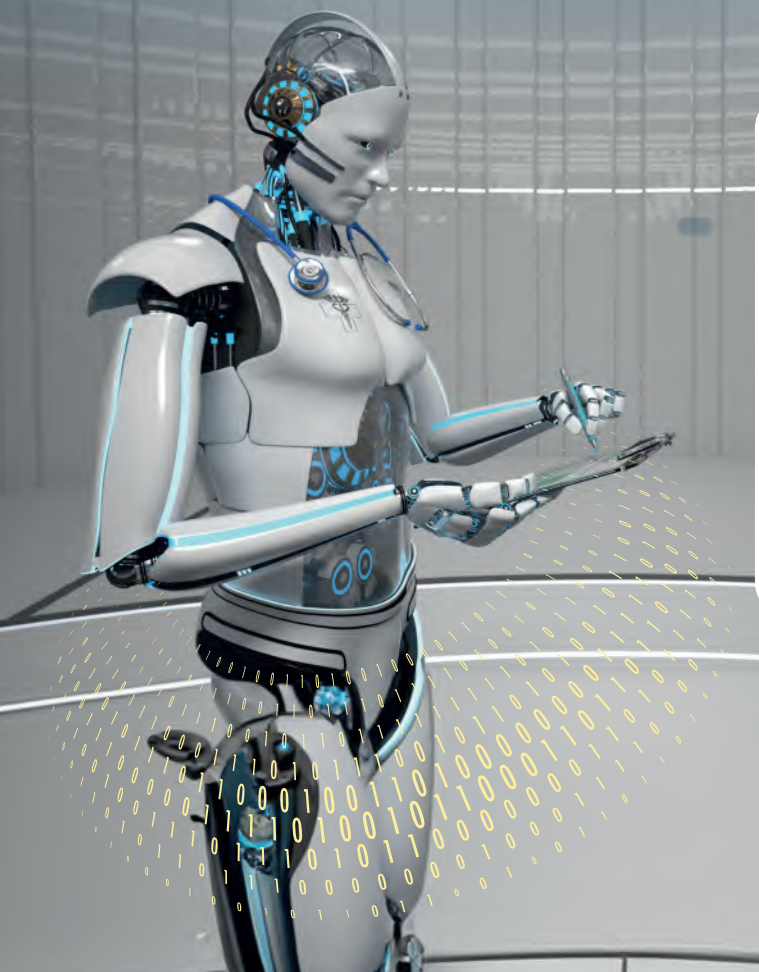
คู่มือ
พัฒนาแอปพลิเคชันแบบ Multi-Platform ด้วย

.NET MAUI

Multi-platform One codebase Productive



- เรียนรู้พื้นฐานและเทคนิคการพัฒนาแอปพลิเคชัน .NET ข้ามแพลตฟอร์ม
- รองรับการพัฒนาแอปพลิเคชันบน Android, MacOS, iOS, Windows และ Web Base
- พัฒนาโค้ดเพียงครั้งเดียวสามารถใช้งานได้ทุกแพลตฟอร์ม
- มีโค้ดตัวอย่างและคำอธิบายอย่างละเอียดอ่านเข้าใจง่าย
- อธิบายเป็นขั้นตอน เหมาะสำหรับผู้เริ่มต้น

ไฟล์ตัวอย่างภายในเล่ม
https://serazu.com/9786164874732

ศุภชัย สมพานิช

IDC
PREMIER

มีเพียง “**ความรู้**” เท่านั้นที่มนุษย์ใช้พลิก “**โลก**”
และเปลี่ยน “**ชีวิต**” เราจึงสร้างสรรค์ และส่งมอบ “**ความรู้**”
ในรูปแบบที่ดีกว่า เพื่อให้คนไทย “**เรียนรู้**” ได้ตลอดชีวิต

Only “**Knowledge**” can help human
change “**The World**” and “**Their Lives**”.
With this truth, it drives us to deliver
“**Knowledge**” for Thai being able to
“**Learn**” better everyday.

Think Beyond
INFOPRESS
Read comics
Little HEART

คู่มือพัฒนาแอปพลิเคชัน Multi-Platform ด้วย .NET MAUI

Author
ศุภชัย สมพานิช
Editorial
กิตินันท์ พลสวัสดิ์ kitinan_p@idcpremier.com
Production Manager
วรพล ณธิกุล
Graphic Designer
ชวนันท์ รัตนะ
Page Layout
ปริเยศ จันทร์จำปา
Proofreader
พรรณรัตน์ ขุราษี
Publishing Coordinators
สุรีย์รัตน์ จิอู๋, ฉัตรดนัย ศรีสวัสดิ์
Product Specialist
ศรัณย์ คมขำ

PUBLISHED AND DISTRIBUTED BY

บริษัท ไอดีซี พรีเมียร์ จำกัด
200 หมู่ 4 ชั้น 19 ห้อง 1901 อาคารจัสมินอินเตอร์เนชั่นแนล ทาวเวอร์ ถ.แจ้งวัฒนะ อ.ปากเกร็ด จ.นนทบุรี 11120
โทรศัพท์ 0-2962-1081 (อัตโนมัติ 10 คู่สาย)
โทรสาร 0-2962-1084
สมาชิกสัมพันธ์
โทรศัพท์ 0-2962-1081-3 ต่อ 121
โทรสาร 0-2962-1084
ร้านค้าและตัวแทนจำหน่าย
โทรศัพท์ 0-2962-1081-3 ต่อ 112-114
โทรสาร 0-2962-1084

สร้างสรรค์โดย

พิมพ์ครั้งที่ 1 กันยายน 2566

ข้อมูลทางบรรณานุกรม
ศุภชัย สมพานิช
คู่มือพัฒนาแอปพลิเคชัน Multi-Platform ด้วย .NET MAUI
นนทบุรี : ไอดีซีฯ, 2566
448 หน้า
1. ภาษาที่ใช้ในการเขียนโปรแกรม
I ชื่อเรื่อง
005.262
Barcode 885-916-101-058-6

ราคา 480 บาท

พิมพ์ที่ **บริษัท ดี.เค.ปริ้นติ้ง จำกัด**
441/56 หมู่ 2 ต.บางบ่อ อ.บางบ่อ จ.สมุทรปราการ 10650
โทรศัพท์ 0-2115-9105 โทรสาร 0-2115-9044

คำนำ

ในยุคปัจจุบันการพัฒนาแอปแบบสร้างเพียง 1 โปรเจ็กต์แต่ได้แอปหลายประเภทกำลังได้รับความนิยมเป็นอย่างสูง ในแพลตฟอร์มของ .NET มีโปรเจ็กต์ประเภทนี้เช่นกัน เรียกว่า .NET MAUI ผลผลิตของโปรเจ็กต์ประเภท .NET MAUI คุณจะได้ทั้ง Desktop App, Web Apps และ Mobile Apps (iOS และ Android) ในเวลาเดียวกัน โดยที่หนังสือเล่มนี้ยึดถือการพัฒนา .NET MAUI บน Windows เป็นหลัก แต่สามารถใช้ใน macOS ได้เช่นกัน อาจจะมีข้อแตกต่างในแง่ขั้นตอนใช้งานบ้าง แต่โค้ดยังคงใช้ได้เกือบทั้งหมด เพราะว่ามีโค้ดบางส่วนถูกสร้างขึ้นมาเฉพาะ Windows หรือเฉพาะ macOS ถือเป็นข้อจำกัดทั่วไป

คุณผู้อ่านท่านใดที่ติดปัญหาหรือต้องการสอบถาม, ให้ข้อแนะนำ, ติติง รวมถึงข้อผิดพลาดต่างๆ ที่อาจจะเกิดขึ้นได้ สามารถติดต่อกับผู้เขียนได้ 5 ช่องทาง คือ

1. แฟนเพจใน Facebook ชื่อว่า Thaivb.NET (https://www.facebook.com/thaivb.net)
2. กลุ่มพูดคุยเรื่องเขียนโปรแกรมใน Facebook ชื่อว่า Thaivb.NET (https://www.facebook.com/groups/Thaivb.NET)
3. ช่องใน YouTube ชื่อว่า Thaivb.NET (https://www.youtube.com/@thaivb)
4. กลุ่มใน LINE ชื่อว่า Thaivb.NET
5. แฟนเพจใน Blockdit ชื่อว่า Thaivb.NET (https://www.blockdit.com/thaivb.net)

ศุภชัย สมพานิช

กันยายน 2023


.NET MAUI

สารบัญ


.NET MAUI

บทที่ 1

การพัฒนาแอปด้วย .NET MAUI

.NET MAUI เป็นความก้าวหน้าครั้งสำคัญอีกก้าวหนึ่งของแพลตฟอร์ม .NET เป็นโปรเจ็กต์ที่ช่วยให้นักพัฒนาสามารถสร้างแอปหลายแพลตฟอร์มในเวลาเดียวกันด้วยโครงสร้างโปรเจ็กต์เพียงหนึ่งเดียว ช่วยให้ลดการดูแลรักษาโปรเจ็กต์ในระยะยาวได้อีกทางหนึ่งด้วย

ผลผลิตของโปรเจ็กต์ .NET MAUI สามารถสร้างแอปได้ 5 แบบ คือ

1. **Windows Apps** เป็นแอปประเภท Desktop บนเครื่อง PC
2. **Android Apps** เป็นแอปบนอุปกรณ์ของค่าย Google
3. **iOS Apps** เป็นแอปบนอุปกรณ์มือถือของค่าย Apple
4. **Mac Catalyst Apps** เป็นแอปประเภท Desktop บนเครื่อง MAC
5. **Tizen** เป็นแอปบนระบบปฏิบัติการของซัมซุง

เนื้อหาในหนังสือเล่มนี้เป็นการใช้งานโปรแกรม Visual Studio บนเครื่องคอม PC ส่งผลให้ผู้เขียนสามารถนำเสนอแอปได้ 2 ประเภท คือ Windows Apps และ Android Apps

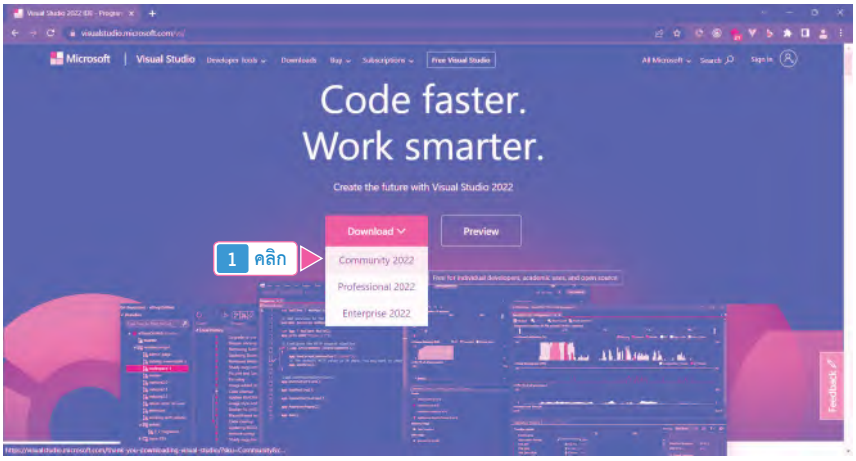
เนื้อหาที่นำเสนอในหนังสือเล่มนี้ครอบคลุมการพัฒนาแอปในรูปแบบ Front End กับ Back End กล่าวคือ

1. **งานฝั่งหน้าบ้าน (Front End)** เป็นหน้าที่ของโปรเจ็กต์ .NET MAUI ทำหน้าที่สร้างส่วนแสดงผลต่างๆ (User Interface – UI)
2. **งานหลังบ้าน (Back End)** เป็นหน้าที่ของโปรเจ็กต์ประเภท ASP.NET Core Web API ทำหน้าที่ติดต่อกับส่วนของข้อมูลเพื่อให้บริการพื้นฐานในรูปแบบ Web API ได้แก่ งานประเภทเพิ่ม, อ่าน, อัพเดต และลบข้อมูล (CRUD)

การดาวน์โหลดและติดตั้งโปรแกรม Visual Studio 2022

การพัฒนาโปรแกรมในโลกของ .NET มีหลายวิธี หนังสือเล่มนี้อาศัยโปรแกรมที่ชื่อว่า Visual Studio 2022 มี 2 แบบ คือ

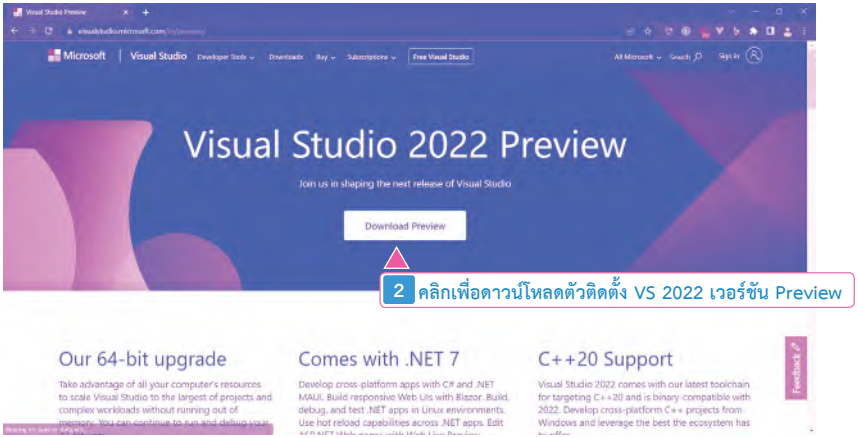
1. **โปรแกรม Visual Studio เวอร์ชันปกติ** เป็นเวอร์ชันที่เหมาะกับการนำไปใช้ในงาน Production จริง สามารถดาวน์โหลดมาใช้งานได้ฟรีที่ https://visualstudio.microsoft.com/vs/



รูปที่ 1-1 แสดงเว็บสำหรับดาวน์โหลดโปรแกรม Visual Studio เวอร์ชันปกติ

2. **โปรแกรม Visual Studio เวอร์ชัน Preview** เป็นเวอร์ชันที่เหมาะสมกับกรณีที่ต้องการทดสอบฟีเจอร์ใหม่ๆ ก่อนใคร มีความเสี่ยงกับบั๊กและ Error ต่างๆ มากกว่าเวอร์ชันปกติ ไม่แนะนำให้ใช้เวอร์ชันนี้กับงาน Production จริง ใช้แค่ในระดับศึกษาเท่านั้น สามารถโหลดมาใช้งานฟรีได้ที่ https://visualstudio.microsoft.com/vs/preview/

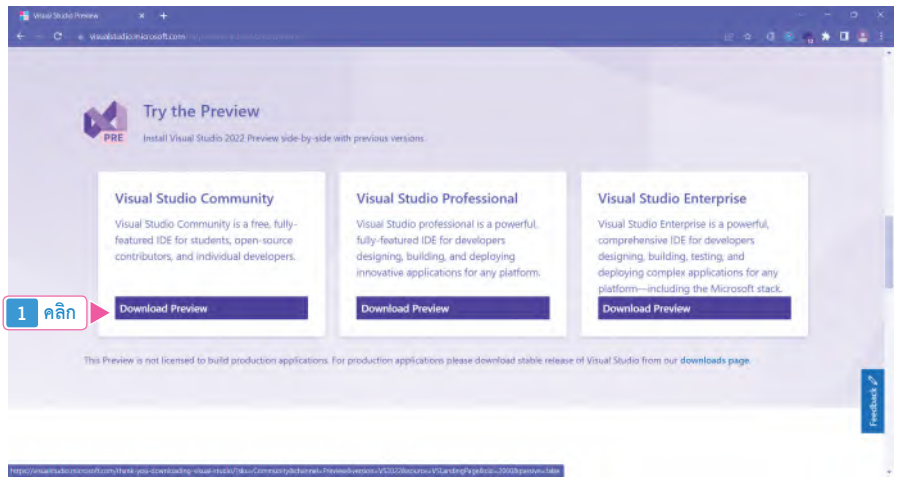
ฟีเจอร์ต่างๆ ที่ถูกนำมาทดสอบในเวอร์ชันนี้ เมื่อผ่านการทดสอบแล้วไมโครซอฟท์จะนำไปใช้งานกับเวอร์ชันปกติในลำดับถัดไป ดังรูปที่ 1-2



รูปที่ 1-2 แสดงเว็บสำหรับดาวน์โหลดโปรแกรม Visual Studio 2022 เวอร์ชัน Preview

ผู้อ่านสามารถติดตั้งโปรแกรม Visual Studio 2022 ทั้ง 2 เวอร์ชันได้ในเครื่องเดียวกัน ต่างคนต่างอยู่ มีวิธีการติดตั้งเหมือนกันดังนี้

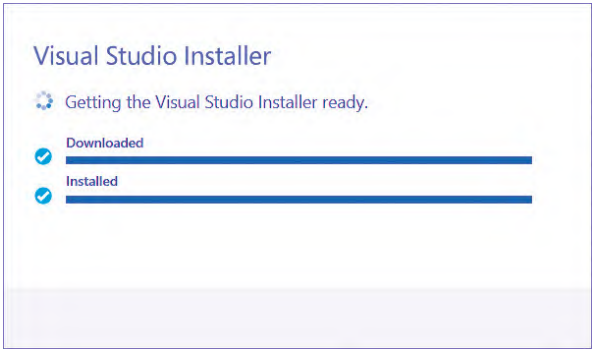
1. ให้ผู้อ่านเลือกดาวน์โหลดตัวติดตั้งโปรแกรม Visual Studio 2022 Community Edition มาก่อน ดังรูปที่ 1-3

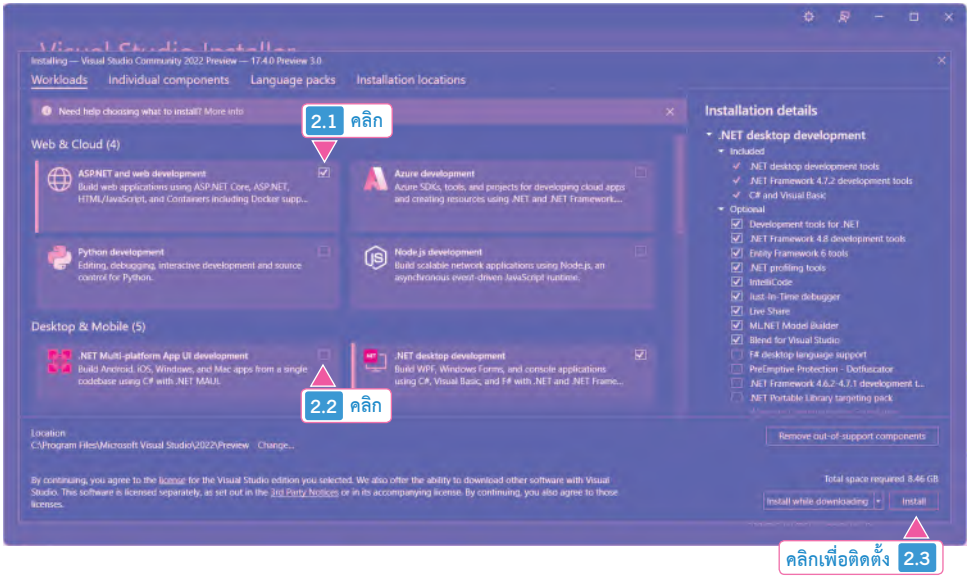


รูปที่ 1-3 แสดงการดาวน์โหลดตัวติดตั้งโปรแกรม Visual Studio 2022

2. ให้ผู้อ่านดับเบิลคลิกไฟล์ที่ดาวน์โหลดมา เพื่อเข้าสู่ขั้นตอนการติดตั้งโปรแกรม Visual Studio 2022 ผู้อ่านสามารถเลือกประเภทโปรเจ็กต์ที่สนใจได้อย่างอิสระ ขอบเขตการนำเสนอเนื้อหาของหนังสือเล่มนี้ประกอบด้วย

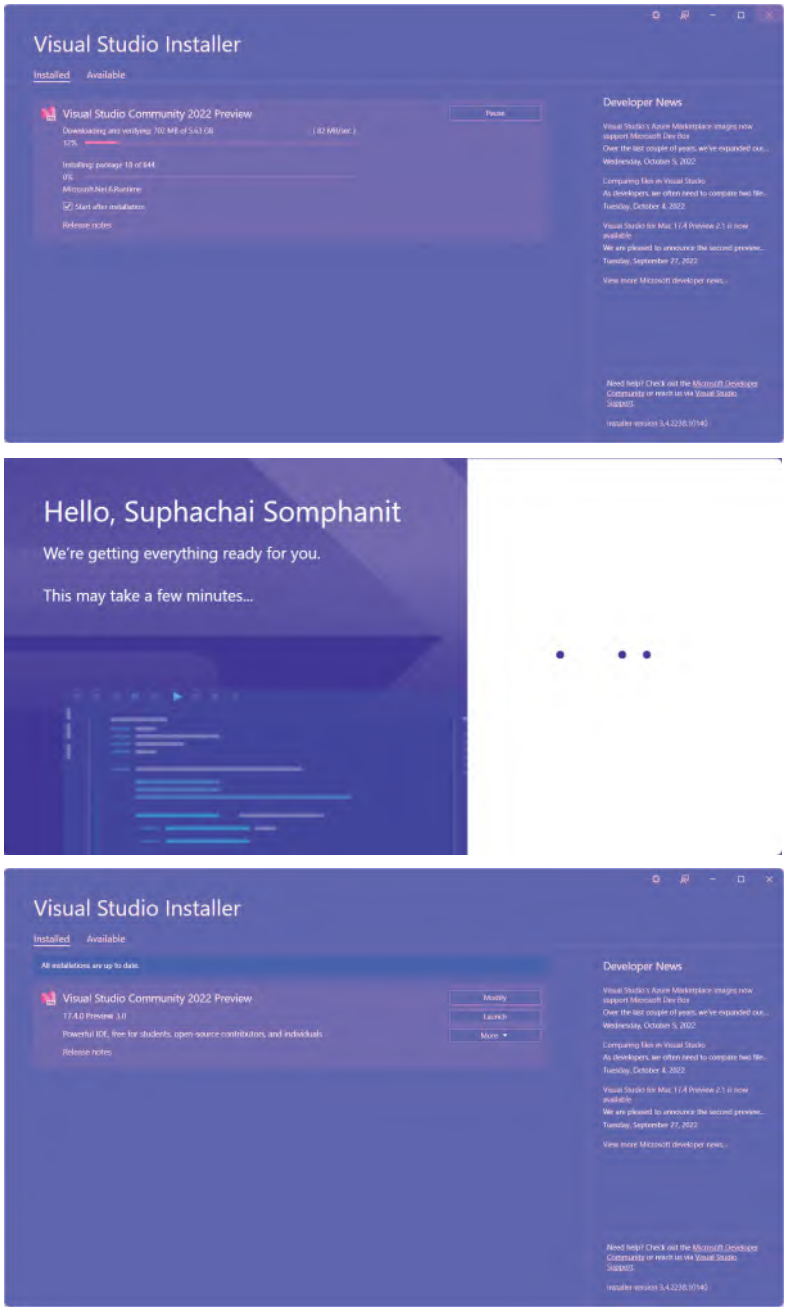
   - **ASP.NET and web development** สำหรับพัฒนาโปรเจ็กต์ประเภท Web Apps ในกรณีนี้สนใจเฉพาะโปรเจ็กต์ ASP.NET Core Web API
   - **.NET Multi-platform App UI development** สำหรับพัฒนาโปรเจ็กต์ประเภท .NET MAUI เป็นเนื้อหาหลักของหนังสือเล่มนี้





รูปที่ 1-4 แสดงหน้าจอเลือกติดตั้งประเภทโปรเจ็กต์

3. ท้ายที่สุดให้ผู้อ่านรอการดาวน์โหลดและติดตั้งรายการต่างๆ ตามที่เลือกจนเสร็จสมบูรณ์ ถือว่าโปรแกรม Visual Studio 2022 พร้อมใช้งานแล้ว ดังรูปที่ 1-5

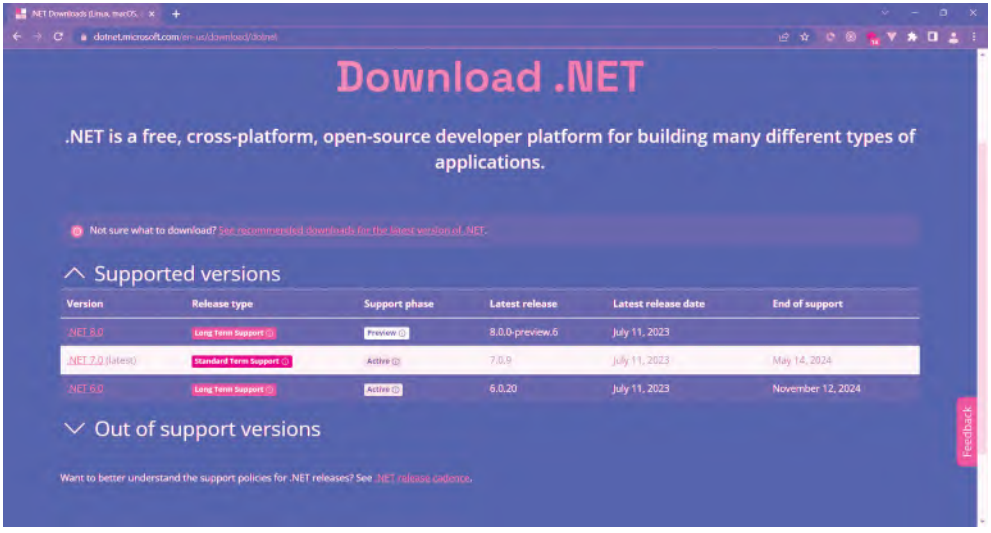


รูปที่ 1-5 แสดงการติดตั้งโปรแกรม Visual Studio เสร็จสมบูรณ์

สถานะของ .NET SDK

หลังจากที่ไมโครซอฟท์รวม .NET Framework และ .NET Core เข้าด้วยกัน กลายเป็นยุค .NET ในปัจจุบัน มีนโยบายออก .NET SDK เวอร์ชันใหม่ปีละ 1 รุ่น ทุกๆ เดือนพฤศจิกายน มีสถานะแตกต่างกัน กล่าวคือ

- **.NET SDK เวอร์ชันเลขคี่** ได้แก่ เวอร์ชัน 5, 7 ... มีระยะเวลาสนับสนุนอัพเดตแพตช์แก้ไข 18 เดือน เป็นเวอร์ชันที่เรียกว่า **Standard-term support** เรียกย่อๆ ว่า **STS**
- **.NET SDK เวอร์ชันเลขคู่** ได้แก่ เวอร์ชัน 6, 8 ... มีระยะเวลาสนับสนุนอัพเดตแพตช์แก้ไข 3 ปี เป็นเวอร์ชันที่เรียกว่า **Long-term support** เรียกย่อๆ ว่า **LTS** การพัฒนาโปรเจ็กต์ในระดับ Production จริง ผู้เขียนแนะนำให้เลือกใช้เวอร์ชัน LTS เหมาะสมที่สุด



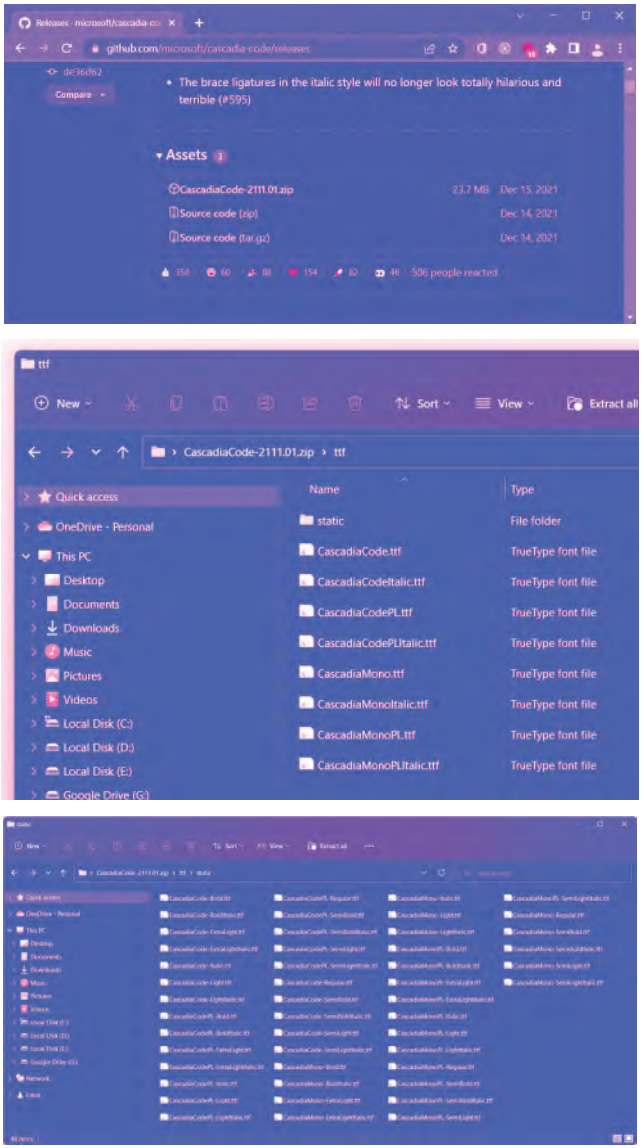
รูปที่ 1-6 แสดงรายละเอียดของ .NET SDK แต่ละรุ่น

.NET SDK เวอร์ชันใดก็ตามที่ใกล้หมดระยะเวลา Support เรียกว่า ระยะ Maintenance ถ้าโปรเจ็กต์ของเราใช้เวอร์ชันที่มีสถานะนี้ ควรที่จะอัพเดตไปใช้เวอร์ชันที่ใหม่กว่าให้เร็วที่สุด

ส่วน .NET SDK เวอร์ชันเก่า จะถูกจัดให้อยู่ในกลุ่ม Out of support versions เป็นเวอร์ชันที่ไม่แนะนำให้ใช้งานในระดับ Production จริงแล้ว ผู้อ่านสามารถเลือกติดตั้ง .NET SDK ได้หลายเวอร์ชันในเวลาเดียวกัน

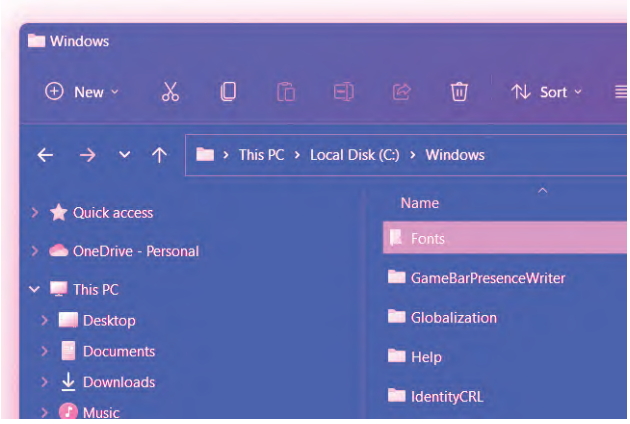
การติดตั้งฟอนต์ Cascadia Code สำหรับเขียนโค้ดโดยเฉพาะ

ไมโครซอฟท์สร้างฟอนต์สำหรับเขียนโค้ดโดยเฉพาะขึ้นมา ชื่อว่า **Cascadia Code** สามารถดาวน์โหลดมาใช้งานได้ที่เว็บไซต์ https://github.com/microsoft/cascadia-code/releases ดังรูปที่ 1-7



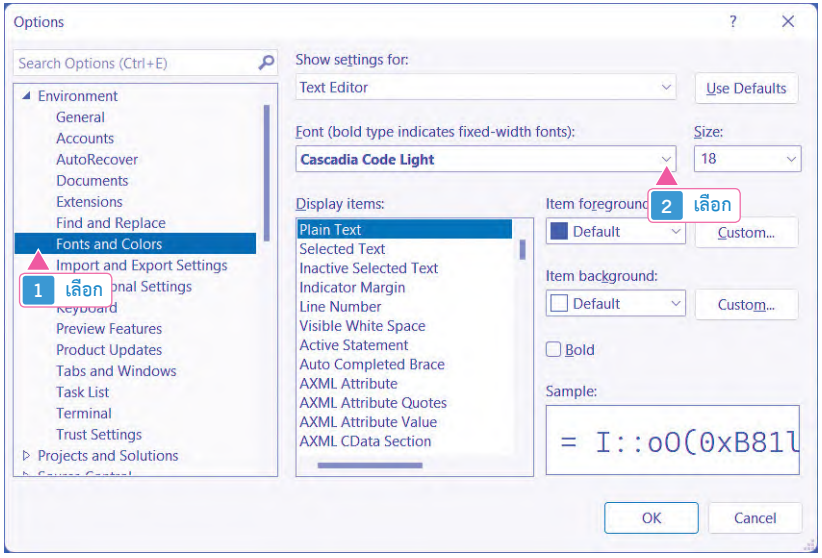
รูปที่ 1-7 แสดงเว็บไซต์ที่ให้ดาวน์โหลดฟอนต์เขียนโค้ด

จากรูปที่ 1-7 ให้แตกไฟล์ zip ที่ได้มา เราสนใจฟอนต์ที่มีนามสกุล .ttf เท่านั้น อยู่ในโฟลเดอร์ที่ชื่อว่า static ด้วย จากนั้นให้ผู้อ่าน Copy ฟอนต์ที่ได้มาทั้งหมดไปวางไว้ในพาธ C:\Windows\Fonts



รูปที่ 1-8 แสดงโฟลเดอร์ที่จัดเก็บรายการฟอนต์ในระบบ Windows 10/11

ให้ผู้อ่านเปิดโปรแกรม Visual Studio ขึ้นมา เลือกเมนู Tools > Options ที่หัวข้อ Environment > Fonts and Colors เลือกใช้ฟอนต์ที่ชื่อว่า **Cascadia Code Light**

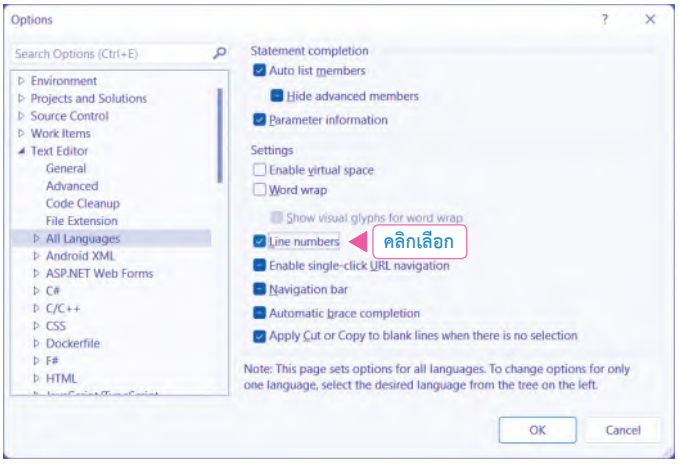


รูปที่ 1-9 แสดงการกำหนดให้โปรแกรม Visual Studio ใช้ฟอนต์ Cascadia Code Light

ข้อแตกต่างของฟอนต์ Cascadia Code กับฟอนต์อื่นๆ ที่น่าสนใจ คือ สามารถแสดงสัญลักษณ์พิเศษทางภาษาได้ เช่น เครื่องหมายหัวลูกศร เกิดมาจากการพิมพ์เครื่องหมาย = ติดกับเครื่องหมาย มากกว่า >

การกำหนดให้แสดงหมายเลขบรรทัด

ส่วนการกำหนดให้ VS 2022 แสดงหมายเลขกำกับโค้ดในแต่ละบรรทัด โดยการเลือกเมนู Tools > Options... ให้ผู้อ่านไปที่หัวข้อ Text Editor > All Languages เลือกตัวเลือก Line numbers ดังรูปที่ 1-10

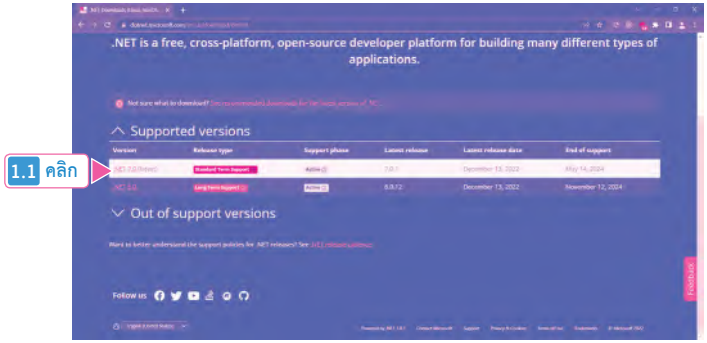


รูปที่ 1-10 แสดงการกำหนดให้แสดงหมายเลขกำกับโค้ดแต่ละบรรทัด

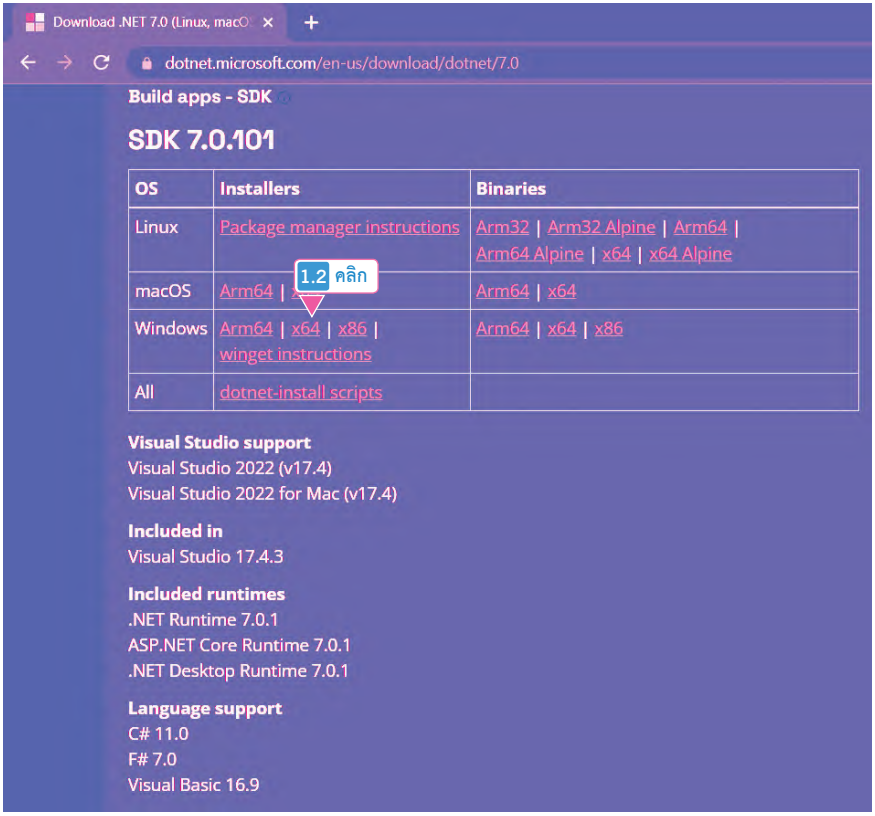
การติดตั้ง .NET SDK เพิ่มเติม

เนื้อหาในหนังสือเล่มนี้ต้องการ .NET SDK เวอร์ชัน 7 ขึ้นไป ผู้อ่านสามารถติดตั้ง .NET SDK หลายเวอร์ชันในเวลาเดียวกัน ส่งผลให้โปรแกรม Visual Studio 2022 ของผู้อ่านสามารถสร้างโปรเจ็กต์ได้ตามเวอร์ชัน .NET SDK ที่ถูกติดตั้ง มีขั้นตอนดังนี้

1. ให้ผู้อ่านไปที่เว็บไซต์ https://dotnet.microsoft.com/en-us/download/dotnet เพื่อดาวน์โหลด .NET 7 SDK เพิ่มเติม ดังรูปที่ 1-11 และรูปที่ 1-12



รูปที่ 1-11 แสดงการดาวน์โหลด .NET 7 SDK



รูปที่ 1-12 แสดงการดาวน์โหลด .NET 7 SDK (ต่อ)

จากรูปที่ 1-12 ผู้เขียนใช้งาน Windows 10/11 แบบ 64 บิต จึงเลือก .NET 7 SDK แบบ x64 ผู้อ่านต้องตรวจสอบด้วยว่า .NET SDK ที่ดาวน์โหลดมาติดตั้งเพิ่มเติมเอง ต้องการโปรแกรม Visual Studio เวอร์ชันใด ในโปรแกรม Visual Studio ตรวจสอบได้ที่เมนู Help > About Visual Studio

NOTE

เมื่อมีการอัพเดต .NET SDK เวอร์ชันใหม่ ก็จะมีการอัพเดตโปรแกรม Visual Studio ควบคู่มาด้วยเสมอ เพื่อให้สามารถใช้งาน .NET SDK เวอร์ชันใหม่ได้นั่นเอง โดยปกติแล้วโปรแกรม Visual Studio เวอร์ชัน Preview มักจะได้สิทธิ์ทดสอบ .NET SDK เวอร์ชันใหม่ก่อนเวอร์ชันปกติ

2. ท้ายที่สุดให้ผู้อ่านดับเบิลคลิกไฟล์ที่ดาวน์โหลดมา เพื่อเข้าสู่ขั้นตอนการติดตั้ง .NET 7 SDK ใช้ค่าเริ่มต้นที่มากับตัวติดตั้งทั้งหมด ส่งผลให้โปรแกรม Visual Studio 2022 สามารถพัฒนาโปรเจ็กต์บน .NET SDK ได้หลายเวอร์ชัน

การอัพเดตโปรแกรม Visual Studio

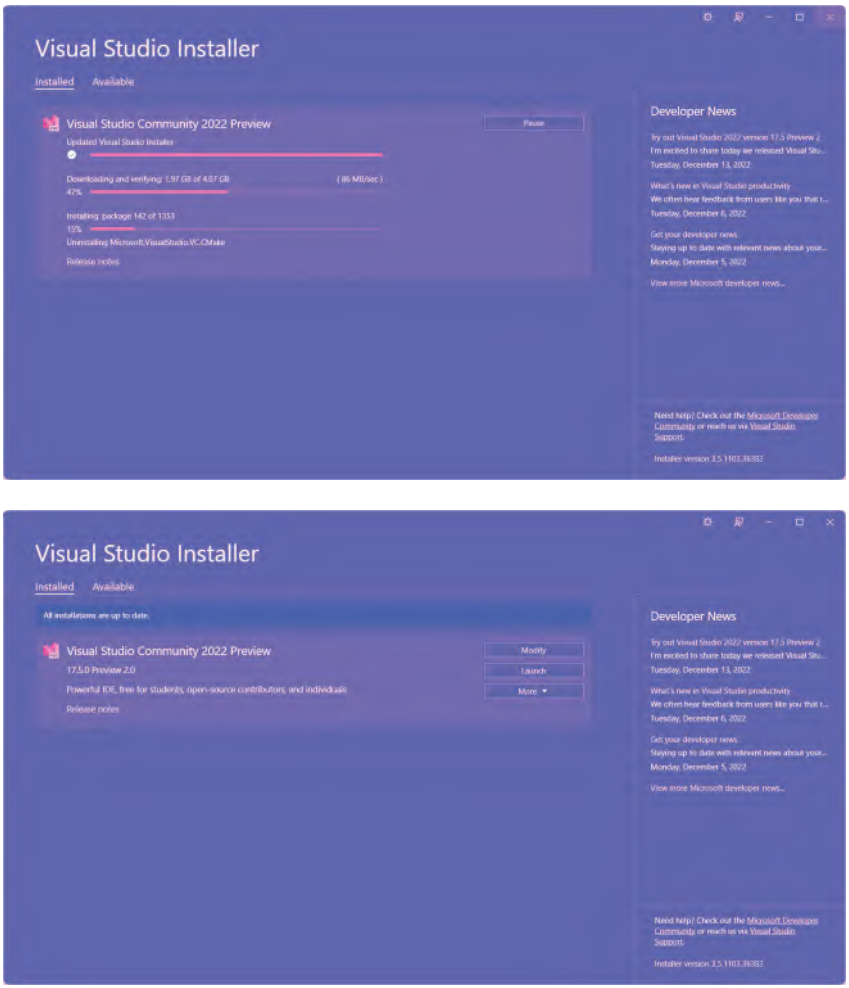
ตัวโปรแกรม Visual Studio มีการอัพเดตอยู่เสมอ อาจจะเป็นการแก้ไขข้อผิดพลาดหรือเพิ่มเติมฟีเจอร์ใหม่ๆ ก็ได้ แยกออกเป็น 2 กรณี

- **Visual Studio เวอร์ชัน Preview** เป็นเวอร์ชันที่ใช้สำหรับทดสอบฟีเจอร์ใหม่ๆ ของ .NET SDK อยู่แล้ว ส่งผลให้มีการอัพเดตค่อนข้างบ่อย
- **Visual Studio เวอร์ชัน Stable** เป็นเวอร์ชันที่เราใช้งานจริงใน Production ของเรา หลังจากฟีเจอร์ต่างๆ ถูกทดสอบในเวอร์ชัน Preview จนกระทั่งสมบูรณ์แล้วก็จะนำมาอัพเดตในเวอร์ชัน Stable มีระยะการอัพเดตห่างกว่าเวอร์ชัน Preview

1. เมื่อตัวโปรแกรม Visual Studio มีอัพเดตใหม่ จะมีการแจ้งเตือน ดังรูปที่ 1-13

รูปที่ 1-13 แสดงการแจ้งเตือนอัพเดตของโปรแกรม Visual Studio

2. การดาวน์โหลดและติดตั้งอัพเดตต่างๆ ให้ปิดตัวโปรแกรม Visual Studio ก่อน ก็จะเข้าสู่ขั้นตอนการอัพเดตโดยอัตโนมัติทั้งหมด ดังรูปที่ 1-14

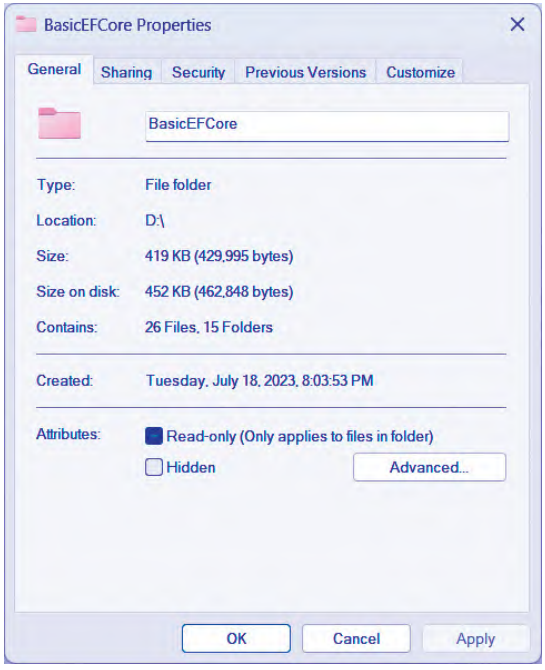


รูปที่ 1-14 แสดงการอัพเดต Visual Studio เสร็จสมบูรณ์

วิธีการจัดเก็บโปรเจ็กต์ .NET

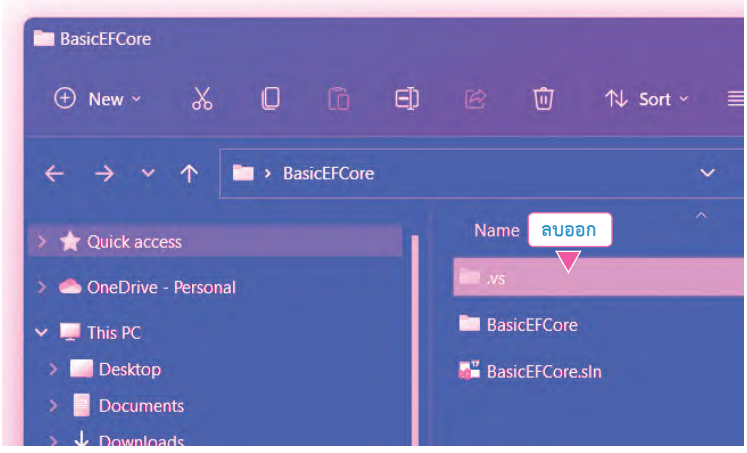
เป้าหมายของวิธีนี้คือ ต้องการลดขนาดไฟล์โปรเจ็กต์ให้มีขนาดเล็กลง โปรเจ็กต์ทุกประเภทของ .NET สามารถใช้วิธีการจัดเก็บโปรเจ็กต์แบบนี้ได้ มีขั้นตอนดังนี้

1. ให้ผู้อ่านตรวจสอบขนาดโปรเจ็กต์ก่อนปรับปรุง โดยการคลิกขวาเลือกคำสั่ง Properties พบว่าขนาดโปรเจ็กต์ 883 KB แสดงดังรูปที่ 1-15



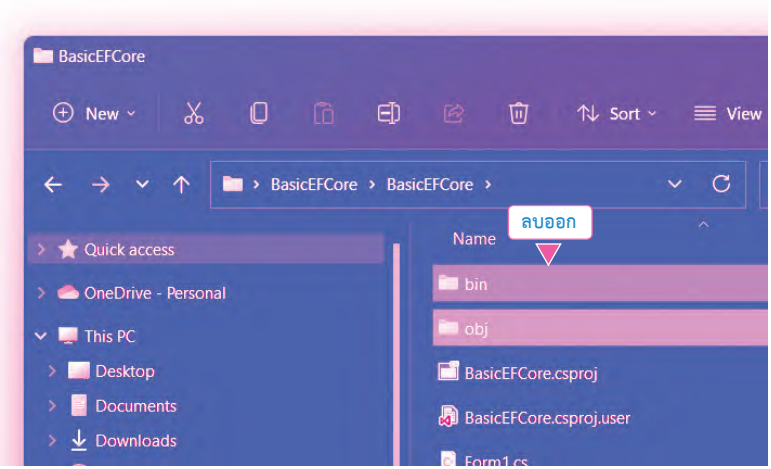
รูปที่ 1-15 แสดงขนาดโปรเจ็กต์ก่อนปรับปรุง

2. การปรับปรุงขนาดโปรเจ็กต์ .NET ทำได้โดยการเข้าไปในโฟลเดอร์ย่อยระดับแรกก่อน ผู้อ่านสามารถลบโฟลเดอร์ .vs ออกได้เลย ดังรูปที่ 1-16



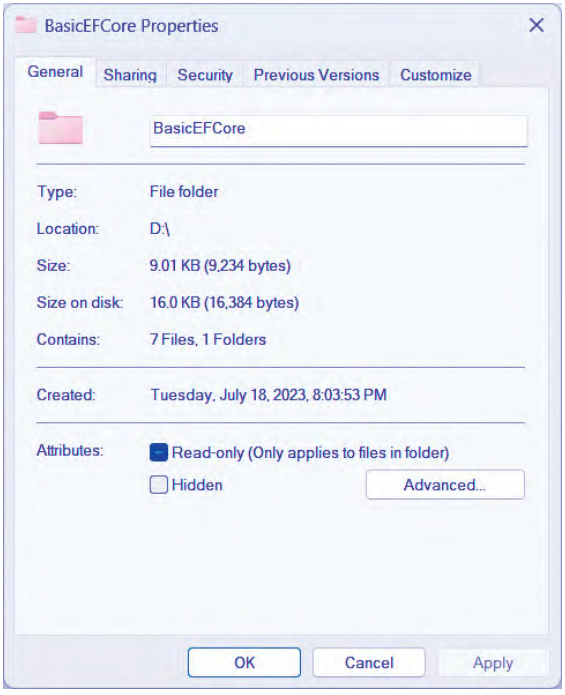
รูปที่ 1-16 แสดงการลบโฟลเดอร์ .vs

3. เข้าไปในโฟลเดอร์ย่อยอีก 1 ระดับ ให้ลบโฟลเดอร์ bin และ obj ออก ถือว่าเสร็จสิ้นแล้ว ดังรูปที่ 1-17



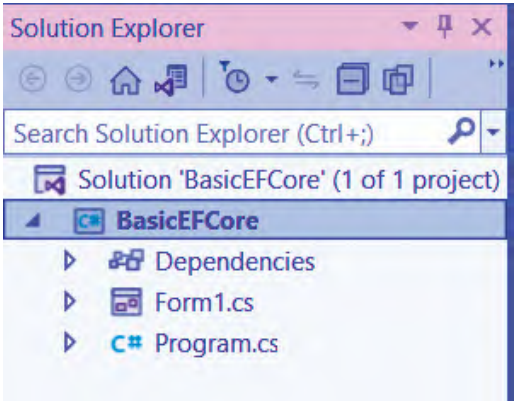
รูปที่ 1-17 แสดงการลบโฟลเดอร์ย่อย bin และ obj

4. ให้ลองตรวจสอบขนาดโปรเจ็กต์อีกครั้ง พบว่าขนาดโปรเจ็กต์จะลดลงแล้ว ผู้อ่านสามารถนำโปรเจ็กต์ .NET ไปจัดเก็บได้แล้ว ดังรูปที่ 1-18



รูปที่ 1-18 แสดงขนาดโปรเจ็กต์หลังปรับปรุงแล้ว

5. การเปิดโปรเจ็กต์ที่มีการปรับปรุงแบบนี้ ต้องรอให้เครื่องหมายตกใจ (!) ในหน้าต่าง Solution Explorer หายไปก่อน ก็จะใช้งานโปรเจ็กต์ได้ตามปกติ

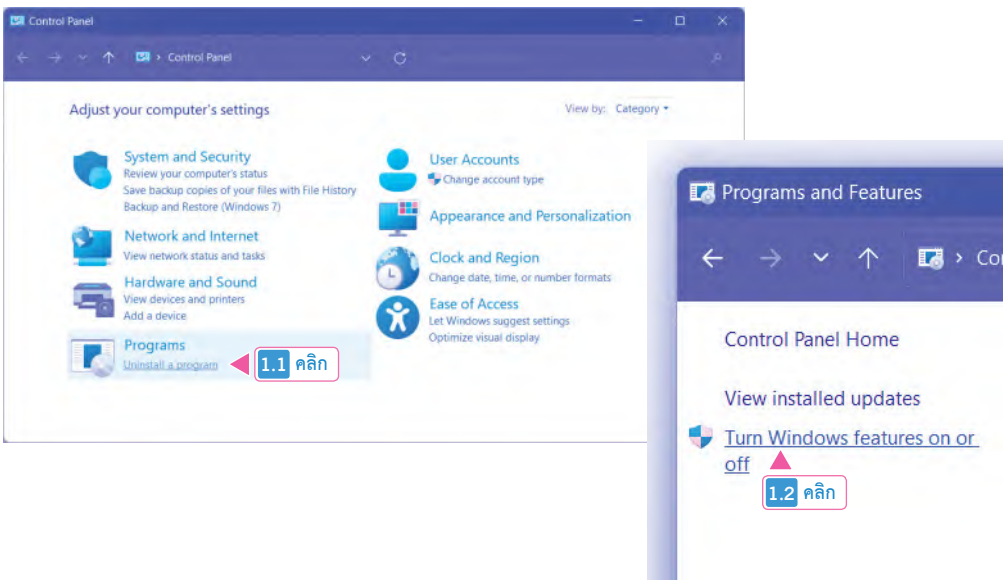


รูปที่ 1-19 แสดงการเปิดโปรเจ็กต์ที่มีการปรับปรุงขนาด

การดาวน์โหลด Android Emulator เพิ่มเติมเพื่อทดสอบโปรเจ็กต์แบบ Android Apps

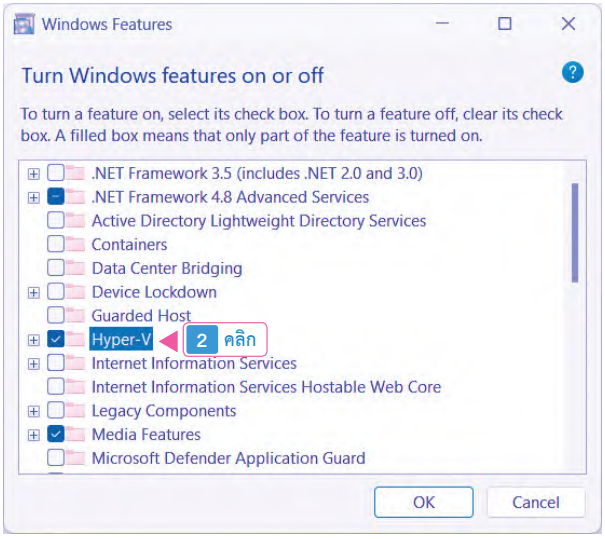
โปรเจ็กต์ประเภท .NET MAUI กับ .NET MAUI Blazor สามารถทดสอบแอปในรูปแบบ Android Apps ได้เช่นกัน โดยที่ผู้อ่านสามารถสร้าง Android Emulator ขึ้นมาเพื่อทดสอบโปรเจ็กต์ในขั้นต้นได้อีกด้วย มีขั้นตอนดังนี้

1. ใน Windows 10/11 ไปที่ Control Panel เลือกรายการ Uninstall a program > Turn Windows features on or off ดังรูปที่ 1-20



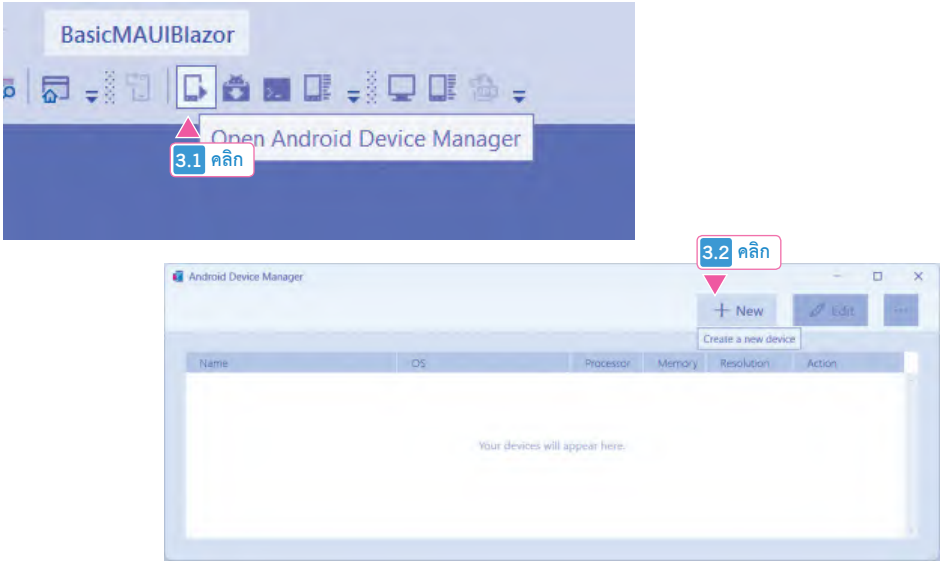
รูปที่ 1-20 แสดงการเลือกคำสั่งเปิด/ปิดฟีเจอร์ของ Windows 10/11

2. ที่หน้าต่าง Windows Features ให้เปิดฟีเจอร์ Hyper-V แล้ว Restart เครื่อง 1 ครั้งก่อน ดังรูปที่ 1-21



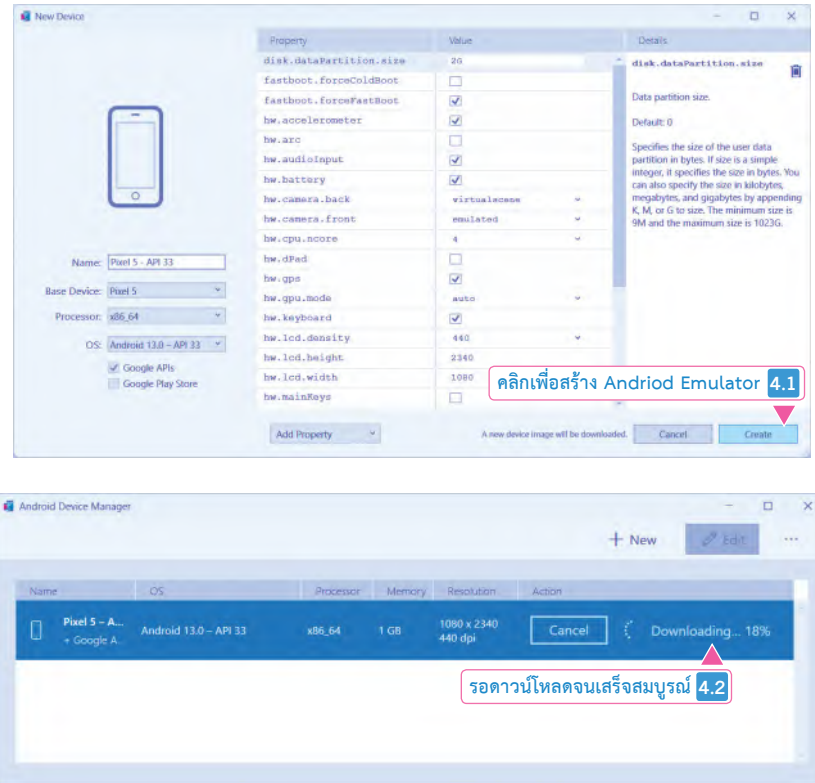
รูปที่ 1-21 แสดงการเปิดฟีเจอร์ Hyper-V ของ Windows 10/11

3. ที่โปรแกรม Visual Studio ให้เปิด Android Device Manager ขึ้นมาก่อน จากนั้นคลิกที่ปุ่ม + New เพื่อเริ่มต้นดาวน์โหลดและติดตั้ง Android Emulator ดังรูปที่ 1-22



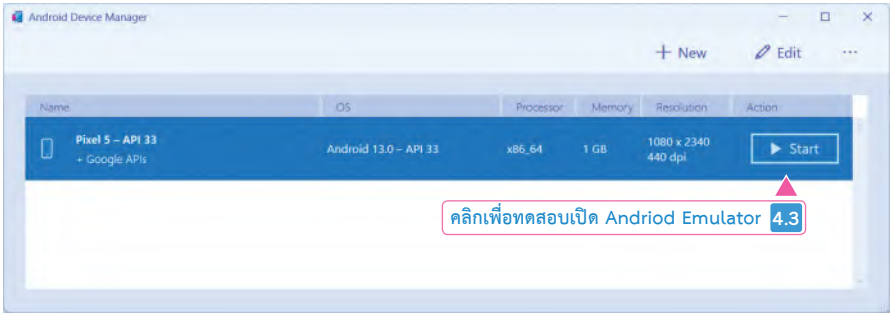
รูปที่ 1-22 หน้าต่างแสดงรายการ Android Emulator ในเครื่องของเรา

4. ผู้อ่านสามารถกำหนดรายละเอียดให้กับ Android Emulator ที่จะสร้างขึ้นมาได้อย่างอิสระ แต่ผู้เขียนมีข้อแนะนำว่า ใช้ค่า default ก็ได้เช่นกัน ดังรูปที่ 1-23



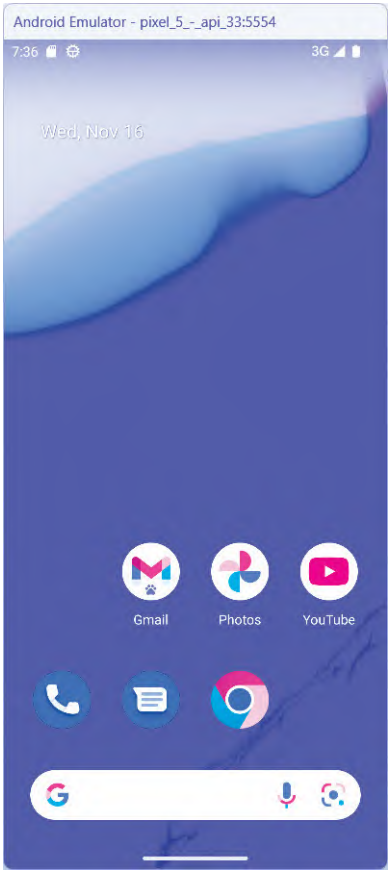
รูปที่ 1-23 แสดงการดาวน์โหลด Android Emulator รายการแรกของเครื่องนี้

จากรูปที่ 1-23 การดาวน์โหลดและติดตั้ง Android Emulator เป็นแบบอัตโนมัติทั้งหมด ผู้อ่านสามารถสร้าง Android Emulator ได้หลายรายการ เพื่อทดสอบส่วนแสดงผลหลายขนาดได้อีกด้วย



รูปที่ 1-24 แสดง Android Emulator ที่ได้มา

5. ให้ทดสอบเปิด Android Emulator ขึ้นมา การทำงานครั้งแรกจะใช้เวลานานมากกว่าปกติ ดังรูปที่ 1-25



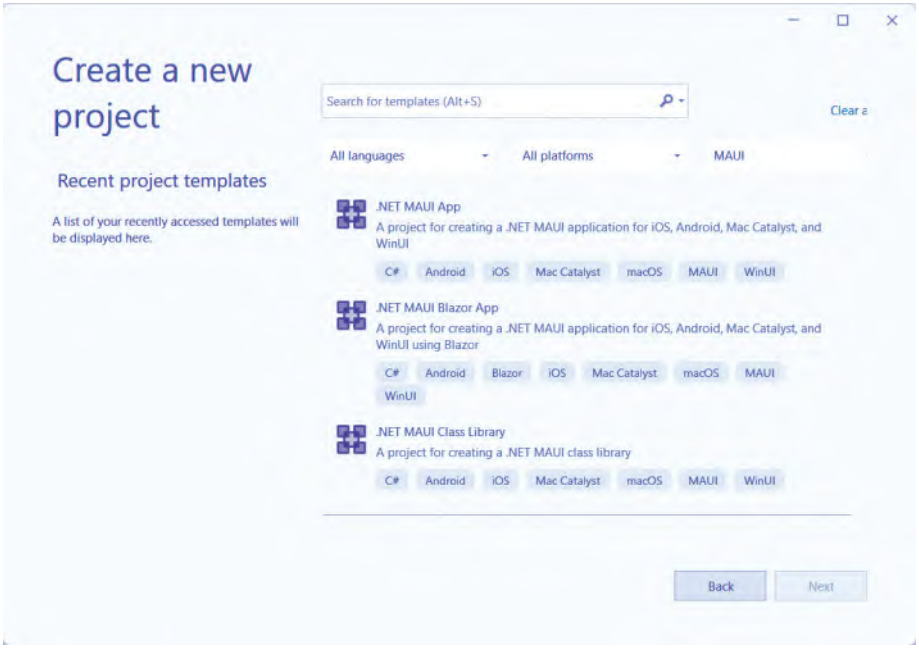
รูปที่ 1-25 แสดงการเปิด Android Emulator

การติดตั้ง Android Emulator มากกว่า 1 เวอร์ชันในเวลาเดียวกันถือเป็นเรื่องปกติ ใช้สำหรับทดสอบโปรเจ็กต์ในขั้นต้น และต้องการตรวจสอบว่าส่วนแสดงผล (UI) ในหน้าจอที่มีความละเอียดแตกต่างกันให้ผลเป็นอย่างไร

ประเภทโปรเจ็กต์ .NET MAUI

การพัฒนาแอปแบบ .NET MAUI สามารถแบ่งออกได้เป็น 2 ประเภทหลัก คือ

1. **.NET MAUI App** ประกอบด้วย
   - **สร้างส่วนแสดงผล** (User Interface – UI) ด้วยสคริปต์ภาษา XAML มีไฟล์นามสกุล .xaml
   - **โค้ดการทำงาน** อาศัยภาษา C# มีไฟล์นามสกุล .cs
2. **.NET MAUI Blazor App** เป็นโปรเจ็กต์ที่อาศัยเทคโนโลยีฝั่งเว็บเป็นแกนหลัก ใน .NET เรียกว่า โปรเจ็กต์ Blazor ประกอบด้วย
   - **สร้างส่วนแสดงผล** ด้วยสคริปต์ภาษา HTML + ส่วนขยายเพิ่มเติม มีไฟล์นามสกุล .razor
   - **โค้ดการทำงาน** อาศัยภาษา C# มีไฟล์นามสกุล .cs

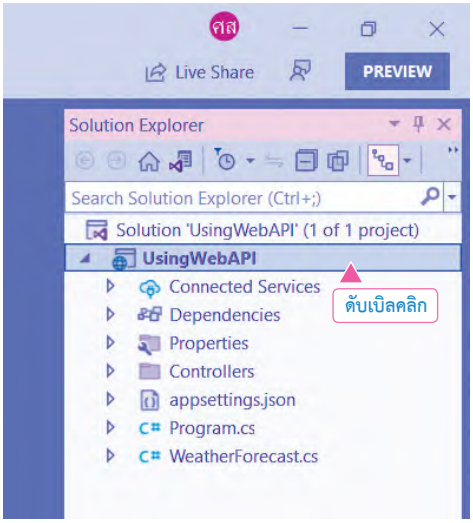


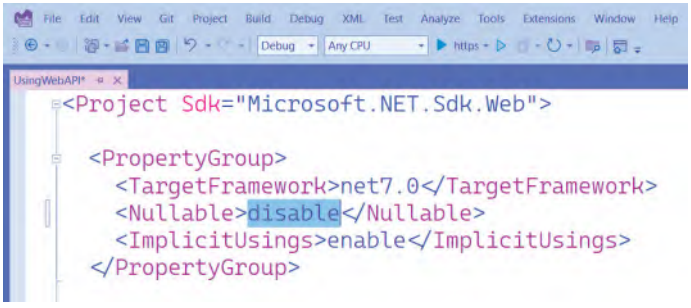
รูปที่ 1-26 แสดงหน้าจอสร้างโปรเจ็กต์ .NET MAUI ทั้ง 2 ประเภท

เนื้อหาของหนังสือเล่มนี้นำเสนอเฉพาะโปรเจ็กต์ประเภท .NET MAUI เท่านั้น

การยกเลิกฟีเจอร์ Nullable ของโปรเจ็กต์ ASP.NET Core Web API

สำหรับโปรเจ็กต์ประเภท ASP.NET Core Web API ที่เราจะใช้ในงานฝั่งหลังบ้าน (Back End) ผู้เขียนแนะนำให้ผู้อ่านยกเลิกฟีเจอร์ Nullable เป็นลำดับแรกก่อนเสมอ เพราะว่ายังไม่ได้ใช้ประโยชน์อะไรกับฟีเจอร์นี้ โดยการดับเบิลคลิกที่ชื่อโปรเจ็กต์ก่อน จากนั้นแก้ไขค่าคุณสมบัติ Nullable ให้มีค่าเป็น disable ดังรูปที่ 1-27





รูปที่ 1-27 แสดงการยกเลิกฟีเจอร์ Nullable

จากรูปที่ 1-27 ให้เลื่อนมาด้านล่างจนถึงหัวข้อ Nullable ให้เลือกรายการ Disable เพื่อยกเลิกใช้งานฟีเจอร์นี้

สรุปท้ายบท

เนื้อหาในบทนี้เป็นเพียงการเตรียมสภาพแวดล้อมให้พร้อมกับการพัฒนาแอปด้วยโปรเจ็กต์ประเภท .NET MAUI นำเสนอเป็นเนื้อหาตั้งแต่บทที่ 2 เป็นต้นไป

บทที่ 2

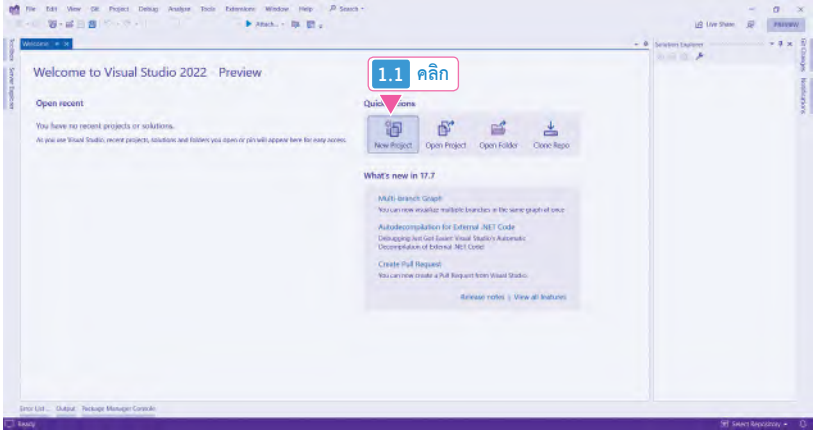
พื้นฐานการพัฒนาแอปด้วย .NET MAUI

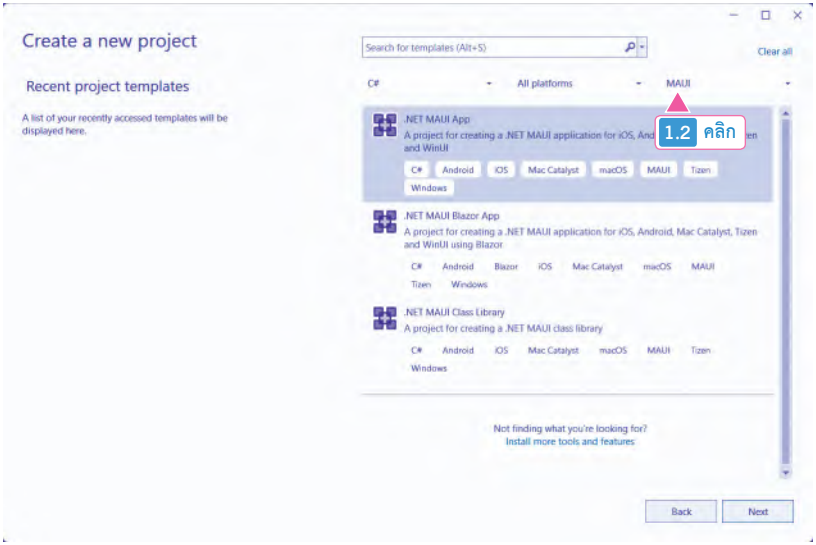
เนื้อหาลำดับแรกของการพัฒนาแอปแบบ .NET MAUI ก็คือ เราจะมาทำความรู้จักกับโครงสร้างโปรเจ็กต์ .NET MAUI ในขั้นต้นก่อนว่ามีโครงสร้างและมีหลักการทำงานอย่างไร

พื้นฐานการสร้างโปรเจ็กต์ .NET MAUI

มีขั้นตอนดังนี้

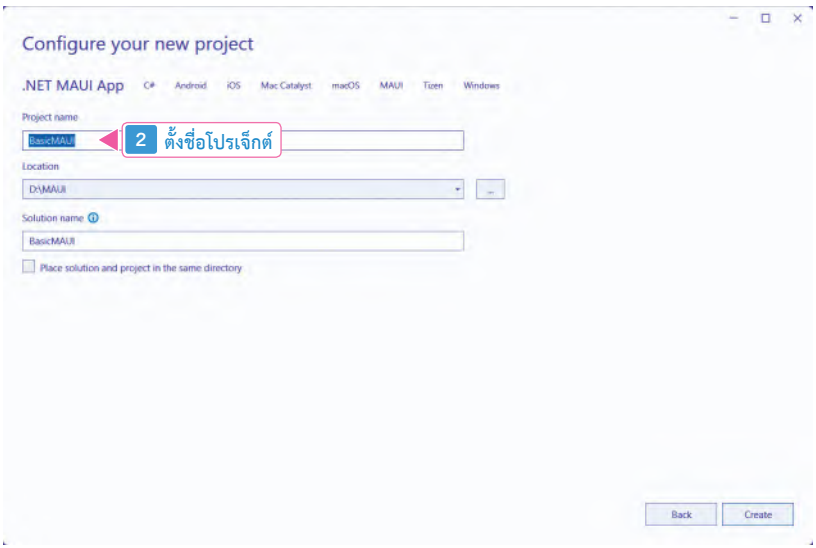
1. ที่หน้าจอเริ่มต้นของโปรแกรม Visual Studio ให้ผู้อ่านเลือก Create a new project เพื่อเริ่มต้นสร้างโปรเจ็กต์ใหม่ ในกรณีนี้ผู้เขียนต้องการสร้างโปรเจ็กต์ประเภท .NET MAUI ทำได้โดยการเลือกที่ช่อง All project types กำหนดให้แสดงรายการโปรเจ็กต์เฉพาะ .NET MAUI เท่านั้น ดังรูปที่ 2-1





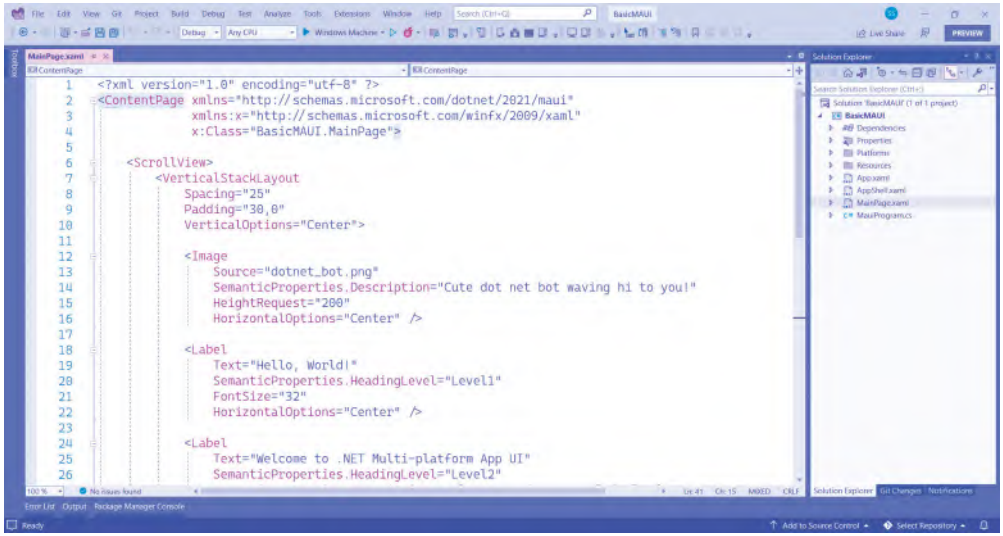
รูปที่ 2-1 แสดงการสร้างโปรเจ็กต์แบบ .NET MAUI App

2. ในกรณีนี้โปรเจ็กต์ .NET MAUI App แรกตั้งชื่อว่า BasicMAUI เก็บไว้ที่พาธ D:\MAUI (หรือพาธอื่นตามที่ผู้อ่านต้องการ) ดังรูปที่ 2-2



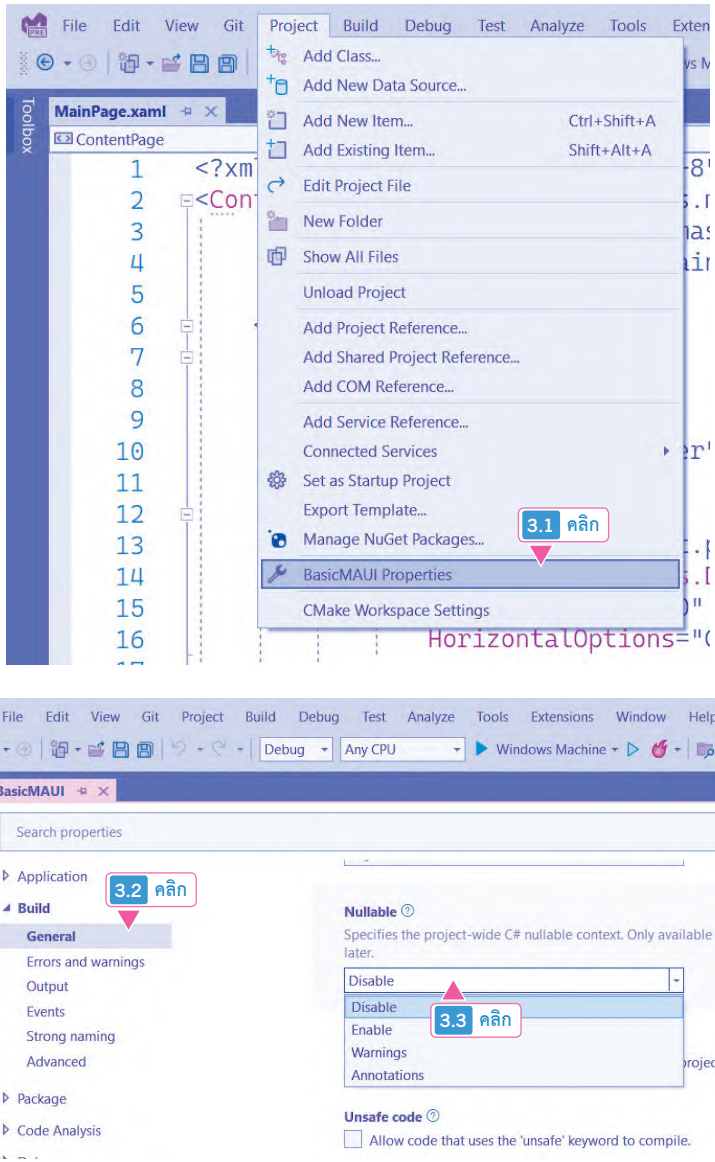
รูปที่ 2-2 แสดงการตั้งชื่อโปรเจ็กต์และพาธที่จัดเก็บ

จากรูปที่ 2-2 ให้รอจนกระทั่งโปรแกรม Visual Studio สร้างโปรเจ็กต์แบบ .NET MAUI จนเสร็จสมบูรณ์ ถือเป็นโครงสร้างโปรเจ็กต์เริ่มต้นของเรา อาจจะมีโครงสร้างโปรเจ็กต์เริ่มต้นเหมือนหรือไม่เหมือนของผู้เขียนก็ได้ ไม่ถือเป็นเรื่องสำคัญ เพราะว่าเมื่อผู้อ่านเข้าใจหลักการทำงานแล้ว โครงสร้างเริ่มต้นโปรเจ็กต์จะมีรูปแบบใดก็ได้



รูปที่ 2-3 แสดงโครงสร้างโปรเจ็กต์เริ่มต้นแบบ .NET MAUI

3. หลังจากที่ผู้อ่านได้โปรเจ็กต์ .NET MAUI เริ่มต้นมาแล้ว ขอให้ปรับแต่งการทำงานในขั้นต้นก่อนทุกครั้ง โดยการเลือกเมนู Project > Properties ดังรูปที่ 2-4



รูปที่ 2-4 แสดงการปรับแต่งโปรเจ็กต์ .NET MAUI เบื้องต้น

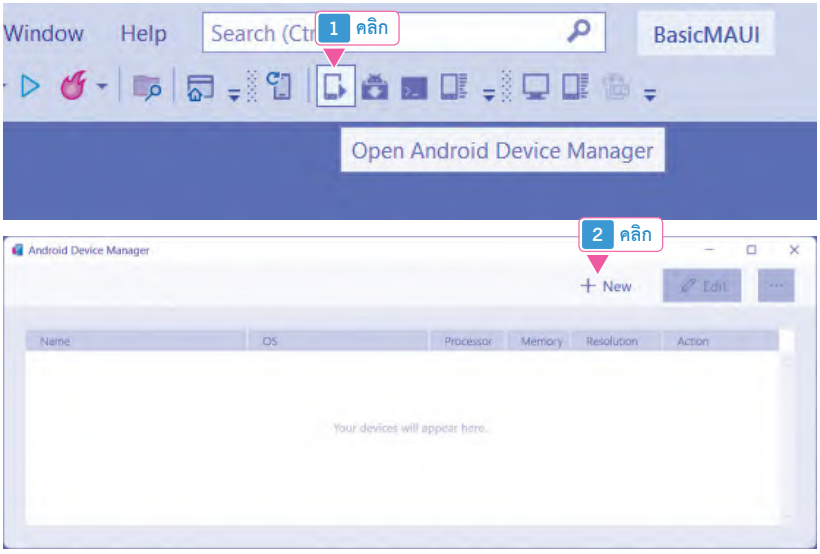
จากรูปที่ 2-4 ที่รายการ Build > General เลื่อนมาที่รายการ Nullable ให้ผู้อ่านยกเลิกใช้งานฟีเจอร์ Nullable โดยการกำหนดค่า Disable ก่อน ถือว่าโปรเจ็กต์ .NET MAUI ของเราพร้อมใช้งานแล้ว (ใน .NET เวอร์ชันใหม่ ไม่ต้องกำหนดแล้ว)

การสร้าง Android Emulator ของ Visual Studio สำหรับทดสอบ Android Apps

ในขั้นต้นนี้เราจะสนใจโปรเจ็กต์ .NET MAUI เฉพาะ Android Apps เป็นลำดับแรกก่อน โดยที่เรายังไม่ต้องแก้ไขหรือเขียนโค้ดใดๆ เพิ่มเติม เราต้องการทดสอบว่า Android Apps ที่ได้มามีหน้าตาเป็นอย่างไร

การทดสอบการทำงานของโปรเจ็กต์ Android Apps ในขั้นต้นคือ อาศัย Android Emulator ที่มากับโปรแกรม Visual Studio มีขั้นตอนดังนี้

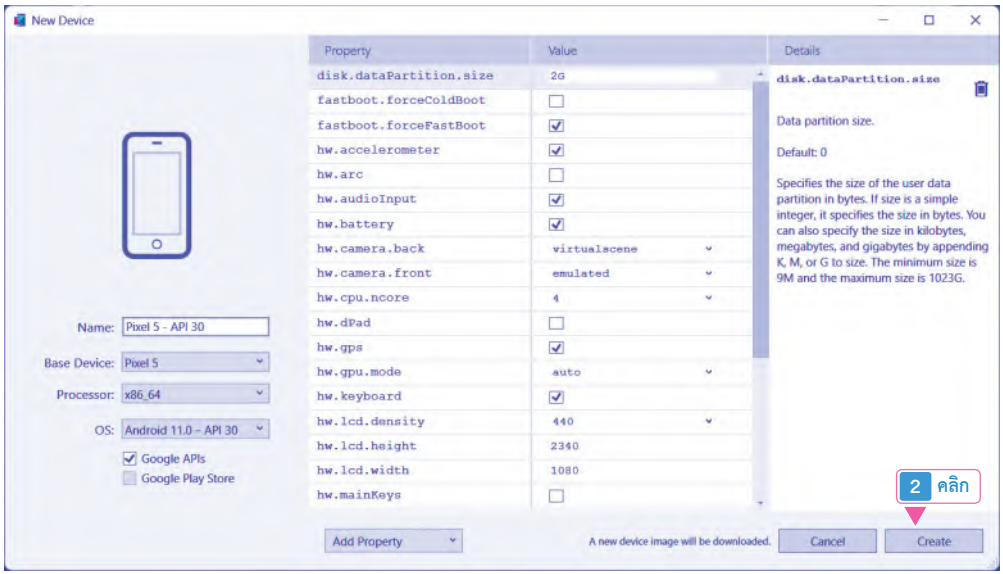
1. ในโปรแกรม Visual Studio ยังไม่มี Android Emulator ใดๆ ทั้งสิ้น ให้ผู้อ่านคลิกที่ปุ่ม  จากนั้นคลิกที่ปุ่ม + New เริ่มต้นสร้าง Android Emulator เพื่อทดสอบโปรเจ็กต์ .NET MAUI ประเภท Android Apps ดังรูปที่ 2-5



รูปที่ 2-5 แสดงการเริ่มต้นสร้าง Android Emulator

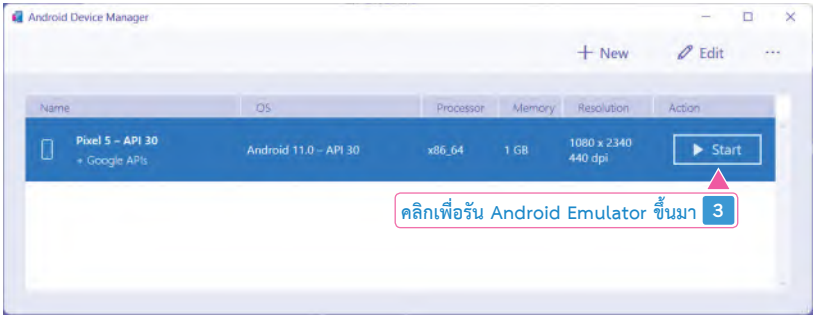
2. เป็นขั้นตอนการกำหนดรายละเอียดให้กับ Android Emulator เพื่อจำลองให้มีการทำงานเหมือนกับมือถือเครื่องจริง ที่ต้องกำหนด ได้แก่
   - **Name:** ชื่อ Android Emulator
   - **Base Device:** ให้ผู้อ่านเลือกรุ่นมือถือที่ต้องการจำลองการทำงาน
   - **Processor:** ให้ผู้อ่านเลือก CPU แบบ x86_x64 ตรงกับ CPU ที่ใช้งานในเครื่อง PC ของเรา

- **OS:** เลือกเวอร์ชันของระบบปฏิบัติการ Android ตามที่ผู้อ่านต้องการ
- ตัวเลือก Google APIs และ Google Play Store เป็นฟีเจอร์หลักของมือถือตระกูล Android



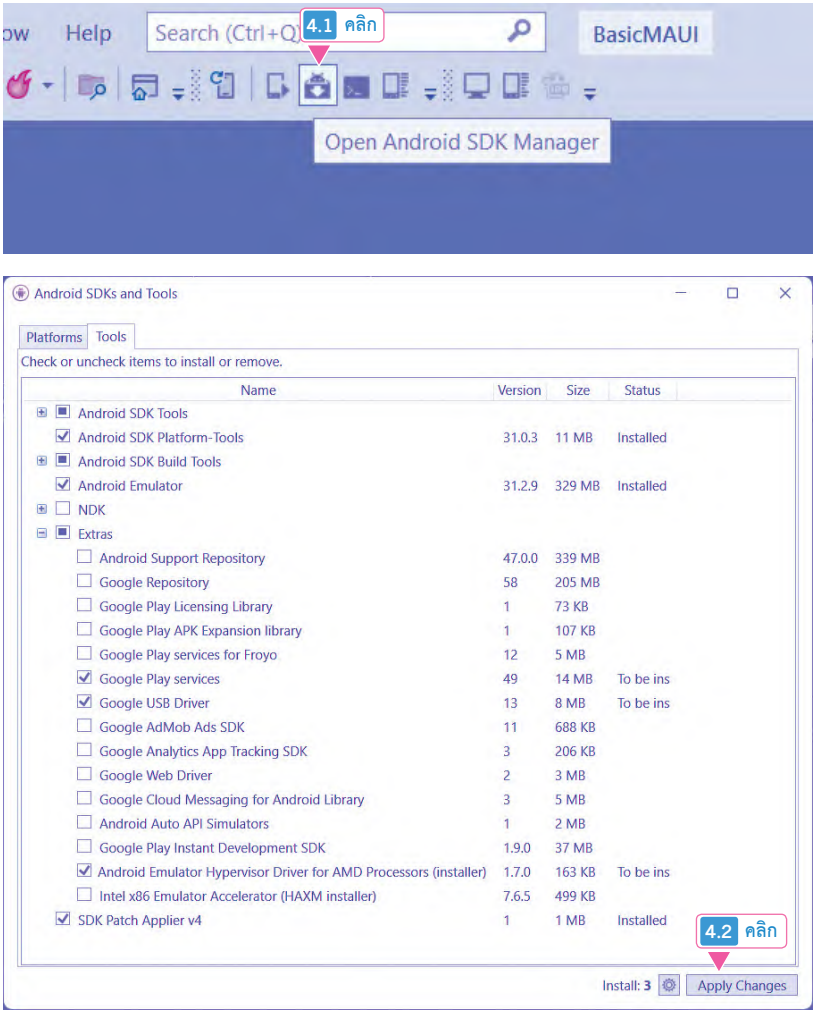
รูปที่ 2-6 แสดงการกำหนดรายละเอียดให้กับ Android Emulator

3. เข้าสู่ขั้นตอนการดาวน์โหลดรายการต่างๆ เพื่อสร้าง Android Emulator ขึ้นมาตามที่ผู้อ่านกำหนดไว้ เป็นขั้นตอนอัตโนมัติทั้งหมด ให้ผู้อ่านรอจนเสร็จสมบูรณ์ก็จะได้ Android Emulator สำหรับทดสอบโปรเจ็กต์ .NET MAUI แบบ Android ในขั้นต้น ดังรูปที่ 2-7



รูปที่ 2-7 แสดง Android Emulator ที่ได้มา

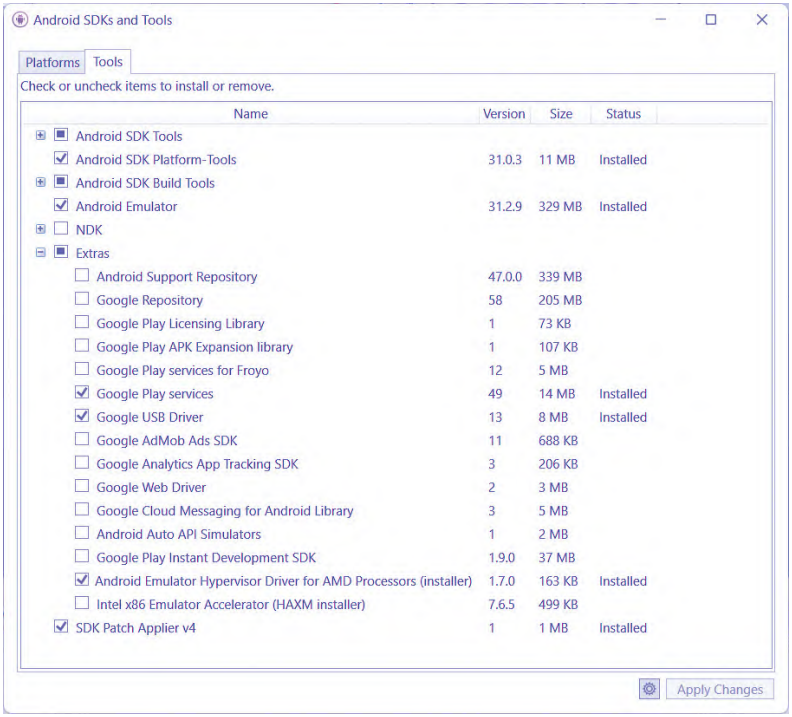
4. ในโปรแกรม Visual Studio ให้ผู้อ่านคลิกปุ่ม  ที่แท็บ Tools เลือกรายการติดตั้งเพิ่มเติมสำหรับโปรเจ็กต์ Android Apps ดังรูปที่ 2-8



รูปที่ 2-8 แสดงรายการติดตั้งเพิ่มเติมสำหรับโปรเจ็กต์ Android Apps

จากรูปที่ 2-8 รายการที่ต้องติดตั้ง ประกอบด้วย

- Google Play services
- Google USB Driver
- Intel x86 Emulator Accelerator (HAXM installer) สำหรับ CPU ของ Intel หรือรายการ Android Emulator Hypervisor Driver for AMD Processors (installer) สำหรับ CPU ของ AMD ให้เลือกติดตั้งตาม CPU ที่ผู้อ่านใช้งานเพียงรายการเดียว



รูปที่ 2-9 แสดงรายการไฟล์ติดตั้งเพิ่มเติมสำหรับโปรเจ็กต์ Android Apps

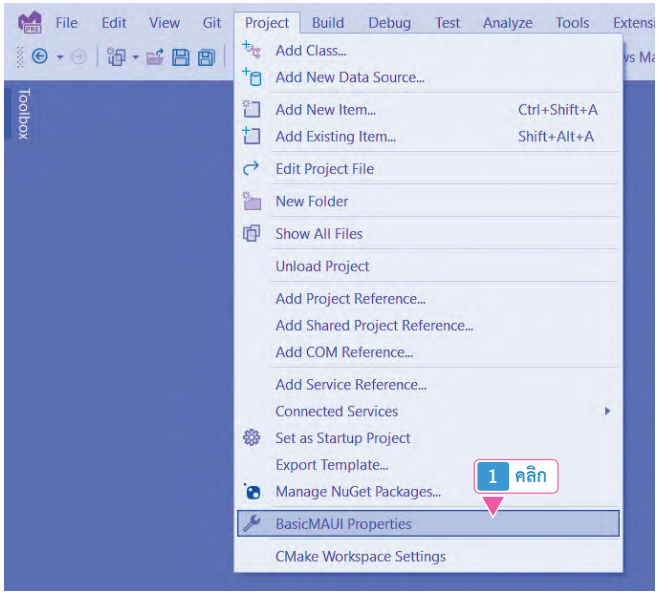
จากรูปที่ 2-9 ให้ผู้อ่านรอขั้นตอนการดาวน์โหลดและติดตั้งจนเสร็จสมบูรณ์

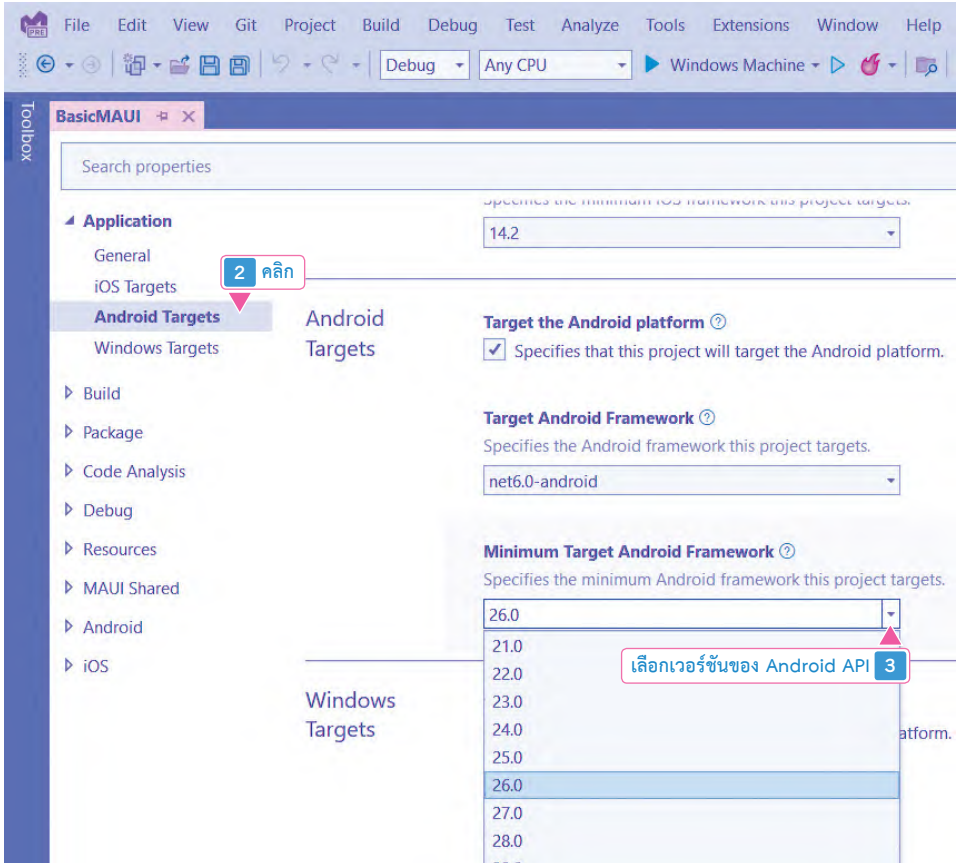
การปรับช่วงเวอร์ชันของ Android API

ในกรณีที่มีการอัพเดต Android SDK เวอร์ชันใหม่เข้ามา ผู้อ่านสามารถปรับแต่งช่วงเวอร์ชันโปรเจ็กต์ Android ของผู้อ่านให้รองรับเวอร์ชันใหม่ๆ ได้เช่นกัน แต่การปรับปรุงโปรเจ็กต์ให้เป็นเวอร์ชันใหม่ดังกล่าว ไม่ได้หมายความว่าผู้อ่านสามารถใช้ฟีเจอร์ใหม่ๆ เหล่านี้ได้ทันที อาจจะต้องรออัพเดต .NET MAUI ให้เป็นเวอร์ชันใหม่ควบคู่กันไปด้วย (ผู้อ่านต้องรออัพเดตผ่านทาง Visual Studio) จึงจะสามารถใช้งานฟีเจอร์ใหม่ๆ ได้นั่นเอง

การปรับช่วงเวอร์ชันของ Android API ให้ผู้อ่านเลือกเมนู Project > Properties ที่แท็บ Android Targets โดยที่

- **ช่อง Minimum Target Android Framework** หมายถึง Android เวอร์ชันต่ำสุดที่สามารถติดตั้งแอปของเราได้
- **ช่อง Target Android Framework** หมายถึง กำหนดเวอร์ชันหลักสำหรับ compile โปรเจ็กต์ เป็นไปตามเวอร์ชันที่ผู้อ่านดาวน์โหลดมา





รูปที่ 2-10 แสดงการปรับแต่งช่วงเวอร์ชันของโปรเจ็กต์ Android Apps

NOTE

การกำหนดเวอร์ชันต่ำสุดของ Android API ผู้เขียนมีข้อแนะนำเพิ่มเติมว่า ขอให้ผู้อ่านติดตามการประกาศจากฝั่ง Google เสมอว่า Play Store มีเงื่อนไขใดๆ เพิ่มเติมหรือไม่ ขอให้ผู้อ่านทำตามประกาศดังกล่าวอย่างเคร่งครัด ในช่วงเวลาที่ผู้เขียนแต่งต้นฉบับหนังสือเล่มนี้อยู่มีข้อแนะนำว่า ควรจะปรับเวอร์ชันต่ำสุดคือ Android 8.0 (API Level – 26) ขึ้นไป

ผลของการปรับเวอร์ชันต่ำสุดเป็น Android 8.0 หมายความว่า Android Apps ที่ผู้อ่านสร้างขึ้นมาสามารถติดตั้งบนระบบปฏิบัติการ Android 8.0 ขึ้นไปเท่านั้น

การอัพเดต Android SDK ของ Visual Studio

สำหรับโปรเจ็กต์ Android ของ Visual Studio ผู้อ่านสามารถอัพเดต Android SDK เป็นเวอร์ชันใหม่ล่าสุด เพื่อให้ Android Apps ของผู้อ่านรองรับฟีเจอร์ใหม่ๆ ได้นั่นเอง

Warning

การอัพเดต Android SDK “ต้องทำในขณะที่โปรแกรม Visual Studio และ .NET MAUI มีการอัพเดตให้รองรับ Android SDK เวอร์ชันใหม่แล้วเท่านั้น” โดยการติดตามข่าวสารจากไมโครซอฟท์อยู่เสมอ ยึดตามช่วงเวลาของผู้อ่านเป็นหลักว่าในช่วงเวลาของผู้อ่านมีการอัพเดตโปรแกรม Visual Studio และ .NET MAUI รองรับ Android SDK ถึงเวอร์ชันใดแล้ว

หมายความว่าถ้าผู้อ่านอัพเดต Android SDK เวอร์ชันใหม่ แต่โปรแกรม Visual Studio และ .NET MAUI ไม่รองรับ การอัพเดตดังกล่าวไม่มีประโยชน์แต่อย่างใด อาจจะก่อให้เกิดข้อผิดพลาดระหว่างเวอร์ชันได้อีกด้วย

การทดสอบโปรเจ็กต์ .NET MAUI แบบ Android Apps ด้วย Android Emulator

เนื้อหาที่นำเสนอในหนังสือเล่มนี้เป็นการใช้งาน .NET MAUI บนเครื่อง PC เป็นหลัก ส่งผลให้การทดสอบโปรเจ็กต์ทั้งหมดรันแบบ Android Apps มีความเหมาะสมที่สุด

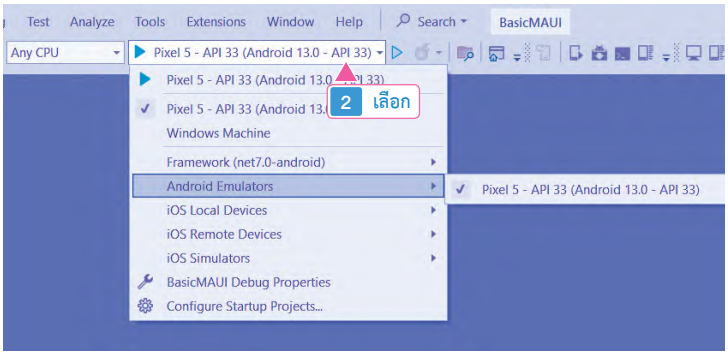
การทดสอบโปรเจ็กต์ .NET MAUI แบบ Android Apps มี 2 วิธี คือ

1. **ทดสอบด้วย Android Emulator** เป็นการทดสอบการทำงานโปรเจ็กต์ในขั้นต้น มีข้อดีก็คือ เราสามารถเห็นผลการทำงานใน PC ของเราได้เลย แต่มีข้อเสียก็คือ ความเร็วการทดสอบเป็นไปตามสเปคเครื่องของเรา และมีข้อจำกัดของฟีเจอร์ต่างๆ ในระดับหนึ่ง ในยุคปัจจุบันผู้เขียนไม่แนะนำให้ทดสอบโปรเจ็กต์ด้วย Android Emulator แล้ว ใช้เท่าที่จำเป็น

2. **ทดสอบด้วยมือถือ Android จริง** เป็นวิธีที่ผู้เขียนแนะนำให้ใช้เป็นหลัก เพราะว่าเราได้ทดสอบและเห็นผลการทำงานบนมือถือจริง และยังช่วยลดระยะเวลาทดสอบโปรเจ็กต์ในแต่ละครั้งอีกด้วย เกิดมาจากการที่เราไม่ต้องสร้าง Android Emulator ใน PC ของเรานั่นเอง

ลำดับแรกเราจะมาทดสอบโปรเจ็กต์ BasicMAUI ด้วย Android Emulator ก่อน โดยการคลิกที่หัวลูกศรในปุ่ม 

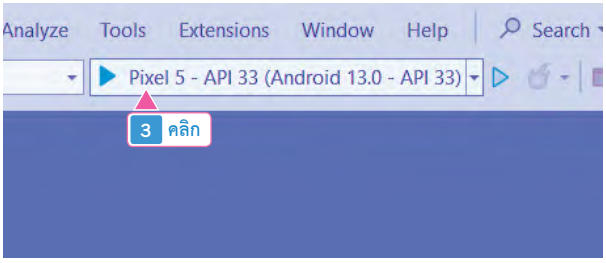
จากนั้นเลือกโฟกัสโปรเจ็กต์ .NET MAUI เป็นแบบ Android Emulators ดังรูปที่ 2-11

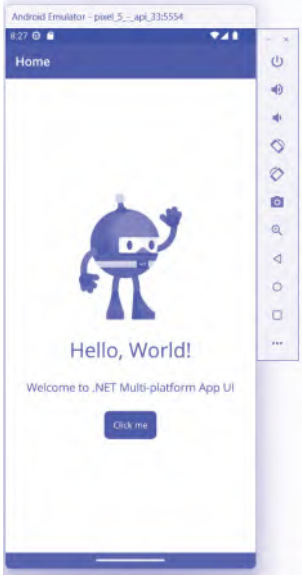


รูปที่ 2-11 แสดงการทดสอบโปรเจ็กต์ .NET MAUI แบบ Android Apps

3. ให้คลิกที่ปุ่ม 

เพื่อรันโปรเจ็กต์ .NET MAUI แบบ Android Apps ดังรูปที่ 2-12





รูปที่ 2-12 ผลการรันโปรเจ็กต์ BasicMAUI ใน Android Emulator

จากรูปที่ 2-12 ถือว่าโปรเจ็กต์ .NET MAUI ของเรารันในรูปแบบ Android Apps ได้แล้ว

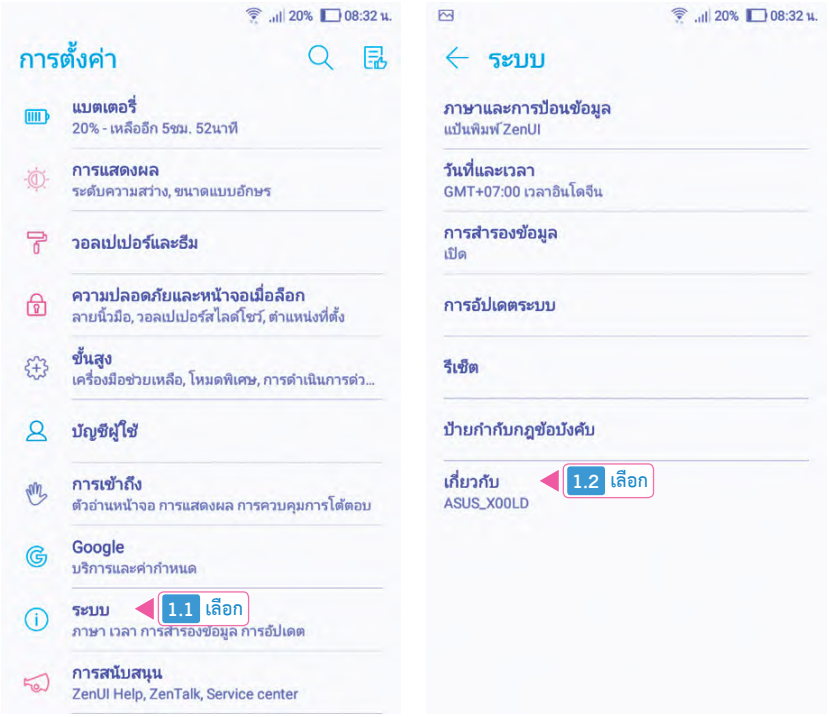
การทดสอบโปรเจ็กต์ Android Apps บนเครื่องจริง (ใช้กับ Android 8.0 ขึ้นไปเท่านั้น)

เพื่อให้การทดสอบผลการทำงานของ Android Apps สมบูรณ์ยิ่งขึ้น การทดสอบโปรเจ็กต์ Android Apps บนเครื่องจริงถือเป็นช่องทางที่ดีที่สุดที่จะทดสอบการทำงานและฟีเจอร์ต่างๆ ในแอปของผู้อ่าน โดยเฉพาะอย่างยิ่งการใช้งานกล้อง, แผนที่ เป็นต้น

มือถือหรือแท็บเล็ต Android ที่จะนำมาใช้ทดสอบต้องมีการเปิดโหมดนักพัฒนา (Developer options) ก่อน ถ้ามือถือหรือแท็บเล็ตของผู้อ่านเคยทำมาก่อนแล้ว ไม่จำเป็นต้องทำซ้ำ สามารถใช้ทดสอบโปรเจ็กต์ได้เลย

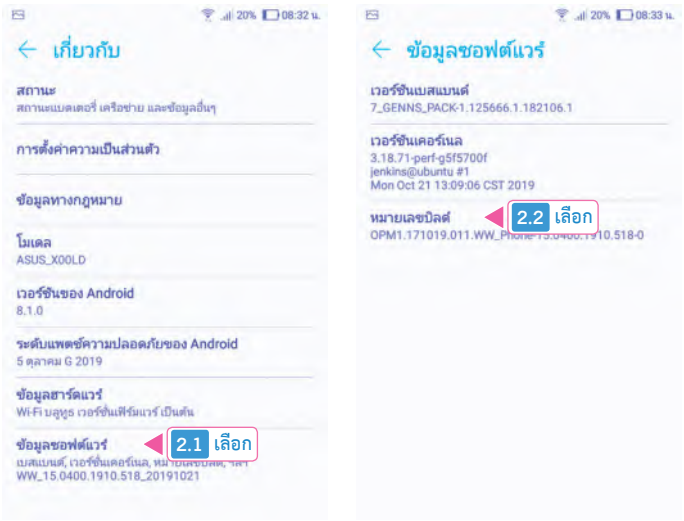
ขั้นตอนการกำหนดในเครื่องจริงของแต่ละยี่ห้อ แต่ละรุ่น อาจจะมีรายละเอียดปลีกย่อยแตกต่างกันไปบ้าง อาจจะใช้คำที่ไม่เหมือนกัน แต่โดยรวมแล้วใช้หลักการเดียวกันหมด นั่นคือเปิดโหมด Debug ขึ้นมาในเครื่อง มีขั้นตอนดังนี้

1. ให้ผู้อ่านเลือก Settings (ตั้งค่า) > ระบบ (System) > About (เกี่ยวกับ) ดังรูปที่ 2-13



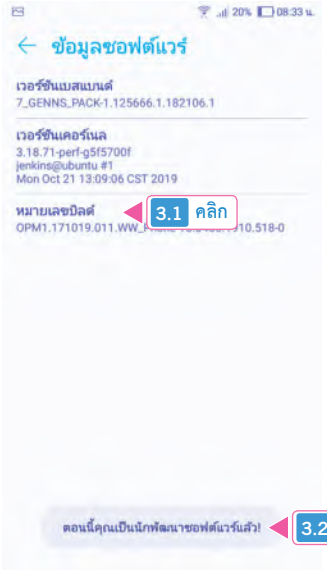
รูปที่ 2-13 แสดงการดูข้อมูลเครื่องที่จะนำมาทดสอบ

2. ให้ผู้อ่านเลือกข้อมูลซอฟต์แวร์ (Software Infomation) > หมายเลขบิลด์ (Build number) ดังรูปที่ 2-14



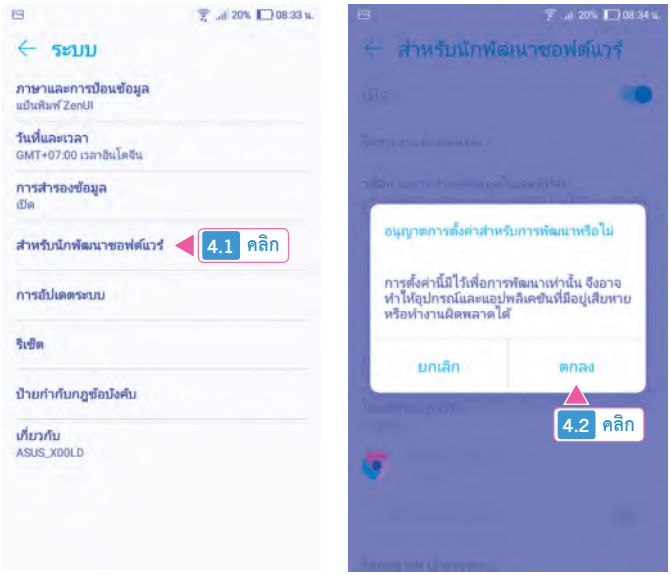
รูปที่ 2-14 แสดงหมายเลข Build number

3. ให้ผู้อ่านสัมผัสรายการ Build number (หมายเลขบิลด์) 7 ครั้ง ก็จะมีข้อความแจ้งว่าเปิดโหมดนักพัฒนาขึ้นมาให้ผู้อ่านทราบ ถือว่าเสร็จสมบูรณ์แล้ว หมายความว่าเครื่องของผู้อ่านเปิดโหมดนักพัฒนา ส่งผลให้สามารถทดสอบโปรเจ็กต์ Android Apps จากภายนอกได้แล้ว ดังรูปที่ 2-15



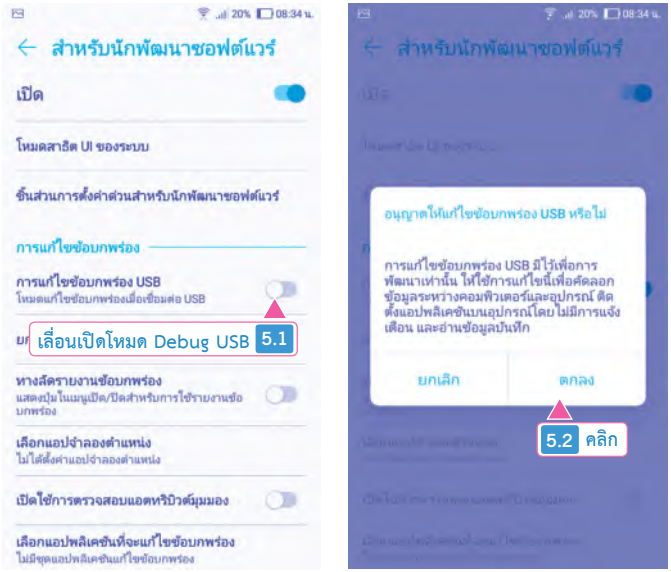
รูปที่ 2-15 แสดงการเปิดโหมดนักพัฒนาในเครื่องจริง

4. ให้ผู้อ่านกดปุ่ม Back กลับมา จะพบว่ามีรายการสำหรับนักพัฒนาซอฟต์แวร์เพิ่มขึ้นมาใหม่ ให้เข้าไปเปิดโหมดสำหรับนักพัฒนา ดังรูปที่ 2-16



รูปที่ 2-16 แสดงการเปิดโหมดสำหรับนักพัฒนา

5. ให้ผู้อ่านเปิดโหมด Debug ขึ้นมา ถือว่าเครื่องของผู้อ่านพร้อมทดสอบโปรเจ็กต์ Android Apps แล้ว ดังรูปที่ 2-17



รูปที่ 2-17 แสดงการเปิดโหมด Debug ขึ้นมา