

Professional Visual Basic 2015



- อ่านเข้าใจง่าย อธิบายอย่างเป็นขั้นตอน มีตัวอย่างประกอบทุกหัวข้อ
- รองรับการพัฒนาทั้ง PC, Mobile และ Web Site
- เหมาะสำหรับนักเรียน นักศึกษา และผู้ที่สนใจพัฒนาแอปพลิเคชัน



ลิงค์ดาวน์โหลด

<http://www.infopress.co.th/devbook/CodeVB2015.zip>

ศุภชัย สพพานิช



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



Think
Beyond



Professional Visual Basic 2015

ผู้แต่ง	ศุภชัย สมพานิช	http://facebook.com/thaivb.net
บรรณาธิการ	กิตินันท์ พลสวัสดิ์	kitinan_p@idcpremier.com
ออกแบบปก	วสันต์ ฟิ่งพูลผล, ยุทธนา เกิดประดิษฐ์	
ออกแบบและจัดรูปเล่ม	วุฒิพันธ์ สมพระเมฆ, สิริลักษณ์ วาระเลิศ	
พิสูจน์อักษร	มนฤดี ศรีอุทโยภาส	
ประสานงานการผลิต	สุพิศรา อาจปรุ, ปฐมพล ธรรมศรีสกุล	

Visual Studio 2015 เป็นเครื่องหมายการค้าของบริษัท Microsoft และเครื่องหมายการค้าอื่นๆ ที่อ้างถึงเป็นของบริษัทอื่นๆ

สงวนลิขสิทธิ์ตามพระราชบัญญัติลิขสิทธิ์ พ.ศ. 2537 โดยบริษัท ไอดีซี พรีเมียร์ จำกัด ห้ามลอกเลียนไม่ว่าส่วนใดส่วนหนึ่งของหนังสือเล่มนี้ ไม่ว่าในรูปแบบใดๆ นอกจากจะได้รับอนุญาตเป็นลายลักษณ์อักษรจากผู้จัดพิมพ์เท่านั้น

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

ข้อมูลทางบรรณานุกรม



ศุภชัย สมพานิช
Professional Visual Basic 2015

นนทบุรี : ไอดีซีฯ, 2559
424 หน้า

1. วิชาลสตูดิโอ (โปรแกรมคอมพิวเตอร์)

2. การเขียนโปรแกรม

I ชื่อเรื่อง

005.133

ISBN 885-916-100-436-3

พิมพ์ครั้งที่ 1 พฤษภาคม 2559

จัดพิมพ์และจัดจำหน่ายโดย

บริษัท ไอดีซี พรีเมียร์ จำกัด

200 หมู่ 4 ชั้น 19 ห้อง 1901 อาคารจัสมินอินเตอร์เนชั่นแนลทาวเวอร์
ถ.แจ้งวัฒนะ อ.ปากเกร็ด จ.นนทบุรี 11120

โทรศัพท์ 0-2962-1081 (อัตโนมัติ 10 คู่สาย) โทรสาร 0-2962-1084

ราคา 350 บาท

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



ปัจจุบันโปรแกรมหรือแอปพลิเคชันไม่ได้มีแต่เพียงแค่บนเครื่องคอมพิวเตอร์ตั้งโต๊ะเพียงอย่างเดียว แต่ได้มีการพัฒนาไปยังเครื่องมืออิเล็กทรอนิกส์ต่างๆ เป็นจำนวนมาก โดยเฉพาะบนโทรศัพท์เคลื่อนที่ หรือที่เรียกกันว่า สมาร์ทโฟน และแท็บเล็ต

จึงเป็นที่มาของ Visual Studio 2015 ที่ไมโครซอฟท์ตั้งใจพัฒนาขึ้นมาเพื่อรองรับการพัฒนาโปรแกรมเกือบจะทุก ๆ ช่องทาง ทั้งการพัฒนาโปรแกรมสำหรับ PC (หรือ Notebook), การพัฒนาโปรแกรมบนเว็บ (ไม่ขึ้นกับแพลตฟอร์ม), การพัฒนาโปรแกรมสำหรับมือถือและแท็บเล็ต เนื้อหาในหนังสือเล่มนี้จึงเรียบเรียงขึ้นบนแนวความคิดที่ว่า “เมื่อผู้อ่านอ่านหนังสือเล่มนี้ ต้องสามารถพัฒนาโปรแกรมบน Desktop, Web, มือถือ และแท็บเล็ตได้ด้วย Visual Basic 2015” ซึ่งทางสำนักพิมพ์เชื่อว่าผู้อ่านสามารถทำได้จริง เพราะผู้เขียนได้อธิบายทุกขั้นตอนอย่างละเอียด เป็นขั้นเป็นตอน สามารถทำตามได้จริง

ทั้งนี้ ทางสำนักพิมพ์หวังเป็นอย่างยิ่งว่า หนังสือเล่มนี้จะสามารถช่วยให้ท่านผู้อ่านทุกท่านพัฒนาโปรแกรมได้จริง และหากหนังสือเล่มนี้มีข้อผิดพลาดประการใดต้องขออภัยมา ณ โอกาสนี้ด้วย ทางสำนักพิมพ์พร้อมน้อมรับทุกคำติชม เพื่อนำไปใช้ในการปรับปรุงในครั้งต่อไป

กิตินันท์ พลสวัสดิ์
พฤษภาคม 2559

คำนำ

ในปัจจุบันโปรแกรม Visual Studio ถูกพัฒนามาถึงเวอร์ชัน 2015 ประกอบไปด้วยฟีเจอร์ต่างๆ มากมาย นับได้ว่าเป็นเวอร์ชันที่พัฒนาแบบก้าวกระโดดอีกครั้งหนึ่งในโลกของ .NET

“หนังสือ Professional Visual Basic 2015” เล่มนี้เป็นการนำเสนอเนื้อหาพัฒนาโปรแกรมทั้งสิ้น 4 ด้าน คือ Desktop Apps, Database Apps, Web Apps และ Mobile Apps เพื่อให้ตรงกับโลกของความเป็นจริงในช่วงเวลานี้

รูปแบบการนำเสนอใช้วิธีการนำเสนอตั้งแต่ระดับพื้นฐานที่สุด ไประดับไปเรื่อยๆ โดยอาศัยความรู้ของแต่ละหัวข้อสะสมในทุกๆ บท

คุณผู้อ่านท่านใดที่ต้องการสอบถามปัญหา, ดิชม หรือให้คำแนะนำ มี 2 ช่องทาง คือ

- แฟนเพจของผู้เขียนที่ <https://www.facebook.com/thaivb.net>
- ช่อง Youtube ค้นหาช่องที่ชื่อว่า Thaivb.NET

หากเนื้อหาในส่วนใดเกิดข้อผิดพลาด หรือมีการอัปเดตเพิ่มเติม สามารถติดตามได้จากแฟนเพจผู้เขียนข้างต้น

ศุภชัย สมพานิช
พฤษภาคม 2559



Professional Visual Basic 2015

บทที่ 1 เตรียมความพร้อมก่อนเริ่มต้นเขียนโปรแกรมด้วยภาษา Visual Basic 2015....1

การดาวน์โหลดและติดตั้ง Visual Studio 2015.....2

การตั้งค่า Visual Studio 2015 เบื้องต้น.....3

 การกำหนดให้แสดงไอคอนเลือกเลือกโปรเจกต์ (New Project Dialog).....3

 การกำหนดให้แสดงหมายเลขกำกับโค้ดในแต่ละบรรทัด.....4

การสร้างโปรเจกต์ใหม่.....5

 ทำความเข้าใจกับโปรเจกต์ประเภทอื่นๆ ของ Visual Studio 2015.....7

 ทำความเข้าใจกับสภาพแวดล้อม (IDE) ของ VS ในขั้นต้น..... 10

 มือใหม่ต้องทราบก่อนเขียนโปรแกรมในโลกของ .NET..... 11

 การออกแบบส่วนแสดงผลด้วยคอนโทรล (User Interface)..... 11

 การปรับแต่งคอนโทรลด้วยวิธีแก้ไขคุณสมบัติ (Property)..... 12

 ทำความเข้าใจกับฟอร์ม (Form) 14

 เริ่มต้นสร้างแอปพลิเคชันแรก..... 15

 การทดสอบโปรเจกต์..... 17

 วิธีการสั่งให้เปิดฟอร์มใหม่โดยการเขียนโค้ด..... 18

 หลักการทำงานของภาษา VB..... 18

 สรุปท้ายบท 20

บทที่ 2 พื้นฐานการเขียนโปรแกรมด้วยภาษา Visual Basic 21

คำสั่ง Option Explicit On กับ Option Strict On..... 21

การประกาศตัวแปรสำหรับใช้เก็บข้อมูล 22

 กฎการตั้งชื่อตัวแปร..... 22

 Reserved Words..... 22

ขอบเขตของตัวแปร (Variable Scope)..... 24

 ตัวแปรระดับ Local และระดับฟอร์ม 25

 ตัวแปรระดับ Block..... 27

ชนิดข้อมูล (Type)..... 28

 ขอบเขตการจัดเก็บข้อมูลแต่ละชนิดข้อมูลและขนาดหน่วยความจำที่ใช้..... 28

 การใช้งานข้อมูลชนิดตัวเลข 31

 การกำหนดชนิดข้อมูลโดยอัตโนมัติ (Type Inference)..... 32

การใช้งานเลขจำนวนเต็มชนิด BigInteger.....	34
การจัดรูปแบบของตัวเลขทศนิยม.....	36
รู้จักและใช้งานชนิดข้อมูลข้อความ (String)	38
วิธีการตรวจสอบจำนวนตัวอักษรใน String.....	39
การต่อข้อความเข้าด้วยกัน.....	39
การแปลงตัวเลขที่อยู่ในฐานะข้อความ String ให้กลายเป็นข้อมูลชนิดตัวเลข	41
การแทนที่ข้อความด้วยเมธอด Replace()	42
การแบ่งข้อความด้วยฟังก์ชัน Split()	43
การใช้งานข้อมูลชนิดตัวอักษร Char	43
การแปลงชนิดข้อความ String เป็นชนิดข้อมูลอาร์เรย์ของ Char	44
การดักจับข้อผิดพลาดด้วย Try Catch.....	45
สรุปท้ายบท	46

บทที่ 3

พื้นฐานการใช้งานคอนโทรล 47

ทำความเข้าใจกับแถบคอนโทรล	47
แสดงข้อความด้วยคอนโทรล Label.....	48
คุณสมบัติที่น่าสนใจของคอนโทรล Label.....	48
การสร้างปุ่มกดด้วยคอนโทรล Button.....	49
คุณสมบัติที่น่าสนใจของคอนโทรล Button.....	49
รับข้อความจากผู้ใช้งานด้วยคอนโทรล TextBox.....	50
คุณสมบัติที่ควรรู้ของคอนโทรล TextBox.....	53
การแสดงผลภาพด้วยคอนโทรล PictureBox.....	54
คุณสมบัติที่ควรรู้ของคอนโทรล PictureBox.....	56
การสร้างส่วนแสดงผลแบบ list ด้วย ComboBox และ ListBox.....	57
การเพิ่ม ถอด และล้างรายการใน ListBox.....	61
การสร้างส่วนแสดงผลแบบรายการด้วยคอนโทรล ListView.....	65
การสร้างไฮเปอร์ลิงค์ด้วย LinkLabel.....	69
คุณสมบัติที่น่าสนใจของคอนโทรล LinkLabel.....	70
การแสดงความคืบหน้าในการทำงานด้วยคอนโทรล ProgressBar.....	71
การแสดงคำอธิบายด้วย ToolTip.....	73

การแสดงผลแบบลำดับชั้นด้วยคอนโทรล TreeView.....	74
คุณสมบัติที่น่าสนใจของคอนโทรล TreeView.....	78
การสร้างตัวจับเวลาด้วย Timer	79
การสร้างตัวเลือกแบบเลือกได้ 1 อย่างด้วยคอนโทรล RadioButton.....	80
การสร้างตัวเลือกแบบหลายตัวเลือกด้วยคอนโทรล CheckBox.....	81
การจัดกลุ่มด้วยคอนโทรล GroupBox.....	83
การใช้งานตัวบรรจุ Panel.....	84
การสร้างส่วนแสดงผลแบบแท็บด้วยคอนโทรล TabControl.....	84
การสร้างตัวเลือกวันที่ด้วย DateTimePicker.....	85
คุณสมบัติที่น่าสนใจของคอนโทรล DateTimePicker.....	87
การสร้างปฏิทินด้วยคอนโทรล MonthCalendar.....	87
คุณสมบัติที่น่าสนใจของคอนโทรล MonthCalendar.....	89
การสร้างรายการแบบมีตัวเลือก CheckBox ด้วยคอนโทรล CheckedListBox.....	90
การกำหนดรูปแบบข้อมูลที่ได้รับเข้ามาด้วย MaskedTextBox.....	92
การสร้างตัวเลือกแบบตัวเลขด้วย NumericUpDowns	94
ทำงานกับไฟล์ข้อความด้วยคอนโทรล RichTextBox, OpenFileDialog และ SaveFileDialog.....	95
รายการคุณสมบัติของคอนโทรล RichTextBox.....	95
รายการคุณสมบัติของคอนโทรล OpenFileDialog.....	95
รายการคุณสมบัติของคอนโทรล SaveFileDialog.....	96
สรุปท้ายบท	100

บทที่ 4 การตรวจสอบและควบคุม..... 101

ตัวดำเนินการด้านคณิตศาสตร์.....	101
ตัวดำเนินการด้านเปรียบเทียบ	102
การตรวจสอบเงื่อนไขด้วยคำสั่ง If...Then.....	103
การตรวจสอบเงื่อนไขด้วยคำสั่ง If...Then...Else.....	104
การตรวจสอบเงื่อนไขด้วยคำสั่ง If...Then...Elseif.....	105
การตรวจสอบเงื่อนไขด้วยคำสั่ง Select Case.....	106
การใช้ตัวดำเนินการ “และ” (AndAlso) และตัวดำเนินการ “หรือ” (OrElse).....	108
การตรวจสอบเงื่อนไขร่วมกับตัวดำเนินการด้านตรรกะ.....	109

การวนลูปด้วยคำสั่ง For...Next	111
การวนลูปด้วยคำสั่ง While...End While	112
การวนลูปด้วยคำสั่ง Do...Loop While	114
การวนลูปด้วยคำสั่ง Do Until...Loop	116
การวนลูปด้วยคำสั่ง Do...Loop Until	117
สรุปท้ายบท	118

บทที่ 5 การใช้งานฟังก์ชันและพารามิเตอร์ (Function & Parameter).....119

รูปแบบฟังก์ชันในภาษา VB	119
รูปแบบพื้นฐานของฟังก์ชันในภาษา VB มี 3 แบบ คือ.....	120
การสร้างฟังก์ชันที่มีชื่อเหมือนกันด้วยการทำ Overload Method	123
ลักษณะของพารามิเตอร์ในฟังก์ชัน.....	126
การกำหนดลักษณะเพิ่มเติมของพารามิเตอร์.....	128
พารามิเตอร์แบบทางเลือก (Optional Parameter)	128
การส่งค่าพารามิเตอร์แบบระบุชื่อ	130
การส่งพารามิเตอร์แบบ Params.....	131
สรุปท้ายบท	132

บทที่ 6 พื้นฐานการเขียนโปรแกรมเชิงวัตถุ (Object Oriented Programming - OOP)...133

การเขียนโปรแกรมเชิงวัตถุ (OOP) คืออะไร	133
พื้นฐานการสร้างคลาสขึ้นมาใช้งานเอง	134
ทำความเข้าใจกับชนิดข้อมูลแบบ Anonymous Type.....	139
การจัดประเภทคลาสด้วยระบบเนมสเปซ (Namespaces)	142
วิธีการเรียกใช้เนมสเปซ	142
แบบที่ 1 อาศัยคำสั่ง Imports	143
แบบที่ 2 อาศัยการระบุชื่อเต็ม.....	144
แบบที่ 3 อาศัยการตั้งชื่อ Alias.....	144
แบบที่ 4 อาศัยคลาส Global.....	145
ทำความเข้าใจกับ Constructor, การระบุสมาชิกด้วยคำสั่ง Me และการกำหนดค่าเริ่มต้น ด้วย Object_INITIALIZER	145
สรุปท้ายบท	150

บทที่ 7	อาร์เรย์และสตรัคเจอร์ (Array and Structure).....	151
	พื้นฐานการใช้งานตัวแปรชุดแบบอาร์เรย์.....	151
	การกลับด้านข้อมูลในอาร์เรย์.....	154
	การถอดค่าซ้ำกันในอาร์เรย์.....	156
	การหาค่ามากที่สุดและน้อยสุดในอาร์เรย์.....	157
	การสร้างอาร์เรย์ด้วยคลาส ArrayList.....	161
	การวนลูปในอาร์เรย์แบบหลายมิติด้วยลูปแบบ For Each.....	163
	พื้นฐานการใช้โครงสร้างแบบ Structure.....	164
	การสร้างคุณสมบัติและเมธอดใน Structure.....	166
	สรุปท้ายบท.....	168
บทที่ 8	การเขียนโปรแกรมแบบ Generic.....	169
	ทำความเข้าใจกับการเขียนโปรแกรมแบบ Generic ภายใน 30 นาที.....	169
	ทำไม .NET จึงต้องมี Generic.....	173
	การกำหนด Type ในขณะรัน (Run-Time).....	174
	การสร้างคลาสแบบ Generic.....	175
	การสร้างรายการ list แบบ Generic (Generic List).....	177
	การทำงานกับ Generic List ที่น่าสนใจ.....	178
	การทำงานกับโครงสร้างข้อมูลแบบเข้าก่อน-ออกก่อน หรือ FIFO (First-In First-Out)	
	ด้วยคลาส Queue.....	180
	การทำงานกับโครงสร้างข้อมูลแบบเข้าหลัง-ออกก่อน หรือ LIFO (Last-In First-Out)	
	ด้วยคลาส Stack.....	184
	สรุปท้ายบท.....	186
บทที่ 9	พื้นฐานการใช้งานระบบฐานข้อมูล SQL Server 2014 Express Edition.....	187
	การเตรียมสภาพแวดล้อมให้พร้อมใช้งาน.....	187
	การดาวน์โหลดและติดตั้ง SQL Server 2014 Express Edition.....	188
	ขั้นตอนการติดตั้งฐานข้อมูล SQL Server 2014 Express Edition.....	190
	พื้นฐานการใช้งานฐานข้อมูล SQL Server 2014 Express Edition.....	192
	การสร้างฐานข้อมูล.....	193
	การสร้างตาราง.....	194
	การกำหนด Primary Key.....	194

	การสร้างความสัมพันธ์ระหว่างตารางในฐานข้อมูล SQL Server 2014	195
	การ Backup และ Restore ฐานข้อมูล SQL Server 2014	197
	การ Backup ฐานข้อมูล.....	197
	การ Restore ฐานข้อมูล	199
	การยกเลิกการป้องกันการแก้ไขโครงสร้างตาราง	200
	การเพิ่มฐานข้อมูล Northwind เข้าไปใน SQL Server 2014 Express Edition.....	201
	สรุปท้ายบท	202
บทที่ 10	พื้นฐานการทำงานกับฐานข้อมูล SQL Server ด้วย LINQ to SQL.....	203
	พื้นฐานการติดต่อฐานข้อมูล SQL Server ด้วย LINQ to SQL.....	203
	การเลือกดูข้อมูลบางฟิลด์โดยการทำ Anonymous Type.....	208
	วิธีการกำหนดโครงสร้างข้อมูลขึ้นมาใช้งานด้วย Anonymous Type.....	210
	การใช้งาน Anonymous Type ในฐานและแหล่งข้อมูลร่วมกับ LINQ to SQL.....	213
	การจัดโครงสร้างข้อมูลใหม่ด้วยคำสั่ง Let.....	216
	พื้นฐานการค้นหาข้อมูลด้วย LINQ to SQL.....	219
	พื้นฐานการกรองข้อมูลด้วย LINQ to SQL.....	221
	การแสดงผลข้อมูลร่วมกันตั้งแต่ 2 ตารางขึ้นไป	224
	สรุปท้ายบท	226
บทที่ 11	การจัดการข้อมูลในฐานข้อมูลด้วย LINQ to SQL.....	227
	พื้นฐานการจัดการข้อมูลโดยการทำ Transaction.....	227
	การเพิ่มและลบข้อมูลแบบ 1 to Many (Master-Details)	237
	การทำแบบฟอร์มป้อนข้อมูล (Data Entry) โดยอาศัย LINQ to SQL.....	242
	การแก้ไขข้อมูลระหว่างฟอร์ม.....	254
	สรุปท้ายบท	260
บทที่ 12	พื้นฐานการพัฒนา Web Application ด้วย ASP.NET.....	261
	HTML5 โครงสร้างพื้นฐานของเว็บเพจ	262
	วิธีการเข้ารหัสหน้าเว็บเพจให้สามารถแสดงผลภาษาไทย.....	262
	โปรเจกต์ ASP.NET แรก.....	263
	ทำความเข้าใจกับสภาพแวดล้อมของ VS 2015 แบบ ASP.NET Web Application.....	266
	การตั้งเว็บเพจหน้าแรก.....	269

	การทดสอบรันโปรแกรม.....	269
	พื้นฐานการใช้งานเว็บคอนโทรลของ ASP.NET.....	270
	ความสัมพันธ์ระหว่างเว็บคอนโทรลของ ASP.NET กับภาษา HTML5.....	271
	สรุปท้ายบท.....	271
บทที่ 13	พื้นฐานการเขียนโปรแกรมใน ASP.NET.....	273
	ลำดับเหตุการณ์ของคลาส Page.....	273
	ทำความรู้จักกับคุณสมบัติ IsPostBack ของคลาส Page.....	275
	การกำหนดให้เว็บคอนโทรลมีการส่งค่ากลับ.....	277
	พื้นฐานการเก็บข้อมูลในเครื่องผู้ใช้งานด้วยคลาส Cookies.....	279
	การใช้งาน Cookies แบบมี Keys.....	282
	การทำคำทักทายผู้เยี่ยมชมเว็บเพจโดยใช้ Cookies.....	285
	พื้นฐานการเก็บข้อมูลผู้ใช้งานแต่ละคนด้วยตัวแปรแบบ Session.....	289
	สรุปท้ายบท.....	292
บทที่ 14	ASP.NET กับฐานข้อมูลด้วย LINQ to SQL.....	293
	พื้นฐานการติดต่อฐานข้อมูลด้วยเว็บคอนโทรล LinqData Source.....	293
	การจัดเรียงและแบ่งหน้าข้อมูลในเว็บคอนโทรล GridView.....	301
	การคิวรีข้อมูลจากฐานข้อมูลด้วย LINQ to SQL แบบเขียนโค้ดโดยตรง.....	303
	การสั่งให้เมธอดทำงานโดยอัตโนมัติ.....	305
	เขียนโค้ดใช้งาน LINQ to SQL แบบกำหนดคอลัมน์เอง.....	308
	การเขียน LINQ to SQL โดยอาศัยไวยากรณ์แบบ Lambda Expression.....	312
	ไวยากรณ์แบบผสมด้วย LINQ to SQL.....	313
	การแสดงผลข้อมูลจาก 2 ตารางขึ้นไปด้วย LINQ to SQL.....	315
	การแสดงผลข้อมูลจากหลายตารางด้วยคำสั่ง Join.....	319
	พื้นฐานการแสดงผลแบบรายการด้วยเว็บคอนโทรล DropDownList.....	320
	สรุปท้ายบท.....	323
บทที่ 15	การเตรียมสภาพแวดล้อมสำหรับพัฒนา Universal Apps.....	325
	การลงทะเบียนขอ Microsoft account.....	326
	การขอสิทธิ์ชั่วคราวในการพัฒนา Windows 10 Universal Apps.....	326
	เริ่มต้นสร้างแอปฯ แรกให้กับ Windows 10.....	327

การทดสอบรันโปรแกรมของ Windows 10 Apps.....	329
การรันโปรแกรมสำหรับหน้าจอใหญ่ (PC, Notebook และแท็บเล็ต)	329
การรันโปรแกรมสำหรับหน้าจอเล็ก (มือถือที่ติดตั้ง Windows 10 for phones).....	330
ข้อผิดพลาดของการทดสอบโปรแกรมแบบ Windows 10 for phones.....	332
การอ้างอิงคอนโทรลหรืออิลิเมนต์.....	333
วิธีการเขียนอิลิเมนต์ของ XAML.....	333
การกำหนดเพจเริ่มต้น.....	334
การจัดตำแหน่งคอนโทรลด้วยคุณสมบัติด้าน Alignment.....	335
การจัดระยะห่างด้วยคุณสมบัติ Margin.....	336
การ Copy คอนโทรลด้วยปุ่ม Alt บนคีย์บอร์ด.....	338
การยกเลิกการแก้ไขคุณสมบัติ	338
การเลือกโฟกัสคอนโทรลที่สนใจอยู่.....	339
พื้นฐานการกำหนดสีพื้นหลังให้กับส่วนแสดงผล	340
พื้นฐานการใช้คุณสมบัติที่เกี่ยวกับสี.....	342
การลงสีเดี่ยวแบบ Solid color brush.....	342
การลงสีแบบไล่โทนสี Gradient.....	343
ขั้นตอนการเพิ่มเพจใหม่เข้ามาในโปรแกรม.....	345
การสร้างเหตุการณ์ในสคริปต์ XAML.....	347
สรุปท้ายบท	350
บทที่ 16 การสร้างส่วนแสดงผลแบบรายการด้วยกลุ่มคอนโทรลประเภท list.....	351
การแสดงผลรายการด้วยคอนโทรล ComboBox.....	351
การเพิ่มรายการในคอนโทรล ComboBox ด้วยอิลิเมนต์	
<ComboBox.Items>...</ComboBox.Items>	353
การสร้างส่วนแสดงผลแบบรายการด้วยคอนโทรล ListBox.....	355
การแสดงรูปภาพในรายการของคอนโทรล ListBox.....	359
การเพิ่มและลบรายการในคอนโทรล ListBox.....	364
สรุปท้ายบท	366
บทที่ 17 การเขียนโปรแกรมแบบซิงโครนัสกับระบบเมนูและไดอะล็อก	367
การแจ้งเตือนแบบ Toast Notification.....	367
แนวความคิดของการเขียนโปรแกรมแบบซิงโครนัสได้ (Asynchronous Programming).....	371

การสร้างกรอบข้อความด้วยคลาส MatDialog.....	372
การใช้ MatDialog แบบมีการตรวจสอบปุ่มที่ถูกเลือก.....	373
การสร้าง MatDialog ก่อนออกจาก App.....	375
การสร้างเมนูจากการคลิกขวา (ContextMenu).....	377
การแสดงผลแบบ Popup ด้วยคอนโทรล Popup.....	379
พื้นฐานการสร้างแถบเมนูแบบ AppBar.....	381
การสร้างเมนูให้กับแถบ AppBar.....	383
สรุปท้ายบท.....	386

บทที่ 18 การพัฒนา Windows Store Apps แบบหลายหน้าจอ..... 387

การสร้างส่วนแสดงผลแบบเปลี่ยนหน้าได้.....	387
การรับ-ส่งข้อมูลระหว่างเพจ.....	390
การรักษาสถานะข้อมูลด้วยระบบแคช (Cache).....	394
การสร้างส่วนนำเสนอข้อมูลด้วยคอนโทรล FlipView.....	396
สรุปท้ายเล่ม.....	405

Index 406

เตรียมความพร้อมก่อนเริ่มต้นเขียนโปรแกรมด้วยภาษา Visual Basic 2015

โลกของการเขียนโปรแกรมไม่ได้มีแต่เพียงแค่พัฒนาโปรแกรมบน PC เพียงอย่างเดียวเท่านั้น แต่ยังรวมไปถึงมือถือและแท็บเล็ตอีกด้วย

ไมโครซอฟต์พัฒนา Visual Studio 2015 (เรียกสั้นๆ ว่า VS 2015) ขึ้นมา เพื่อรองรับการพัฒนาโปรแกรม 3 ช่องทาง คือ

1. พัฒนาโปรแกรมสำหรับ PC (หรือ Notebook)
2. พัฒนาโปรแกรมบนเว็บ (ไม่ขึ้นกับแพลตฟอร์ม)
3. พัฒนาโปรแกรมสำหรับมือถือและแท็บเล็ต

ในปัจจุบันไมโครซอฟต์พัฒนา Windows 10 ขึ้นมา ทำให้โลกของโปรแกรมบน PC, มือถือและแท็บเล็ตกลายเป็นโลกเดียวกัน ส่งผลให้นักพัฒนาไม่ต้องเสียเวลาเรียนรู้ซ้ำซ้อนหลายขั้นตอนอีกต่อไป

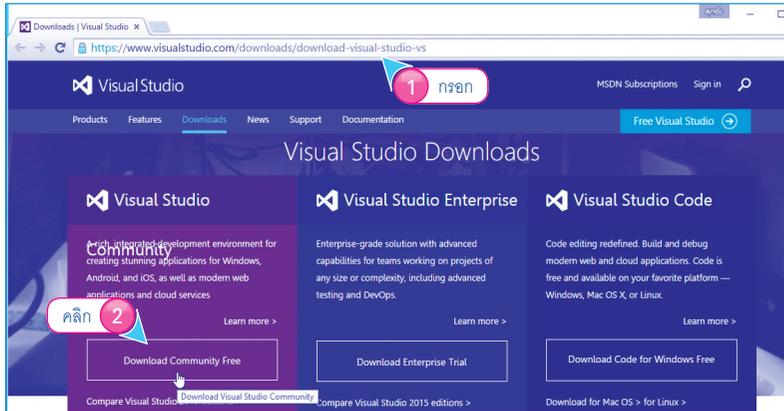
เนื้อหาที่จะนำเสนอในหนังสือเล่มนี้ ครอบคลุมการพัฒนาโปรแกรมบน Desktop, Web, มือถือ และแท็บเล็ต โดยอาศัยภาษา Visual Basic 2015 (เรียกสั้นๆ ว่า VB 2015) ที่มีเนื้อหาสาระเพียงพอที่จะต่อยอดเป็นโปรแกรมเมอร์มืออาชีพในลำดับต่อไปได้ไม่ยาก

การดาวน์โหลดและติดตั้ง Visual Studio 2015

ผู้อ่านสามารถเขียนโปรแกรมในตระกูล .NET ได้โดยการใช้โปรแกรมที่ชื่อว่า Visual Studio ซึ่งมีขั้นตอนดังนี้

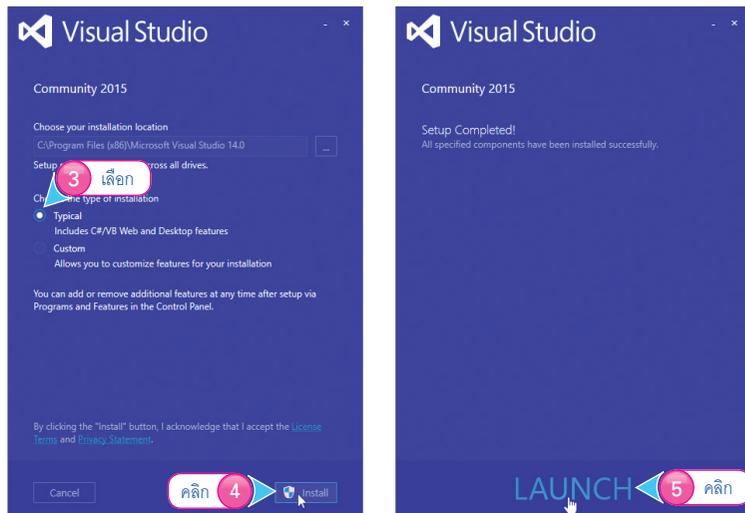
1. ให้ไปที่เว็บไซต์ <https://www.visualstudio.com/downloads/download-visual-studio-vs> เพื่อดาวน์โหลดโปรแกรม Visual Studio เวอร์ชันล่าสุดเท่าที่มีอยู่ในปัจจุบัน ดังรูปที่ 1-1
2. ให้ผู้อ่านคลิกดาวน์โหลดเวอร์ชัน Community Edition มาใช้งานได้ฟรี

รูปที่ 1-1
แสดงหน้าเว็บเพจ
ที่ให้ดาวน์โหลด
Visual Studio



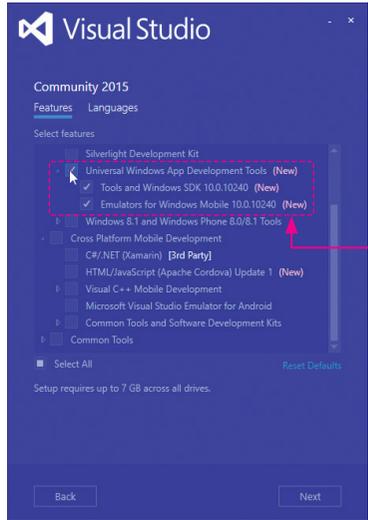
3. การติดตั้ง Visual Studio 2015 ให้เลือกแบบ Typical ซึ่งเป็นการติดตั้งแบบปกติ (ไม่รวมโปรเจกต์ประเภท Windows 10 Apps)
4. คลิกปุ่ม  Install
5. คลิกปุ่ม LAUNCH

รูปที่ 1-2
แสดงการติดตั้งโปรแกรม
Visual Studio 2015
Community Edition



ในกรณีที่ผู้อ่านต้องการพัฒนา Windows 10 Apps ด้วย ให้ติดตั้ง Visual Studio 2015 บน Windows 10 และเลือกติดตั้งแบบ Custom โดยเลือกตัวเลือกดังรูป

รูปที่ 1-3
กรณีติดตั้ง Visual Studio แบบ Custom



จากรูปที่ 1-3 ที่รายการ Universal Windows App Development Tools ให้ผู้อ่านเลือกติดตั้งเพิ่มเติม คือ ตัว Windows 10 SDK กับ Emulator สำหรับรุ่นมือถือคือ Windows 10

รายการที่ต้องติดตั้งเพิ่มเติม

การตั้งค่า Visual Studio 2015 เบื้องต้น

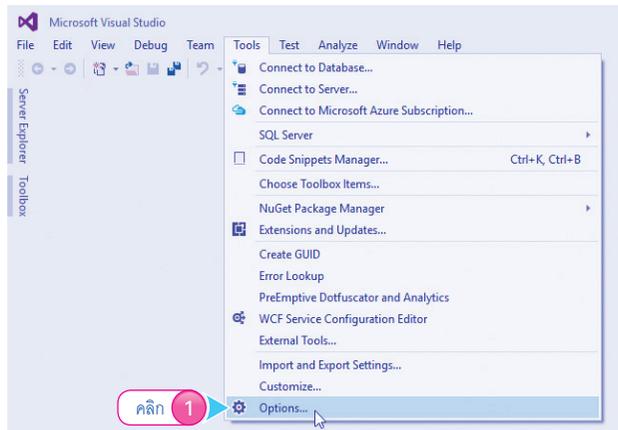
ก่อนเริ่มต้นเขียนเรียนรู้การพัฒนาแอปพลิเคชันด้วย Visual Studio 2015 ผู้เขียนอยากจะแนะนำให้ผู้อ่านรู้จักการตั้งค่าต่างๆ ที่ช่วยให้การพัฒนาแอปพลิเคชันต่างๆ ง่ายขึ้นดังนี้

การกำหนดให้แสดงไดอะล็อกเลือกโปรเจกต์ (New Project Dialog)

เพื่อเป็นการอำนวยความสะดวกในการสร้างโปรเจกต์ใหม่ทุกครั้ง ผู้อ่านสามารถกำหนดให้ VS แสดงไดอะล็อกเลือกสร้างโปรเจกต์ทุกครั้ง เมื่อมีการเปิด Visual Studio 2015 ขึ้นมา โดยมีขั้นตอนดังนี้

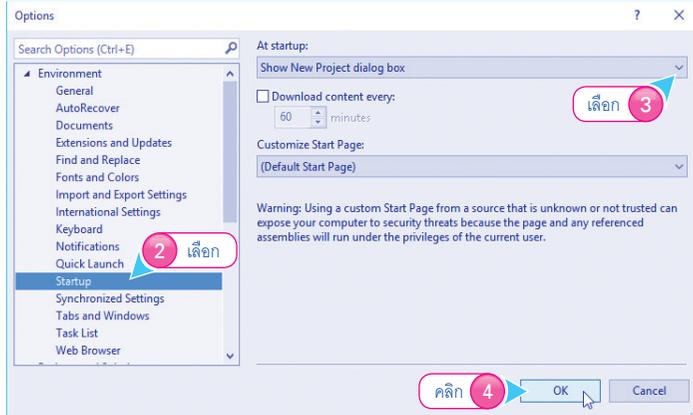
1. คลิกเมนู Tools > Options...
2. จะปรากฏไดอะล็อก Options ให้คลิกเลือก Environment > Startup
3. คลิกเลือก Show New Project dialog box
4. คลิกปุ่ม

รูปที่ 1-4
แสดงการเลือกรายการ Show New Project dialog box



รูปที่ 1-4 (ต่อ)

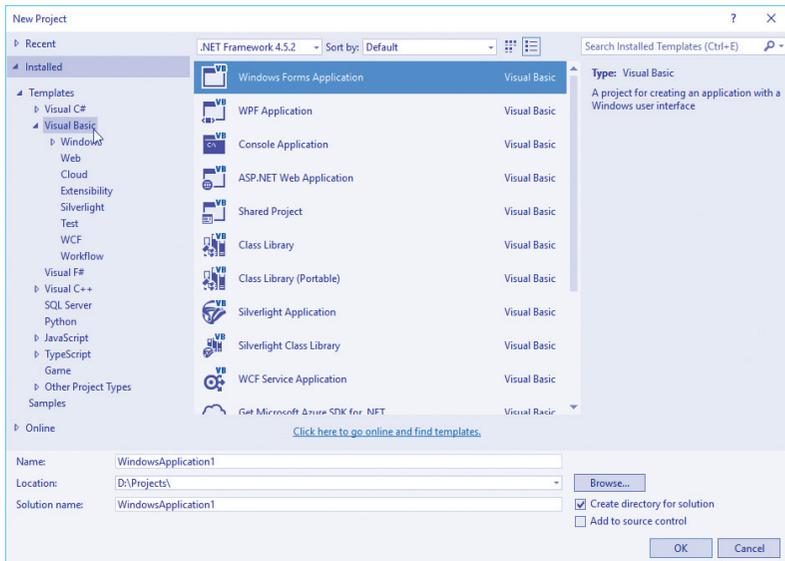
แสดงการเลือกรายการ Show New Project dialog box



5. จะปรากฏไดอะล็อกสำหรับสร้างโปรเจกต์ใหม่ดังรูป

รูปที่ 1-5

แสดงไดอะล็อกเลือกสร้างโปรเจกต์ใหม่



จากรูปที่ 1-5 ผู้อ่านสามารถเลือกสร้างโปรเจกต์ใน Visual Studio 2015 ได้ตามที่ต้องการจากไดอะล็อกนี้

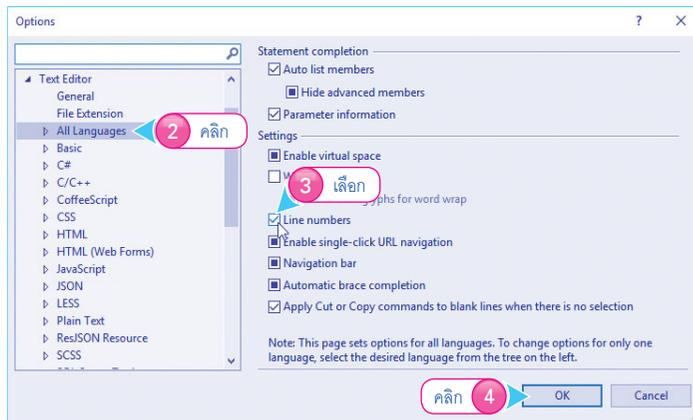
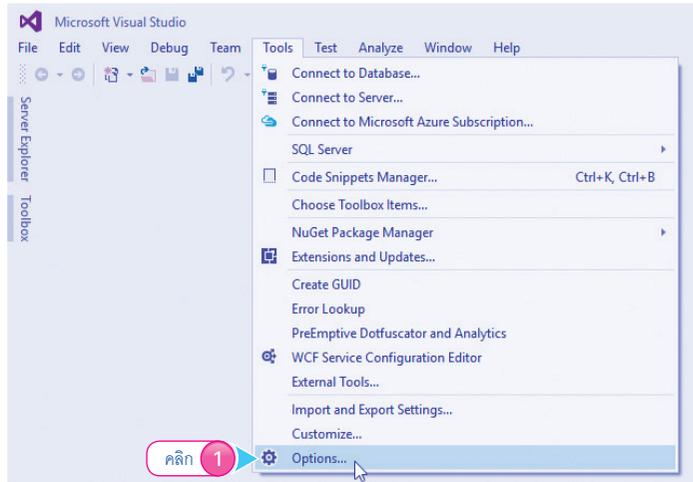
การกำหนดให้แสดงหมายเลขกำกับโค้ดในแต่ละบรรทัด

การกำหนดให้ Visual Studio 2015 แสดงหมายเลขกำกับโค้ดในแต่ละบรรทัด ผู้อ่านสามารถทำได้โดยมีขั้นตอนดังนี้

1. คลิกเมนู Tools > Options...
2. จะปรากฏไดอะล็อก Options ให้คลิกเลือก Text Editor > All Languages
3. คลิกเลือก Line numbers
4. คลิกปุ่ม

รูปที่ 1-6

แสดงการกำหนดให้แสดงหมายเลขกำกับโค้ดแต่ละบรรทัด



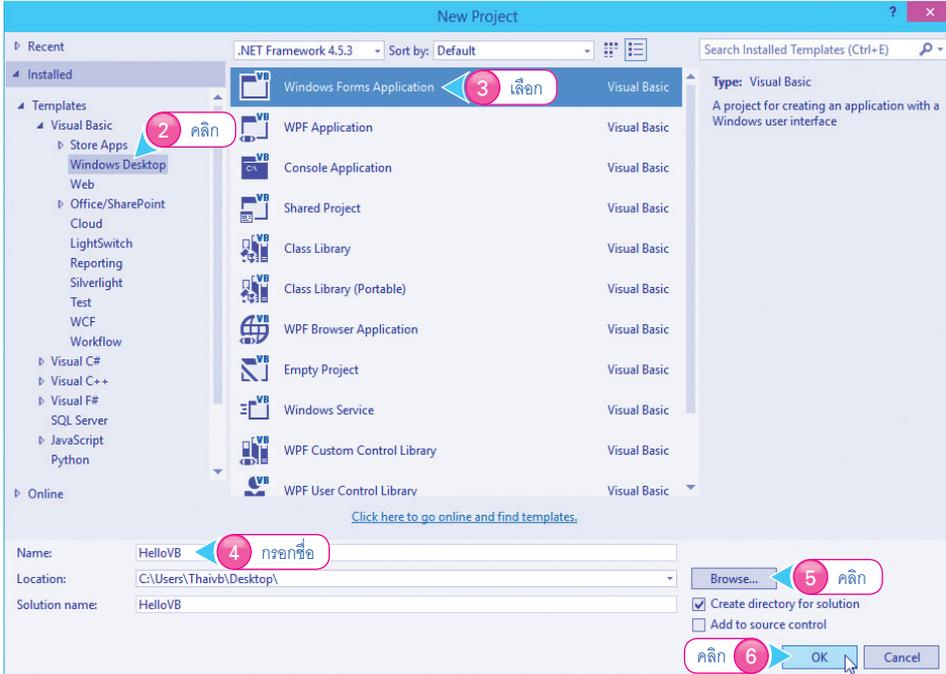
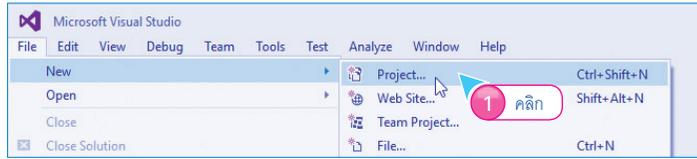
การสร้างโปรเจกต์ใหม่

การสร้างโปรเจกต์ใหม่ถือเป็นจุดเริ่มต้นของการพัฒนาแอปพลิเคชันต่าง ๆ ในที่นี้จะเริ่มต้นจาก Windows Forms Application ซึ่งเป็นโปรเจกต์หลักของ Visual Basic 2015 โดยมีขั้นตอนดังนี้

1. ให้ผู้อ่านคลิกเมนู File > New > Project...
2. คลิกเลือก Visual Basic > Store Apps > Windows Desktop
3. เลือกสร้างโปรเจกต์แบบ Windows Forms Application ซึ่งเป็นโปรเจกต์ที่ใช้สร้างโปรแกรมสำหรับทำงานบน Windows
4. กรอกชื่อโปรเจกต์
5. เลือกตำแหน่งที่อยู่ของโปรเจกต์
6. คลิกปุ่ม

รูปที่ 1-7

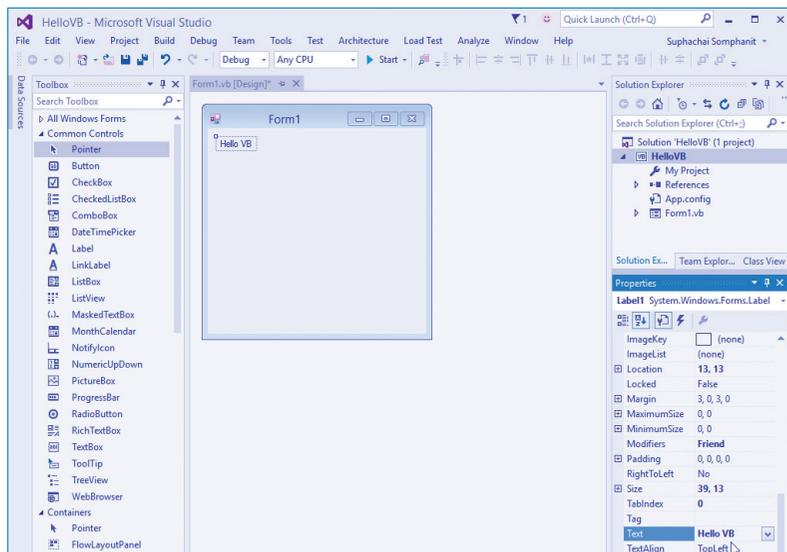
แสดงการสร้างโปรเจกต์
แบบ Windows Forms
Application



7. จะปรากฏหน้าต่าง Windows Forms Application สำหรับสร้างโปรเจกต์ดังรูป

รูปที่ 1-8

แสดงสภาพ
แวดล้อมของ VS
แบบ Windows
Forms Application

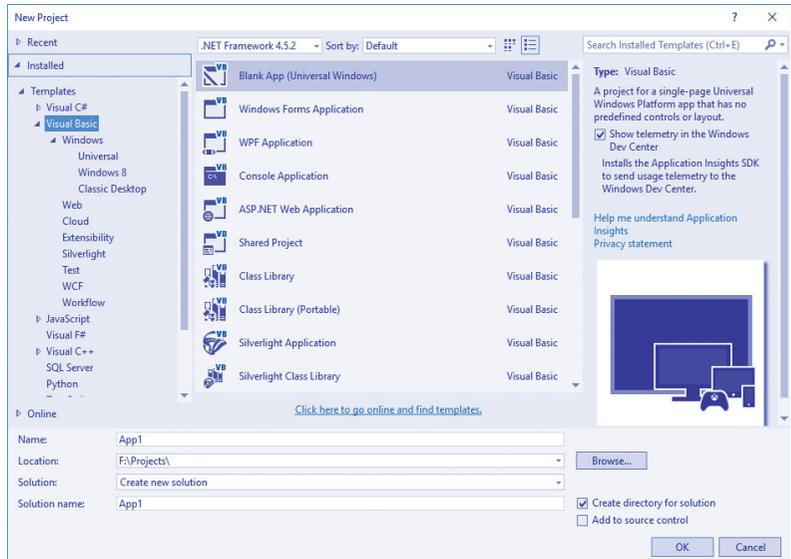


ทำความเข้าใจกับโปรเจกต์ประเภทอื่นๆ ของ Visual Studio 2015

นอกเหนือไปจากโปรเจกต์ประเภท Windows Forms Application แล้ว ตัวโปรแกรม Visual Studio 2015 ยังรองรับการพัฒนาแอปฯ ประเภทต่างๆ อีกมากมาย

รูปที่ 1-9

แสดงไดอะล็อก
สร้างโปรเจกต์
ประเภทต่างๆ
ของ Visual
Studio 2015



โปรเจกต์ประเภท Windows

โปรเจกต์ประเภท Windows ยังถูกแบ่งออกเป็น 3 ประเภท คือ

- **Universal** เป็นการพัฒนาแอปฯ ที่รันบนระบบปฏิบัติการ Windows 10 โดยสามารถสร้างโปรเจกต์ขึ้นมาเพียง 1 โปรเจกต์ แต่แอปฯ ที่ได้นำมารันได้ทั้ง PC แท็บเล็ต และมือถือ
- **Windows 8** เป็นการพัฒนาแอปฯ ที่รันบนระบบปฏิบัติการ Windows 8/8.1 เท่านั้น
- **Classic Desktop** เป็นการพัฒนาแอปแบบที่เราคุ้นเคยกันเป็นอย่างดี แอปฯ ที่ได้มาก็คือ ไฟล์ exe สามารถติดตั้งในระบบปฏิบัติการ Windows ต่างๆ ที่มีการติดตั้ง .NET Framework ตามเวอร์ชันที่เราเป็นผู้สร้างโปรเจกต์ขึ้นมา

โปรเจกต์ประเภท Web

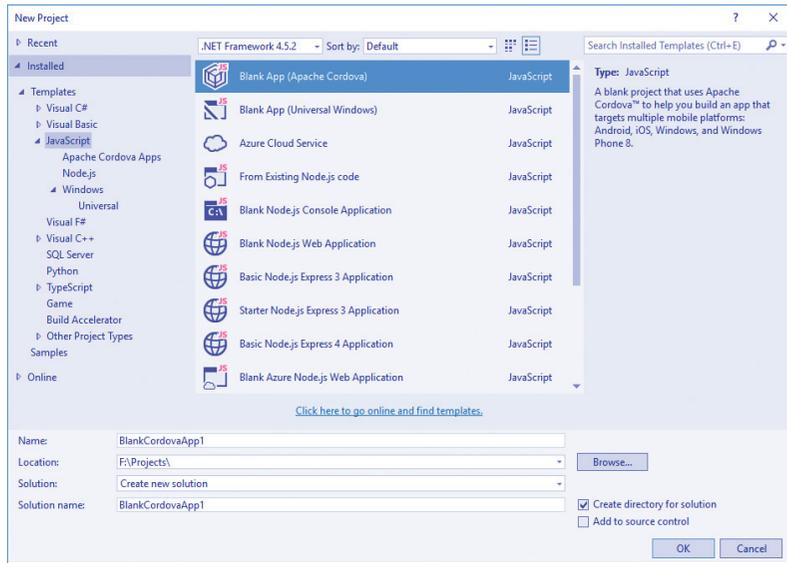
ถ้าต้องการสร้างเว็บไซต์ต่างๆ เช่น เว็บแสดงรูปภาพ, เว็บบอร์ด, เว็บข่าวสาร ฯลฯ แสดงผลในเบราว์เซอร์ เราถือว่าเป็นโปรเจกต์ประเภท Web อยู่ในความรับผิดชอบของภาษา ASP.NET

โปรเจกต์ประเภท JavaScript

เดิมทีเดี่ยวภาษา JavaScript ถูกนำมาใช้พัฒนาด้าน Web Application เพียงอย่างเดียวเท่านั้น แต่ในปัจจุบันภาษา JavaScript ถูกนำมาพัฒนาแอปฯ ได้ทั้งแบบ Web Apps กับ Mobile Apps อีกด้วย

รูปที่ 1-10

แสดงโปรเจกต์ประเภท JavaScript



จากรูปที่ 1-10 ภาษา JavaScript ใน Visual Studio 2015 ในขั้นต้นสามารถสร้างโปรเจกต์ได้ 3 แบบ คือ

- **Apache Cordova Apps** เป็นการใช้นภาษา JavaScript ร่วมกับภาษา HTML5 เพื่อสร้างแอปพที่สามารถรันบนมือถือ โดยอาศัย Cordova (อาจจะเรียกว่า PhoneGap ก็ได้)
- **Node.js** เป็นการพัฒนา Web Apps ที่รันได้ทั้งฝั่ง Server/Client โดยอาศัยภาษา JavaScript เพียงภาษาเดียว
- **Windows – Universal** เป็นการพัฒนาแอปพ ที่รันบน Windows 10 (PC, แท็บเล็ตและมือถือ) โดยอาศัยภาษา HTML5 ร่วมกับภาษา JavaScript

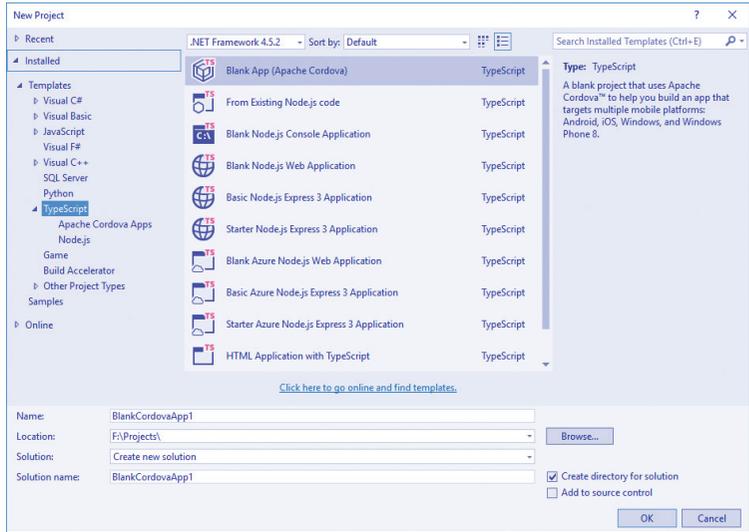
โปรเจกต์ประเภท TypeScript

ภาษา TypeScript เป็นภาษาใหม่ที่ไม่โครซอฟท์สร้างขึ้นมา โดยมีหลักการที่ว่าทุกสิ่งทุกอย่างของภาษา JavaScript คือ ภาษา TypeScript และเพิ่มเติมความสามารถอื่นๆ ในตัวภาษา TypeScript อีกด้วย เช่น การกำหนดประเภทข้อมูล, การสร้างคลาสขึ้นมาใช้งานได้ เป็นต้น ปัจจุบันเป็นภาษาสคริปต์ที่กำลังได้รับความนิยมเป็นอย่างมาก

จากรูปที่ 1-11 ภาษา TypeScript ใน Visual Studio 2015 ในขั้นต้นสามารถสร้างโปรเจกต์ได้ 2 แบบ คือ

- **Apache Cordova Apps** เป็นการใช้นภาษา TypeScript ร่วมกับภาษา HTML5 เพื่อสร้างแอปพที่สามารถรันบนมือถือ โดยอาศัย Cordova (อาจจะเรียกว่า PhoneGap ก็ได้)
- **Node.js** เป็นการพัฒนา Web Apps ที่รันได้ทั้งฝั่ง Server/Client โดยอาศัยภาษา TypeScript เพียงภาษาเดียว

รูปที่ 1-11
แสดงโปรเจกต์ประเภท TypeScript



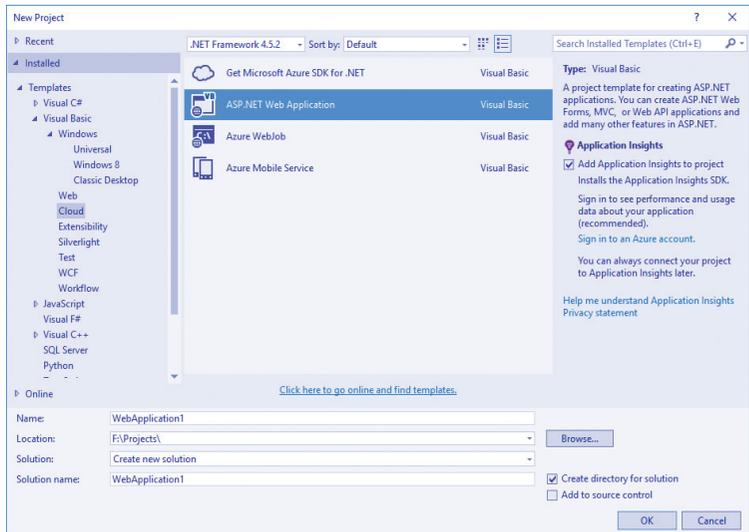
โปรเจกต์ประเภท Cloud

ในปัจจุบันการให้บริการต่างๆ บนก้อนเมฆ (Cloud) ได้รับความนิยมเป็นอย่างสูง และบริการ Cloud ของไมโครซอฟท์ชื่อว่า **Azure** เราสามารถสร้างโปรเจกต์ได้ 2 รูปแบบใหญ่ๆ คือ

- **โปรเจกต์ประเภท ASP.NET Web Application** เป็นการพัฒนา Web Apps ด้วยภาษา ASP.NET เพื่อขอใช้บริการต่างๆ ใน Azure
- **โปรเจกต์ประเภท Azure Mobile Service** เป็นการสร้างบริการที่เชื่อมต่อกับ Azure ให้บริการด้านต่างๆ โดยอาศัยโครงสร้างของ Web API สำหรับอุปกรณ์ประเภท Mobile Devices ต่างๆ

ประเภทโปรเจกต์ที่ผู้เขียนนำเสนอรายละเอียดในขั้นตอนนี้ เป็นการติดตั้งเพิ่มเติมแยกต่างหาก เพื่อแสดงให้เห็นความสามารถของตัวโปรแกรม Visual Studio 2015 ที่มีอยู่มากมาย

รูปที่ 1-12
แสดงโปรเจกต์ประเภท Cloud

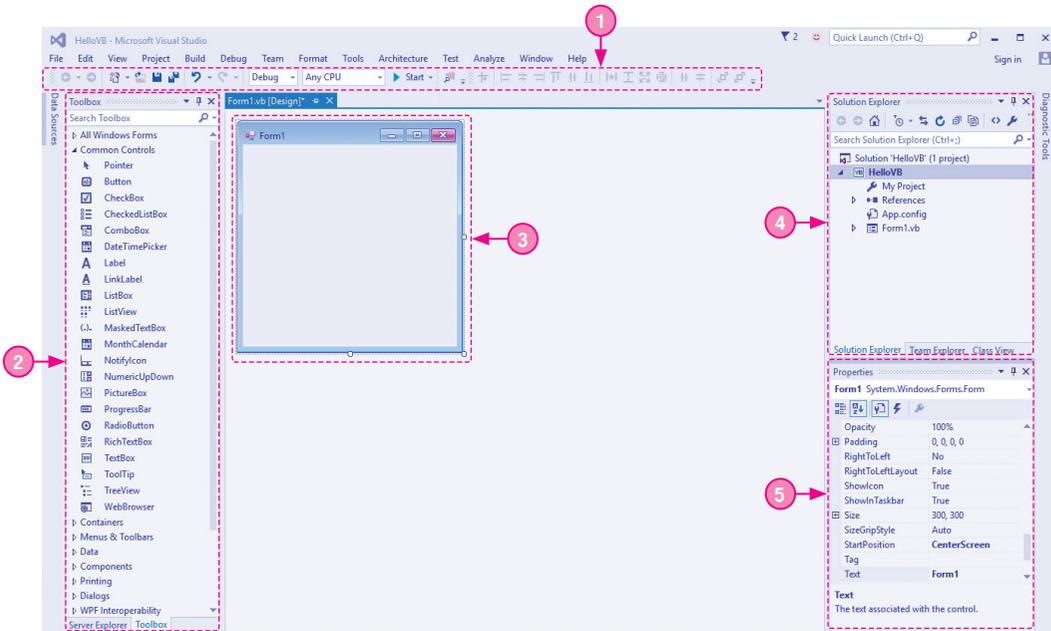


ทำความเข้าใจกับสภาพแวดล้อม (IDE) ของ VS ในขั้นต้น

VS 2015 รองรับการพัฒนาแอปพลิเคชันหลายแพลตฟอร์ม (PC, มือถือ และแท็บเล็ต) ผู้อ่านสามารถเลือกสร้างโปรเจกต์ได้ตามต้องการ และสภาพแวดล้อมของโปรเจกต์แต่ละประเภทจะมีลักษณะคล้ายกัน เพื่อให้ นักพัฒนาคุ้นเคยนั่นเอง

รูปที่ 1-13
แสดงสภาพแวดล้อม IDE ของ VS แบบ Windows Forms Application

ผู้เขียนขอยกตัวอย่างสภาพแวดล้อมของโปรเจกต์ประเภท Windows Forms Application มาอธิบาย ดังรูปที่ 1-13



- **หมายเลข 1** แถบเครื่องมือสำหรับทดสอบโปรเจกต์ปัจจุบัน
- **หมายเลข 2** แถบคอนโทรลสำเร็จรูปทำหน้าที่ออกแบบส่วนแสดงผล (User Interface เรียกย่อๆ ว่า UI) ภายในโปรเจกต์ของผู้อ่าน
- **หมายเลข 3** ส่วนแสดงผลจำลอง ผู้อ่านสามารถออกแบบส่วนแสดงผลให้กับหน้าจอต่างๆ ได้ที่นี่ โดยการลากคอนโทรลต่างๆ มาวางตามที่ต้องการ
- **หมายเลข 4** เรียกว่า **หน้าต่าง Solution Explorer** ทำหน้าที่แสดงโครงสร้างโปรเจกต์ปัจจุบันของผู้อ่านว่า ประกอบไปด้วยไฟล์อะไรบ้าง
- **หมายเลข 5** เรียกว่า **หน้าต่างคุณสมบัติ (Properties)** ทำหน้าที่แสดงคุณสมบัติต่างๆ ของคอนโทรลที่กำลังถูกโฟกัสในส่วนแสดงผลจำลองว่ามีสมบัติอะไรบ้าง สามารถแก้ไขคุณสมบัติต่างๆ ได้ตามที่ต้องการเช่นกัน เช่น ตั้งชื่อคอนโทรลได้ที่คุณสมบัติ Name, กำหนดสีพื้นหลังได้ที่คุณสมบัติ BackColor เป็นต้น

มือใหม่ต้องทราบก่อนเขียนโปรแกรมในโลกของ .NET

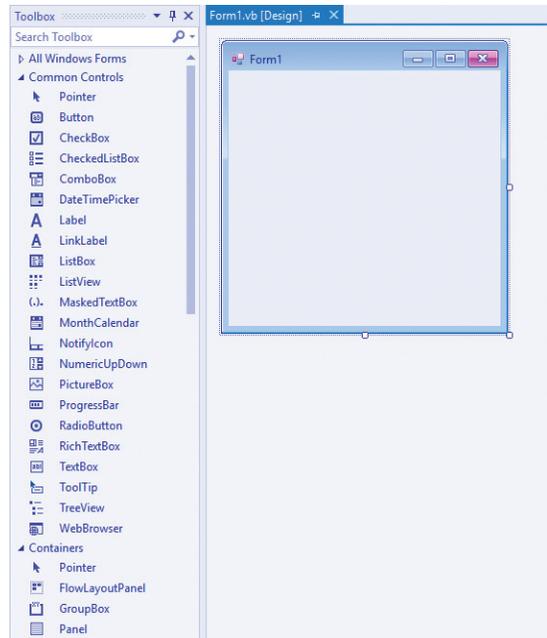
ก่อนที่จะเข้าสู่รายละเอียดของการเขียนโปรแกรมในโลกของ .NET Framework ผู้เขียนจะขอแนะนำเสนอหลักการพื้นฐานในภาพกว้างที่มือใหม่ต้องทราบในขั้นต้นเสียก่อน เพื่อปูพื้นฐานก่อนที่จะเข้าสู่รายละเอียดเนื้อหาส่วนต่างๆ ในแต่ละบทต่อไป

การออกแบบส่วนแสดงผลด้วยคอนโทรล (User Interface)

ไม่ว่าจะสร้างโปรเจกต์ประเภทใดก็ตามใน Visual Studio 2015 ไมโครซอฟท์กำหนดให้สภาพแวดล้อมของตัวโปรแกรม Visual Studio 2015 มีลักษณะใกล้เคียงกันมากที่สุดเท่าที่จะเป็นไปได้ เพื่อให้ นักพัฒนาคุ้นเคยกับตัวโปรแกรมได้ไม่ยาก

แม้ว่าภาพประกอบผู้เขียนจะใช้โปรเจกต์ประเภท Windows Forms Application ก็จริงอยู่ แต่สามารถเทียบเคียงได้กับโปรเจกต์ประเภทอื่นๆ ได้

ก่อนอื่นขอให้ผู้อ่านสร้างโปรเจกต์แบบ Windows Forms Application ว่างๆ ขึ้นมา 1 โปรเจกต์ ดังรูปที่ 1-14



รูปที่ 1-14

แสดงแถบ
Toolbox

จากรูปที่ 1-14 แถบคอนโทรล (Control) อยู่ด้านซ้ายมือ ทำหน้าที่ออกแบบส่วนแสดงผล (User Interface เรียกว่า UI) ในขั้นต้นนี้ผู้อ่านสามารถใช้งานคอนโทรลนี้ได้ 2 วิธี

1. ดับเบิลคลิกที่ตัวคอนโทรลเพื่อกำหนดให้ Visual Studio ใช้งานคอนโทรลดังกล่าว ในหน้าจออกแบบโดยอัตโนมัติ เช่น ผู้เขียนต้องการใช้ปุ่มกด Button1 ดังรูปที่ 1-15

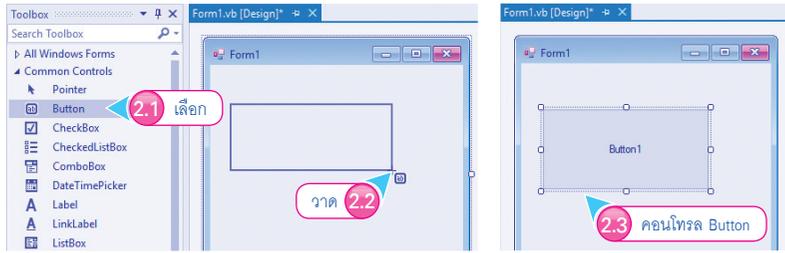
รูปที่ 1-15

แสดงการใช้ปุ่มกด
ด้วยวิธีการ
ดับเบิลคลิกใน
แถบ Toolbox



2. คลิกเลือกคอนโทรลที่ต้องการใช้งานแล้วนำไปวางในส่วนของฟอร์ม (Form) ดังรูปที่ 1-16

รูปที่ 1-16
แสดงการใช้ปุ่มกด
ด้วยวิธีวาด

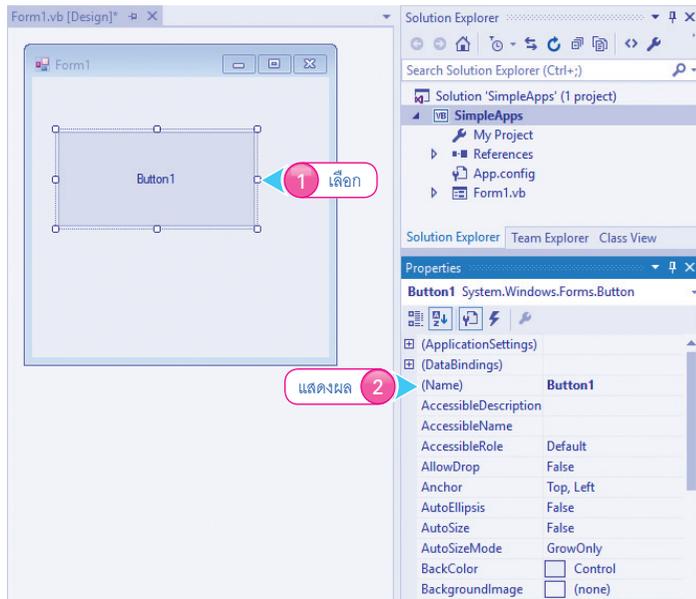


จากรูปที่ 1-16 เราได้ส่วนแสดงผลง่าย ๆ มาแล้ว 1 หน้าจอ โดยมีปุ่มกด Button 1 ปุ่ม

การปรับแต่งคอนโทรลด้วยวิธีแก้ไขคุณสมบัติ (Property)

คอนโทรลต่างๆ ที่นำมาใช้งาน สามารถปรับแต่งหรือแก้ไขได้ในหน้าต่างคุณสมบัติ (Properties) กล่าวคือ เมื่อโฟกัสที่คอนโทรลตัวใดก็ตาม หน้าต่างคุณสมบัติจะเปลี่ยนไปโดยอัตโนมัติ

รูปที่ 1-17
แสดงรายการ
คุณสมบัติของปุ่ม
กด Button



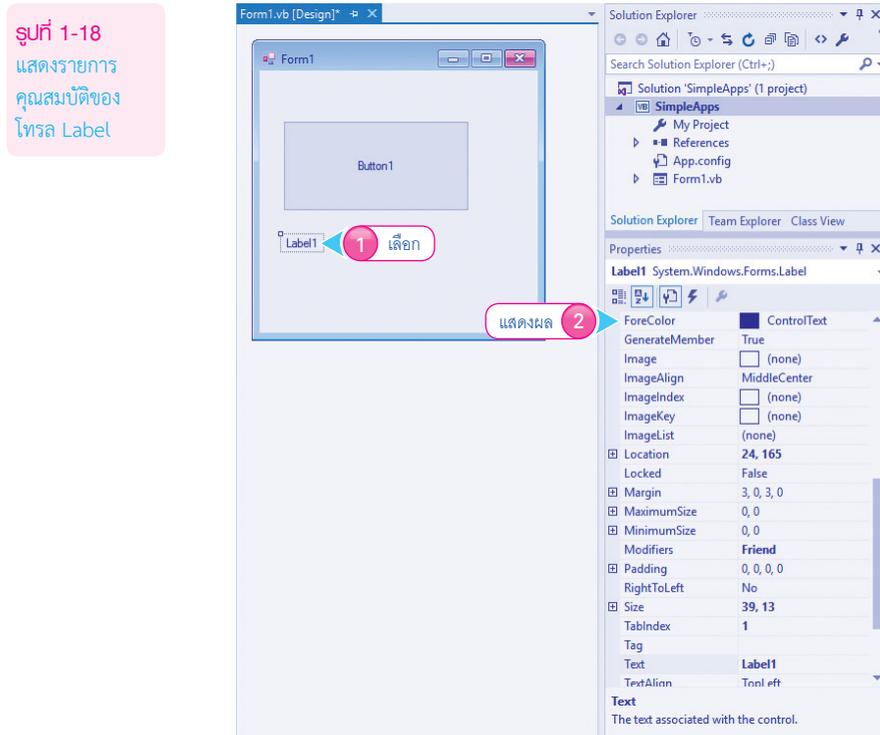
จากรูปที่ 1-17 ส่วนแสดงผลปัจจุบันของผู้เขียนมีเพียง 1 ปุ่ม เมื่อถูกโฟกัสแล้วหน้าต่างคุณสมบัติก็จะแสดงรายการคุณสมบัติต่างๆ ของคอนโทรล Button ให้เราโดยอัตโนมัติ

รายการคุณสมบัติของคอนโทรล Button ที่น่าสนใจ แสดงดังตารางต่อไปนี้

คุณสมบัติ	หน้าที่
Name	ทำหน้าที่ตั้งชื่อให้กับตัวคอนโทรล ใช้สำหรับอ้างอิงตอนเขียนโค้ด
Text	กำหนดข้อความที่จะแสดงในปุ่มกด
BackColor	กำหนดสีพื้นหลังให้กับปุ่มกด
ForeColor	กำหนดสีตัวอักษรที่อยู่ในปุ่มกด

คุณสมบัติใดก็ตามที่ถูกแก้ไขจะแสดงด้วยตัวอักษรตัวหนา ส่วนตัวอักษรปกติ หมายถึง เป็นค่าเริ่มต้นที่โปรแกรม Visual Studio ตั้งไว้ ในการใช้งานคอนโทรลไม่จำเป็นต้องแก้ไขให้ครบทุกคุณสมบัติ โดยเฉพาะเท่าที่ต้องการเปลี่ยนแปลงเท่านั้น

ต่อมา ผู้เขียนลองใช้คอนโทรล Label ซึ่งทำหน้าที่แสดงข้อความ เมื่อคอนโทรล Label ได้รับโฟกัส พบว่าที่หน้าต่างคุณสมบัติก็จะแสดงรายการคุณสมบัติของคอนโทรล Label ให้เราโดยอัตโนมัติ



รายการคุณสมบัติที่น่าสนใจแสดงดังตารางต่อไปนี้

คุณสมบัติ	หน้าที่
Name	ทำหน้าที่ตั้งชื่อให้กับตัวคอนโทรลใช้สำหรับอ้างอิงตอนเขียนโค้ด
Text	กำหนดข้อความที่จะแสดงใน Label
BackColor	กำหนดสีพื้นหลังให้กับคอนโทรล Label
ForeColor	กำหนดสีตัวอักษรที่อยู่ในคอนโทรล Label
AutoSize	กำหนดขนาดของคอนโทรล Label มี 2 แบบ คือ <ul style="list-style-type: none"> • True หมายถึง กำหนดให้ Label มีขนาดเท่าที่ข้อความแสดงอยู่ • False หมายถึง คุณสามารถกำหนดขนาดของ Label ได้อย่างอิสระ

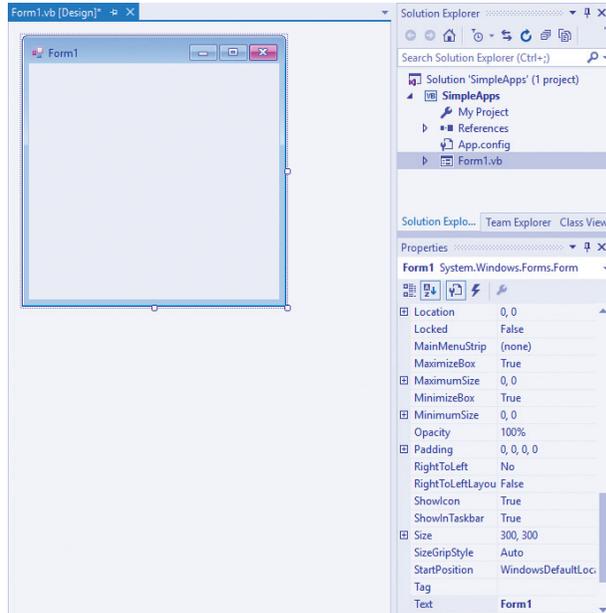
คอนโทรลแต่ละตัวก็จะมีรายการคุณสมบัติทั้งที่เหมือนกันและต่างกันไปตามหน้าที่ของมันเอง

ทำความเข้าใจกับฟอร์ม (Form)

ไม่ว่าจะสร้างโปรเจกต์ประเภทใดก็ตาม โดยส่วนใหญ่ (โปรเจกต์บางประเภทไม่ต้องมีส่วนแสดงผลเริ่มต้น) จะมีส่วนแสดงผลเริ่มต้นมาให้ใช้ออกแบบหน้าจอแรก เช่น ในกรณีนี้เราสร้างโปรเจกต์แบบ Windows Forms Application ก็จะได้ฟอร์ม (Form) ที่มีชื่อว่า Form1 เป็นหน้าจอเริ่มต้น ดังรูปที่ 1-19

รูปที่ 1-19

แสดงรายการคุณสมบัติของฟอร์ม

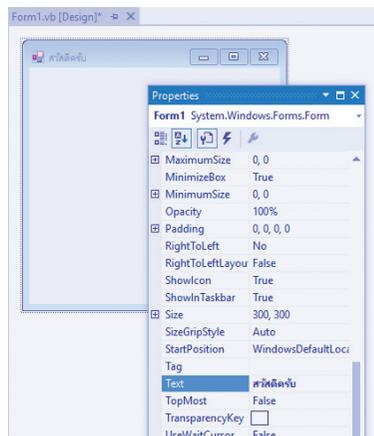


จากรูปที่ 1-19 เมื่อโฟกัสที่ Form1 หน้าต่างคุณสมบัติก็จะแสดงรายการคุณสมบัติของ Form1 เช่นกัน โดยคุณสมบัติที่น่าสนใจก็คือ

คุณสมบัติ	หน้าที่
Name	ทำหน้าที่ตั้งชื่อให้กับฟอร์มใช้สำหรับอ้างอิงตอนเขียนโค้ด
Text	กำหนดข้อความที่จะแสดงในแถบ Title Bar
BackColor	กำหนดสีพื้นหลังให้กับฟอร์ม
ForeColor	กำหนดสีตัวอักษรที่อยู่ในฟอร์ม

รูปที่ 1-20

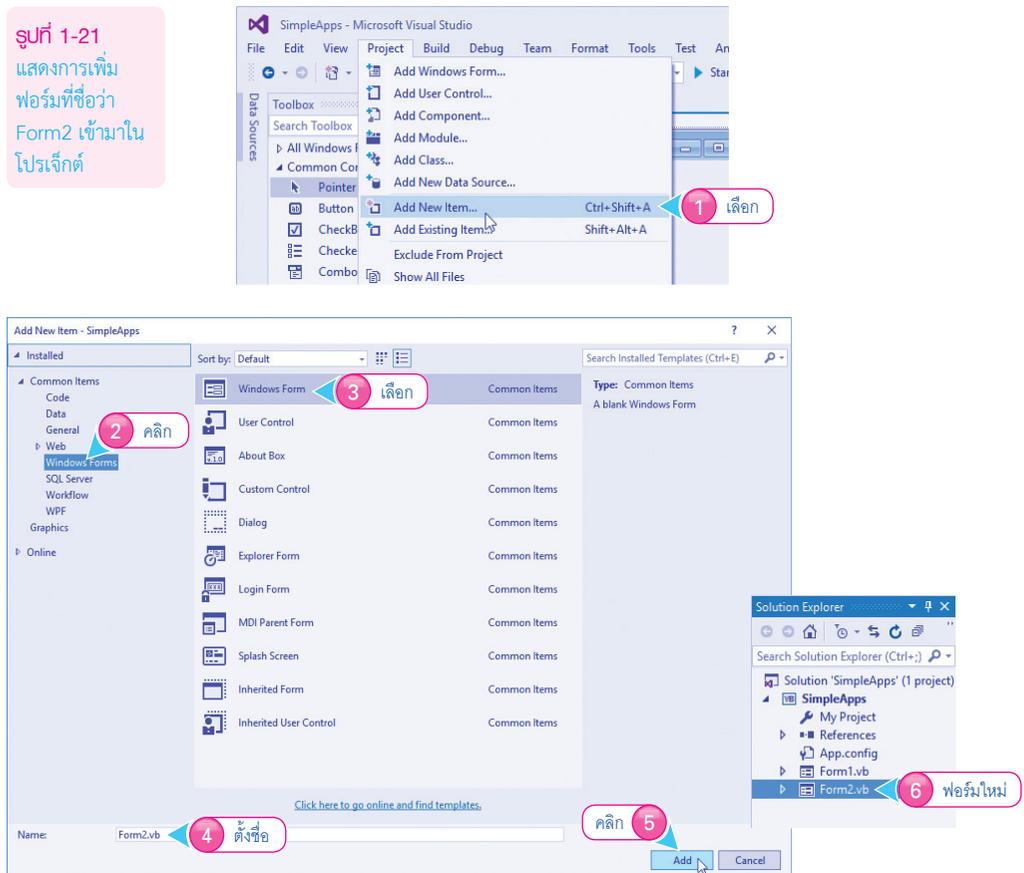
แสดงการแก้ไขคุณสมบัติ Text ของ Form1



การเพิ่มฟอร์มใหม่เข้ามาในโปรเจกต์

ในการพัฒนาแอปฯ ขึ้นมาใช้งาน โดยส่วนใหญ่จะประกอบไปด้วยส่วนแสดงผลตั้งแต่ 2 หน้าจอขึ้นไป วิธีการเพิ่มหน้าจอใหม่เข้ามา ให้ผู้อ่านคลิกเมนู Project > Add New Item... ในกรณีนี้เราต้องการเพิ่มหน้าจอใหม่ประเภท Windows Forms Application โดยตั้งชื่อว่า Form2 ดังรูปที่ 1-21

รูปที่ 1-21
แสดงการเพิ่ม
ฟอร์มที่ชื่อว่า
Form2 เข้ามาใน
โปรเจกต์



จากรูปที่ 1-21 ณ จุดนี้ โปรเจกต์ของเรามี 2 ฟอร์มแล้ว

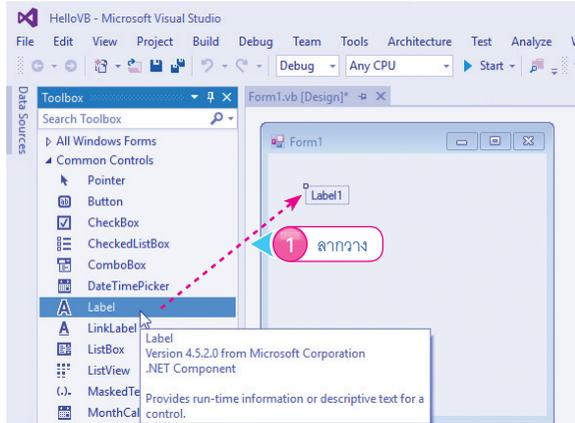
เริ่มต้นสร้างแอปพลิเคชันแรก

ในหัวข้อนี้ผู้อ่านจะได้ใช้งาน Visual Studio 2015 เขียนโปรแกรมด้วยภาษา Visual Basic 2015 เป็นครั้งแรก โดยเป้าหมายของแอปพลิเคชันนี้ก็คือ แสดงข้อความ “Hello VB” โดยมีขั้นตอนดังนี้

1. ให้ผู้อ่านเลือกใช้คอนโทรล Label เข้ามาทำหน้าที่แสดงข้อความ “Hello VB” โดยการลาก & วางใน Form1 ซึ่งทำหน้าที่เป็นส่วนแสดงผล ดังรูปที่ 1-22

รูปที่ 1-22

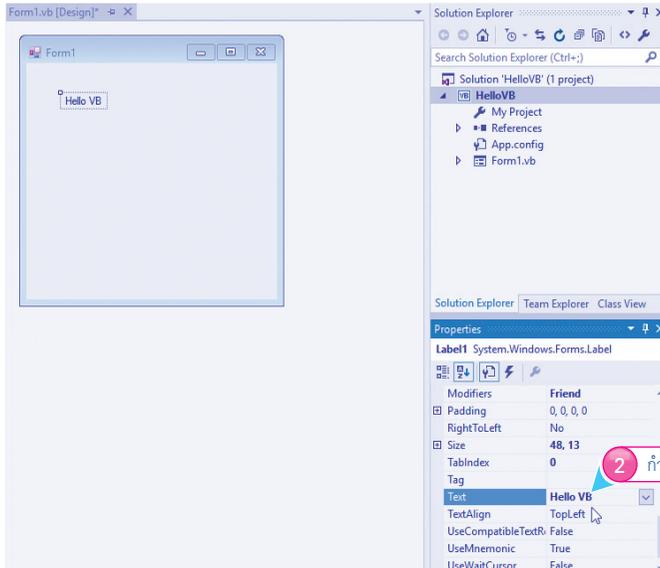
แสดงการใช้งาน
คอนโทรล Label
ใน Form1



- กำหนดข้อความใน Label ตัวนี้ ให้ผู้อ่านกำหนดที่คุณสมบัติ Text ในหน้าต่าง Properties เป็นข้อความตามที่ต้องการ

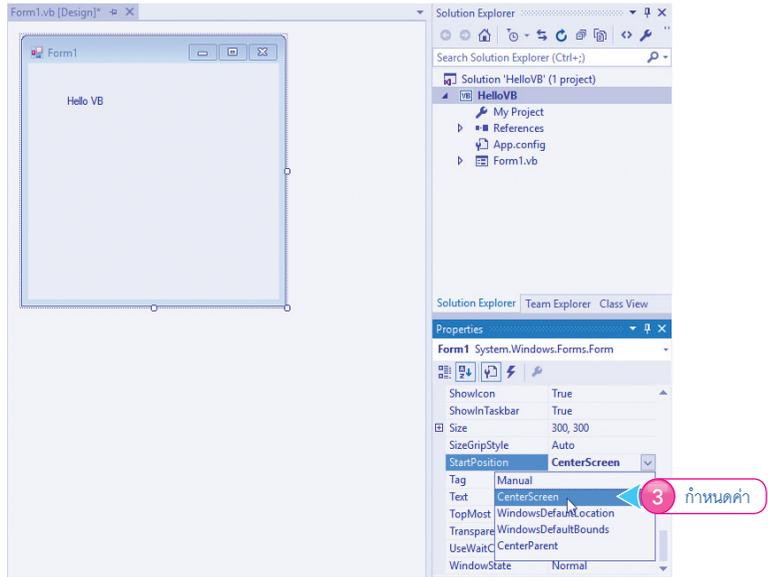
รูปที่ 1-23

แสดงการแก้ไข
คุณสมบัติ Text
ของคอนโทรล
Label ตัวนี้



- กำหนดให้ Form1 ปรากฏอยู่ที่กึ่งกลางหน้าจอ โดยโฟกัสที่ Form1 ก่อน (คลิกเลือกที่ Form1) ในหน้าต่าง Properties ที่คุณสมบัต **StartPosition** ซึ่งทำหน้าที่กำหนดตำแหน่งของ Form1 ให้ผู้เขียนเลือกเป็นแบบ CenterScreen คือ เมื่อโปรแกรมทำงานกำหนดให้ Form1 ปรากฏขึ้นมาที่กึ่งกลางหน้าจอ ดังรูปที่ 1-24

รูปที่ 1-24
แสดงการแก้ไข
ตำแหน่งของ
ฟอร์ม ให้อยู่
กึ่งกลางหน้าจอ

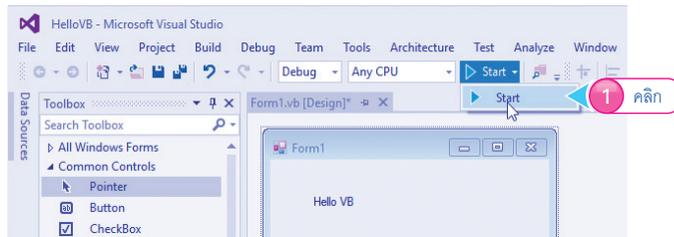


จากรูปที่ 1-24 ณ จุดนี้ เราได้โปรเจกต์ตามที่เราต้องการแล้ว

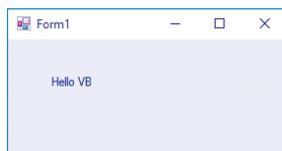
การทดสอบโปรเจกต์

หลังจากที่ผู้อ่านออกแบบและเขียนโค้ดแล้ว ก็จะเข้าสู่ขั้นตอนการทดสอบโปรเจกต์เพื่อดูแอปพลิเคชันที่สร้างขึ้นมีผลการทำงานเป็นไปตามที่เราต้องการหรือไม่ โดยการคลิกปุ่ม **Start** เพื่อเริ่มต้นทดสอบรันโปรเจกต์

รูปที่ 1-25
แสดงการเริ่ม
ทดสอบโปรเจกต์



รูปที่ 1-26
ผลการทำงานของ
โปรเจกต์ปัจจุบัน



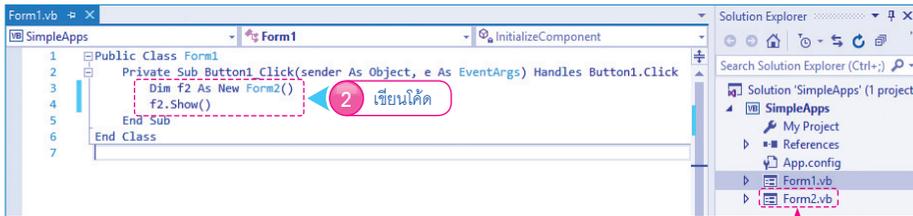
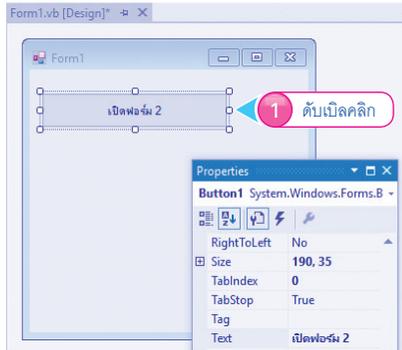
จากรูปที่ 1-26 เห็นได้ว่า เราได้ข้อความ “Hello VB” ปรากฏขึ้นมาใน Form1 ตามที่เราต้องการแล้ว

วิธีการสั่งให้เปิดฟอร์มใหม่โดยการเขียนโค้ด

สำหรับวิธีการสั่งให้เปิดฟอร์มใหม่ที่เราเพิ่มเข้ามามีขั้นตอนดังนี้

1. ดับเบิลคลิกปุ่ม Button1 เพื่อสร้างเหตุการณ์ Click()
2. เขียนโค้ด ดังรูปที่ 1-27

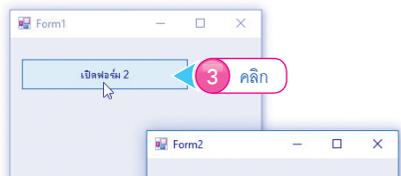
รูปที่ 1-27
แสดงโค้ดที่ใช้
สำหรับเปิด
Form2



ฟอร์มใหม่ถูกเพิ่มเข้ามาโดยการคลิกเมนู Project > Add Windows Forms...

3. ทดสอบรันโปรแกรม พบว่า เราสามารถเปิด Form2 ได้แล้ว ดังรูปที่ 1-28

รูปที่ 1-28
แสดงโค้ดที่ใช้
สำหรับเปิด
Form2



หลักการทำงานของภาษา VB

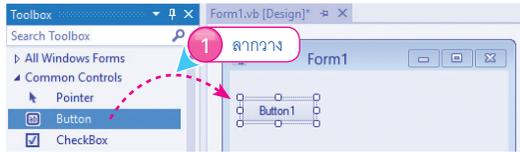
หลักการทำงานของภาษา VB เรียกว่า **การเขียนโปรแกรมตอบสนองเหตุการณ์ที่เกิดขึ้น (Event Driven Programming)** กล่าวคือ ผู้อ่านเป็นผู้กำหนดเองว่า เมื่อเกิดเหตุการณ์ Click() ให้ทำอะไร, เมื่อเกิดเหตุการณ์ Load() ให้ทำอะไร เป็นต้น

หมายความว่า แท้ที่จริงแล้วการกระทำต่างๆ ที่ผู้ใช้งานกระทำต่อโปรแกรมของผู้อ่าน ผู้อ่านเป็นผู้กำหนดขึ้นมาเองทั้งสิ้นว่า “ให้ทำอะไร” ประเด็นที่ตามมา ก็คือ วิธีการการสร้างเหตุการณ์ให้กับคอนโทรลแต่ละตัวอย่างไร

สมมติว่า ผู้เขียนต้องการสร้างเหตุการณ์ Click() ให้กับปุ่มกด Button จะมีวิธีการดังนี้

1. ให้ผู้อ่านลากปุ่มกด Button มาวางใน Form1 ดังรูปที่ 1-29

รูปที่ 1-29
แสดงการใช้งาน
คอนโทรล Button
ใน Form1



โดยการสร้างเหตุการณ์ให้กับปุ่มกด Button1 นี้ มี 2 วิธี ดังข้อที่ 2 กับ 3

2. ดับเบิลคลิกที่คอนโทรล Button1 เพื่อสั่งให้ VS สร้างเหตุการณ์ประจำตัวของคอนโทรล Button ขึ้นมา พบว่า เหตุการณ์ประจำตัวของคอนโทรล Button ก็คือ เหตุการณ์ Click() เพราะคอนโทรล Button ถูกสร้างขึ้นมาเพื่อทำหน้าที่รองรับการคลิกจากผู้ใช้งาน

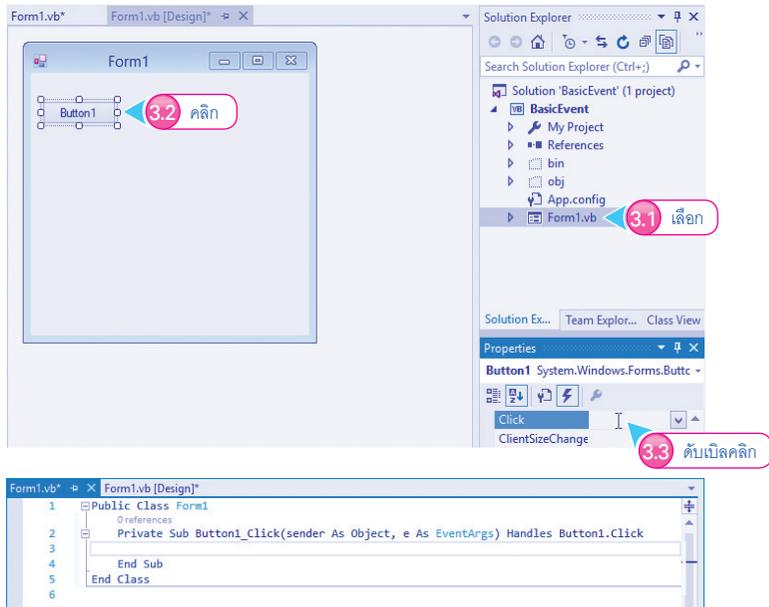
รูปที่ 1-30
แสดงการสร้าง
เหตุการณ์ Click()
ด้วยวิธีดับเบิลคลิก



แต่ละคอนโทรลมีเหตุการณ์ประจำตัวแตกต่างกัน ขึ้นอยู่กับว่าคอนโทรลตัวนั้นๆ ถูกสร้างขึ้นมาเพื่อ “ทำหน้าที่อะไร”

3. อีกวิธีหนึ่ง ก็คือ ให้ผู้อ่านโฟกัสที่คอนโทรล Button1 ก่อน จากนั้นคลิกปุ่ม  ในหน้าต่าง Properties เพื่อเลือกสร้างเหตุการณ์ที่เราต้องการ จะเห็นได้ว่า ปุ่มกด Button สามารถรองรับได้หลายเหตุการณ์ ให้ผู้อ่านดับเบิลคลิกที่ Click เพื่อสร้างเหตุการณ์ Click() ขึ้นมาสำหรับปุ่มกด Button1 ปุ่มนี้

รูปที่ 1-31
แสดงเหตุการณ์
Click() ที่ได้จาก
วิธีเลือกสร้าง
เหตุการณ์



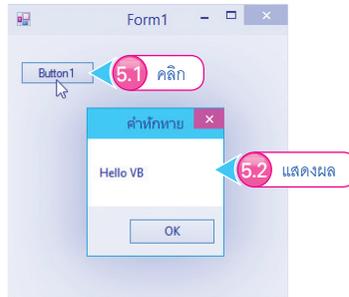
- ผู้เขียนลองตั้งโจทย์ง่ายๆ ว่า ต้องการแสดงข้อความ “Hello VB” เมื่อผู้ใช้งานคลิกปุ่ม Button1 ซึ่งสามารถทำได้โดยการเขียนโค้ดดังต่อไปนี้

```

Form1.vb
Public Class Form1
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        MessageBox.Show("Hello VB", "คำทักทาย")
    End Sub
End Class
    
```

- จากโค้ดข้างต้นเป็นการสั่งให้แสดงข้อความ “Hello VB” โดยอาศัยคลาส MessageBox เข้ามาทำหน้าที่สร้างไอคอนแสดงข้อความ เมื่อรันโปรแกรมและมีการคลิกปุ่ม Button1 ดังรูปที่ 1-32

รูปที่ 1-32
แสดงข้อความที่ได้



- ในกรณีที่ผู้อ่านดับเบิลคลิกบริเวณพื้นที่ว่างๆ ของ Form 1 ส่งผลให้ VB สร้างเหตุการณ์ Load() ขึ้นมาเพื่อให้กำหนดว่า เมื่อ Form1 ถูกโหลดขึ้นมาแล้วให้ทำอะไร เพราะเหตุการณ์ประจำตัวของ Form ก็คือ เหตุการณ์ Load()

รูปที่ 1-33
แสดงที่จัดเก็บโค้ด
ภาษา VB



รูปที่ 1-33 ที่หน้าต่าง Solution Explorer ทำหน้าที่แสดงโครงสร้างของโปรเจกต์ปัจจุบัน ให้ผู้อ่านคลิกที่รายการ Form1 (.vb) เพื่อแสดงโค้ดทั้งหมดที่อยู่ใน Form 1

สรุปท้ายบท

เนื้อหาในบทนี้เป็นเพียงขั้นตอนการเตรียมและทดสอบสภาพแวดล้อมให้เครื่องของผู้อ่านพร้อมใช้งานเท่านั้น เนื้อหาในบทต่อไปจะเข้าสู่โลกของการเขียนโปรแกรมอย่างแท้จริง

พื้นฐานการเขียนโปรแกรมด้วยภาษา Visual Basic

เนื้อหาในบทนี้จะเข้าสู่โลกของการเขียนโปรแกรมกับภาษา VB 2015 ด้วย Visual Studio 2015 ผู้อ่านจะได้ศึกษาพื้นฐานและไวยากรณ์ทางภาษาที่ต้องทราบเป็นลำดับแรกก่อนที่จะเข้าสู่หัวข้อต่อไป

คำสั่ง Option Explicit On กับ Option Strict On

ทุกครั้งที่ผู้อ่านเริ่มเขียนโค้ดของภาษา VB ควรกำหนดให้ตัว Editor ของ VS ใช้คำสั่ง Option Explicit On และคำสั่ง Option Strict On บริเวณตำแหน่งบนสุดเสมอ ดังรูปที่ 2-1

รูปที่ 2-1

แสดงตำแหน่งการ
เขียนคำสั่งทั้ง 2

```

1 Option Explicit On
2 Option Strict On
3
4 Public Class Form1
5     Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
6
7     End Sub
8 End Class
9

```

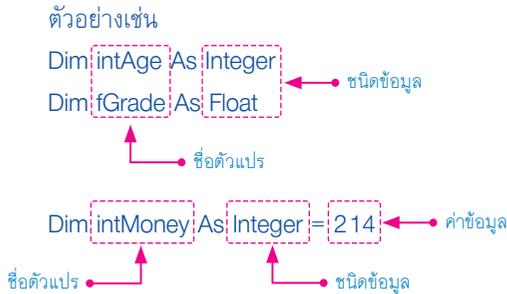
โดยความหมายของคำสั่งทั้ง 2 ข้างต้น มีดังนี้

- **คำสั่ง Option Explicit On** หมายถึง ตัวแปรทุกตัวที่ใช้งานต้องมีการประกาศใช้งานด้วยคำสั่ง Dim ก่อนเสมอ
- **คำสั่ง Option Strict On** หมายถึง กำหนดให้ตัวแปลภาษาของ VB มีความเข้มงวดและเคร่งครัดไวยากรณ์ในภาษา VB เทียบเท่ากับภาษา VC# เพื่อให้ได้โค้ดที่มีคุณภาพนั่นเอง

การประกาศตัวแปรสำหรับใช้เก็บข้อมูล

ไม่ว่าผู้อ่านจะเริ่มต้นศึกษาภาษาใดก็ตาม ไวยากรณ์ที่ต้องทำความเข้าใจเป็นลำดับแรก คือ การประกาศตัวแปร (Variable) สำหรับทำหน้าที่เก็บข้อมูล โดยการประกาศตัวแปรในภาษา VB จะใช้คำสั่ง **Dim** ซึ่งมีไวยากรณ์ดังต่อไปนี้

VB 2015
Dim ชื่อตัวแปร As ชนิดข้อมูล [= ค่าข้อมูล]



กฎการตั้งชื่อตัวแปร

ในภาษา VB 2015 การตั้งชื่อตัวแปรต่างๆ ขึ้นใช้งาน มีกฎข้อบังคับดังนี้

- ต้องประกอบด้วยอักขระ A-Z, a-z, 0-9 หรือ _ เท่านั้น
- อักขระตัวแรกต้องเป็นตัว A-Z, a-z หรือ _ อย่างไม่อย่างหนึ่งเท่านั้น (เป็นตัวเลขไม่ได้)
- มีความยาวไม่เกิน 1,023 ตัว
- ต้องไม่ซ้ำกับ Reserved Words ของ VB

Reserved Words

ในภาษา VB 2015 มี Reserved Words ขึ้นต้นที่กำหนดให้ห้ามใช้เป็นชื่อตัวแปรต่างๆ ดังนี้

AddHandler	AddressOf	Alias	And
AndAlso	As	Boolean	ByRef
Byte	ByVal	Call	Case
Catch	CBool	CByte	CChar
CDate	CDbl	CDec	Char
CInt	Class		CLng
CObj	Const	Continue	CSByte
CShort	CSng	CStr	CType
CUInt	CULng	CUShort	Date
Decimal	Declare	Default	Delegate

Dim	DirectCast	Do	Double
Each	Else	Elseif	End
	EndIf	Enum	Erase
Error	Event	Exit	
Finally	For		Friend
Function	Get	GetType	
Global	GoSub	GoTo	Handles
If		Implements	
Imports		In	
Inherits	Integer	Interface	Is
IsNot	Let	Lib	Like
Long	Loop	Me	Mod
Module		MustInherit	MustOverride
MyBase		Namespace	Narrowing
New		Next	
Not	Nothing	NotInheritable	NotOverridable
Object	Of	On	Operator
Option	Optional	Or	OrElse
Out	Overloads	Overridable	Overrides
ParamArray	Partial	Private	Property
Protected	Public	RaiseEvent	ReadOnly
ReDim	REM	RemoveHandler	Resume
Return	SByte	Select	Set
Shadows	Shared	Short	Single
Static	Step	Stop	String
Structure		Sub	SyncLock
Then	Throw	To	TRUE
Try	TryCast		UInteger
ULong	UShort	Using	Variant
Wend	When	While	Widening
With	WithEvents	WriteOnly	Xor
#Const	#Else	#Elseif	#End

ขอบเขตของตัวแปร (Variable Scope)

หลังจากที่ผู้อ่านประกาศตัวแปรที่ต้องการใช้งานแล้ว เนื้อหาลำดับถัดมาที่ควรรับทราบก็คือ ตัวแปรดังกล่าวมีขอบเขตที่สามารถเรียกใช้งานได้แค่ไหน ซึ่งมีขอบเขตอยู่ 3 ระดับ คือ

1. **ระดับฟอร์ม** ถ้าผู้อ่านประกาศตัวแปรชนิดนี้ไว้ในฟอร์มจะมีขอบเขตกว้างมากที่สุด กล่าวคือ ทุกเหตุการณ์ที่อยู่ในฟอร์มนั้นๆ จะสามารถเรียกใช้งานได้
2. **ระดับ Local** หรือเรียกอีกอย่างว่า **ระดับ Procedure** มีขอบเขตขนาดกลาง เป็นระดับที่เหมาะสมกับการใช้งานมากที่สุด คือ มีขอบเขตอยู่ในเหตุการณ์ (Event) แต่ละเหตุการณ์
3. **ระดับ Block** มีขอบเขตขนาดเล็กที่สุด มักที่จะใช้เก็บค่าข้อมูลแค่ชั่วคราว หรือใช้ในวนลูป (คำสั่ง For, While) เสียเป็นส่วนใหญ่ เช่น ตัวแปรที่อยู่ในบล็อกของคำสั่งต่างๆ เป็นต้น

สำหรับการเลือกใช้ตัวแปรระดับใดก็ตาม ไม่มีข้อกำหนดหรือข้อบังคับใดๆ ทั้งสิ้น ขึ้นอยู่กับลักษณะของโปรเจกต์และความเหมาะสม

NOTE



คำว่า เหมาะสม หมายถึง ในการเรียกใช้งานตัวแปรทั้งหมดที่อยู่ในโปรเจกต์ของผู้อ่าน สิ่งหนึ่งที่ต้องคำนึงถึงก็คือ หน่วยความจำ (RAM) การเขียนโปรแกรมที่ดีจะต้องมีการใช้งานทรัพยากรของระบบได้เหมาะสมกับความสามารถของตัวเอง

ขอบเขตของตัวแปรจะเป็นตัวบ่งบอกว่า ตัวแปรนั้นๆ จะมีอายุในการทำงานนานเท่าไร เพราะเมื่อสิ้นสุดการทำงานของตัวแปรนั้นๆ แอปพลิเคชันจะคืนทรัพยากรที่จองมาใช้งานกลับสู่ระบบ ถ้าผู้อ่านเลือกใช้ขอบเขตและประเภทของตัวแปรได้อย่างเหมาะสมแล้ว โปรเจกต์ที่ได้ก็จะมีทั้งประสิทธิภาพในด้านการทำงานและการใช้ทรัพยากรระบบ ตัวอย่างเช่น

ผู้เขียนประกาศตัวแปรขึ้นมา 2 ตัว คือ ตัวแปร num กำหนดให้มีชนิดข้อมูลเป็นตัวเลขจำนวนเต็ม Integer และกำหนดค่าเริ่มต้นให้เก็บค่า 1000 ส่วนตัวแปร str กำหนดให้มีชนิดข้อมูลเป็นข้อความ String และกำหนดค่าเริ่มต้นเป็นชื่อ-สกุลผู้เขียน

```

VB 2015
Option Explicit On
Option Strict On

Public Class Form1
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Dim num As Integer = 1000
        Dim str As String = "Suphachai Somphanit"
    End Sub
End Class
    
```

ตัวแปรทั้ง 2 ตัวถูกประกาศให้อยู่ภายในเหตุการณ์ Form1_Load() เราเรียกตัวแปรประเภทนี้ว่า **ตัวแปรที่มีขอบเขตแบบ Local** หมายถึง สามารถใช้งานได้เฉพาะ “ภายในเหตุการณ์ Form1_Load()” เท่านั้น ต่อมาผู้เขียนลองประกาศตัวแปรอีกลักษณะหนึ่ง โดยตำแหน่งที่ประกาศตัวแปรอยู่นอกเหตุการณ์ ต่าง ๆ กล่าวคือ สร้างตัวแปรที่ชื่อว่า str ขึ้นมา และกำหนดให้มีชนิดข้อมูลเป็นข้อความ String

```
VB 2015
Option Explicit On
Option Strict On

Public Class Form1
    Dim str As String = ""

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        str = "Visual Basic"
    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        str = "Visual Basic"
    End Sub
End Class
```

ในกรณีนี้ตัวแปร str ถูกประกาศให้อยู่นอกเหตุการณ์ต่างๆ เรียกว่า ตัวแปรที่มีขอบเขตระดับฟอร์ม ผู้อ่านสามารถเรียกใช้ตัวแปร str นี้ในเหตุการณ์ต่างๆ ได้ ในกรณีนี้ คือ เหตุการณ์ Form1_Load() กับเหตุการณ์ Button1_Click()

ตัวแปรระดับ Local และระดับฟอร์ม

เพื่อเพิ่มความเข้าใจในการใช้งานตัวแปรระดับ Local และระดับฟอร์ม ให้ผู้อ่านศึกษาการใช้งานตัวแปร จากตัวอย่างต่อไปนี้

1. ออกแบบฟอร์มดังรูป 2-2
2. เขียนโค้ดในเหตุการณ์ Form1_Load() ดังนี้

```
ตัวอย่างที่ 2-1 ตัวแปรระดับ Local และระดับฟอร์ม (Form1.vb)
Option Explicit On
Option Strict On
Public Class Form1
    Dim str As String = "ตัวแปรระดับฟอร์ม"

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Dim str As String = "ตัวแปรระดับ Local"
        Label1.Text = str
    End Sub
End Class
```

3. เขียนโค้ดในเหตุการณ์ Button1_Click() ดังนี้

ตัวอย่างที่ 2-1 ตัวแปรระดับ Local และระดับฟอร์ม (Form1.vb)

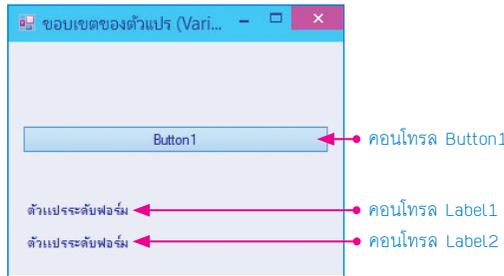
```
Option Explicit On
Option Strict On
Public Class Form1
    Dim str As String = "ตัวแปรระดับฟอร์ม"

    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        'Dim str As String = "ตัวแปรระดับ Local"
        Label1.Text = str
    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Label2.Text = str
    End Sub
End Class
```

4. เมื่อรันโปรแกรมจะได้ผลลัพธ์ดังนี้

รูปที่ 2-2 ผลการรันตัวอย่างที่ 2-1



จากรูปที่ 2-2 ผู้เขียนประกาศตัวแปรในระดับฟอร์มที่ชื่อว่า str เห็นได้ว่า คอนโทรล Label1 ถูกกำหนดค่าในเหตุการณ์ Form1_Load() ส่วนคอนโทรล Label2 ถูกกำหนดค่าในเหตุการณ์ Button1_Click() เห็นได้ว่าการประกาศตัวแปรในลักษณะนี้ ผู้อ่านสามารถใช้งานตัวแปรตัวนี้ได้หลายเหตุการณ์

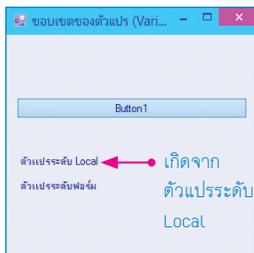
5. ให้ผู้อ่านนำคอมเมนต์โค้ดบรรทัดต่อไปนี้ออก แล้วลองรันตัวอย่างอีกครั้ง ดังรูปที่ 2-3

VB 2015

```
Dim str As String = "ตัวแปรระดับ Local"
```

6. เมื่อรันโปรแกรมจะได้ผลลัพธ์ดังนี้

รูปที่ 2-3 หลังจากเอาหมายเหตุออก



จากรูปที่ 2-3 ผู้เขียนสร้างตัวแปรอีก 1 ตัว มีขอบเขตระดับ Local จงใจให้มีชื่อเดียวกับตัวแปรระดับฟอร์มข้างต้น พบว่า คอนโทรล Label1 แสดงข้อความของตัวแปรในระดับ Local

สรุปได้ว่า ถ้ามีตัวแปรชื่อเดียวกันทั้งในระดับฟอร์มกับระดับ Local ตัวแปรที่อยู่ในระดับ Local มีความสำคัญมากกว่าตัวแปรที่อยู่ในระดับฟอร์มนั่นเอง

ตัวแปรระดับ Block

เพื่อเพิ่มความเข้าใจในการใช้งานตัวแปรระดับ Block ซึ่งเป็นตัวแปรที่มีขอบเขตแคบที่สุด ให้ผู้อ่านศึกษาการใช้งานตัวแปรจากตัวอย่างต่อไปนี้

1. ออกแบบฟอร์มดังรูป 2-4
2. เขียนโค้ดในเหตุการณ์ Form1_Load() ดังนี้

ตัวอย่างที่ 2-2 ตัวแปรระดับ Block (Form1.vb)

```
Option Explicit On
Option Strict On

Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Dim i As Integer
        For i = 0 To 2
            Dim strVal As String = "ตัวแปรระดับ Block"
            Label1.Text = strVal
        Next
        "Label1.Text = strVal"
    End Sub
End Class
```

Diagram annotations in the code block:
- "ชนิดข้อมูล" points to `Dim i As Integer`
- "ค่าข้อมูล" points to `strVal = "ตัวแปรระดับ Block"`
- "ตัวแปรระดับ Block" points to the variable `strVal`
- "ทำหน้าที่เหตุเอาไว้ด้วยเครื่องหมาย " points to `"Label1.Text = strVal"`

3. เมื่อรันโปรแกรมจะได้ผลลัพธ์ดังนี้

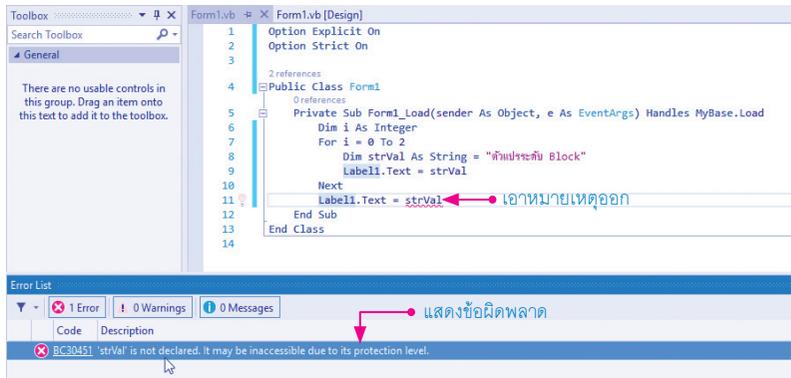
รูปที่ 2-4
ผลการรันตัวอย่างที่ 2-2



จากรูปที่ 2-4 ผู้เขียนประกาศตัวแปรที่ชื่อว่า `strVal` ใหม่ลึกลับคำสั่งของลูป `For` ส่งผลให้ตัวแปรตัวนี้มีขอบเขตการใช้งานเฉพาะภายในบล็อกของลูป `For Next` นี้เท่านั้น

4. ให้ผู้อ่านเอาคอมเมนต์ออก พบว่า ไม่สามารถอ่านค่าของตัวแปร `strVal` ตัวนี้นอกลูป `For` ได้เลย ดังรูปที่ 2-5

รูปที่ 2-5
แสดงการใช้งานตัวแปร `strVal` นอกบล็อกคำสั่ง



จากรูปที่ 2-5 ข้อผิดพลาดที่ปรากฏขึ้นมา เกิดจากตัวแปร `strVal` ยังไม่มีการประกาศใช้งาน เพราะว่าเราเรียกใช้งานตัวแปรตัวนี้เกินขอบเขตของมันนั่นเอง