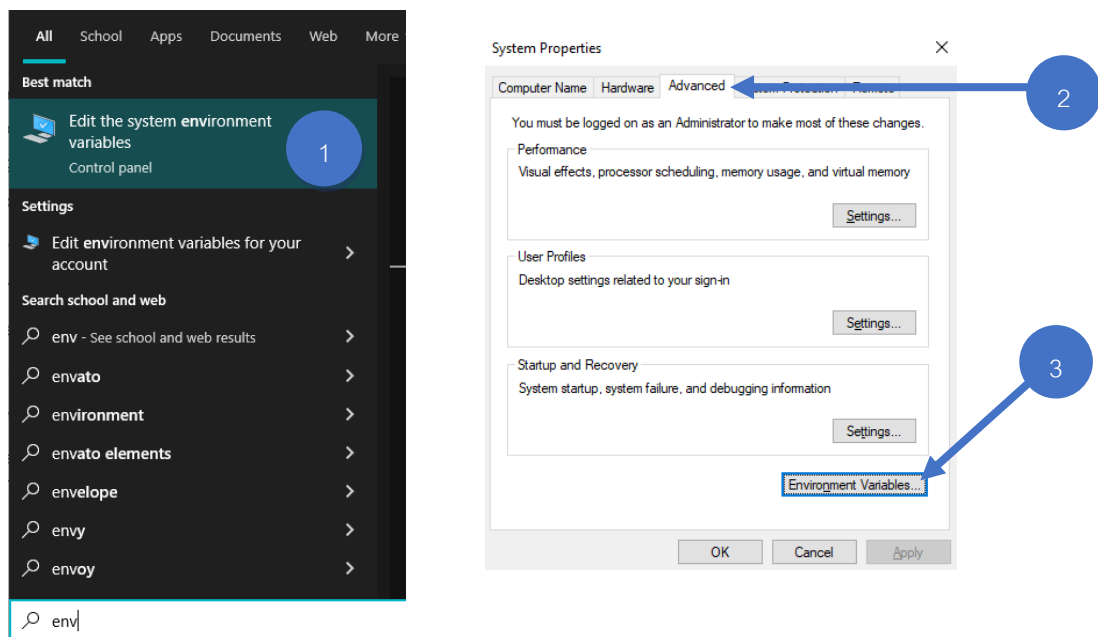
รูปที่ 2.1 Flutter SDK

2. ไฟล์ที่ดาวน์โหลดมาจะอยู่ในแบบ .zip เราสามารถแตกไฟล์ไปเก็บไว้ที่ไดรฟ์ C:\ ได้ หลังจากการแตกไฟล์จะได้เป็น C:\flutter ตามรูปที่ 2.2



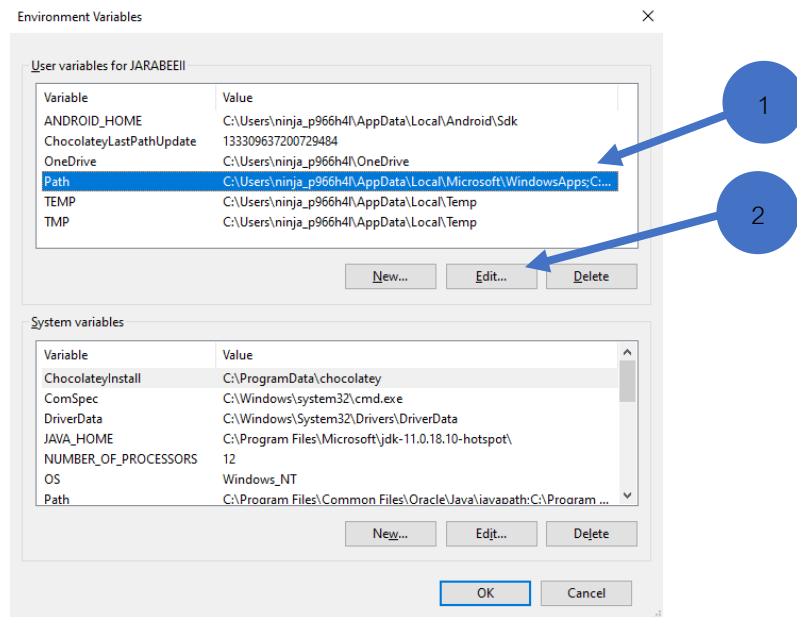
รูปที่ 2.2 Flutter ลงบนโดรฟ์

3. พิมพ์ที่ search bar ว่า env เลือก Edit environment variables for your account ในหน้า System Properties เลือกหัวข้อ Advanced เลือกไปที่ Environment Variables ตามรูปที่ 2.3



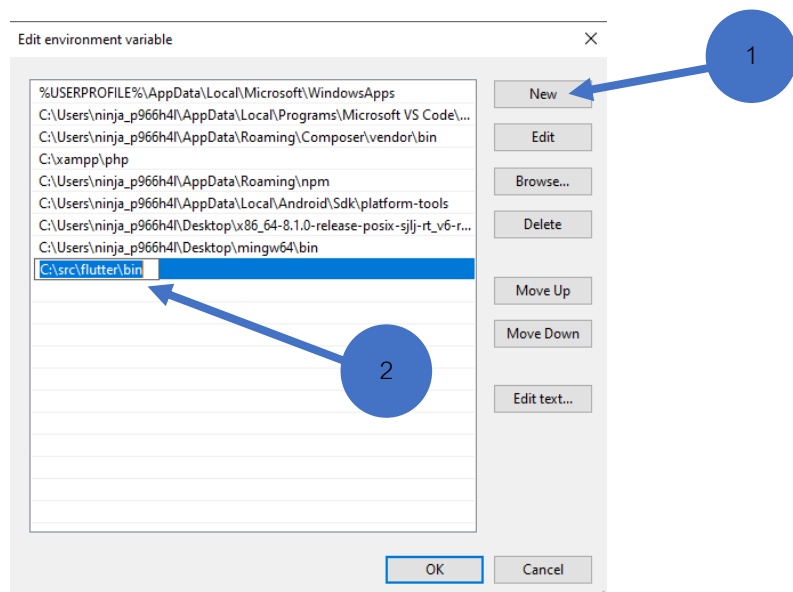
รูปที่ 2.3 การลงพาสตัวแปร

4. ในหน้า Environment Variables ส่วนของ User variables ให้เลือก Path และกด Edit ตามรูปที่ 2.4



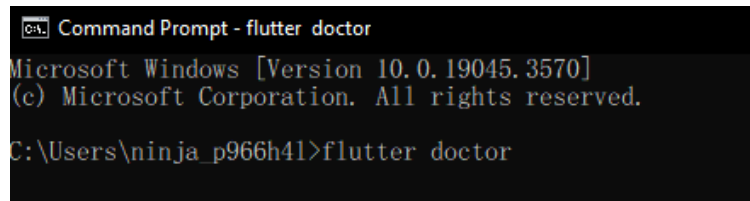
รูปที่ 2.4 หน้า Environment Variables

5. ในหน้า Edit environment variable ให้กด New หลังจากนั้นใส่ path ที่เราวาง Flutter SDK ไว้ ตามในตัวอย่างคือ C:\flutter และให้ใส่ \bin ตามหลังเพื่อเรียกใช้ข้อมูลได้ จะได้เป็น C:\flutter\bin ถือว่าเสร็จสิ้นกระบวนการการตั้งค่า PATH ดังรูปที่ 2.5



รูปที่ 2.5 Edit Environment variable

6. ใช้คำสั่ง flutter doctor ใน Command Prompt ตามรูปที่ 1.7 เพื่อเป็นการตรวจสอบความต้องการของ Flutter ว่าต้องการให้ติดตั้งโปรแกรมใดเพิ่มเติม ดังรูปที่ 2.6



```
C:\Users\ninja_p966h4l>flutter doctor
```

[!] Android toolchain - develop for Android devices

- Android SDK at D:\Android\sdk

X Android SDK is missing command line tools; download from <https://goo.gl/XxQghQ>

- Try re-installing or updating your Android SDK,

visit <https://docs.flutter.dev/setup/#android-setup> for detailed instructions.

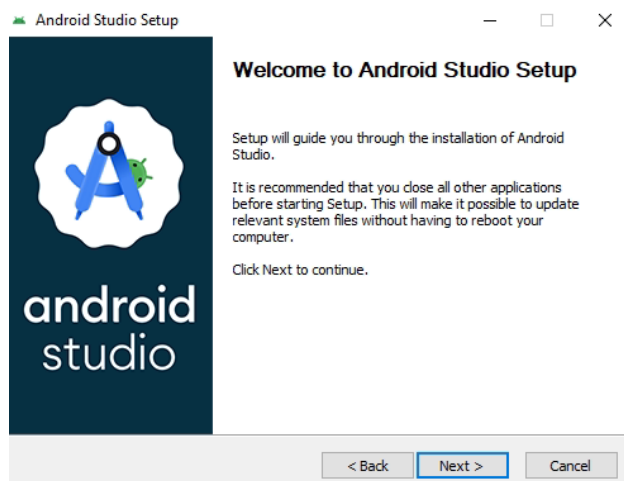
รูปที่ 2.6 Flutter Doctor

2.2.3 การติดตั้ง Android Studio

Android Studio คือโปรแกรมประเภท IDE ที่ใช้เขียนและทดสอบแอป ถือเป็นเครื่องมือตัวหลักที่จำเป็นสำหรับ Flutter แต่หากผู้อ่านได้ติดตั้งเอาไว้แล้ว ก็สามารถข้ามหัวข้อนี้ได้เลย ซึ่งขั้นตอนการติดตั้งพอสังเขปของ Android Studio มีดังนี้

1. สามารถดาวน์โหลด Android Studio ได้ที่ <https://developer.android.com/studio> หลังจากเสร็จเรียบร้อยแล้ว ก็ให้เริ่มติดตั้งโดยดับเบิลคลิกไฟล์ที่ดาวน์โหลดมา

2. ขั้นตอนที่สำคัญในระหว่างการติดตั้งเป็นตามรูปที่ 2.7



รูปที่ 2.7 ติดตั้ง Android Studio

ชั้น)

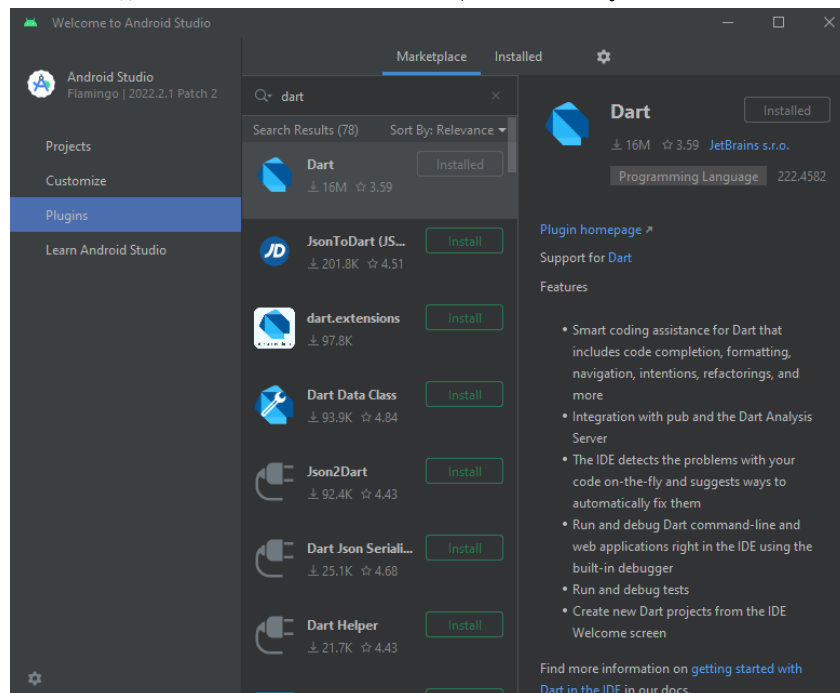
3. หากเราเปิดเข้าสู่ Android Studio ได้แล้ว จะขึ้นหน้าจอ Welcome ตามรูปที่ 2.8 (ต่างกันตามแต่เวอร์



รูปที่ 2.8 หน้า Welcome

2.2.4 การติดตั้งปลั๊กอินของ Dart และ Flutter

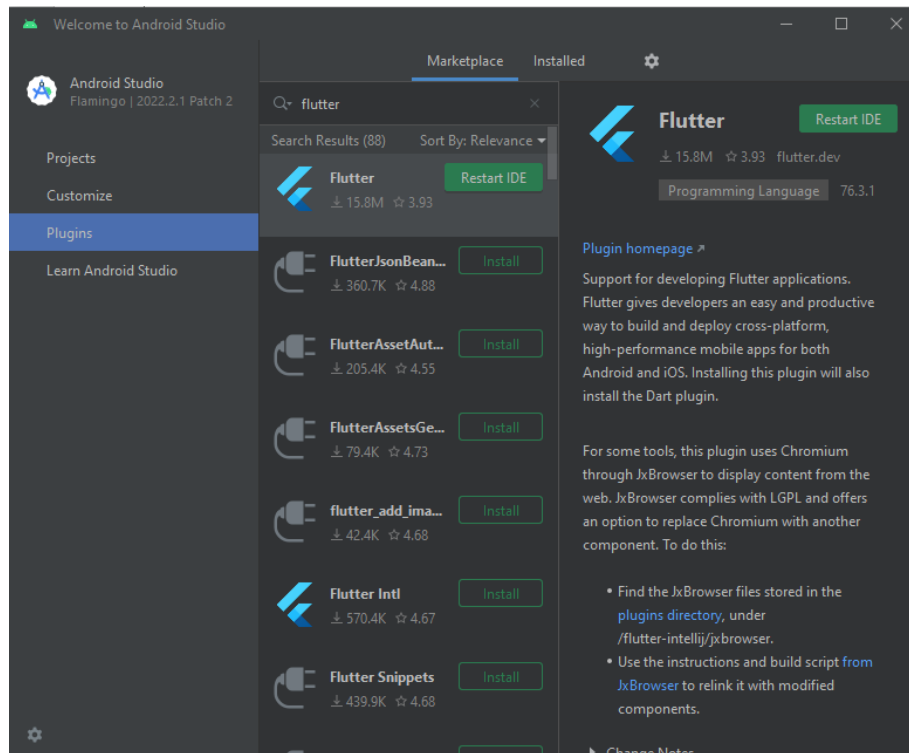
1. เปิดเข้าสู่ Android Studio แล้วคลิกเมนู Plugins
2. เลือกแท็บ Marketplace
3. เราต้องติดตั้งปลั๊กอินของ Dart ก่อน Flutter แล้วพิมพ์คำว่า dart ลงในช่องค้นหา
4. เมื่อปรากฏรายการให้คลิก ที่ Dart แล้วคลิกปุ่ม Install ตามรูปที่ 2.9



รูปที่ 2.9 ติดตั้ง Dart

Android

5. ต่อไปให้ค้นหาปลั๊กอิน Flutter แล้วคลิกปุ่ม Install แล้วหลังจากติดตั้ง จำเป็นต้องเริ่มต้นโปรแกรม Studio ใหม่ ทั้งนี้เมื่อติดตั้งเสร็จแล้วจะมีปุ่มหรือไอคอนล็อกขึ้นมาสอบถาม ก็ให้เลือก Restart IDE



รูปที่ 2.10 ติดตั้ง Flutter

2.2.5 การสร้างโปรเจกต์ใหม่

เมื่อจะเข้าสู่ขั้นตอนการสร้างแอป เราต้องเริ่มจากการสร้างโปรเจกต์ใหม่ก่อน ซึ่งมีวิธีการพอสังเขปดังนี้

1. หลังจากเปิดเข้าสู่ Android Studio เมื่อปรากฏหน้าจอ Welcome คลิกที่แท็บ Projects จากนั้นคลิกปุ่ม New Flutter Project

2. ในหน้าจอถัดไป ที่แท็บด้านซ้าย ให้เลือกชนิดโปรเซสเซอร์ Flutter และช่อง Folders path ให้เรา กำหนดตำแหน่งที่ได้ติดตั้ง Flutter SDK เอาไว้ ซึ่งตามการอ้างอิงในหัวข้อที่ผ่านมาคือ C:\flutter (ถ้าติดตั้งไว้ที่ตำแหน่งอื่นให้กำหนดค่าไปตามนั้น) และสุดท้ายก็คลิกปุ่ม Next

3. หน้าจอถัดไป ตรงช่อง Project name ให้เราระบุชื่อโปรเจกต์ (ชื่อแอป) และช่อง Project location ให้ระบุตำแหน่งที่จะจัดเก็บโปรเจกต์

- การตั้งชื่อโปรเจกต์จะใช้รูปแบบ app<chapter><0x> นั่นคือ จะขึ้นต้นด้วยคำว่า app แล้วตามด้วยหมายเลขบท แล้วต่อท้ายด้วยลำดับตัวอย่างในบทนั้น (ถ้าเป็นเลขหลักเดียวให้ขึ้นต้นด้วยเลข 0) เช่น
- โปรเจกต์ของบทที่ 6 ตัวอย่างลำดับที่ 1 จะตั้งชื่อเป็น app601

- โปรเจกต์ของบทที่ 6 ตัวอย่างลำดับที่ 2 จะตั้งชื่อเป็น app602

- โปรเจกต์ของบทที่ 7 ตัวอย่างลำดับที่ 1 จะตั้งชื่อเป็น app701

- โปรเจกต์ของบทที่ 7 ตัวอย่างลำดับที่ 2 จะตั้งชื่อเป็น app702

- ตำแหน่งในการจัดเก็บโปรเจกต์จะเก็บไว้ที่ C:\Users\ชื่อผู้ใช้ Flutter project_.name เช่น โปรเจกต์ app601 จะจัดเก็บไว้ที่ C:\Users\ชื่อผู้ใช้ Flutter\app601

4. จากขั้นตอนในข้อ 3 หลังจากที่เรากำหนดข้อมูลจนครบแล้ว เมื่อคลิกปุ่ม Finish ก็จะเข้าสู่หน้าจอหลักของ Android Studio ในมุมมอง Project

หลังจากที่เปิดเข้าสู่โปรเจกต์แล้ว หากต้องการปิดหรือออกจากโปรเจกต์ ให้เลือกที่เมนู File > Close Project ต่อไปก็จะย้อนกลับไปยังหน้าจอ Welcome และถ้าเราต้องการปิดโปรแกรม Android studio ก็ปิดที่หน้าจอนั้นได้เลย หรือหากจะปิดขณะอยู่ในมุมมองโปรเจกต์ ก็อาจเลือก ที่เมนู File > Exit

2.2.6 การติดตั้งเครื่องจำลอง (AVD/Emulator)

หากเราใช้ Android Studio บนระบบ Windows จะทดสอบการใช้งานได้เฉพาะ Web และแอปสำหรับ Android เท่านั้น ซึ่งโดยส่วนใหญ่แล้ว หากแอปสำหรับ Android ใช้งานได้ แอปสำหรับแพลตฟอร์มอื่น ๆ ก็มักจะใช้งานได้เช่นกัน (แต่ไม่ใช่ทุกกรณีเสมอไป) ดังนั้นถ้าไม่มีเครื่อง Mac สำหรับทดสอบแอปสำหรับ iOS ก็สามารถทดสอบแค่แอปสำหรับ Windows ไปก่อน แล้วเมื่อมีเครื่อง Mac พร้อมใช้งานก็ค่อยไปทดสอบแอปสำหรับ iOS ในภายหลังก็ได้

อย่างไรก็ตาม ก่อนที่เราจะทดสอบแอปได้ จำเป็นต้องติดตั้งเครื่องจำลอง (Android Virtual Device: AVD or Emulator) เพิ่มเข้ามาใน Android Studio เสียก่อน โดยมีแนวทางดังนี้

1. ถ้าเราอยู่ที่หน้าจอ Welcome (ยังไม่เปิดเข้าสู่โปรเจกต์) สามารถคลิกที่ปุ่ม 3 จุดตรงมุมขวาบน แล้วเลือก **Virtual Device Manager** หรือถ้าเราเปิดเข้าสู่มุมมองของโปรเจกต์แล้ว อาจเลือกวิธีใดวิธีหนึ่งคือ

- วิธีที่ 1 คลิกแท็บ **Device Manager** ที่ริมขอบด้านขวามือ
- วิธีที่ 2 เลือกที่เมนู **Tools > Device Manager**
- วิธีที่ 3 เลือกที่เมนู **View > Tool Windows > Device Manager**

2. ไม่ว่าเราจะเข้าสู่ Device Manager ด้วยวิธีใดดังที่กล่าวมา หลังจากหน้าจอนี้ปรากฏขึ้น ให้คลิกปุ่ม **Create device**

3. ในหน้าจอ Select Hardware ที่กรอบ Category ให้เราเลือกว่าจะติดตั้งเครื่องจำลองแบบ Phone หรือแบบ Tablet จากนั้นคลิกชื่อเครื่องจำลองที่ต้องการ (รายชื่อจะขึ้นกับ Category ที่เราเลือก) จากนั้นคลิกปุ่ม Next

4. ในหน้าจอถัดไปก็เลือก API Level (ถ้ามีหลายตัวเลือก ควรเลือกเวอร์ชันสูงสุดที่เสถียร)

5. ในหน้าจอถัดไปเป็นการตั้งชื่อเครื่องจำลอง เราอาจตั้งชื่อตามต้องการหรือจะใช้ชื่อที่ Android studio ตั้งเป็นค่าที่พอลต์มาให้ล่วงหน้าก็ได้ แล้วคลิกปุ่ม Finish จากนั้นเครื่องจำลองที่เราเลือกก็จะถูกดาวโหลดมาติดตั้งให้โดยอัตโนมัติ

เราสามารถติดตั้งเครื่องจำลองชนิดอื่นเพิ่มเติมได้ โดยเฉพาะอย่างยิ่งเครื่องจำลองที่มีขนาดหน้าจอแตกต่างกันออกไป ซึ่งก็ย้อนกลับไปเริ่มที่ขั้นตอนเดิมอีกครั้ง ทั้งนี้ เราสามารถติดตั้งเครื่องจำลองจำนวนเท่าไรก็ได้ แต่ไม่ควรมากเกินไป และให้ครอบคลุมทั้งแบบ Phone และ Tablet

2.2.7 การเปิดเครื่องจำลอง

ก่อนที่เราจะใช้งานเกี่ยวกับแอป เช่น รันทดสอบ จะต้องเปิดเครื่องจำลองที่ได้ติดตั้งเอาไว้ขึ้นมาทำงานก่อน ซึ่งการเปิดเครื่องจำลองอาจเลือกใช้วิธีใดวิธีหนึ่งคือ

1. เลือกจากรายชื่อเครื่องจำลองที่เราได้ติดตั้งเอาไว้ โดยคลิกเปิด Dropdown ที่แถบทูลบาร์ด้านบน แล้วเลือกเครื่องที่ต้องการเปิด แต่ถ้าเปิด Dropdown แล้วไม่ปรากฏรายชื่อเครื่องจำลอง ก็ให้คลิกที่รายการ Refresh (ภาพที่แล้วด้านซ้ายมือ)

2. เปิด Device Manager ด้วยวิธีใดก็ตามที่แนะนำไปในหัวข้อที่แล้ว ซึ่งเมื่อปรากฏเครื่องจำลองที่เราได้ติดตั้งเอาไว้ ก็ให้เราคลิกที่ไอคอนหัวลูกศรตรงรายชื่อเครื่องที่ต้องการเปิด (ภาพที่แล้วด้านขวามือ)

เมื่อเครื่องจำลองที่ถูกเปิดขึ้นมาแล้ว จะปรากฏอยู่เช่นนี้ ถึงแม้จะปิดโปรเจกต์เข้าสู่หน้าจอ Welcome เมื่อเปิดโปรเจกต์ใหม่ เครื่องจำลองก็ยังคงอยู่ จนกว่าเราจะออกจากโปรแกรม Android studio เครื่องจำลองจึงจะถูกปิดไปโดยอัตโนมัติ และนอกจากนี้ ยังมีหลายกรณีที่ที่น่าสนใจอื่นๆ เกี่ยวกับเครื่องจำลองที่เราควรรู้เพิ่มเติมคือ

- ถ้าเราต้องการปิดเครื่องจำลอง ให้คลิกแท็บด้านบนของกรอบวินโดวที่บรรจุเครื่องจำลอง
- ถ้าเราสั่งเปิดเครื่องจำลองแล้ว แต่มันไม่ปรากฏให้เห็น อาจเนื่องจากกรอบวินโดวที่บรรจุเครื่องจำลองนั้นถูกซ่อน ก็ให้คลิกที่แท็บ Emulator ที่ขอบด้านขวามือ
- ภายในกรอบวินโดวของเครื่องจำลองนั้น จะมีแถบเครื่องมือสำหรับการควบคุม ซึ่งก็คล้ายกับเครื่องจริง ดังนั้นขอให้ผู้อ่านทดลองใช้งานด้วยตัวเอง
- บางครั้ง เครื่องจำลองอาจมีปัญหา เช่น ค้าง หรือทำงานผิดพลาด เราอาจลองปิดเครื่องจำลองดังที่กล่าวมาแล้วรอสักครู่ จากนั้นค่อยเปิดใหม่ในแบบ Cold Boot (วิธีนี้จะใช้เวลาในการบูตนานกว่าปกติ) โดยเปิด Device Manager แล้วคลิกที่ไอคอน 3 จุดตรงรายการชื่อเครื่องจำลองที่ต้องการ จากนั้นเลือกเมนู **Cold Boot Now**
- แต่ถ้าเครื่องจำลองทำงานผิดพลาด และไม่มีทางแก้ไขเป็นอย่างอื่น เราอาจลบเครื่องจำลองนั้นทิ้งไป เพื่อติดตั้งใหม่ โดยเปิด Device Manager แล้วคลิกที่ไอคอน 3 จุดตรงรายการชื่อเครื่องจำลองที่ต้องการลบ จากนั้นเลือกเมนู Delete ซึ่งหลังจากที่ลบแล้ว หากเราต้องการติดตั้งใหม่ ให้เริ่มทำตามขั้นตอนที่กล่าวไปแล้วในหัวข้อ "การติดตั้งเครื่องจำลอง"

2.2.8 การรันทดสอบแอป

การรัน คือ การประมวลผลโค้ดที่เราเขียนเพื่อสร้าง (Build) เป็นแอป ซึ่งหากไม่เกิดข้อผิดพลาด แอปนั้นจะถูกติดตั้งลงในเครื่องจำลอง เหมือนกับที่เราติดตั้งบนเครื่องจริงผ่าน Store หรือไฟล์ APK แต่กรณีนี้ การติดตั้งจะดำเนินไปเองโดยอัตโนมัติหลังจากเราสั่งรัน แต่ก่อนที่เราจะรันทดสอบแอปได้ เราต้องเปิดเครื่องจำลอง แล้วรอให้หน้าจอ Home ปรากฏจนครบ

สมบูรณ์ก่อน ตามวิธีการที่ได้กล่าวไปในหัวข้อที่ผ่านมา (เทียบได้กับกรณีที่เราเปิดใช้เครื่องจริงนั้นแล้วเมื่อเครื่องจำลองพร้อมใช้งาน และโค้ดของแอปที่เราเขียนพร้อมวันทดสอบได้ ก็มีขั้นตอนที่เกี่ยวข้องดังนี้

- การรัน อาจทำได้วิธีหนึ่งคือ
 - คลิกที่ปุ่มไอคอนหัวลูกศรบนแถบทูลบาร์ซึ่งตามปกติ ถ้าแอปอยู่ในสถานะที่พร้อมวันได้ ไอคอนจะมีสีเขียว แต่ถ้าเป็นสีเทาแสดงว่าเรารันค้างอยู่ ต้องหยุดรันก่อน
 - หรือเลือกที่เมนู Run > Run
- หลังจากสั่งรัน หากต้องการหยุด ให้คลิกปุ่มไอคอนสี่เหลี่ยม (Stop) ที่อยู่ถัดไปจากปุ่ม Run ซึ่งปกติจะเป็นสีแดง และสามารถคลิกหยุดได้หลังจากที่สั่งรันไปแล้วเท่านั้น
- หากการรันมีปัญหา เช่น โค้ดมีข้อผิดพลาด มันจะหยุดโดยอัตโนมัติ ถ้าต้องการดูสาเหตุข้อผิดพลาด ให้คลิกแท็บ Run ที่ขอบด้านล่าง
- ในขณะการรันยังค้างอยู่ หากเราแก้ไขโค้ด แล้วต้องการให้สิ่งที่แก้ไขมีผลทันทีโดยไม่ต้องเริ่มต้นรันใหม่ ให้ลองคลิกปุ่ม Hot Reload แต่อาจไม่ได้ผลในทุกกรณีเสมอไป ทั้งนี้ขึ้นอยู่กับโค้ดส่วนที่เราแก้ไข ซึ่งถ้าใช้ Hot Reload แล้วไม่ได้ผล ก็ให้คลิกที่ปุ่ม Run ซ้ำอีกครั้ง
- หากเราจะปิดหรือออกจากโปรเจกต์ ถ้าขณะนั้นยังรันค้างเอาไว้ ควรหยุดการรันก่อนแล้วค่อยปิดโปรเจกต์ (ปกติจะมีไดอะล็อกขึ้นมาแจ้งเตือนอยู่แล้ว)

คำสั่งให้นักศึกษาเขียนคำตอบตามที่โจทย์กำหนดให้ถูกต้อง

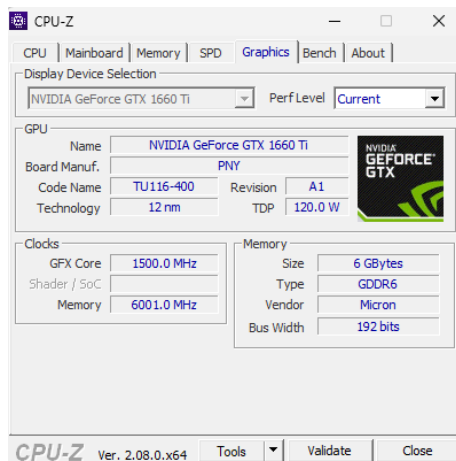
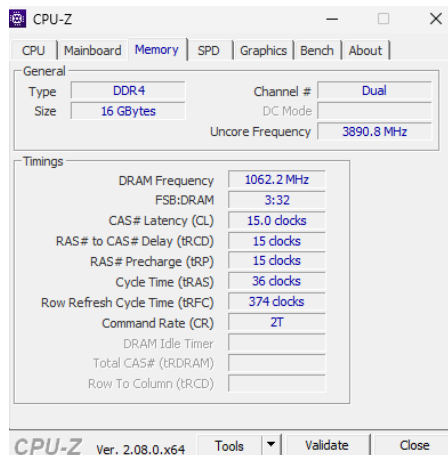
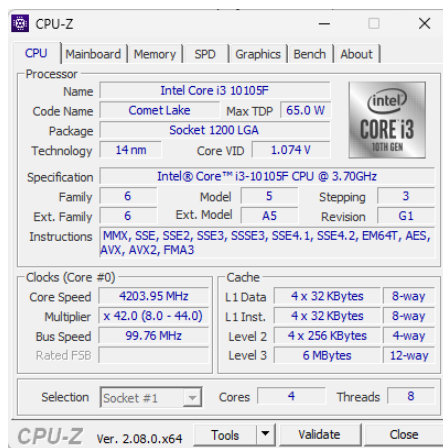
1. ให้บอกรายละเอียดสเปกและฮาร์ดแวร์ของเครื่องคอมพิวเตอร์ตัวเอง และ ตรวจสอบด้วย CPU-Z

1.1. บอกรายละเอียดสเปกและฮาร์ดแวร์ของเครื่องคอมพิวเตอร์

CPU-Z download : <https://www.cpuid.com/softwares/cpu-z.html>

ประเภท คอมพิวเตอร์	คอมพิวเตอร์ตั้งโต๊ะ
CPU	Intel Core i3 10105F
RAM	16 GBytes
GPU	NVIDIA GeForce GTX 1660 Ti

1.2. ตรวจสอบด้วย CPU-Z

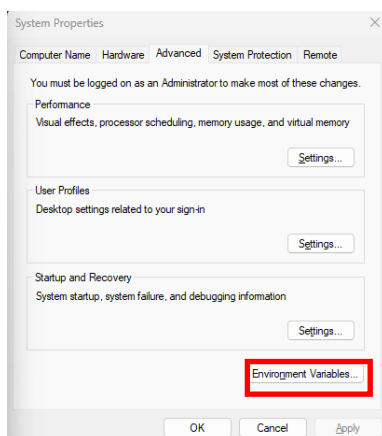
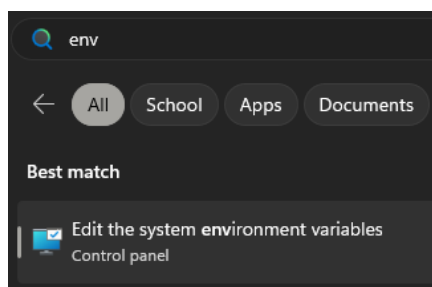
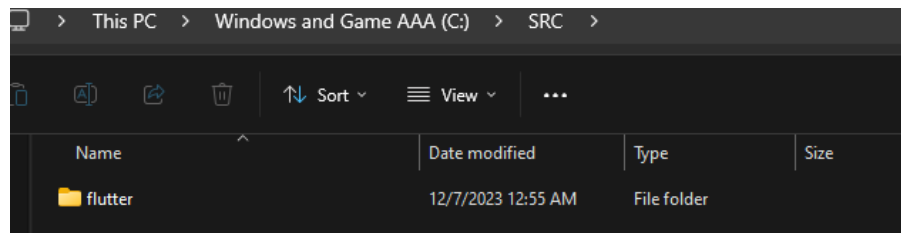
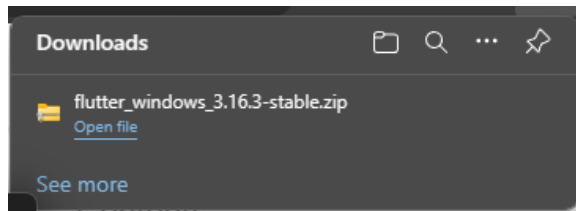


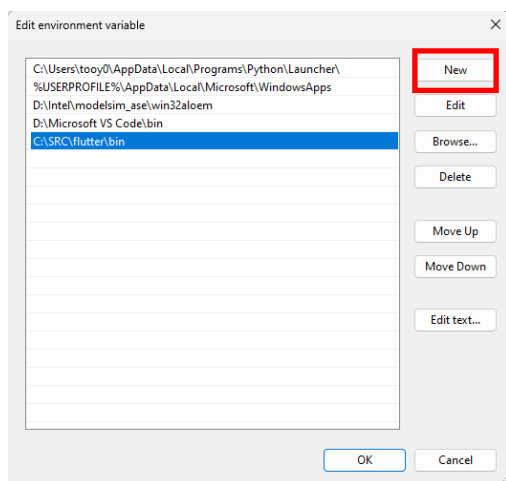
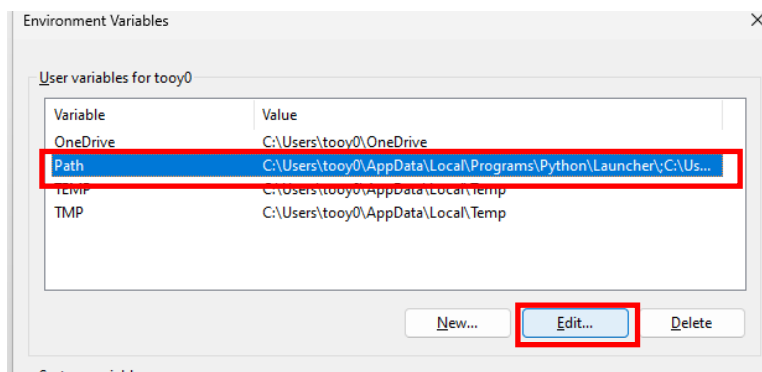
2. ถ่ายวิธีดำเนินการ ตามลำดับขั้นตอนการลง Flutter VScode และ Android SDK

1. Download the following installation bundle to get the latest stable release of the Flutter SDK:

`flutter_windows_3.16.3-stable.zip`

For other release channels, and older builds, check out the [SDK archive](#).





This PC > Windows and Game AAA (C:) > SRC > flutter >


cmd

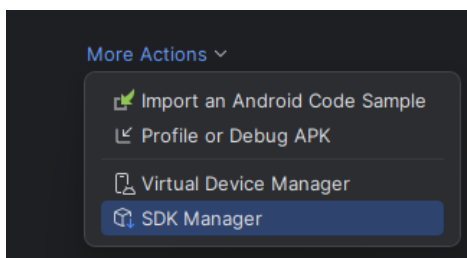
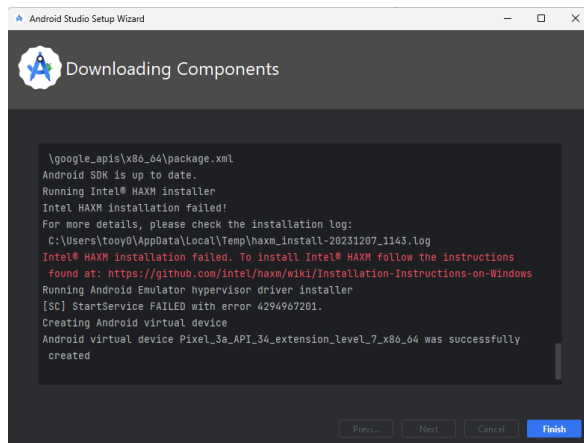
```
C:\SRC\flutter>flutter_console.bat
```

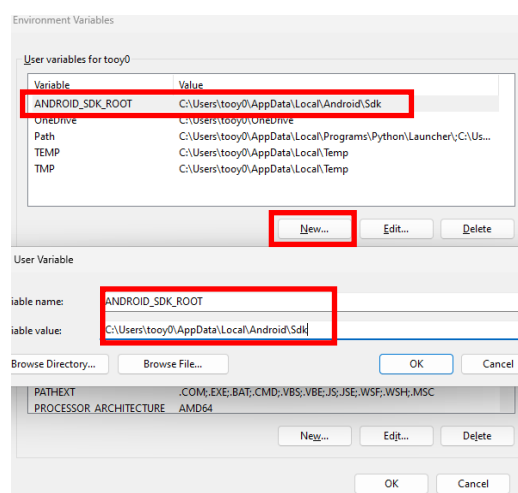
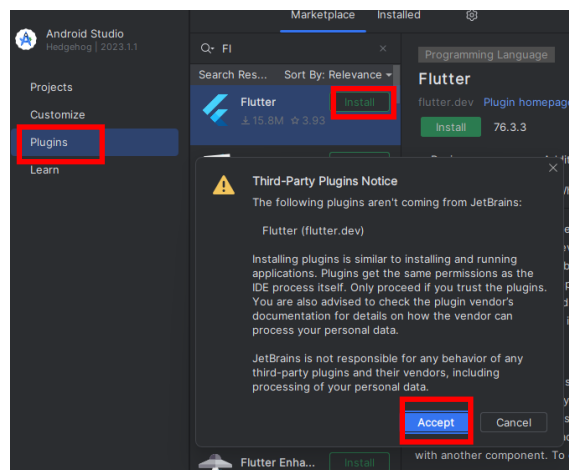
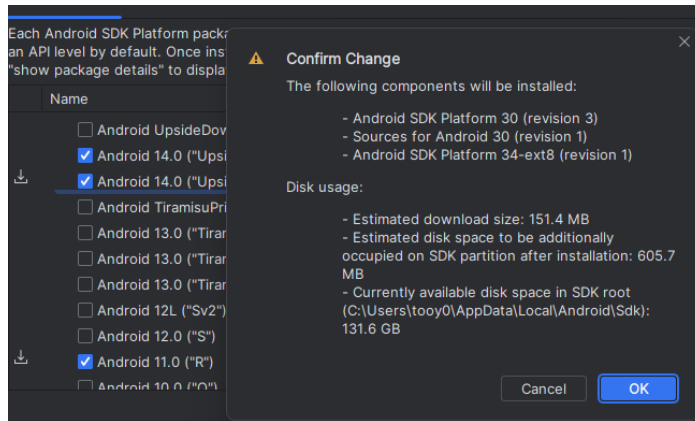
```
C:\Users\tooy0>flutter doctor
```

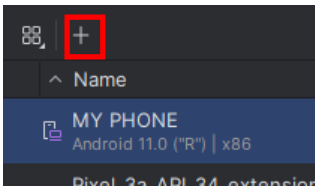
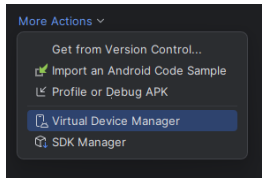
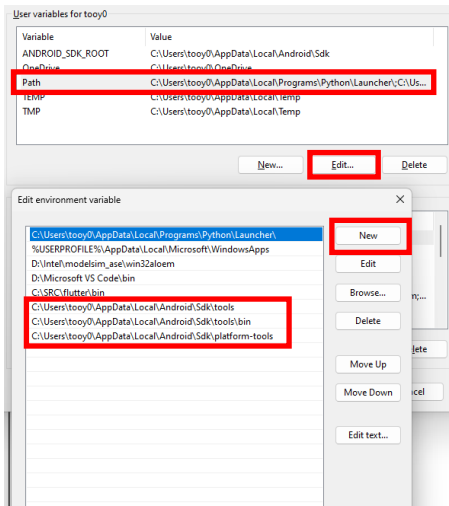
Android Studio

Get the official Integrated Development Environment (IDE) for Android app development.

Download Android Studio Hedgehog 

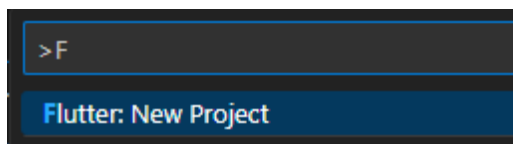
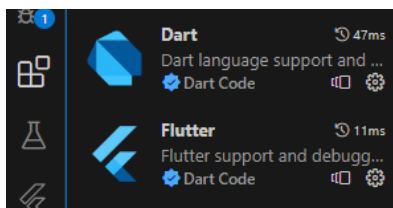
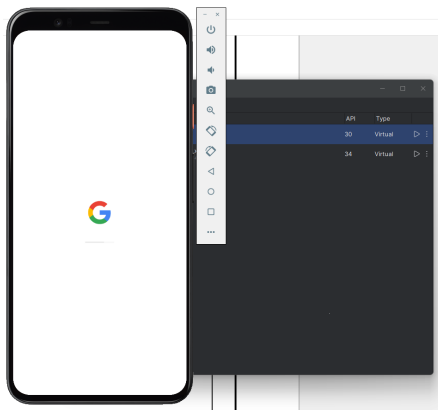
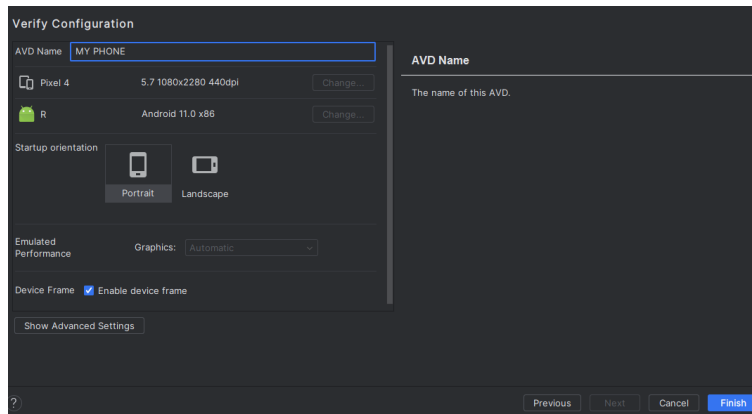






Category	Name	Play Store	Size	Resolution	Density
Phone	Pixell 6		6.4"	1080x2400	420dpi
Tablet	Pixell 5		6.0"	1080x2340	440dpi
Wear OS	Pixell 4a		5.8"	1080x2340	440dpi
Desktop	Pixell 4 XL		6.3"	1440x3040	560dpi
TV	Pixell 4		5.7"	1080x2280	440dpi
Automotive	Pixell 3a XL		6.0"	1080x2160	400dpi

Release Name	API Level	ABI	Target
UpsideDownCakePrivacy...	UpsideDownCake	x86_64	Android API UpsideDownCakeP
TiramisuPrivacySandbox	Tiramisu	x86_64	Android 14.0 (Google Play)
UpsideDownCake	34	x86_64	Android 14.0 (Google Play)
Tiramisu	33	x86_64	Android 13.0 (Google Play)
Sv2	32	x86_64	Android 12L (Google Play)
S	31	x86_64	Android 12.0 (Google Play)
R	30	x86	Android 11.0 (Google Play)
Q	29	x86	Android 10.0 (Google Play)
Pie	28	x86	Android 9.0 (Google Play)
Oreo	27	x86	Android 8.1 (Google Play)
Oreo	26	x86	Android 8.0 (Google Play)



เมื่อทำการ run Error ที่ได้ออก

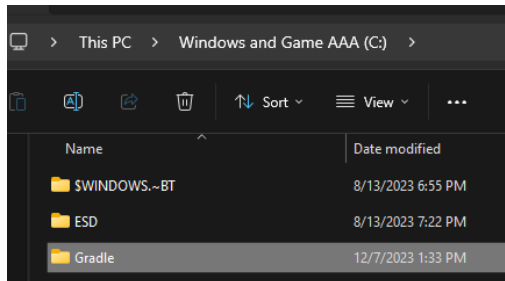
```
Launching lib\main.dart on sdk gphone x86 in debug mode...
Error: Could not find or load main class org.gradle.wrapper.GradleWrapperMain
Caused by: java.lang.ClassNotFoundException: org.gradle.wrapper.GradleWrapperMain
Exception: Gradle task assembleDebug failed with exit code 1
```


<https://gradle.org/releases/>

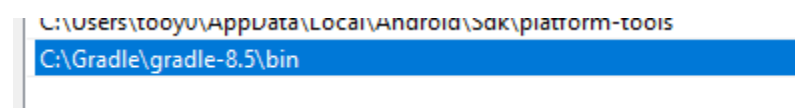
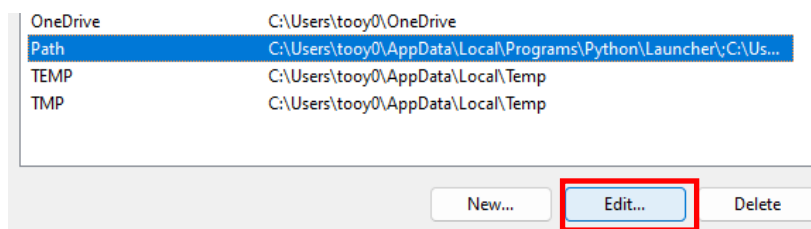
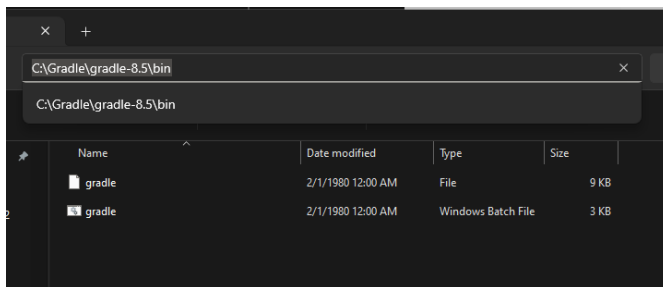
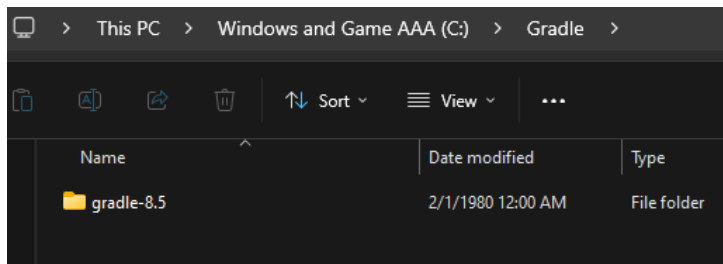
v8.5

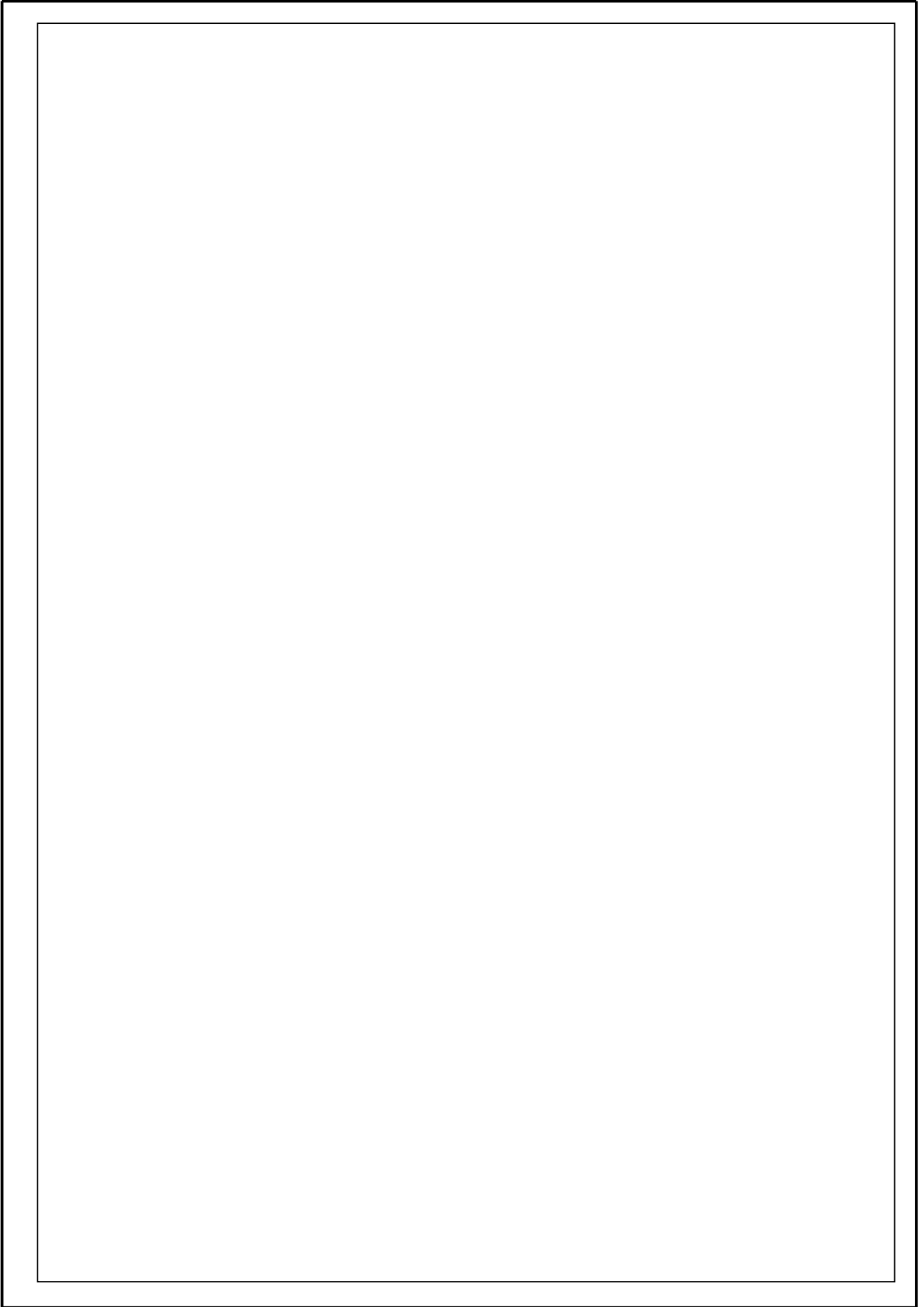
Nov 29, 2023

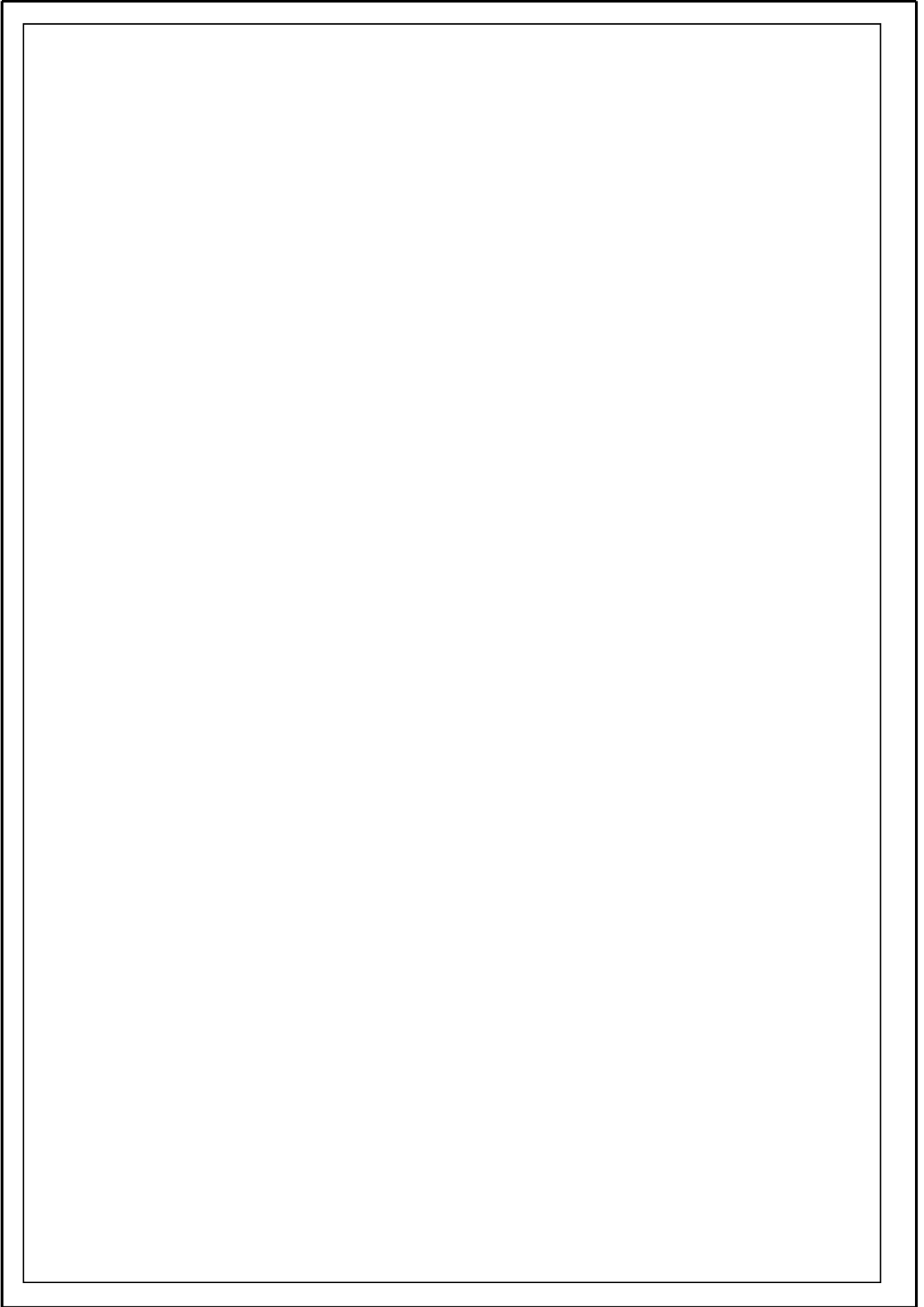
- Download: [binary-only](#) or [complete \(checksums\)](#)



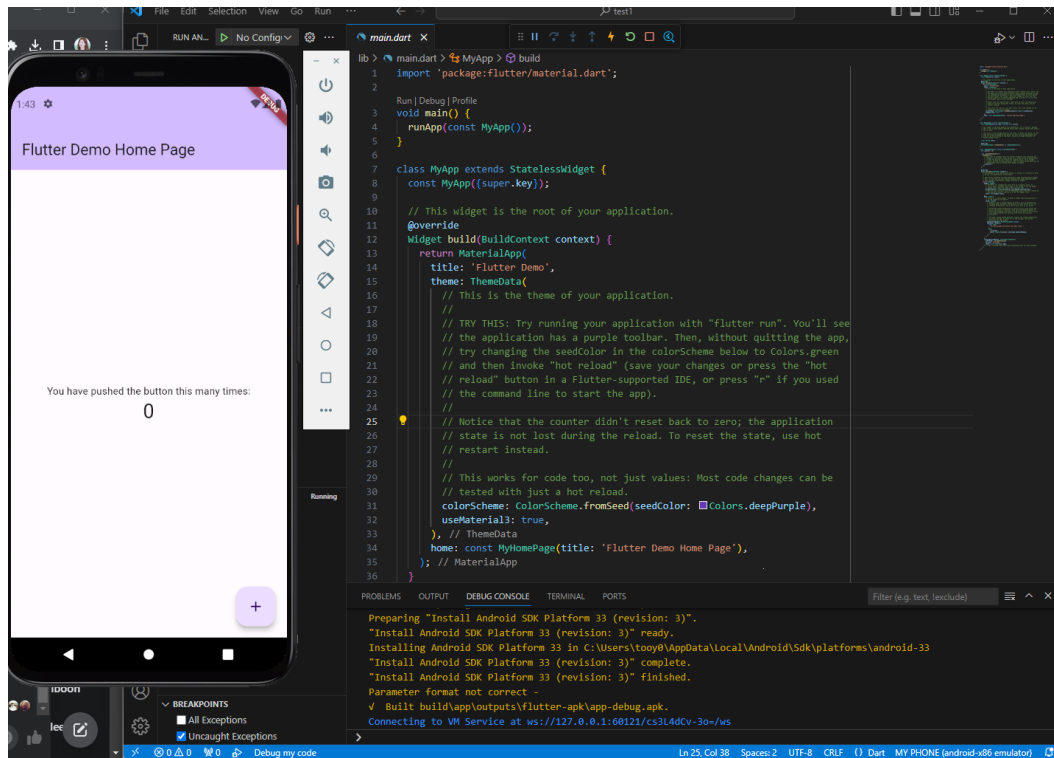
แตกไฟล์ใน Gradle







3.แบบรูปที่แสดงให้เห็นว่า VScode สามารถใช้งาน Flutter ได้แล้วโดยการ Run code counter และแสดงให้ดูใน Virtual Devices ที่เราได้ติดตั้ง



5. สรุปผลการทดลอง

การทดลองในวันนี้เป็นการทดลองการติดตั้งใช้ **Futter** ใน **vs code** โดยได้ดำเนินการตามขั้นตอนของการติดตั้งตามที่อาจารย์ที่สอนไว้ซึ่งได้ทำการติดตั้งจนได้เกิดปัญหาขึ้นโดยสามารถแก้ไขปัญหามาได้โดยการ **downloade** สลิปของ **Gradle** ซึ่งการ **run** จะต้องจำเป็นที่จะได้สลิปตัวนี้โดยเมื่อติดตั้งเสร็จสิ้นผลการ **run** จะสำเร็จตามที่ได้คิดรูปไว้ในข้อที่ 3 ซึ่งการทดลองนี้เป็นการแสดงให้เห็นถึงวิธีการใช้งาน **Futter** ใน **vs code** นั้นเอง