

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



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.



คู่มือพัฒนาแอปพลิเคชัน Multi-Platform ด้วย .NET MAUI

Author

ศุภชัย สมพานิช

Editorial

กิตินันท์ พลสวัสดิ์ kitinan_p@idcpremier.com

Production Manager

วราพล ฅนธิกุล

Graphic Designer

ชวนันท์ รัตนะ

Page Layout

ปรีเยศ จันทร์จำปา

Proofreader

พรรณรัตน์ ชูราณี

Publishing Coordinators

สุรียรัตน์ จิ๋ว, นัตรดนัย ศรีสวัสดิ์

Product Specialist

ศรัณย์ คมขำ

Visual Studio Code เป็นเครื่องหมายการค้าของบริษัท Microsoft Corp., MongoDB เป็นเครื่องหมายการค้าของบริษัท MongoDB และเครื่องหมายการค้าอื่นๆ ที่อ้างถึงเป็นของบริษัทนั้นๆ

สงวนลิขสิทธิ์ตามพระราชบัญญัติลิขสิทธิ์ พ.ศ. 2537 โดย บริษัท ไอดีซี พรีเมียร์ จำกัด ห้ามลอกเลียนไม่ว่าส่วนใด ส่วนหนึ่งของหนังสือเล่มนี้ ไม่ว่าในรูปแบบใดๆ นอกจากจะได้รับการอนุญาตเป็นลายลักษณ์อักษรจากผู้จัดพิมพ์เท่านั้น

บริษัท ไอดีซี พรีเมียร์ จำกัด จัดตั้งขึ้นเพื่อเผยแพร่ความรู้ที่มีคุณภาพสู่ผู้อ่านชาวไทย เรายินดีรับงานเขียนของนักวิชาการ และนักเขียนทุกท่าน ท่านผู้สนใจกรุณาติดต่อผ่านทางอีเมลที่ infopress@idcpremier.com หรือทางโทรศัพท์หมายเลข 0-2962-1081 (อัตราโทร 10 คู่สาย) โทรสาร 0-2962-1084

สร้างสรรคโดย



พิมพ์ครั้งที่ 1 กันยายน 2566

ข้อมูลทางบรรณานุกรม

ศุภชัย สมพานิช

คู่มือพัฒนาแอปพลิเคชัน Multi-Platform ด้วย .NET MAUI
นนทบุรี : ไอดีซี, 2566

448 หน้า

1. ภาษาที่ใช้ในการเขียนโปรแกรม

1 ชื่อเรื่อง

005.262

Barcode 885-916-101-058-6

ราคา 480 บาท

พิมพ์ที่ บริษัท ดี.เค.ปรี้นติ้ง จำกัด

441/56 หมู่ 2 ต.บางบ่อ อ.บางบ่อ จ.สมุทรปราการ 10650

โทรศัพท์ 0-2115-9105 โทรสาร 0-2115-9044

PUBLISHED AND DISTRIBUTED BY



บริษัท ไอดีซี พรีเมียร์ จำกัด

200 หมู่ 4 ชั้น 19 ห้อง 1901 อาคารจัสมินอินเตอร์เนชั่นแนล

ทาวเวอร์ 8.แจ้งวัฒนะ อ.ปากเกร็ด จ.นนทบุรี 11120

โทรศัพท์ 0-2962-1081 (อัตราโทร 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



คำนำ

ในยุคปัจจุบันการพัฒนาแอปแบบสร้างเพียง 1 โปรเจกต์แต่ได้แอปหลายประเภทกำลังได้รับความนิยมเป็นอย่างสูง ในแพลตฟอร์มของ .NET มีโปรเจกต์ประเภทนี้เช่นกัน เรียกว่า .NET MAUI ผลผลิตของโปรเจกต์ประเภท .NET MAUI คุณจะได้ทั้ง Desktop App, Web Apps และ Mobile Apps (iOS และ Android) ในเวลาเดียวกัน โดยที่หนังสือเล่มนี้ยึดถือการพัฒนา .NET MAUI บน Windows เป็นหลัก แต่สามารถใช้ใน macOS ได้เช่นกัน อาจจะมีข้อแตกต่างในแง่ขั้นตอนใช้งานบ้าง แต่ไค้ยังคงใช้ได้เกือบทั้งหมด เพราะว่ามีไค้ดบางส่วนถูกสร้างขึ้นมาเฉพาะ Windows หรือเฉพาะ macOS ถือเป็นข้อจำกัดทั่วไป

คุณผู้อ่านท่านใดที่ติดปัญหาหรือต้องการสอบถาม, ให้ข้อเสนอแนะ, ทติติง 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





สารบัญ

บทที่ 1	การพัฒนาแอปด้วย .NET MAUI	1
	การดาวน์โหลดและติดตั้งโปรแกรม Visual Studio 2022	2
	สถานะของ .NET SDK	6
	การติดตั้งฟอนต์ Cascadia Code สำหรับเขียนโค้ดโดยเฉพาะ	7
	การกำหนดให้แสดงหมายเลขบรรทัด	9
	การติดตั้ง .NET SDK เพิ่มเติม	9
	การอัปเดตโปรแกรม Visual Studio	11
	วิธีการจัดเก็บโปรเจกต์ .NET	12
	การดาวน์โหลด Android Emulator เพิ่มเติมเพื่อทดสอบโปรเจกต์แบบ Android Apps.....	15
	ประเภทโปรเจกต์ .NET MAUI	18
	การยกเลิกฟีเจอร์ Nullable ของโปรเจกต์ ASP.NET Core Web API	19
บทที่ 2	พื้นฐานการพัฒนาแอปด้วย .NET MAUI	21
	พื้นฐานการสร้างโปรเจกต์ .NET MAUI	21
	การสร้าง Android Emulator ของ Visual Studio สำหรับทดสอบ Android Apps	25
	การปรับช่วงเวอร์ชันของ Android API	28
	การอัปเดต Android SDK ของ Visual Studio	30
	การทดสอบโปรเจกต์ .NET MAUI แบบ Android Apps ด้วย Android Emulator	30
	การทดสอบโปรเจกต์ Android Apps บนเครื่องจริง (ใช้กับ Android 8.0 ขึ้นไปเท่านั้น) ...	32
	การทดสอบโปรเจกต์ .NET MAUI แบบ Android Apps บนเครื่องจริง	35
	การรันโปรเจกต์ .NET MAUI ในรูปแบบ Windows Apps	36
	ทำความเข้าใจกับโครงสร้างโปรเจกต์ .NET MAUI	39
	การสร้างเหตุการณ์ (Event) ด้วยวิธีการเขียนโค้ด	50
	ทำความเข้าใจกับฟีเจอร์แสดงผลทันที (Hot Reload)	54
	วิธีการจัดเก็บโปรเจกต์ .NET MAUI	55
	การตรวจสอบแพลตฟอร์ม	57
	การตกแต่งแถบข้อความด้วยอีลีเมนต์ <Border>...</Border>	60
	การลงแสงเงาให้กับรูปภาพ (Image Shadow)	62

บทที่ 3	ตำแหน่งและ Layout ขั้นต้นของ .NET MAUI	64
	การจัดเรียงแนวตั้งด้วย VerticalStackLayout	64
	การจัดเรียงแนวนอนด้วย HorizontalStackLayout	66
	การกำหนดระยะห่างด้วยคุณสมบัติ Margin	67
	การกำหนดระยะห่างภายในคอนโทรลด้วยคุณสมบัติ Padding	69
	การจัดตำแหน่งข้อความด้วยกลุ่มคุณสมบัติ TextAlignment	72
	พื้นฐานการสร้างช่องรับข้อมูลด้วยอีลีเมนต์ <Entry />	73
	การแสดงผลเนื้อหาด้วย ContentView	79
	พื้นฐานการใช้ Layout แบบตาราง Grid	80
	การรวมคอลัมน์ (Grid.ColumnSpan) หรือแถว (Grid.RowSpan) ใน Grid	83
	ความหมายของการกำหนดขนาดแบบ Auto และ * ใน Grid	85
	การสร้าง Layout แบบเลื่อนดูเนื้อหาด้วย ScrollView	88
บทที่ 4	การใช้งานคอนโทรล (Controls) หรืออีลีเมนต์ (Element)	90
	พื้นฐานการตกแต่งคุณสมบัติ (Property) หรือแอตทริบิวต์ (Attribute) ของคอนโทรล ...	90
	การสร้างปุ่มกดด้วยคอนโทรล Button	91
	การสร้างปุ่มกดแบบรูปภาพด้วยอีลีเมนต์ ImageButton	93
	การแสดงผลข้อความธรรมดาด้วย TextCell	96
	การสร้างช่องรับข้อมูลแบบ EntryCell	99
	การสร้างตัวเปิด-ปิดสถานะด้วย SwitchCell	101
	การสร้างตัวเลือกวันที่ด้วย DatePicker	104
	การสร้างตัวเลือกเวลาด้วย TimePicker	108
	การสร้าง Resource ส่วนกลาง	110
	การสร้าง Static Resources แบบกำหนดค่าเดียว	110
	การสร้าง Static Resources แบบกำหนดหลายค่า (Style)	112
	การสร้าง Dynamic Resources	115
	การแจ้งเตือน (DisplayAlert)	118
	การตรวจสอบสถานะเชื่อมต่อ Internet ด้วยคลาส Connectivity	124
	การสร้างตัวเลือกไฟล์ด้วยคอนโทรล FilePicker	127

บทที่ 5	การผูกติดข้อมูล (Data Binding).....	132
	พื้นฐานการเขียนโปรแกรมเชิงวัตถุ (Object Oriented Programming – OOP).....	132
	พื้นฐานการผูกติดข้อมูล.....	133
	การผูกติดระหว่างคอนโทรล (หรืออีลีเมนต์).....	143
	โหมดการผูกติด (Data Binding Mode).....	145
บทที่ 6	การสั่งให้เกิดการทำงานด้วยคลาส Command	149
	การใช้งานคลาส Command	149
	การส่งพารามิเตอร์ให้กับ Command (Command Parameter).....	157
	การกำหนดค่าเริ่มต้นให้กับตัวเก็บสถานะ.....	163
บทที่ 7	การพัฒนาแอปแบบหลายหน้าจอ	166
	พื้นฐานการเพิ่มหน้าจอใหม่เข้ามาในโปรเจกต์	166
	การสร้าง Resource ใน App.xaml สำหรับใช้ได้ทั้งโปรเจกต์.....	170
	การรับ-ส่งข้อมูลระหว่างหน้า	173
	การสร้างส่วนแสดงผลแบบแท็บ (Tab).....	176
	การสร้างเมนูสำหรับ Desktop Apps.....	180
บทที่ 8	การสร้างรายการด้วย ListView	185
	พื้นฐานการสร้างรายการแบบ list ด้วย ListView	185
	การปรับแต่งรายการใน ListView.....	189
	การแบ่งการทำงานแบบ MVVM	197
บทที่ 9	การทำงานกับฟังก์ชันพื้นฐานของอุปกรณ์ Mobile Devices (.NET MAUI Essentials).....	209
	ฟังก์ชันการโทรออก.....	209
	การส่งข้อความ (SMS).....	214
	การตรวจสอบ Hardware ด้วยคลาส DeviceInfo.....	217
	การตรวจสอบรายละเอียดแอปด้วย AppInfo.....	220
	การตรวจสอบสถานะ Battery	224
	การตรวจสอบสถานะการเชื่อมต่ออินเทอร์เน็ต	229
	การเปิดเว็บเพจด้วย Browser	232
	การ Copy & Paste ข้อความจากคลิปบอร์ด.....	234
	การสร้างไฟล์เก็บข้อความส่วนกลาง (Shared Preferences)	237

บทที่ 10	.NET MAUI Blazor	241
	พื้นฐานการสร้างโปรเจกต์ .NET MAUI Blazor App	241
	ทำความรู้จักกับโครงสร้างโปรเจกต์ .NET MAUI Blazor App.....	245
	Layout หลักของโปรเจกต์ .NET MAUI Blazor (MainLayout.razor)	249
	ระบบเมนู (NavMenu.razor).....	251
	ส่วนแสดงผล Index.razor.....	253
	ส่วนแสดงผล Counter.razor.....	256
	ส่วนแสดงผล FetchData.razor.....	257
บทที่ 11	พื้นฐานการสร้าง Web API ด้วย ASP.NET Core Web API	265
	ASP.NET Core Web API คืออะไร.....	265
	การสร้างโปรเจกต์ ASP.NET Core Web AP.....	266
	การทดสอบและหยุดรันโปรเจกต์ ASP.NET Core Web API	269
	ทำความรู้จักกับโครงสร้างโปรเจกต์ ASP.NET Core Web API.....	274
	การสร้าง Web API แรกด้วย ASP.NET Core Web API	279
บทที่ 12	การสร้าง .NET MAUI Apps แบบ CRUD เชื่อมต่อกับ ASP.NET Core Web API (Front End).....	289
	การสร้างโปรเจกต์ .NET MAUI แบบ CRUD.....	289
	Repository Pattern ฝั่ง Front End	297
	การกำหนดให้ Android Emulator เชื่อมต่อกับ localhost	317
	การสร้างหน้าจอแสดงรายการลูกค้า (MainPage.xaml).....	320
บทที่ 13	การสร้าง .NET MAUI Apps แบบ CRUD เชื่อมต่อกับ ASP.NET Core Web API (Back End)	337
	ทำความรู้จักกับโครงสร้างแบบหลายโปรเจกต์	337
	Repository Pattern ฝั่ง Back End	342
	วิธีการรันหลายโปรเจกต์.....	359

บทที่ 14	พื้นฐานการพัฒนา Web Apps แบบคอนเทนเนอร์ด้วย Docker	369
	การดาวน์โหลดและติดตั้ง Windows Subsystem for Linux (WSL)	370
	การดาวน์โหลดและติดตั้ง Docker.....	372
	การทำงานกับ Docker Image/Container ในเบื้องต้น	374
	การลบอิมเมจ.....	381
บทที่ 15	การสร้าง ASP.NET Core Web API ด้วย Docker	383
	การสร้างโปรเจกต์ ASP.NET Core Web API ที่มีการใช้งาน Docker	383
บทที่ 16	การเผยแพร่อิมเมจแบบสาธารณะกับ Docker Registry	392
	การ push อิมเมจสู่ Docker Registry	392
บทที่ 17	การสร้างไฟล์ Signed APK และ Signed AAB ด้วย .NET MAUI	399
	การกำหนด Target Android Framework.....	399
	วิธีการสร้างไฟล์ Key Store ให้กับ Android Apps	400
	การสร้างไฟล์ Signed APK และไฟล์ Signed AAB จากโปรเจกต์ .NET MAUI แบบ Android Apps.....	403
บทที่ 18	การอัปโหลดแอปขึ้น Google Play Store	406
	การสมัครบัญชีนักพัฒนา Android Apps	406
	ขั้นตอนการอัปโหลดไฟล์ Signed AAB ไปสู่ Google Play Store	409
บทที่ 19	การสร้างไฟล์ติดตั้งของ .NET MAUI แบบ Windows Apps	412
	การสร้างไฟล์ .exe จากโปรเจกต์ .NET MAUI แบบ Windows Apps	412
	การสร้างไฟล์ Signed MSIX สำหรับ Windows Apps	416





บทที่ 1

การพัฒนาแอปด้วย .NET MAUI

.NET MAUI เป็นความก้าวหน้าครั้งสำคัญอีกก้าวหนึ่งของแพลตฟอร์ม .NET เป็นโปรเจกต์ที่ช่วยให้นักพัฒนาสามารถสร้างแอปหลายแพลตฟอร์มในเวลาเดียวกันด้วยโครงสร้างโปรเจกต์เพียงหนึ่งเดียว ช่วยให้การดูแลรักษาโปรเจกต์ในระยะยาวได้อีกทางหนึ่งด้วย

ผลผลิตของโปรเจกต์ .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 บนเครื่องคอมพิวเตอร์ PC ส่งผลให้ผู้เขียนสามารถนำเสนอแอปได้ 2 ประเภท คือ Windows Apps และ Android Apps

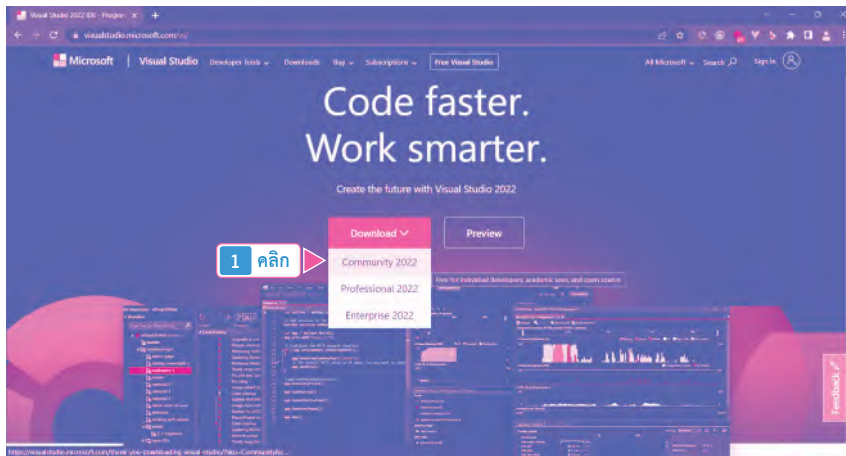
เนื้อหาที่นำเสนอในหนังสือเล่มนี้ครอบคลุมการพัฒนาแอปในรูปแบบ Front End กับ Back End กล่าวคือ

1. **งานฝั่งหน้าบ้าน (Front End)** เป็นหน้าที่ของโปรเจกต์ .NET MAUI ทำหน้าที่สร้างส่วนแสดงผลต่างๆ (User Interface - UI)
2. **งานหลังบ้าน (Back End)** เป็นหน้าที่ของโปรเจกต์ประเภท ASP.NET Core Web API ทำหน้าที่ติดต่อกับส่วนของข้อมูลเพื่อให้บริการพื้นฐานในรูปแบบ Web API ได้แก่ งานประเภทเพิ่ม, อ่าน, อัปเดต และลบข้อมูล (CRUD)

การดาวน์โหลดและติดตั้งโปรแกรม Visual Studio 2022

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

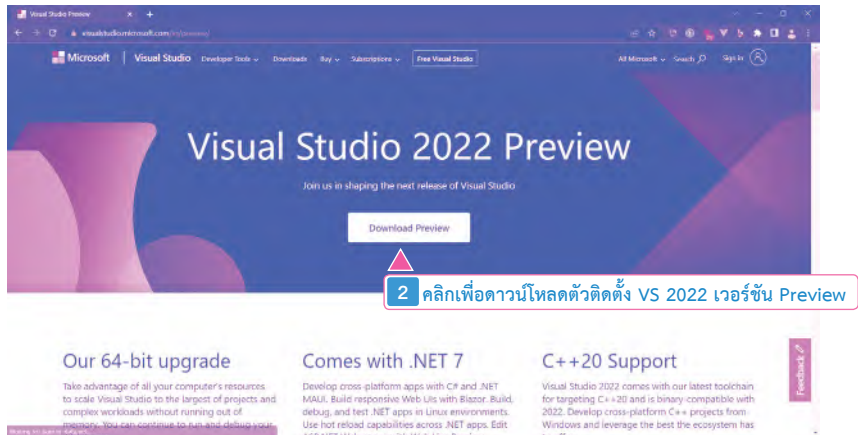
1. **โปรแกรม Visual Studio เวอร์ชันปกติ** เป็นเวอร์ชันที่เหมาะสมกับการนำไปใช้ในงาน Production จริง สามารถดาวน์โหลดมาใช้งานได้ฟรีที่ <https://visualstudio.microsoft.com/vs/>



รูปที่ 1-1 แสดงเว็บสำหรับดาวน์โหลดโปรแกรม Visual Studio เวอร์ชันปกติ

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

ฟีเจอร์ต่างๆ ที่ถูกนำมาทดสอบในเวอร์ชันนี้ เมื่อผ่านการทดสอบแล้วไมโครซอฟท์จะนำไปใช้งานกับเวอร์ชันปกติในลำดับถัดไป ดังรูปที่ 1-2

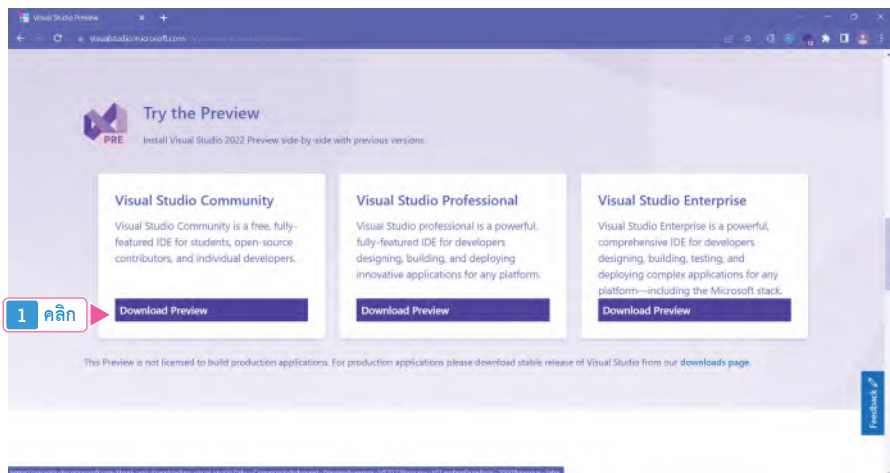


2 คลิกเพื่อดาวน์โหลดตัวติดตั้ง VS 2022 เวอร์ชัน Preview

รูปที่ 1-2 แสดงเว็บไซต์สำหรับดาวน์โหลดโปรแกรม 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

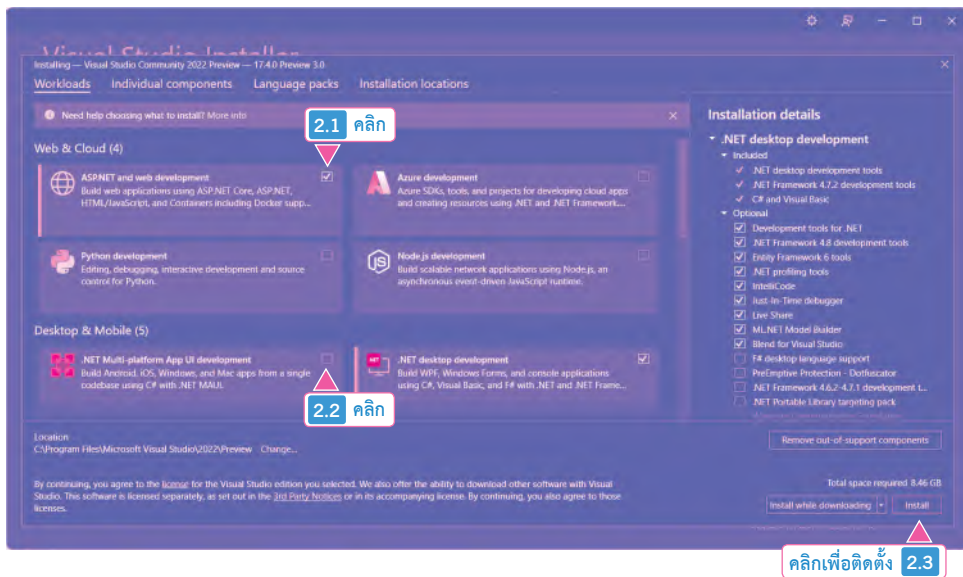
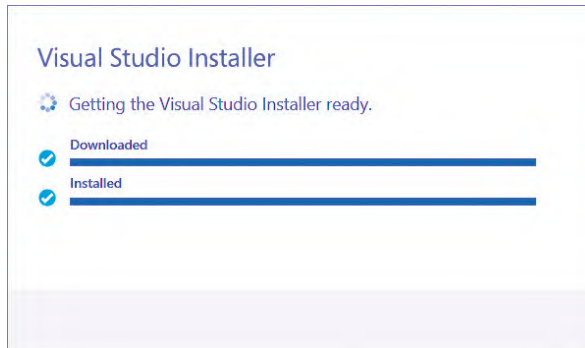


1 คลิก

รูปที่ 1-3 แสดงการดาวน์โหลดตัวติดตั้งโปรแกรม Visual Studio 2022

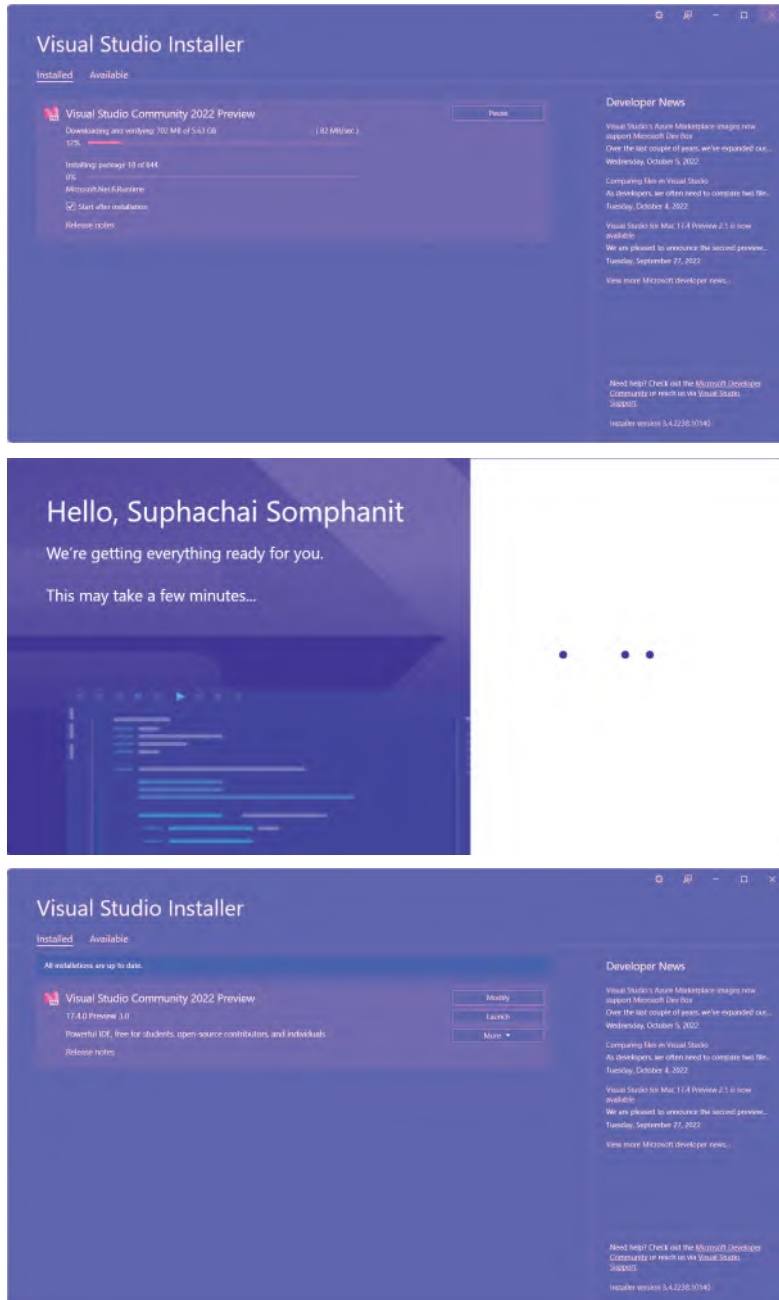
2. ให้ผู้อ่านดับเบิลคลิกไฟล์ที่ดาวน์โหลดมา เพื่อเข้าสู่ขั้นตอนการติดตั้งโปรแกรม Visual Studio 2022 ผู้อ่านสามารถเลือกประเภทโปรเจกต์ที่สนใจได้อย่างอิสระ ขอบเขตการนำเสนอเนื้อหาของหนังสือเล่มนี้ประกอบด้วย

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



รูปที่ 1-4 แสดงหน้าจอเลือกติดตั้งประเภทโปรเจกต์

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

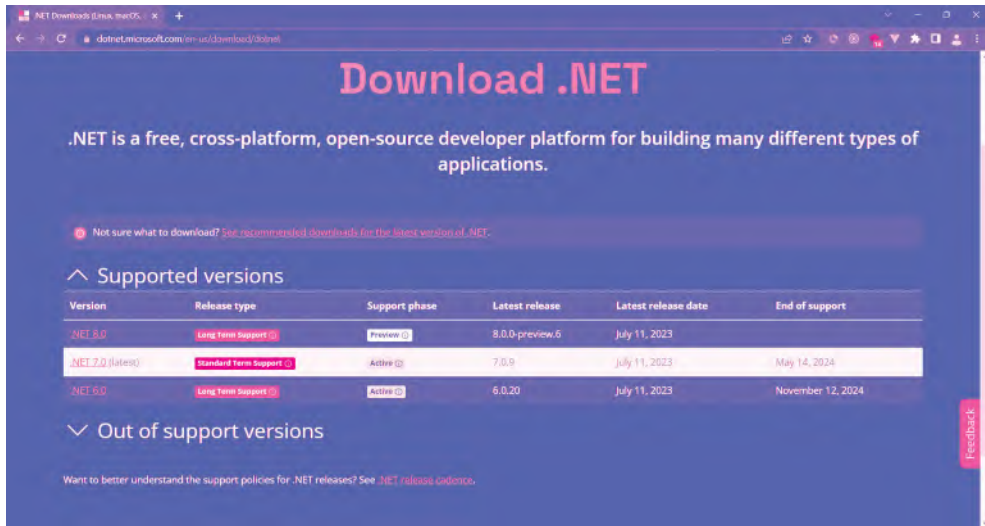


รูปที่ 1-5 แสดงการติดตั้งโปรแกรม Visual Studio เสร็จสมบูรณ์

สถานะของ .NET SDK

หลังจากที่ไม่โครซอฟท์รวม .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 ... มีระยะเวลาสนับสนุนอัปเดตแพตช์แก้ไข 18 เดือน เป็นเวอร์ชันที่เรียกว่า **Standard-term support** เรียกย่อๆ ว่า **STS**
- **.NET SDK เวอร์ชันเลขคู่** ได้แก่ เวอร์ชัน 6, 8 ... มีระยะเวลาสนับสนุนอัปเดตแพตช์แก้ไข 3 ปี เป็นเวอร์ชันที่เรียกว่า **Long-term support** เรียกย่อๆ ว่า **LTS** การพัฒนาโปรเจกต์ในระดับ Production จริง ผู้เขียนแนะนำให้เลือกใช้เวอร์ชัน LTS เหมาะสมที่สุด



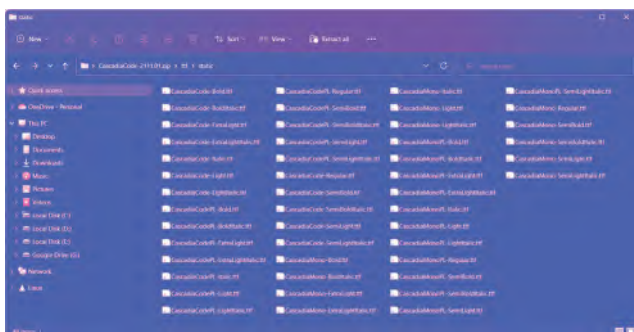
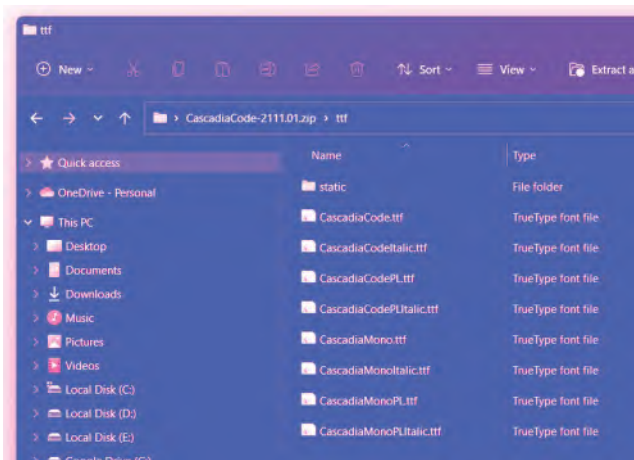
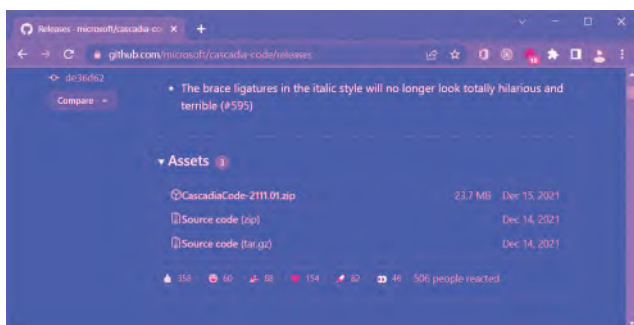
รูปที่ 1-6 แสดงรายละเอียดของ .NET SDK แต่ละรุ่น

.NET SDK เวอร์ชันใดก็ตามที่ใกล้หมดระยะเวลา Support เรียกว่า ระยะเวลา Maintenance ถ้าโปรเจกต์ของเราใช้เวอร์ชันที่มีสถานะนี้ ควรที่จะอัปเดตไปใช้เวอร์ชันที่ใหม่กว่าให้เร็วที่สุด

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

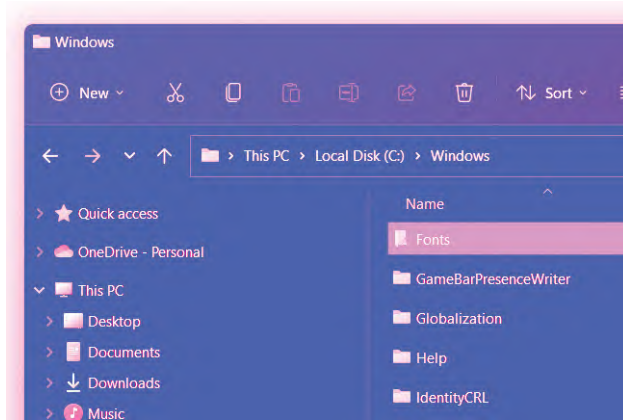
การติดตั้งฟอนต์ Cascadia Code สำหรับเขียนโค้ดโดยเฉพาะ

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



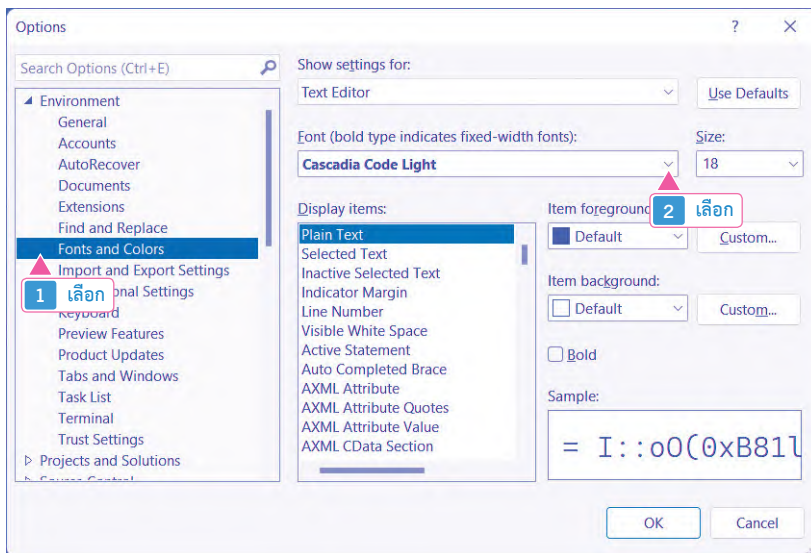
รูปที่ 1-7 แสดงเว็บไซต์ที่นำดาวน์โหลดฟอนต์เขียนโค้ด

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



รูปที่ 1-8 แสดงโฟลเดอร์ที่จัดเก็บรายการฟอนต์ในระบบ Windows 10/11

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

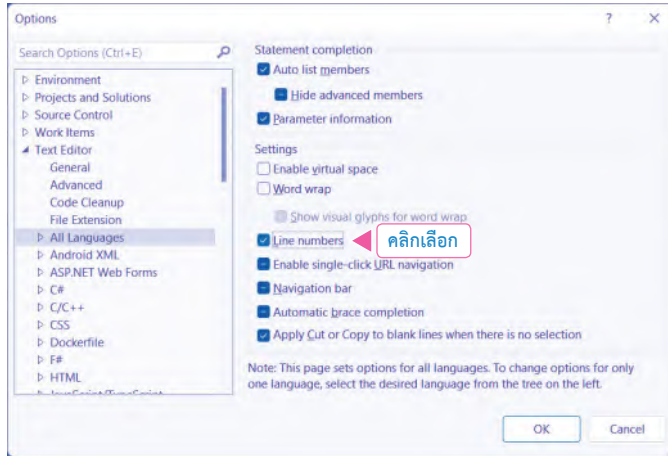


รูปที่ 1-9 แสดงการกำหนดให้โปรแกรม Visual Studio ใช้ฟอนต์ Cascadia Code Light

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

การกำหนดให้แสดงหมายเลขบรรทัด

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

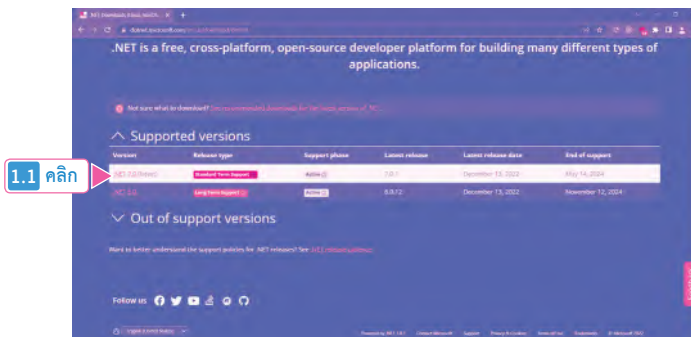


รูปที่ 1-10 แสดงการกำหนดให้แสดงหมายเลขกำกับโค้ดแต่ละบรรทัด

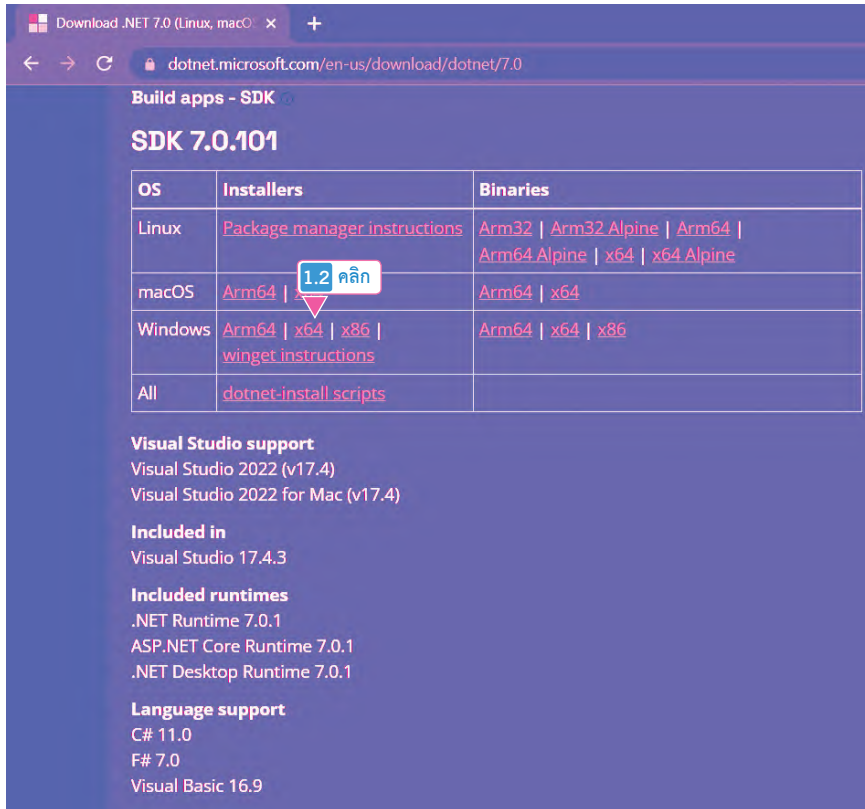
การติดตั้ง .NET SDK เพิ่มเติม

เนื้อหาในหนังสือเล่มนี้ต้องการ .NET SDK เวอร์ชัน 7 ขึ้นไป ผู้อ่านสามารถติดตั้ง .NET SDK หลายเวอร์ชันในเวลาเดียวกัน ส่งผลให้โปรแกรม Visual Studio 2022 ของผู้อ่านสามารถสร้างโปรเจกต์ได้ตามเวอร์ชัน .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



OS	Installers	Binaries
Linux	Package manager instructions	Arm32 Arm32 Alpine Arm64 Arm64 Alpine x64 x64 Alpine
macOS	Arm64 x64	Arm64 x64
Windows	Arm64 x64 x86 winget instructions	Arm64 x64 x86
All	dotnet-install scripts	

Visual Studio support
 Visual Studio 2022 (v17.4)
 Visual Studio 2022 for Mac (v17.4)

Included in
 Visual Studio 17.4.3

Included runtimes
 .NET Runtime 7.0.1
 ASP.NET Core Runtime 7.0.1
 .NET Desktop Runtime 7.0.1

Language support
 C# 11.0
 F# 7.0
 Visual Basic 16.9

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

เมื่อมีการอัปเดต .NET SDK เวอร์ชันใหม่ ก็จะมีการอัปเดตโปรแกรม Visual Studio ควบคู่มาด้วยเสมอ เพื่อให้สามารถใช้งาน .NET SDK เวอร์ชันใหม่ได้นั่นเอง โดยปกติแล้วโปรแกรม Visual Studio เวอร์ชัน Preview มักจะได้สิทธิ์ทดสอบ .NET SDK เวอร์ชันใหม่ก่อนเวอร์ชันปกติ

- ท้ายที่สุดให้ผู้อ่านดับเบิลคลิกไฟล์ที่ดาวน์โหลดมาเพื่อเข้าสู่ขั้นตอนการติดตั้ง .NET 7 SDK ใช้ค่าเริ่มต้นที่มากับตัวติดตั้งทั้งหมด ส่งผลให้โปรแกรม Visual Studio 2022 สามารถพัฒนาโปรเจกต์บน .NET SDK ได้หลายเวอร์ชัน

การอัปเดตโปรแกรม Visual Studio

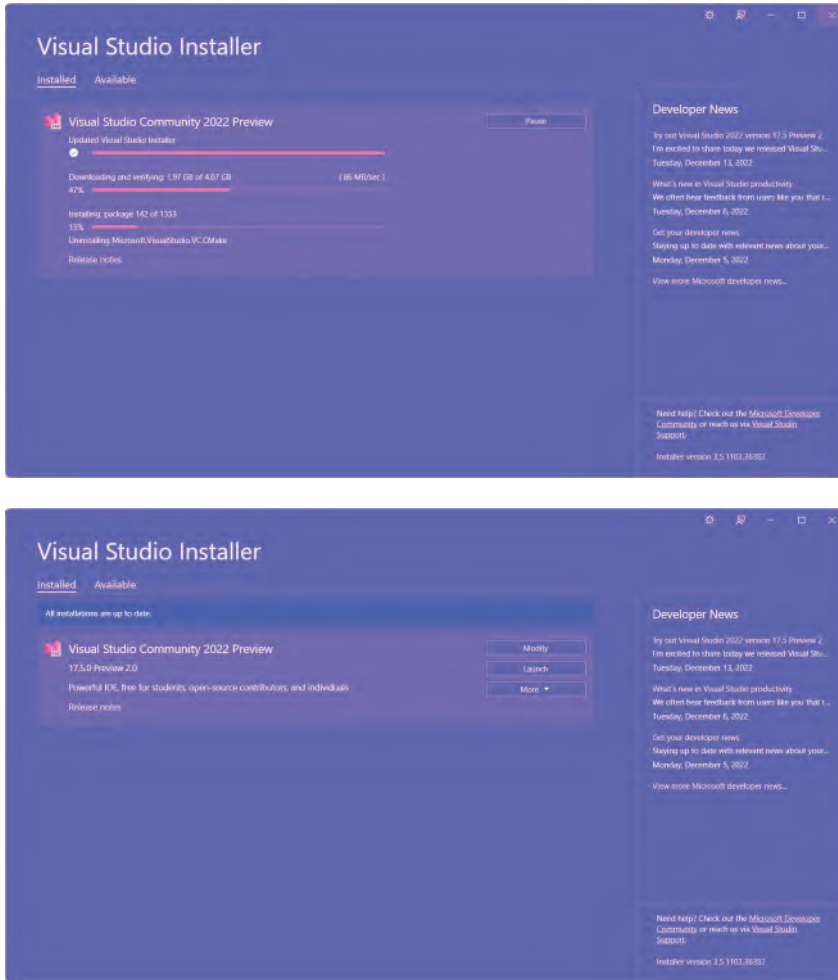
ตัวโปรแกรม Visual Studio มีการอัปเดตอยู่เสมอ อาจจะเป็นการแก้ไขข้อผิดพลาดหรือเพิ่มเติมฟีเจอร์ใหม่ๆ ก็ได้ แยกออกเป็น 2 กรณี

- **Visual Studio เวอร์ชัน Preview** เป็นเวอร์ชันที่ใช้สำหรับทดสอบฟีเจอร์ใหม่ๆ ของ .NET SDK อยู่แล้ว ส่งผลให้มีการอัปเดตค่อนข้างบ่อย
 - **Visual Studio เวอร์ชัน Stable** เป็นเวอร์ชันที่เราใช้งานจริงใน Production ของเรา หลังจากฟีเจอร์ต่างๆ ถูกทดสอบในเวอร์ชัน Preview จนกระทั่งสมบูรณ์แล้วก็จะนำมาอัปเดตในเวอร์ชัน Stable มีระยะการอัปเดตห่างกว่าเวอร์ชัน Preview
1. เมื่อตัวโปรแกรม Visual Studio มีอัปเดตใหม่ จะมีการแจ้งเตือน ดังรูปที่ 1-13



รูปที่ 1-13 แสดงการแจ้งเตือนอัปเดตของโปรแกรม Visual Studio

2. การดาวน์โหลดและติดตั้งอัปเดตต่างๆ ให้ปิดตัวโปรแกรม Visual Studio ก่อน ก็จะเข้าสู่ขั้นตอนการอัปเดตโดยอัตโนมัติทั้งหมด ดังรูปที่ 1-14

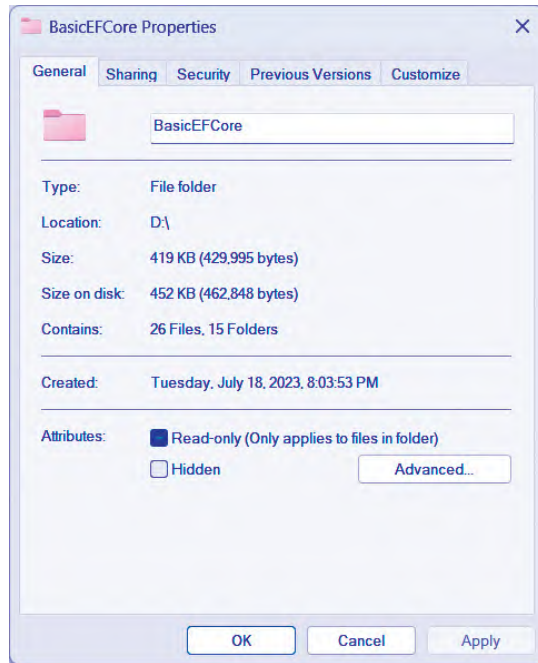


รูปที่ 1-14 แสดงการอัปเดต Visual Studio เสร็จสมบูรณ์

วิธีการจัดเก็บโปรเจกต์ .NET

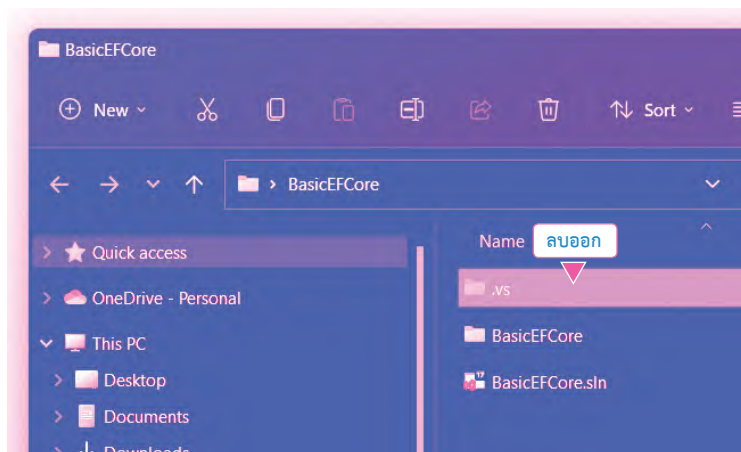
เป้าหมายของวิธีนี้คือ ต้องการลดขนาดไฟล์โปรเจกต์ให้มีขนาดเล็กลง โปรเจกต์ทุกประเภทของ .NET สามารถใช้วิธีการจัดเก็บโปรเจกต์แบบนี้ได้ มีขั้นตอนดังนี้

1. ให้ผู้อ่านตรวจสอบขนาดโปรเจกต์ก่อนปรับปรุง โดยการคลิกขวาเลือกคำสั่ง Properties พบว่าขนาดโปรเจกต์ 883 KB แสดงดังรูปที่ 1-15



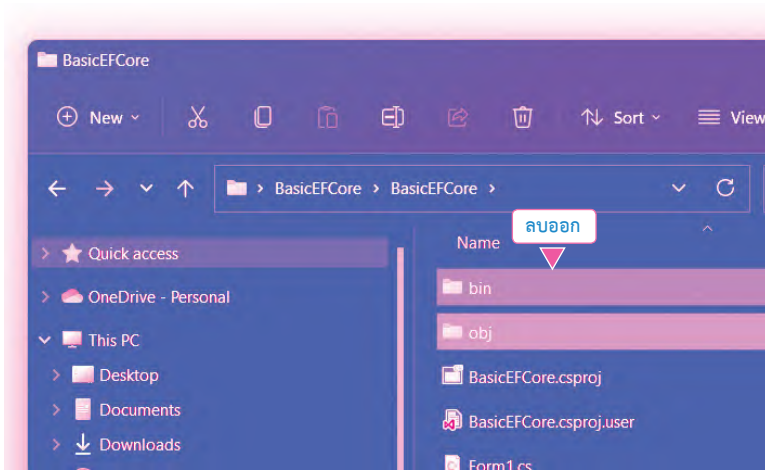
รูปที่ 1-15 แสดงขนาดโปรเจกต์ก่อนปรับปรุง

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



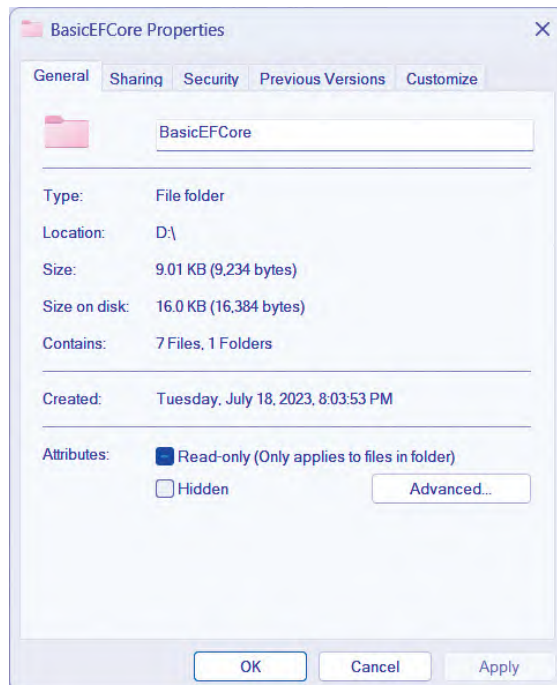
รูปที่ 1-16 แสดงการลบโฟลเดอร์ .vs

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



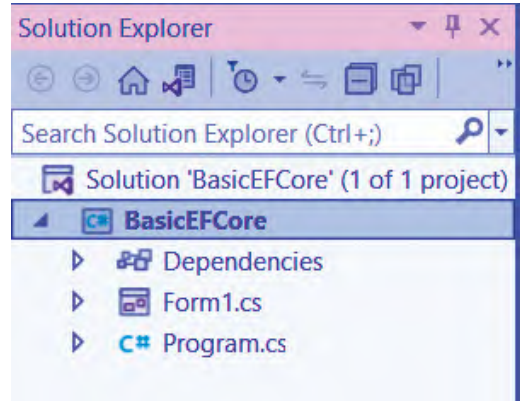
รูปที่ 1-17 แสดงการลบโฟลเดอร์ย่อย bin และ obj

4. ให้ลองตรวจสอบขนาดโปรเจกต์อีกครั้ง พบว่าขนาดโปรเจกต์จะลดลงแล้ว ผู้อ่านสามารถนำโปรเจกต์ .NET ไปจัดเก็บได้แล้ว ดังรูปที่ 1-18



รูปที่ 1-18 แสดงขนาดโปรเจกต์หลังปรับปรุงแล้ว

5. การเปิดโปรเจกต์ที่มีการปรับปรุงแบบนี้ ต้องรอให้เครื่องหมายตกใจ (!) ในหน้าต่าง Solution Explorer หายไปก่อน ก็จะใช้งานโปรเจกต์ได้ตามปกติ

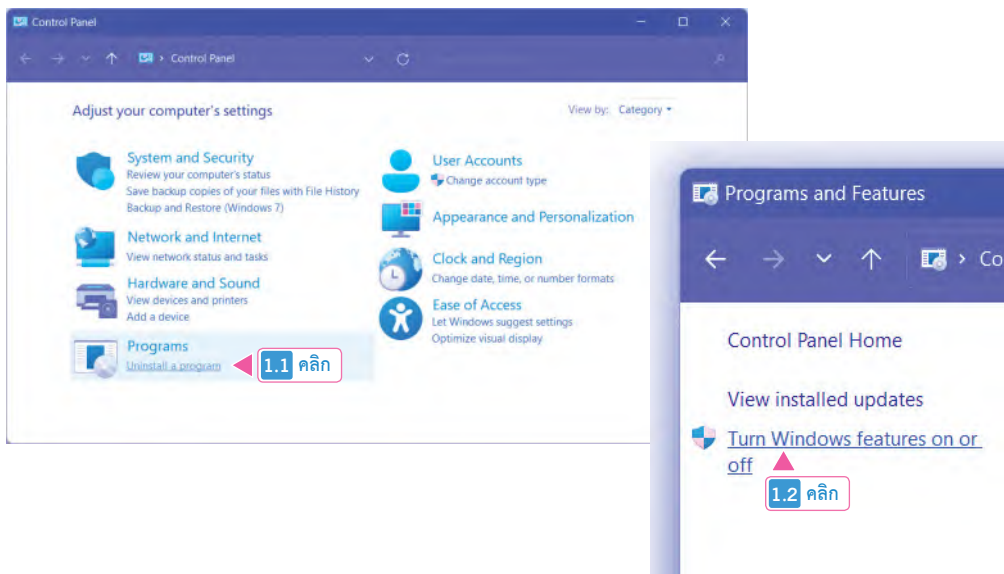


รูปที่ 1-19 แสดงการเปิดโปรเจกต์ที่มีการปรับปรุงขนาด

การดาวน์โหลด Android Emulator เพิ่มเติมเพื่อทดสอบโปรเจกต์แบบ Android Apps

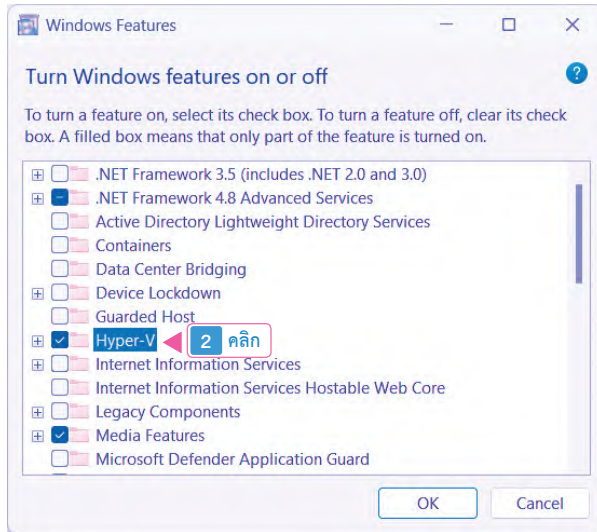
โปรเจกต์ประเภท .NET MAUI กับ .NET MAUI Blazor สามารถทดสอบแอปในรูปแบบ Android Apps ได้เช่นกัน โดยที่ผู้อ่านสามารถสร้าง Android Emulator ขึ้นมาเพื่อทดสอบโปรเจกต์ในขั้นตอนนี้ได้ด้วย มีขั้นตอนดังนี้

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



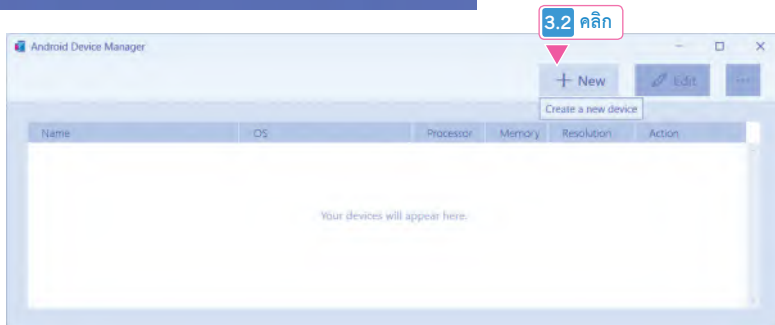
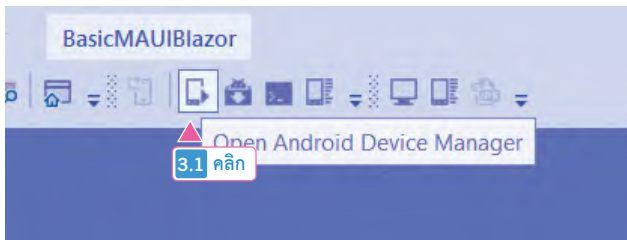
รูปที่ 1-20 แสดงการเลือกคำสั่งเปิด/ปิดฟีเจอร์ของ Windows 10/11

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



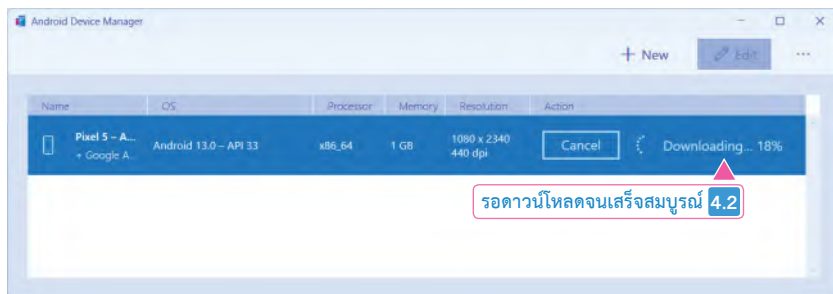
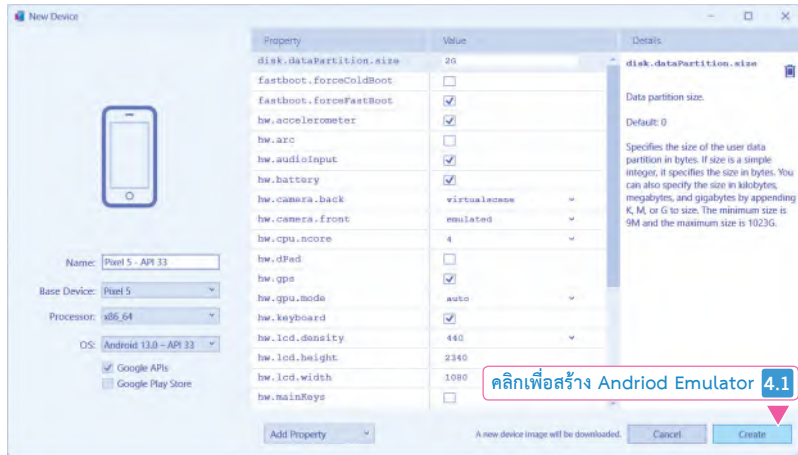
รูปที่ 1-21 แสดงการเปิดฟีเจอร์ Hyper-V ของ Windows 10/11

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



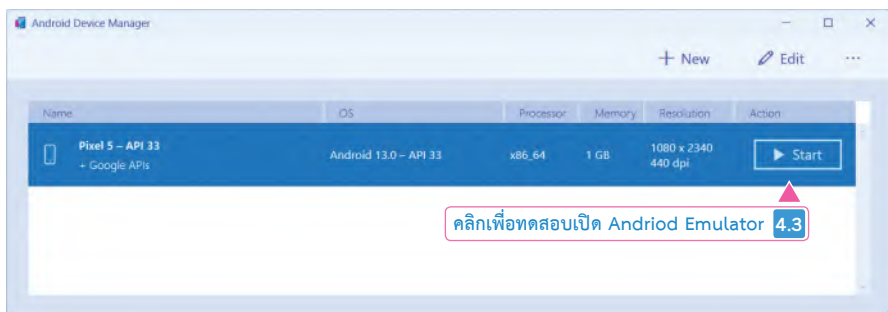
รูปที่ 1-22 หน้าต่างแสดงรายการ Android Emulator ในเครื่องของเรา

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



รูปที่ 1-23 แสดงการดาวน์โหลด Android Emulator รายการแรกของเครื่องนี้

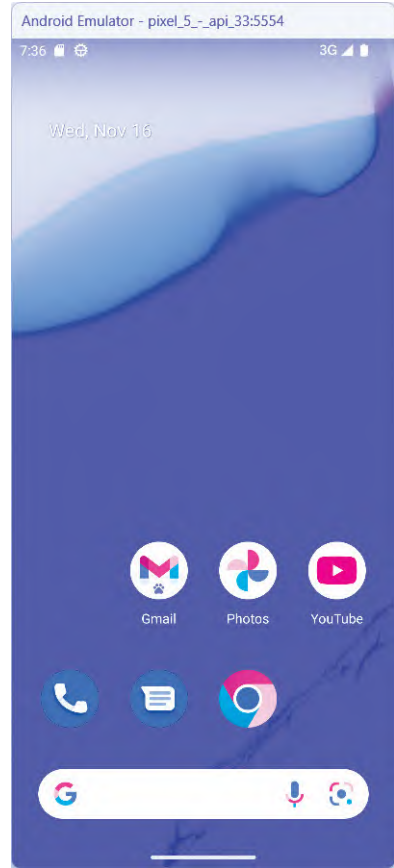
จากรูปที่ 1-23 การดาวน์โหลดและติดตั้ง Android Emulator เป็นแบบอัตโนมัติทั้งหมด ผู้อ่านสามารถสร้าง Android Emulator ได้หลายรายการ เพื่อทดสอบส่วนแสดงผลหลายขนาดได้อีกด้วย



รูปที่ 1-24 แสดง Android Emulator ที่ได้มา

5. ให้ทดสอบเปิด Android Emulator ขึ้นมา การทำงานครั้งแรกจะใช้เวลาานมากกว่าปกติ ดังรูปที่ 1-25

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



รูปที่ 1-25 แสดงการเปิด Android Emulator

ประเภทโปรเจกต์ .NET MAUI

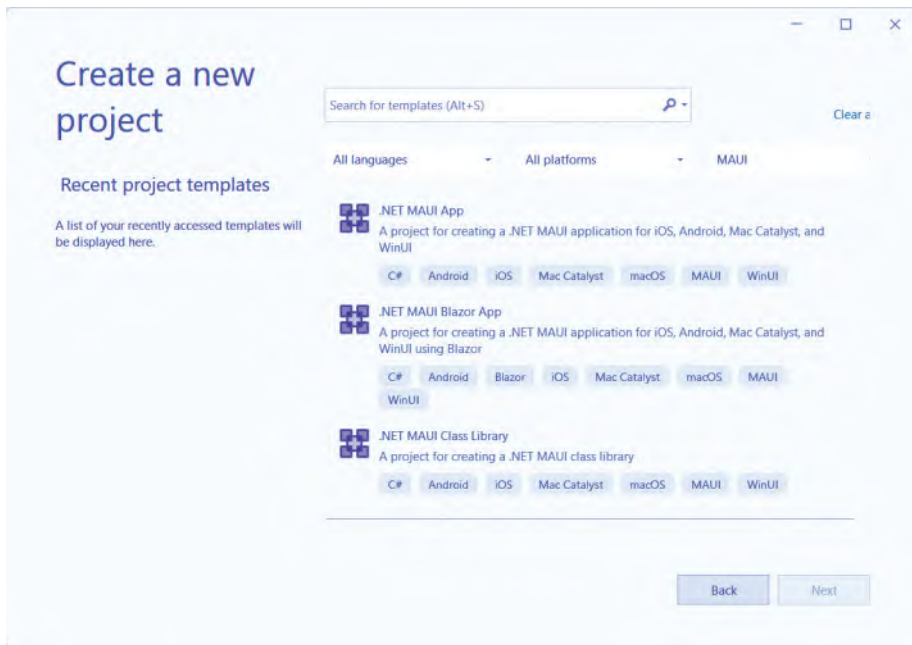
การพัฒนาแอปแบบ .NET MAUI สามารถแบ่งออกได้เป็น 2 ประเภทหลัก คือ

1. .NET MAUI App ประกอบด้วย

- **สร้างส่วนแสดงผล** (User Interface – UI) ด้วยสคริปต์ภาษา XAML มีไฟล์นามสกุล .xaml
- **โค้ดการทำงาน** อาศัยภาษา C# มีไฟล์นามสกุล .cs

2. .NET MAUI Blazor App เป็นโปรเจกต์ที่อาศัยเทคโนโลยีฝั่งเว็บเป็นแกนหลัก ใน .NET เรียกว่า โปรเจกต์ Blazor ประกอบด้วย

- **สร้างส่วนแสดงผล** ด้วยสคริปต์ภาษา HTML + ส่วนขยายเพิ่มเติม มีไฟล์นามสกุล .razor
- **โค้ดการทำงาน** อาศัยภาษา C# มีไฟล์นามสกุล .cs

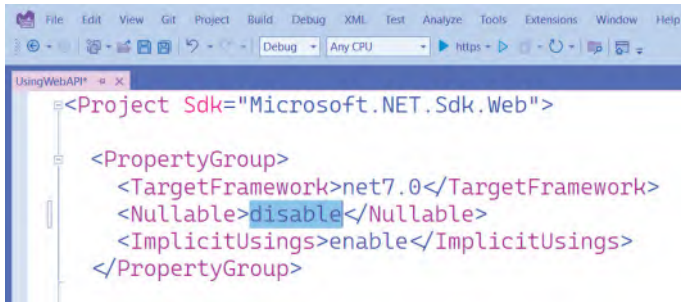
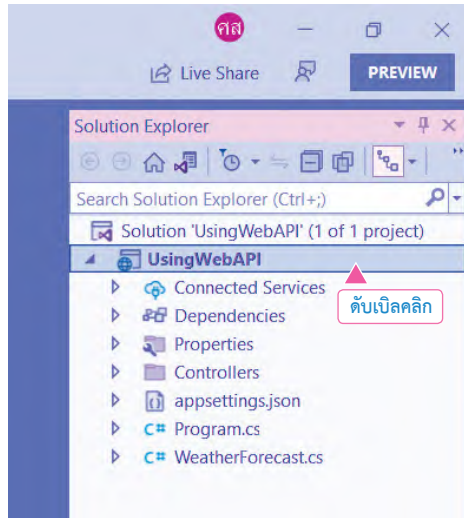


รูปที่ 1-26 แสดงหน้าจอสร้างโปรเจกต์ .NET MAUI ทั้ง 2 ประเภท

เนื้อหาของหนังสือเล่มนี้นำเสนอเฉพาะโปรเจกต์ประเภท .NET MAUI เท่านั้น

การยกเลิกฟีเจอร์ Nullable ของโปรเจกต์ ASP.NET Core Web API

สำหรับโปรเจกต์ประเภท ASP.NET Core Web API ที่เราจะใช้ในงานฝั่งหลังบ้าน (Back End) ผู้เขียนแนะนำให้ผู้อ่านยกเลิกฟีเจอร์ Nullable เป็นลำดับแรกก่อนเสมอ เพราะวยังไม่ได้ใช้ประโยชน์อะไรกับฟีเจอร์นี้ โดยการดับเบิลคลิกที่ชื่อโปรเจกต์ก่อน จากนั้นแก้ไขค่าคุณสมบัติ Nullable ให้มีค่าเป็น disable ดังรูปที่ 1-27



รูปที่ 1-27 แสดงการยกเลิกฟีเจอร์ Nullable

จากรูปที่ 1-27 ให้เลื่อนมาด้านล่างจนถึงหัวข้อ Nullable ให้เลือกรายการ Disable เพื่อยกเลิกใช้งานฟีเจอร์นี้

สรุปท้ายบท

เนื้อหาในบทนี้เป็นเพียงการเตรียมสภาพแวดล้อมให้พร้อมกับการพัฒนาแอปด้วยโปรเจกต์ประเภท .NET MAUI นำเสนอเป็นเนื้อหาตั้งแต่บทที่ 2 เป็นต้นไป



บทที่ 2

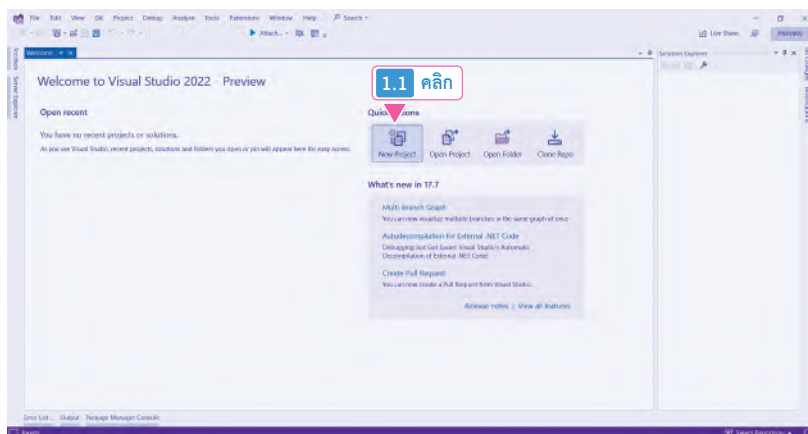
พื้นฐานการพัฒนาแอปด้วย .NET MAUI

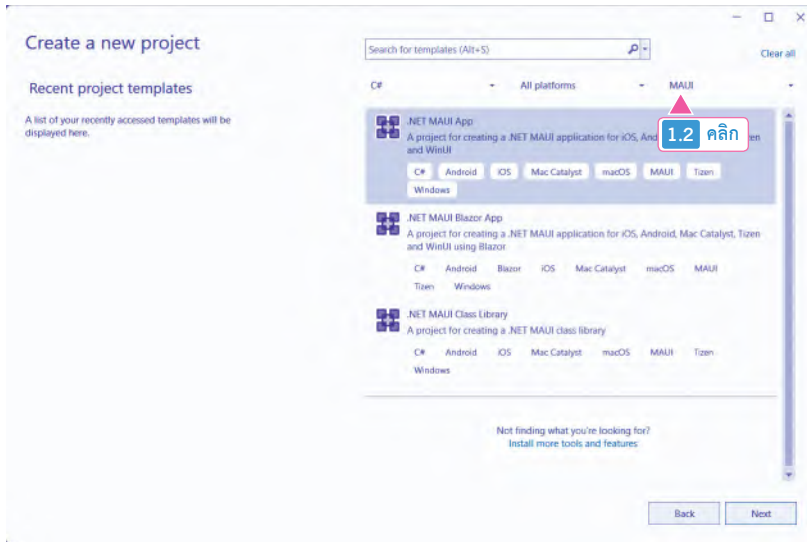
เนื้อหาลำดับแรกของการพัฒนาแอปแบบ .NET MAUI ก็คือ เราจะมาทำความรู้จักกับโครงสร้างโปรเจกต์ .NET MAUI ในขั้นต้นก่อนว่ามีโครงสร้างและมีหลักการทำงานอย่างไร

พื้นฐานการสร้างโปรเจกต์ .NET MAUI

มีขั้นตอนดังนี้

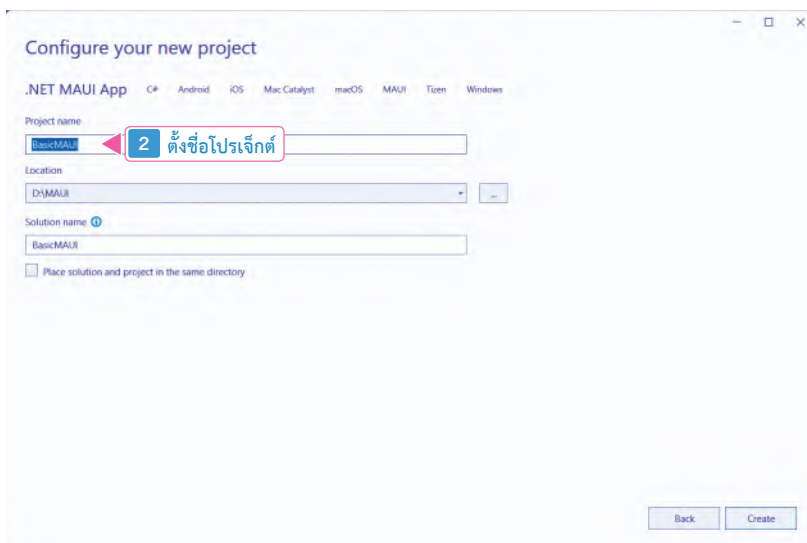
1. ที่หน้าจอเริ่มต้นของโปรแกรม Visual Studio ให้ผู้อ่านเลือก Create a new project เพื่อเริ่มต้นสร้างโปรเจกต์ใหม่ ในกรณีนี้ผู้เขียนต้องการสร้างโปรเจกต์ประเภท .NET MAUI ทำได้โดยการเลือกที่ช่อง All project types กำหนดให้แสดงรายการโปรเจกต์เฉพาะ .NET MAUI เท่านั้น ดังรูปที่ 2-1





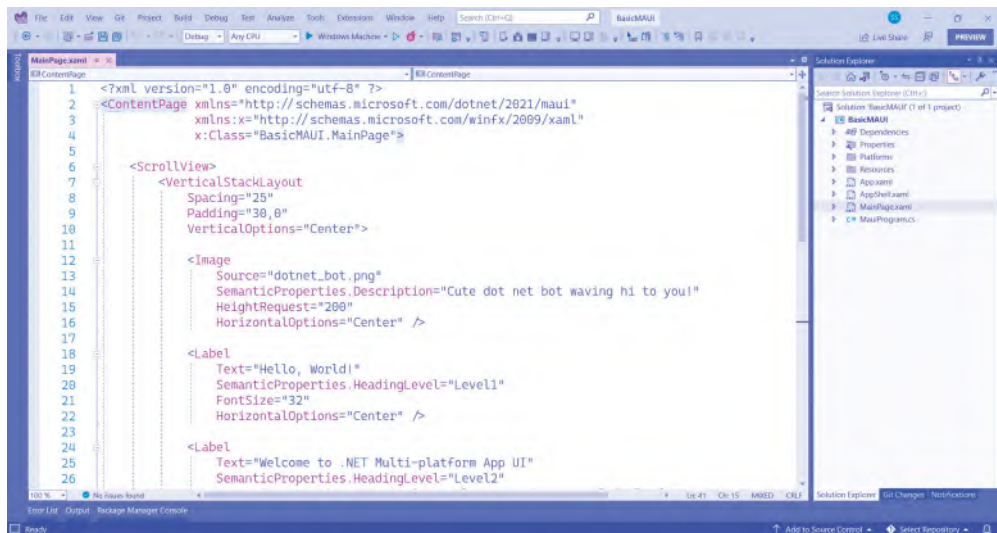
รูปที่ 2-1 แสดงการสร้างโปรเจกต์แบบ .NET MAUI App

2. ในกรณีนี้โปรเจกต์ .NET MAUI App แรกตั้งชื่อว่า BasicMAUI เก็บไว้ที่พาท D:\MAUI (หรือพาทอื่นตามที่ผู้อ่านต้องการ) ดังรูปที่ 2-2



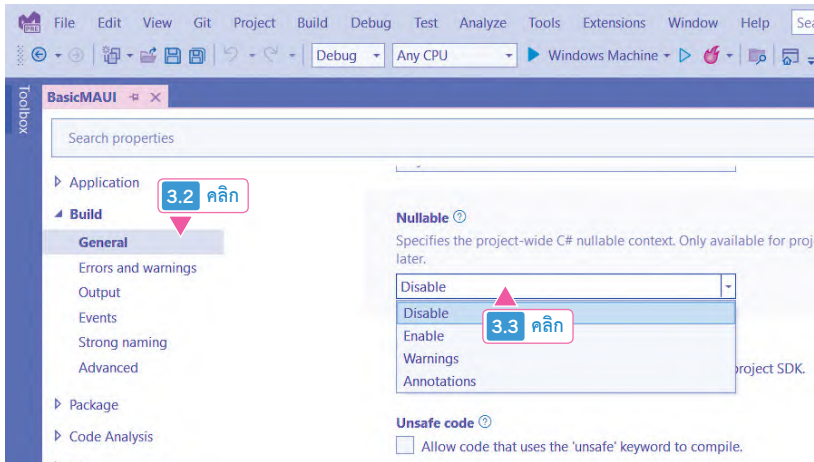
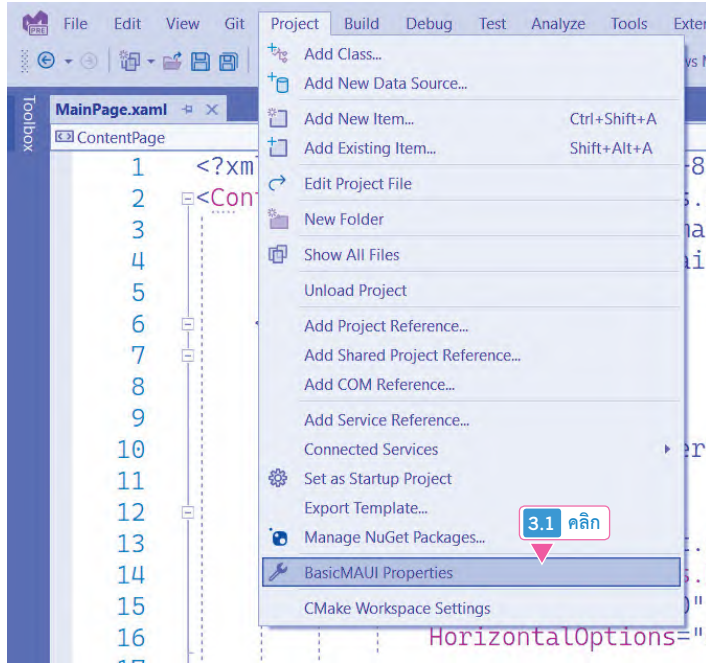
รูปที่ 2-2 แสดงการตั้งชื่อโปรเจกต์และพาทที่จัดเก็บ

จากรูปที่ 2-2 ให้รอนกระทั่งโปรแกรม Visual Studio สร้างโปรเจกต์แบบ .NET MAUI จนเสร็จสมบูรณ์ ถือเป็นโครงสร้างโปรเจกต์เริ่มต้นของเรา อาจจะมีโครงสร้างโปรเจกต์เริ่มต้นเหมือนหรือไม่เหมือนของผู้เขียนก็ได้ ไม่ถือเป็นเรื่องสำคัญ เพราะว่าเมื่อผู้อ่านเข้าใจหลักการทำงานแล้ว โครงสร้างเริ่มต้นโปรเจกต์จะมีรูปแบบใดก็ได้



รูปที่ 2-3 แสดงโครงสร้างโปรเจกต์เริ่มต้นแบบ .NET MAUI

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



รูปที่ 2-4 แสดงการปรับแต่งโปรเจกต์ .NET MAUI เบื้องต้น

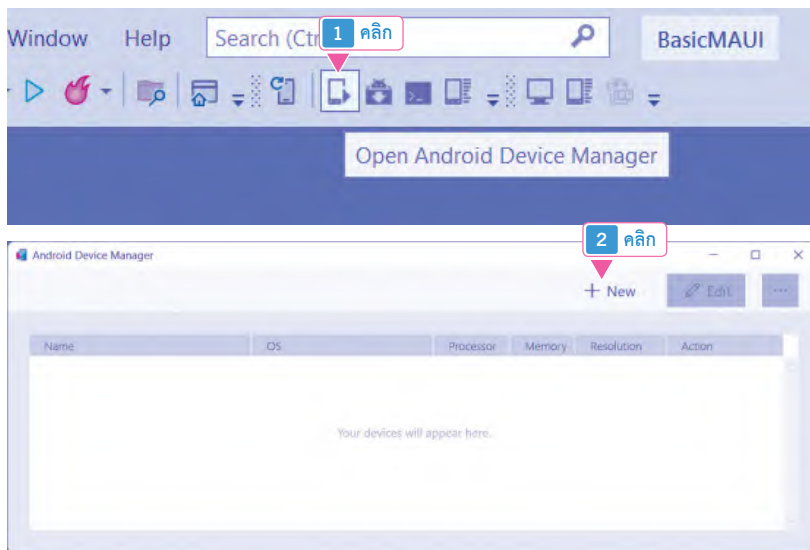
จากรูปที่ 2-4 ที่รายการ Build > General เลื่อนมาที่รายการ Nullable ให้ผู้อ่านยกเลิกใช้งานฟีเจอร์ Nullable โดยการกำหนดค่า Disable ก่อน ถือว่าโปรเจกต์ .NET MAUI ของเราพร้อมใช้งานแล้ว (ใน .NET เวอร์ชันใหม่ ไม่ต้องกำหนดแล้ว)

การสร้าง Android Emulator ของ Visual Studio สำหรับทดสอบ Android Apps

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

การทดสอบการทำงานของโปรเจกต์ Android Apps ในขั้นต้นคือ อาศัย Android Emulator ที่มากับโปรแกรม Visual Studio มีขั้นตอนดังนี้

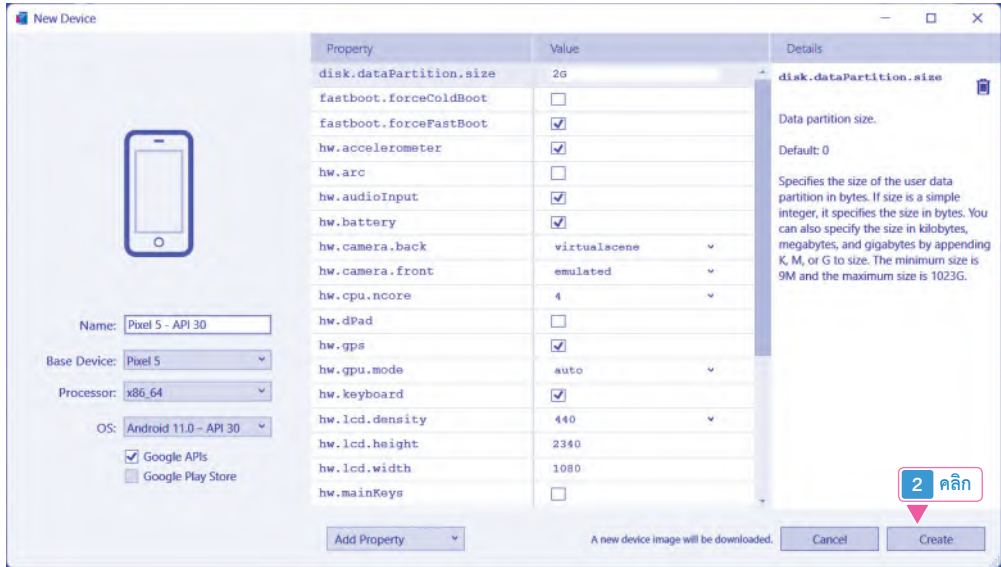
1. ในโปรแกรม Visual Studio ยังไม่มี Android Emulator ใดๆ ทั้งสิ้น ให้ผู้อ่านคลิกที่ปุ่ม  จากนั้นคลิกที่ปุ่ม **+ New** เริ่มต้นสร้าง Android Emulator เพื่อทดสอบโปรเจกต์ .NET MAUI ประเภท Android Apps ดังรูปที่ 2-5



รูปที่ 2-5 แสดงการเริ่มต้นสร้าง Android Emulator

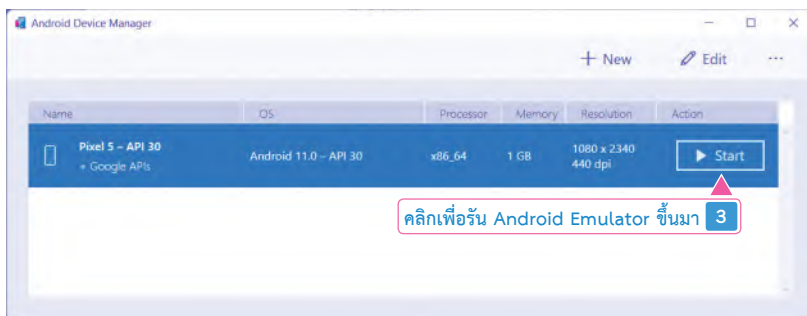
2. เป็นขั้นตอนการกำหนดรายละเอียดให้กับ Android Emulator เพื่อจำลองให้มีความเหมือนมือถือเครื่องจริง ที่ต้องกำหนด ได้แก่
 - **Name:** ชื่อ Android Emulator
 - **Base Device:** ให้ผู้อ่านเลือกรุ่นมือถือที่ต้องการจำลองการทำงาน
 - **Processor:** ให้ผู้อ่านเลือก CPU แบบ x86_x64 ตรงกับ CPU ที่ใช้งานในเครื่อง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

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



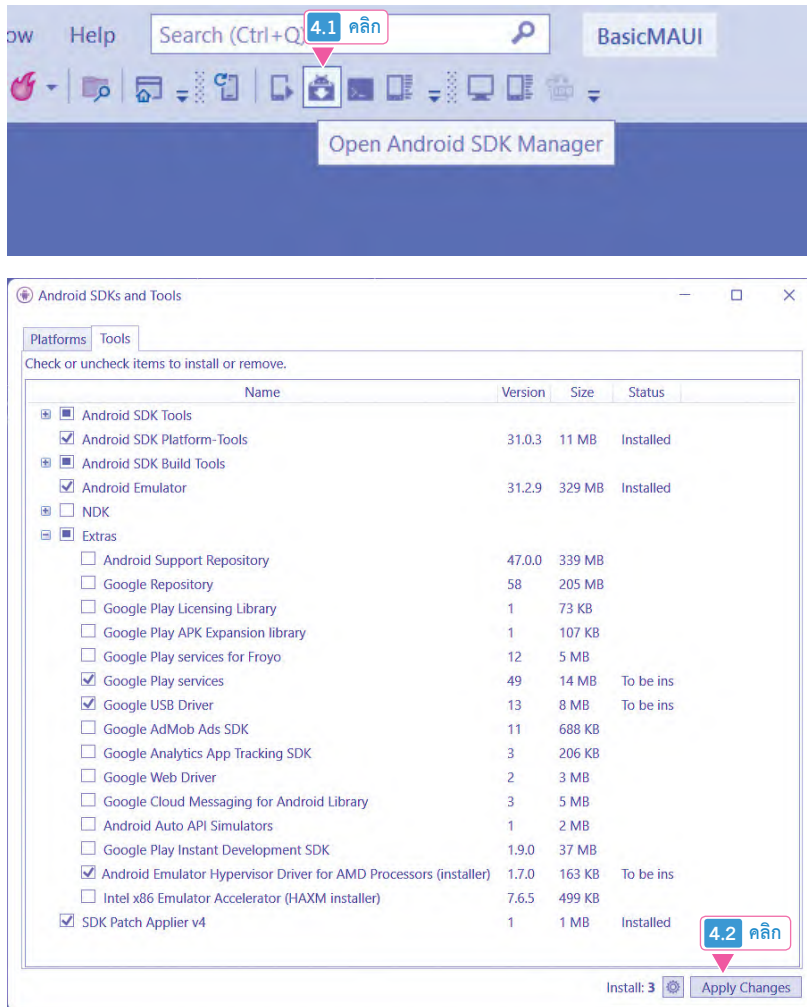
รูปที่ 2-6 แสดงการกำหนดรายละเอียดให้กับ Android Emulator

3. เข้าสู่ขั้นตอนการดาวน์โหลดรายการต่างๆ เพื่อสร้าง Android Emulator ขึ้นมาตามที่ผู้อ่านกำหนดไว้ เป็นขั้นตอนอัตโนมัติทั้งหมดให้ผู้อ่านรอจนเสร็จสมบูรณ์ก็จะได้ Android Emulator สำหรับทดสอบโปรเจกต์ .NET MAUI แบบ Android ในขั้นต้น ดังรูปที่ 2-7



รูปที่ 2-7 แสดง Android Emulator ที่ได้มา

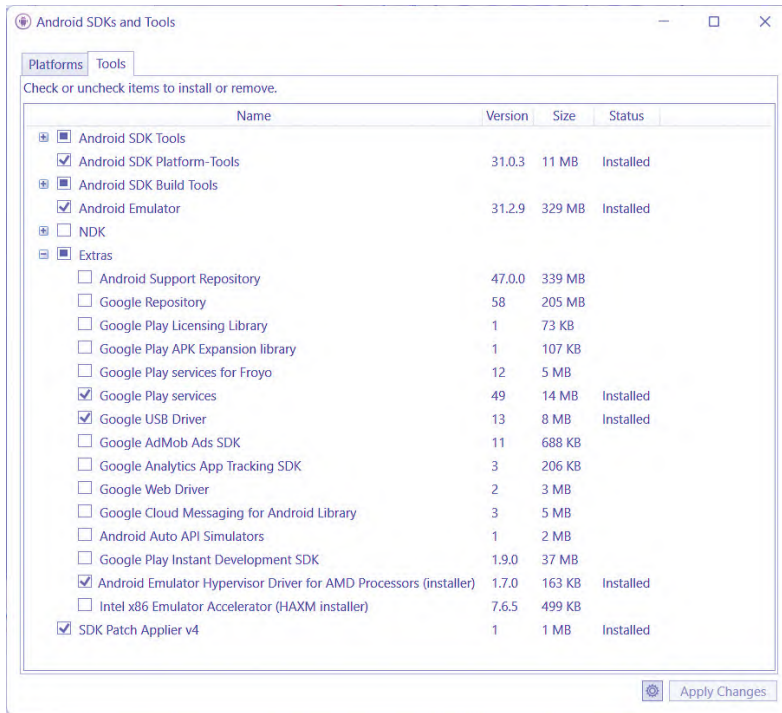
4. ในโปรแกรม Visual Studio ให้ผู้อ่านคลิกปุ่ม  ที่แท็บ Tools เลือกรายการติดตั้งเพิ่มเติมสำหรับโปรเจกต์ Android Apps ดังรูปที่ 2-8



รูปที่ 2-8 แสดงรายการติดตั้งเพิ่มเติมสำหรับโปรเจกต์ 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 แสดงรายการไฟล์ติดตั้งเพิ่มเติมสำหรับโปรเจกต์ Android Apps

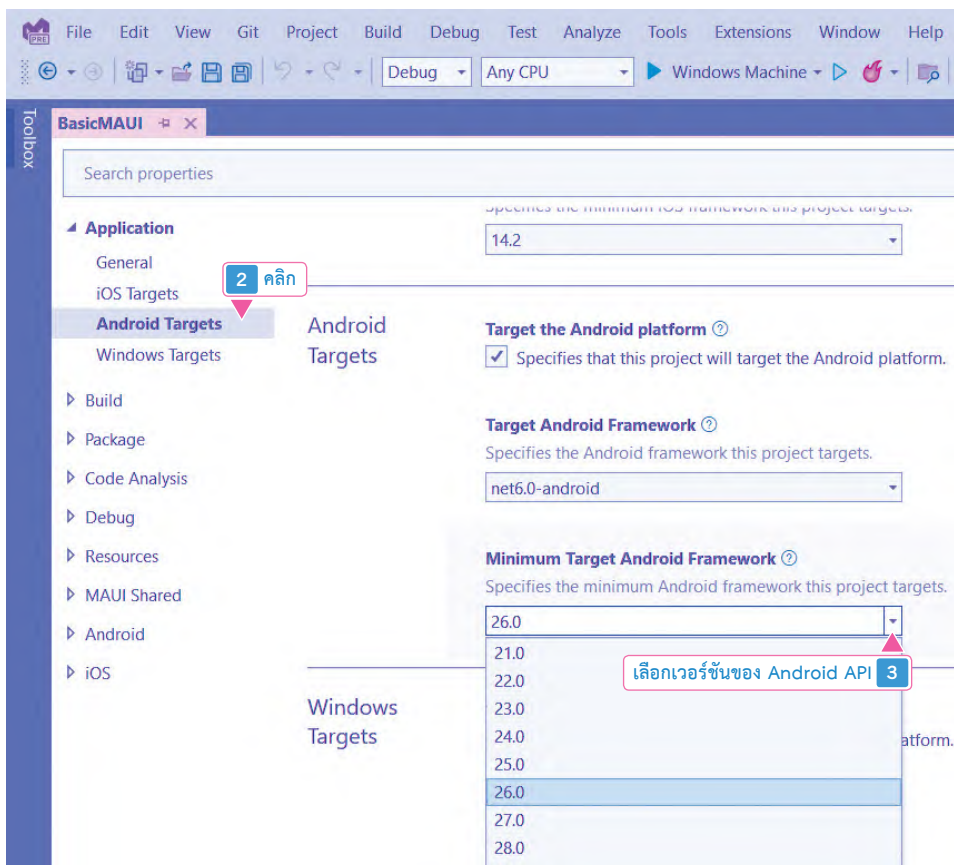
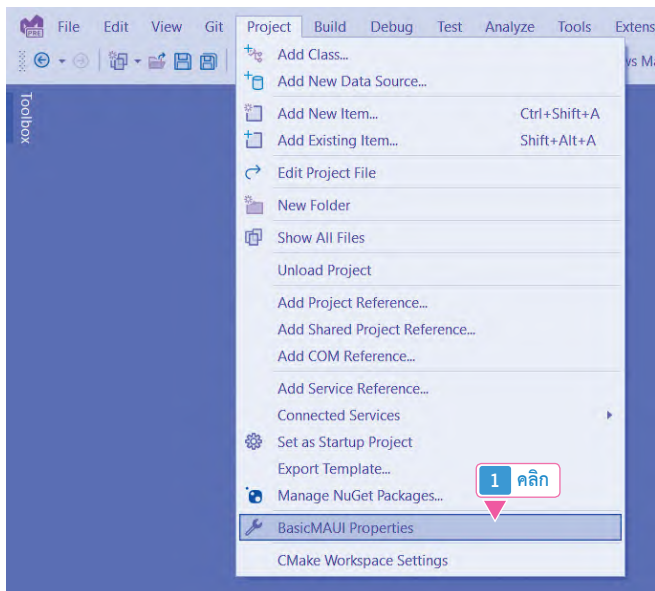
จากรูปที่ 2-9 ให้ผู้อ่านรอขั้นตอนการดาวน์โหลดและติดตั้งจนเสร็จสมบูรณ์

การปรับช่วงเวอร์ชันของ Android API

ในกรณีที่มีการอัปเดต Android SDK เวอร์ชันใหม่เข้ามา ผู้อ่านสามารถปรับแต่งช่วงเวอร์ชันโปรเจกต์ Android ของผู้อ่านให้รองรับเวอร์ชันใหม่ๆ ได้เช่นกัน แต่การปรับปรุงโปรเจกต์ให้เป็นเวอร์ชันใหม่ดังกล่าว ไม่ได้หมายความว่าผู้อ่านสามารถใช้ฟีเจอร์ใหม่ๆ เหล่านี้ได้ทันที อาจจะต้องรออัปเดต .NET MAUI ให้เป็นเวอร์ชันใหม่ควบคู่กันไปด้วย (ผู้อ่านต้องรออัปเดตผ่านทาง 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 โปรเจกต์ เป็นไปตามเวอร์ชันที่ผู้อ่านดาวน์โหลดมา



รูปที่ 2-10 แสดงการปรับแต่งช่วงเวอร์ชันของโปรเจกต์ Android Apps

NOTE

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

ผลของการปรับเวอร์ชันล่าสุดเป็น Android 8.0 หมายความว่า Android Apps ที่ผู้อ่านสร้างขึ้นจะสามารถติดตั้งบนระบบปฏิบัติการ Android 8.0 ขึ้นไปเท่านั้น

การอัปเดต Android SDK ของ Visual Studio

สำหรับโปรเจกต์ Android ของ Visual Studio ผู้อ่านสามารถอัปเดต Android SDK เป็นเวอร์ชันใหม่ล่าสุด เพื่อให้ Android Apps ของผู้อ่านรองรับฟีเจอร์ใหม่ๆ ได้นั่นเอง



Warning

การอัปเดต Android SDK “ต้องทำในขณะที่โปรแกรม Visual Studio และ .NET MAUI มีการอัปเดตให้รองรับ Android SDK เวอร์ชันใหม่แล้วเท่านั้น” โดยการติดตามข่าวสารจากโมโครซอฟท์อยู่เสมอ ยึดตามช่วงเวลาของผู้อ่านเป็นหลักว่าในช่วงเวลาของผู้อ่านมีการอัปเดตโปรแกรม Visual Studio และ .NET MAUI รองรับ Android SDK ถึงเวอร์ชันใดแล้ว

หมายความว่าถ้าผู้อ่านอัปเดต Android SDK เวอร์ชันใหม่ แต่โปรแกรม Visual Studio และ .NET MAUI ไม่รองรับ การอัปเดตดังกล่าวไม่มีประโยชน์แต่อย่างใด อาจจะก่อให้เกิดข้อผิดพลาดระหว่างเวอร์ชันได้อีกด้วย

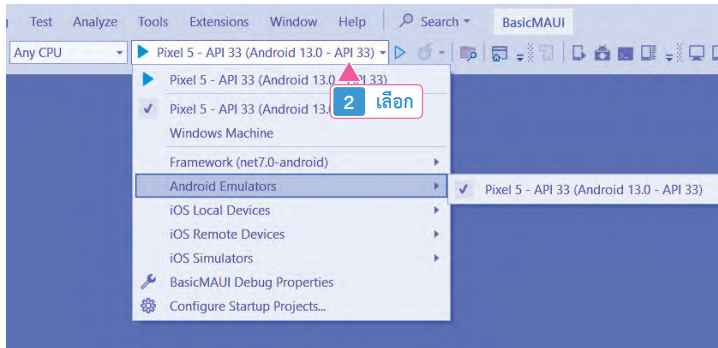
การทดสอบโปรเจกต์ .NET MAUI แบบ Android Apps ด้วย Android Emulator

เนื้อหาที่นำเสนอในหนังสือเล่มนี้เป็นการใช้งาน .NET MAUI บนเครื่อง PC เป็นหลัก ส่งผลให้การทดสอบโปรเจกต์ทั้งหมดรันแบบ Android Apps มีความเหมาะสมที่สุด

การทดสอบโปรเจกต์ .NET MAUI แบบ Android Apps มี 2 วิธี คือ

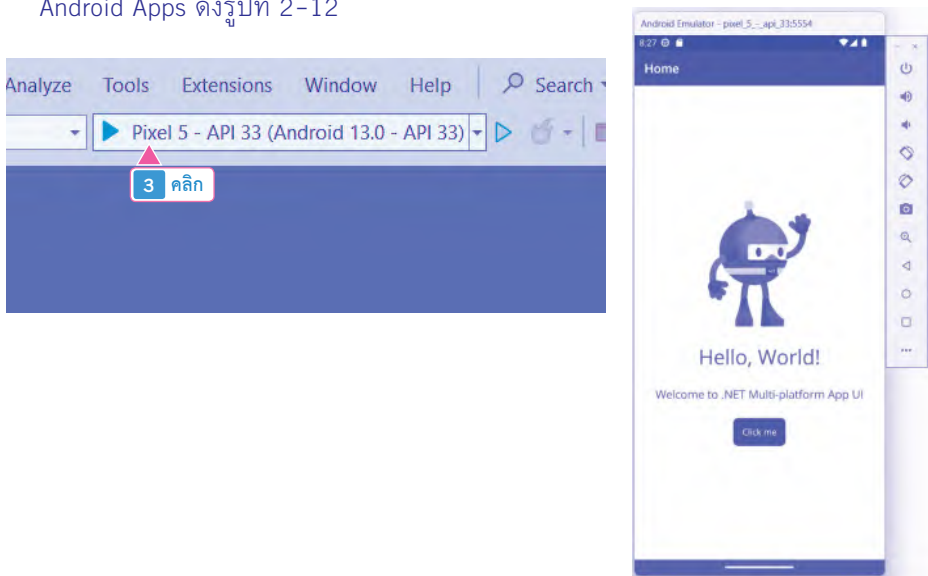
1. **ทดสอบด้วย Android Emulator** เป็นการทดสอบการทำงานของโปรเจกต์ในขั้นต้น มีข้อดีก็คือ เราสามารถเห็นผลการทำงานใน PC ของเราได้เลย แต่มีข้อเสียก็คือ ความเร็วการทดสอบเป็นไปตามสเปคเครื่องของเรา และมีข้อจำกัดของฟีเจอร์ต่างๆ ในระดับหนึ่ง ในยุคปัจจุบัน ผู้เขียนไม่แนะนำให้ทดสอบโปรเจกต์ด้วย Android Emulator แล้ว ใช้เท่าที่จำเป็น

2. ทดสอบด้วยมือถือ Android จริง เป็นวิธีที่ผู้เขียนแนะนำให้ใช้เป็นหลัก เพราะเราได้ทดสอบและเห็นผลการทำงานบนมือถือจริง และยังช่วยลดระยะเวลาทดสอบโปรเจกต์ในแต่ละครั้งอีกด้วย เกิดมาจากการที่เราไม่ต้องสร้าง Android Emulator ใน PC ของเรานั้นเอง ลำดับแรกเราจะมาทดสอบโปรเจกต์ BasicMAUI ด้วย Android Emulator ก่อน โดยการคลิกที่หัวลูกศรในปุ่ม **Pixel 5 - API 30 (Android 11.0 - API 30)** จากนั้นเลือกโฟกัสโปรเจกต์ .NET MAUI เป็นแบบ Android Emulators ดังรูปที่ 2-11



รูปที่ 2-11 แสดงการทดสอบโปรเจกต์ .NET MAUI แบบ Android Apps

3. ให้คลิกที่ปุ่ม **Pixel 5 - API 30 (Android 11.0 - API 30)** เพื่อรันโปรเจกต์ .NET MAUI แบบ Android Apps ดังรูปที่ 2-12



รูปที่ 2-12 ผลการรันโปรเจกต์ BasicMAUI ใน Android Emulator

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

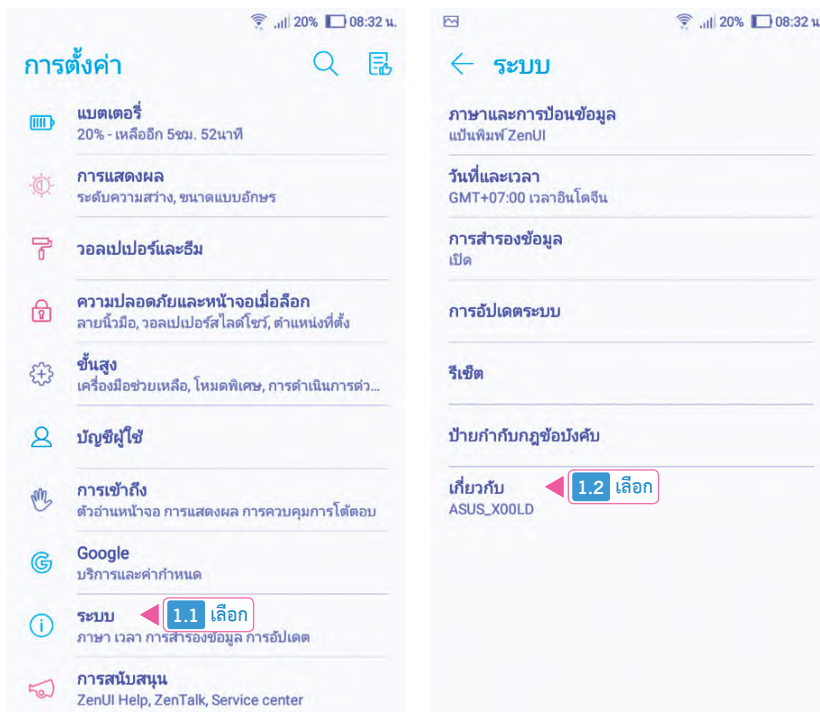
การทดสอบโปรเจกต์ Android Apps บนเครื่องจริง (ใช้กับ Android 8.0 ขึ้นไป เท่านั้น)

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

มือถือหรือแท็บเล็ต Android ที่จะนำมาใช้ทดสอบต้องมีการเปิดโหมดนักพัฒนา (Developer options) ก่อน ถ้ามือถือหรือแท็บเล็ตของผู้อ่านเคยทำมาก่อนแล้ว ไม่จำเป็นต้องทำซ้ำ สามารถใช้ทดสอบโปรเจกต์ได้เลย

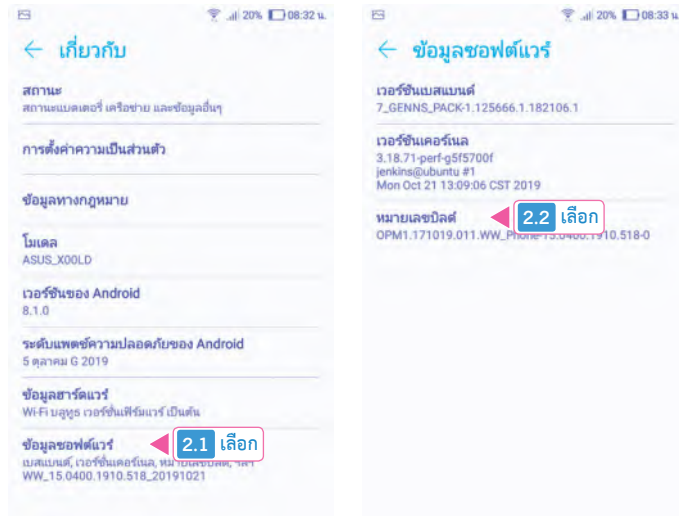
ขั้นตอนการกำหนดในเครื่องจริงของแต่ละยี่ห้อ แต่ละรุ่น อาจจะมีรายละเอียดปลีกย่อย แตกต่างกันไปบ้าง อาจจะใช้คำที่ไม่เหมือนกัน แต่โดยรวมแล้วใช้หลักการเดียวกันหมด นั่นคือ เปิดโหมด Debug ขึ้นมาในเครื่อง มีขั้นตอนดังนี้

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



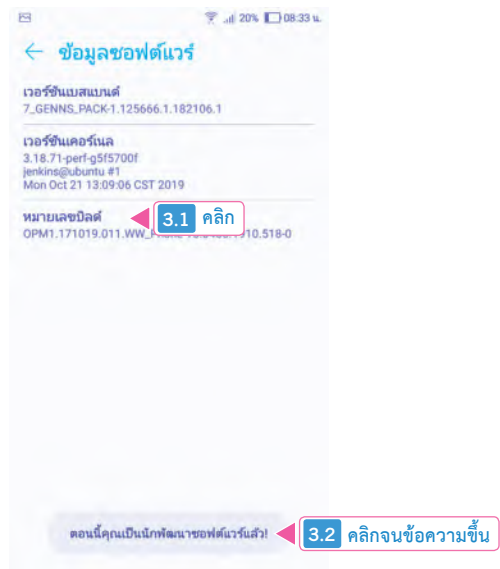
รูปที่ 2-13 แสดงการดูข้อมูลเครื่องที่จะนำมาทดสอบ

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



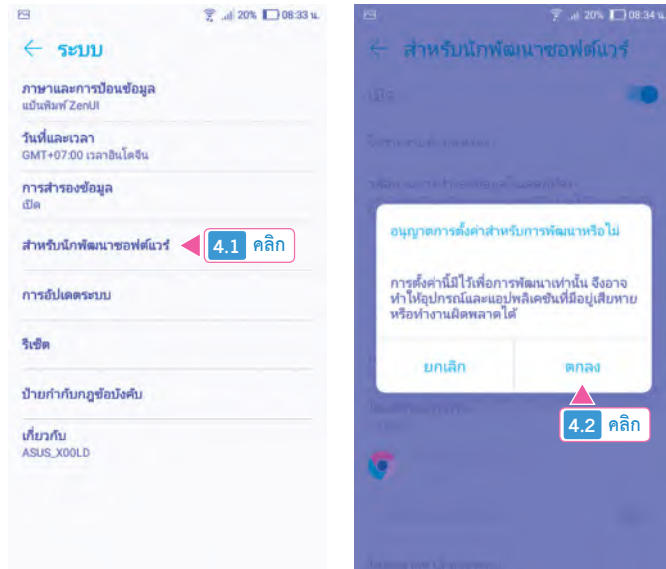
รูปที่ 2-14 แสดงหมายเลข Build number

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



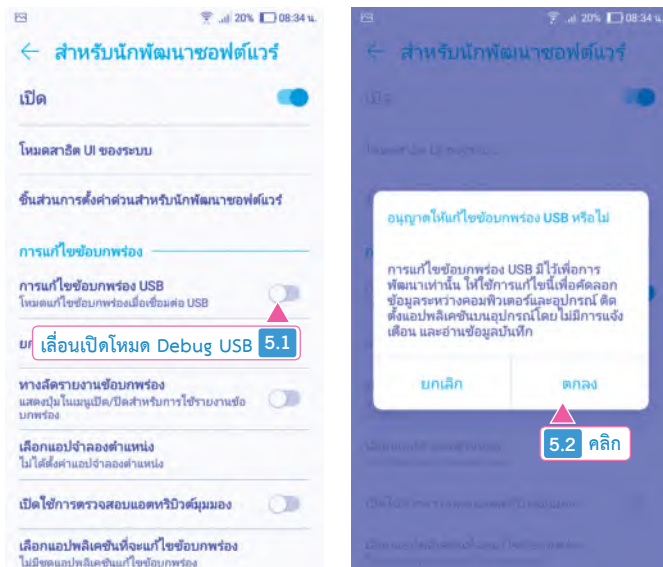
รูปที่ 2-15 แสดงการเปิดโหมดนักพัฒนาในเครื่องจริง

4. ให้ผู้อ่านกดปุ่ม Back กลับมา จะพบว่ามีการสำหรับนักพัฒนาซอฟต์แวร์เพิ่มขึ้นมาใหม่ ให้เข้าไปเปิดโหมดสำหรับนักพัฒนา ดังรูปที่ 2-16



รูปที่ 2-16 แสดงการเปิดโหมดสำหรับนักพัฒนา

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



รูปที่ 2-17 แสดงการเปิดโหมด Debug ขึ้นมา