

บทที่ 1

พื้นฐานของโปรแกรม Visual Studio 2010

Basic of Visual Studio .NET 2010

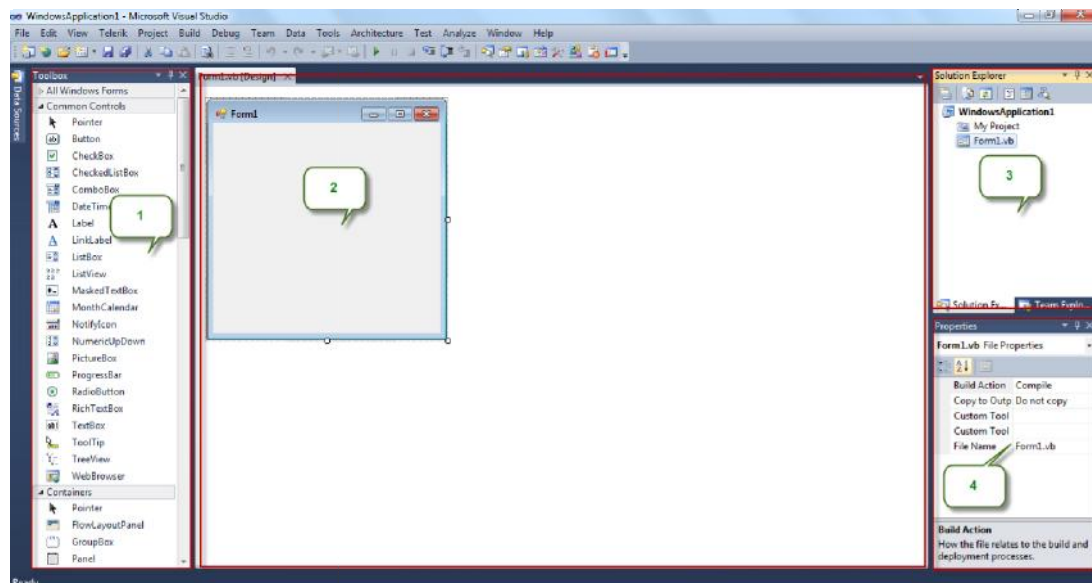
Microsoft Visual Studio .NET 2010 เป็นเครื่องมือสำหรับการพัฒนาโปรแกรมและเว็บไซต์ที่มีประสิทธิภาพมาก มีเครื่องมืออำนวยความสะดวกมากมายรวมทั้งสถาปัตยกรรมที่ทาง Microsoft ได้ออกแบบมานั้นถือว่า เพื่อความมีเสถียรภาพและความยืดหยุ่นสูงมาก

ใน Visual Studio .NET 2010 ประกอบด้วยเครื่องมือต่างๆ ดังนี้

- Visual Basic .NET
- Visual C++ .NET
- Visual VB .NET
- Visual J# .NET
- ASP .NET

ทั้งหมดเป็นเครื่องมือการพัฒนาภายใต้สถาปัตยกรรม .NET Framework ซึ่งก็คือกรอบการทำงานของ การเขียนโปรแกรมที่ Microsoft คิดขึ้น เพื่อรองรับการติดต่อสื่อสาร เพื่อแลกเปลี่ยนข้อมูลระหว่างกัน หรือแลกเปลี่ยนข้อมูลระหว่าง Platform ให้มีความสมบูรณ์ยิ่งขึ้นโดยอาศัยภาษา XML (Extensible Markup Language) ที่ทำหน้าที่เป็นตัวกลางในการแลกเปลี่ยนข้อมูลระหว่าง Platform ที่ต่างกันได้

สภาพแวดล้อมในการทำงานของ Visual Studio 2010

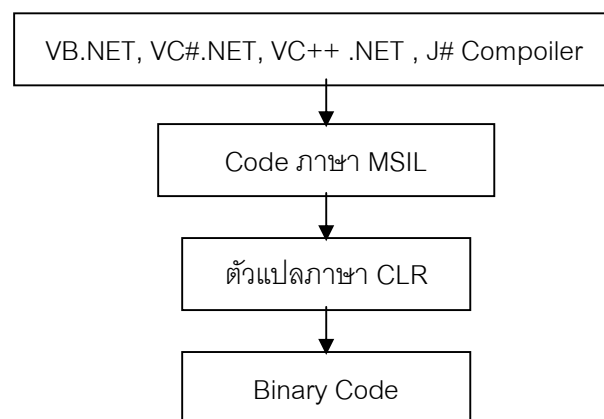


ส่วนสภาพแวดล้อมจะแบ่งออกเป็นส่วนหลักๆ 4 ส่วนดังนี้

- ส่วนที่ 1 ToolBox สำหรับแสดงคอนโทรลต่างๆ ที่ลากมาวางในตัว Document Window ได้ และ Server Explorer สำหรับแสดงบริการต่างๆ ที่มีบนเซิร์ฟเวอร์
- ส่วนที่ 2 Document Window เป็นส่วนหลักในการออกแบบและส่วนแสดงโค้ด
- ส่วนที่ 3 Solution Explorer แสดงไฟล์และโฟลเดอร์ต่างๆ ที่มีในโปรเจค
- ส่วนที่ 4 Properties window เป็นหน้าต่างสำหรับบรรจุคุณสมบัติของ Control ต่างๆ ที่เราใช้ออกแบบโปรแกรม

Getting Start using VB.NET 2010

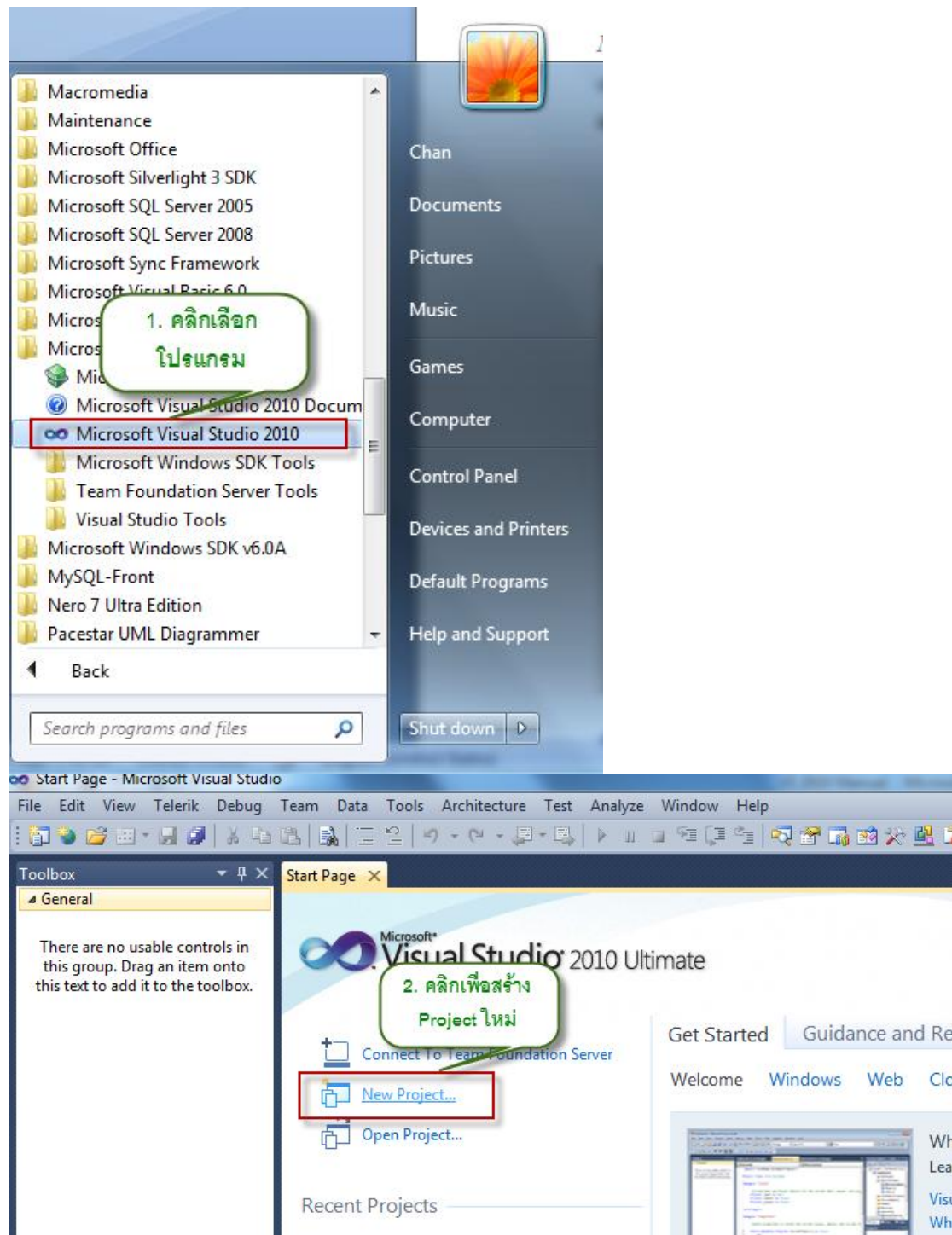
ภาษา VB เป็นภาษาที่ถูกออกแบบมาเพื่อรองรับการทำงานในยุค .NET โดยมีแนวคิดของภาษาที่เป็นแบบการเขียนโปรแกรมเชิงวัตถุสมัยใหม่ ซึ่งทุกภาษาที่อยู่ภายใต้เทคโนโลยี .NET นั้นจะใช้ตัวแปลภาษาตัวเดียวกันคือ Common Language Runtime ดังภาพ

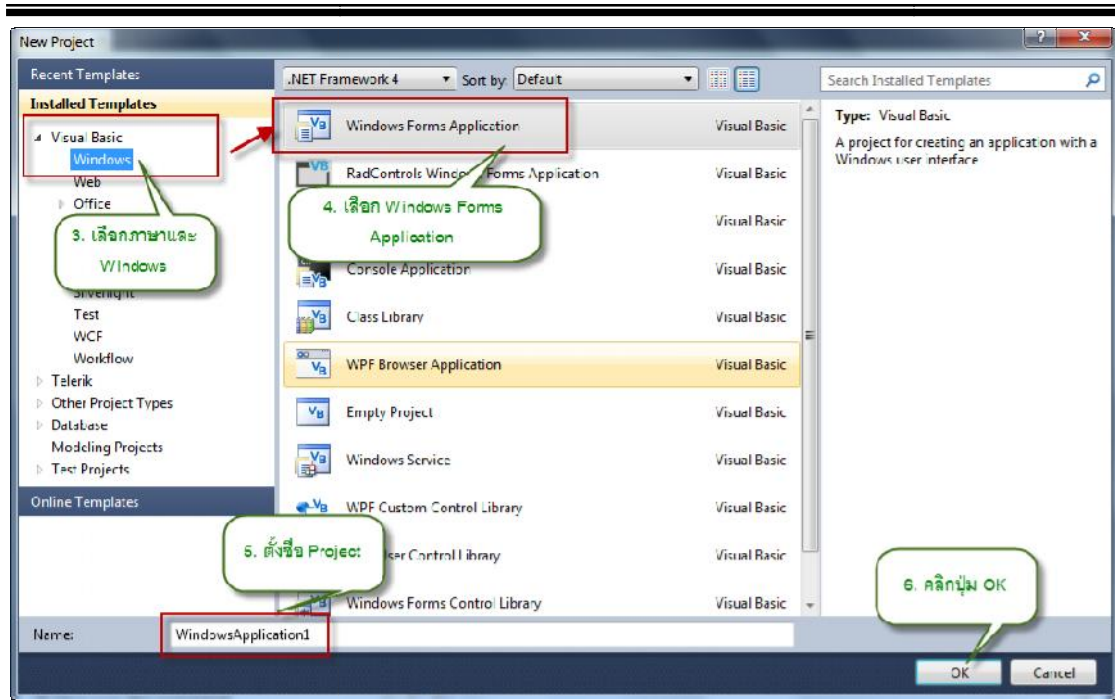


จากรูป จะเห็นได้ว่า เมื่อเกิดการแปลโค้ดที่มาจากภาษาใดๆ ก็ตามใน .NET จะอาศัย CLR ทำหน้าที่แปลออกมาเป็นภาษากลางที่เรียกว่า IL (Intermediate Language) ก่อน เมื่อได้โค้ดของ IL มาแล้ว ถ้าต้องการแปลออกมาเป็นภาษาเครื่อง ก็จะอาศัยหลักการทำงานของเครื่องจักรเสมือน (Virtual Machine) แปลภาษา IL อีกครั้งหนึ่ง โดยอาศัย Compiler JIT (Just-In-Time)

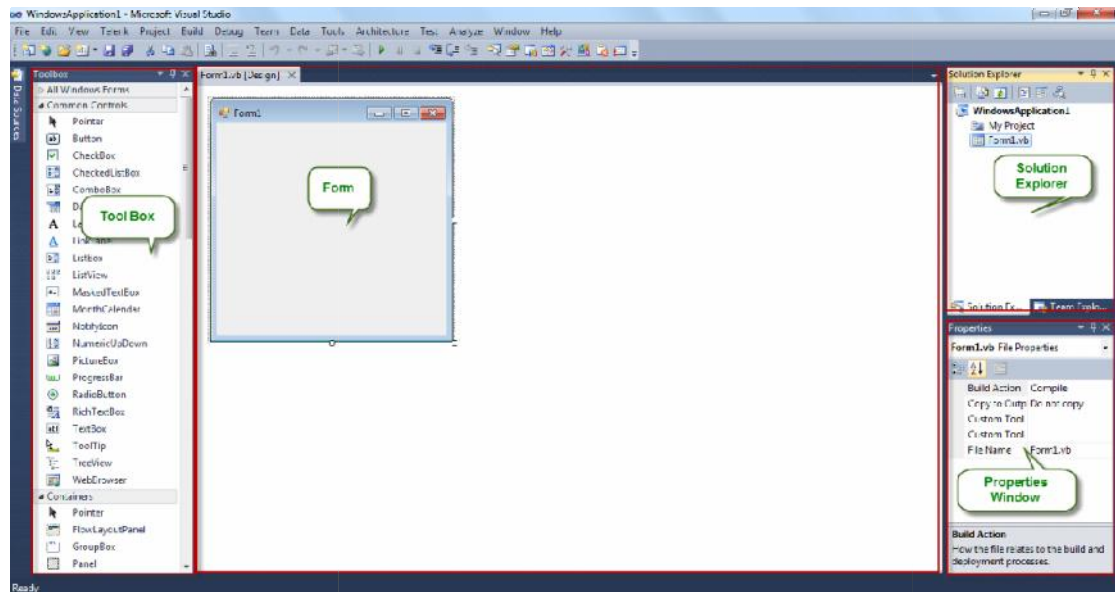
จะเห็นว่าด้วยหลักการทำงานของตัวแปลภาษา CLR ดังกล่าว สามารถตีความได้ว่า ในยุค .NET Microsoft ได้พัฒนาให้ทุกๆ ภาษาเข้าสู่จุดศูนย์กลาง กล่าวคือ ไม่ว่าจะพัฒนา Application ด้วยภาษาใดก็ตาม ท้ายที่สุดแล้ว ก็จะได้โค้ด IL ชุดเดียวกันที่พร้อมจะแปลเป็นภาษาเครื่องได้ทันที ซึ่งข้อดีก็คือ สามารถจะใช้ภาษาใดก็ได้ภายใต้เทคโนโลยี .NET ในการพัฒนา ก็จะได้ผลลัพธ์เหมือนกัน หรืออาจจะสร้าง Application เดียวด้วยหลายๆ ภาษาก็ได้ ทำให้ยืดหยุ่นต่อการพัฒนา Application เป็นทีม

เริ่มต้นการใช้ VB .NET 2010





หลังจากได้สร้าง Application (ใน Visual Studio เรียกว่า Project) ก็จะได้หน้าจอดังรูป

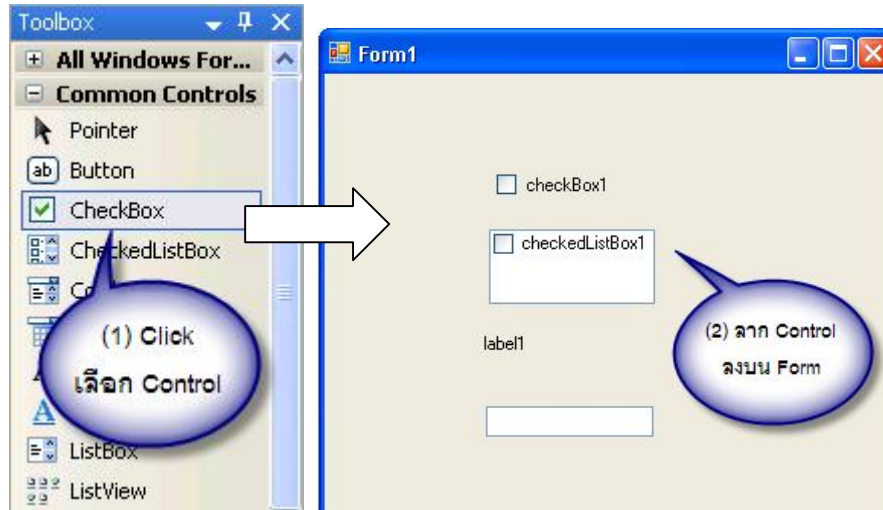


- Toolbox ใช้สำหรับให้ผู้พัฒนา นำ Control ต่างๆ ที่อยู่บน Toolbox มาวางลงบน Form เพื่อสร้างหน้าต่างโปรแกรม
- Form ใช้สำหรับเป็นพื้นที่ให้ Control มาวางลง เพื่อสร้างหน้าต่างโปรแกรม
- Solution Explorer window มีไว้สำหรับแสดงว่า Project เรามีไฟล์อะไรบ้าง ซึ่ง VB นั้นจะมีไฟล์นามสกุลเป็น “.vb”
- Properties window มีไว้สำหรับกำหนดคุณสมบัติต่างๆ ของ Control รวมทั้ง Form

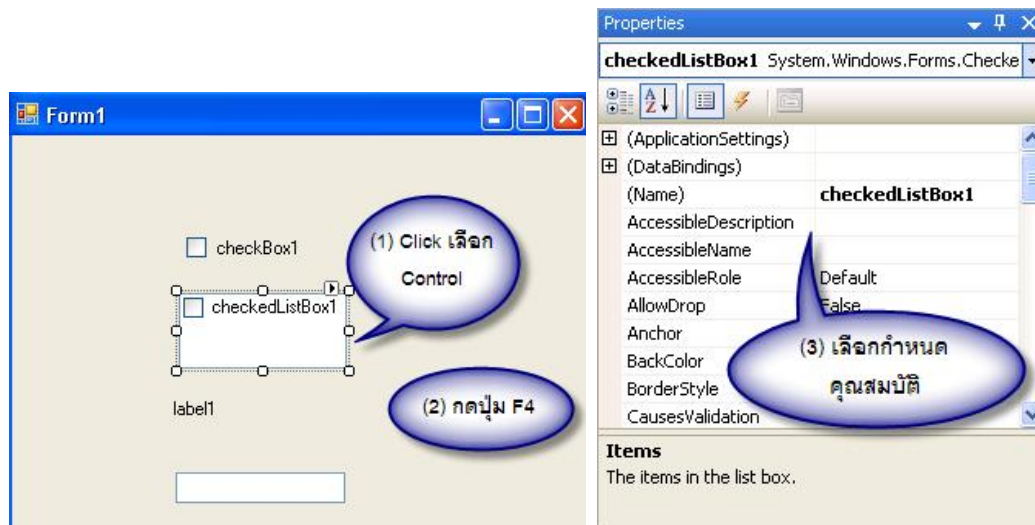
Standard Control

Standard Control คือ Control มาตรฐานที่ Visual Studio จัดมาให้ ซึ่งเป็น Control พื้นฐานที่ผู้ออกแบบสามารถเลือก Control ตัวใด ๆ วางลงฟอร์มได้ ซึ่งจัดไว้อยู่ใน Tool Box นั้นเอง

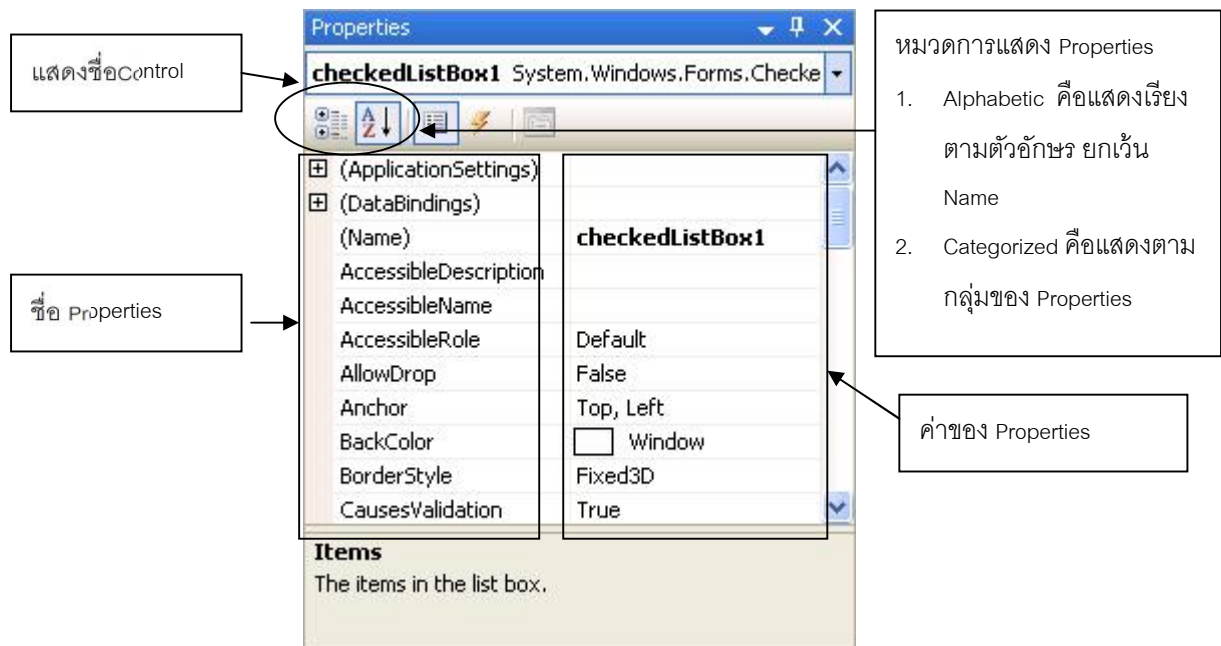
การวาด Control ลงบน Form



การกำหนดคุณสมบัติให้กับ Control หรือ Form



ส่วนประกอบของ Properties window



Properties สำหรับ Standard Control

ความเข้าใจก่อนการกำหนด Properties ในคู่มือเล่มนี้

1. Properties บางตัวของ Control แต่ละตัว จะเห็นการเปลี่ยนแปลงทันที เมื่อผู้ใช้เปลี่ยน Properties แต่ Properties บางตัวจะเห็นก็ต่อเมื่อรันโปรแกรมเท่านั้น เพราะฉะนั้น ก็ให้ลองทั้งสองแบบนะครับ
2. ผู้แต่งไม่ได้จัดมาให้ทุก Properties และของทุก Control ซึ่งหากนักศึกษาที่ต้องการศึกษาเพิ่มเติมก็สามารถไปลองตามกระบวนการข้อ 1 ได้ ในบาง Standard Control ผู้แต่งก็ไม่ได้นำมายกตัวอย่างหรืออธิบายให้ทราบ ซึ่งหากต้องการรายละเอียดจริง ๆ ก็ติดต่อกับผู้แต่งได้โดยตรงนะครับ

ในหัวข้อนี้จะเป็นการกล่าวถึง Properties ของ Control ที่สำคัญ ๆ แต่ขออย่าว่าอธิบาย Properties ไม่ครบทุกตัวนะครับ เอาแต่เพียงส่วนที่สำคัญ ๆ เท่านั้น มีดังนี้ครับ

Form

Form เป็น Control ที่สำคัญของ Visual Studio และจะมีลักษณะพิเศษกว่า Control ตัวอื่น ๆ เนื่องจากว่า Form จะเป็นพื้นที่สำหรับรองรับการวาง Control เพื่อจะสื่อให้กับผู้ใช้งานได้ใช้งานโปรแกรมได้อย่างง่ายดาย สำหรับ Form ใน Visual Studio นั้น จะมีอยู่ 2 ชนิด คือ SDI และ MDI

SDI ฟอรัม คือ ฟอรัมที่เป็นเสมือนโปรแกรมหนึ่ง หากเราออกแบบและรันโปรแกรม เมื่อฟอรัมใด ๆ ถูกเปิดขึ้นมา จะทำให้ใช้เนื้อที่ Taskbar ของ Windows 1 ส่วน ถ้าหากโปรเจกต์ 1 มี หลายฟอรัม และถ้าเปิดหลาย ๆ ฟอรัม ก็จะทำให้เสียพื้นที่ Taskbar อย่างมาก ซึ่งถ้าพิจารณาแล้วจะไม่ค่อยเหมาะสมซักเท่าใดนัก

MDI ฟอรัม คือ ฟอรัมที่มีใช้บรรจุฟอรัมลูกได้ โดย MDI ฟอรัม นั้นโดยส่วนใหญ่จะใช้เป็นการเขียนโปรแกรมที่เป็นมาตรฐาน ซึ่งฟอรัมแบบนี้จะใช้พื้นที่ Taskbar เพียง 1 ส่วนเท่านั้น โดยฟอรัมย่อยที่เปิดขึ้นนั้นจะทำงานอยู่ภายใต้ฟอรัม MDI ตัวอย่างของฟอรัมประเภทนี้จะเห็นได้อย่างมากมาย เช่น โปรแกรมตระกูล Office ซึ่งถ้าลองเปิดใช้งานดูแล้วจะสังเกตเห็นว่าจะมี ปุ่ม 3 ปุ่ม ที่เป็นมาตรฐานของ Windows คือ Minimize, Resize, Close อยู่จำนวน 2 ชุดนั่นเอง

สำหรับในหัวข้อนี้จะขอยกตัวอย่างเฉพาะ SDI เท่านั้น

Properties ของ Form

ชื่อ Properties	ค่าของ Properties
Name	ใช้กำหนดชื่อของ Form ซึ่งต้องให้เป็นไปตามกฎในการตั้งชื่อตัวแปร มีความยาวได้ไม่เกิน 255 ตัวอักษร ข้อแนะนำคือ ควรตั้งชื่อให้เหมาะสมกับลักษณะการทำงานของฟอรัม
BorderStyle	คือ การกำหนดลักษณะของฟอรัมว่าจะให้มีปุ่ม 3 ปุ่มมาตรฐานของ Windows หรือไม่ คือ Minimize, Resize, Close รวมทั้ง Icon บน Title Bar ด้วย ซึ่ง Visual Basic บางเวอร์ชันจะแสดงให้เห็นเมื่อกำหนดค่า Properties ได้เลย แต่บางเวอร์ชันจะเห็นก็ต่อเมื่อรันโปรแกรมเท่านั้น
Caption	เป็นการกำหนดข้อความให้แสดงบน Title Bar
Icon	เป็นการกำหนดรูปภาพให้แสดงบนมุมซ้ายของ Title Bar ซึ่งไฟล์รูปภาพจะต้องเป็นไฟล์ที่มีนามสกุล .ico หรือ .cur เท่านั้น
KeyPreview	เป็นการกำหนดให้มีการตรวจสอบการกดแป้นคีย์บอร์ด ซึ่งจะกำหนดให้เป็น True ก็ต่อเมื่อเราจะทำให้มีการกดแป้น Enter แล้ว เคอร์เซอร์จะเลื่อนไปยัง Control อื่น ๆ ได้
StartPosition	เป็นการกำหนดตำแหน่งฟอรัมเมื่อรันโปรแกรม หรือเมื่อเปิดฟอรัม ซึ่งส่วนใหญ่ก็กำหนดให้เป็น Center Screen คือจะอยู่ตรงกลางหน้าจอทุกครั้งเมื่อเปิดฟอรัมนี้ขึ้นมา
WindowState	เป็นการกำหนดขนาดของฟอรัมเมื่อทำการเปิดฟอรัมขึ้นมาทำงาน ซึ่งก็จะมี Normal = จะมีขนาดเท่ากับที่ได้ออกแบบไว้ Minimize = จะมีขนาดเล็กหรือจะทำการย่อไว้ที่ Task Bar Maximize = จะขยายฟอรัมให้เต็มหน้าจอ

Label

Label เป็น Control ที่ใช้สำหรับแสดงผลข้อความต่าง ๆ เพื่อสื่อกับผู้ใช้งานโปรแกรมได้ง่ายขึ้น จะมี Properties ที่คล้าย ๆ กันกับ Control ตัวอื่น ๆ ในกลุ่มของการแสดงลักษณะของ Control ซึ่งได้แก่ Appearance , Backcolor

ชื่อ Properties	ค่าของ Properties
Name	ใช้กำหนดชื่อให้กับ Label

Autosize	เป็นการกำหนดกรอบของ Label ให้มีขนาดพอดีกับจำนวนตัวอักษร โดยจะปรับขนาดได้เองอัตโนมัติ
BackStyle	คือการกำหนดรูปแบบของพื้นหลัง มีค่า 2 ค่าคือ 0 - Transparent ทำให้เหมือนข้อความลอยอยู่ไม่มีพื้นหลัง 1 - Opaque ทำให้ข้อความมีพื้นหลัง ซึ่งจะสามารถกำหนดสีต่าง ๆ ได้
Text	ใช้กำหนดข้อความ

TextBox เป็น Control ที่อนุญาตให้ผู้ใช้โปรแกรมสามารถระบุข้อมูลต่าง ๆ เพื่อบันทึกค่า หรือส่งค่าลงไปยังโปรแกรม และก็สามารถแสดงข้อมูลที่มีอยู่ในโปรแกรมได้ ซึ่งมี Properties ที่น่าสนใจดังนี้

ชื่อ Properties	ค่าของ Properties
Name	ใช้กำหนดชื่อของ TextBox ซึ่งต้องให้เป็นไปตามกฎในการตั้งชื่อตัวแปร มีความยาวได้ไม่เกิน 255 ตัวอักษร ชื่อแนะนำคือ ควรตั้งชื่อให้เหมาะสมกับลักษณะการทำงานของ TextBox เอง
Appearance	0-Flat หมายถึง การแสดงฟอร์มแบบปกติ โดยไม่ใช้ Visual Effect 1-3D หมายถึง การแสดงในรูป 3 มิติ ซึ่งโดยปกจะใช้แบบนี้
BackColor	คือการกำหนดสีพื้นหลังของฟอร์ม
BorderStyle	คือการกำหนดรูปแบบของกรอบ ซึ่งจะมี 2 ค่าคือ 0 - None คือไม่ให้มีกรอบ 1 - Fixed Single คือให้มีกรอบ
Locked	เป็นการกำหนดให้ TextBox ไม่สามารถระบุข้อมูลใด ๆ ได้
MaxLength	คือการกำหนดให้ TextBox สามารถรับข้อมูลได้กี่ตัวอักษร โดยผู้ออกแบบสามารถระบุตัวเลขลงไปได้เลย แต่ถ้าหากต้องการแบบไม่จำกัด ก็ให้ระบุตัวเลข 0
MultiLine	ใช้กำหนดให้ TextBox สามารถรับข้อมูลได้หลายบรรทัด ซึ่งโดยปกติแล้ว TextBox จะสามารถรับข้อมูลได้เพียงบรรทัดเดียวเท่านั้น ซึ่งจะมี 2 ค่า คือ True คือสามารถรับได้หลายบรรทัด และ False รับได้บรรทัดเดียว
PasswordChar	ใช้กำหนดอักษรแสดงแทนข้อมูลที่ผู้ใช้ระบุ โดย Properties นี้จะใช้ในกรณีผู้ออกแบบจะใช้ TextBox ตัวนี้ระบุ Password
Text	เป็น Properties ประจำตัวของ TextBox ที่เก็บค่าหรือแสดงค่าข้อมูลต่าง ๆ ที่ปรากฏใน TextBox ทั้งหมด

CommandButton

เป็น Control ที่กำหนดให้ผู้ใช้คลิกเลือกเพื่อจะให้โปรแกรมทำงานตามที่ต้องการ หรือเพื่อดำเนินการอย่างใดอย่างหนึ่ง Command Button เป็นมาตรฐานของการเขียนโปรแกรมบน Windows

เป็น Control ที่จะต้องมีในทุก Form ที่ได้ทำการออกแบบไว้ในส่วนของการออกแบบโปรแกรม Properties ที่น่าสนใจมีดังนี้

ชื่อ Properties	ค่าของ Properties
Default	เป็นการกำหนดให้ปุ่มพร้อมที่จะให้ผู้ใช้งานกดแป้น Enter หรือคลิกได้ เสมือนกับว่า Cursor ไปกระพริบอยู่ ณ ปุ่มนั้นเลยก็ว่าได้
Caption	ใช้กำหนดข้อความให้กับปุ่ม เพื่อให้ผู้ใช้รู้ว่าปุ่มนี้ใช้ทำอะไร ซึ่งในการกำหนด Caption นี้ ถ้าต้องการให้มีขีดเส้นใต้ที่ตัวอักษรใด ก็เพียงแต่ใส่สัญลักษณ์ & ไว้ที่หน้าตัวอักษรตัวนั้น แต่ต้องไม่เป็นตัวอักษรตัวเดียวกันบนฟอร์มเดียวกัน เนื่องจากว่า Windows จะอนุญาตให้ผู้ใช้งานกดแป้น Alt+ตัวอักษรตัวที่ขีดเส้นใต้เช่น <u>OK</u> ถ้าหากไม่ต้องการคลิกที่ปุ่มนี้ ก็สามารถกดแป้น Alt+O ได้ เป็นต้น
BackColor,Picture	เป็นส่วนของการแสดงแบบสีพื้น หรือรูปภาพ แทนข้อความหรือรวมกับข้อความบนปุ่มก็ได้ แต่ต้องใช้ควบคู่กับ Properties Style
Style	ใช้เลือกรูปแบบการแสดง ซึ่งมี 2 แบบคือ 0 – Standard เป็นลักษณะปุ่มทั่ว ๆ ไป 1 – Graphic สามารถกำหนดสีพื้นหลัง และ ใส่รูปภาพได้
Text	ใช้กำหนดข้อความให้กับปุ่ม

Option Button

เป็น Control ที่ใช้สำหรับให้ผู้ใช้สามารถเลือกได้เพียงอย่างเดียว ไม่ว่าจะมียกที่ทางเลือกก็ตาม โดยมี Properties ที่น่าสนใจดังนี้

ชื่อ Properties	ค่าของ Properties
Caption	ใช้กำหนดข้อความ เพื่อให้ผู้ใช้ได้ทราบ
Style	ใช้เลือกรูปแบบการแสดง ซึ่งมี 2 แบบคือ 0 – Standard เป็นลักษณะปุ่มทั่ว ๆ ไป 1 – Graphic สามารถกำหนดสีพื้นหลัง และ ใส่รูปภาพได้และจะมีลักษณะเหมือน CommandButton แต่โดยส่วนใหญ่ไม่แนะนำให้ใช้ซะ มันไม่เป็นมาตรฐาน
Value	เป็น Properties ที่ใช้ทดสอบการเลือก ซึ่งมีอยู่ 2 ค่า คือ True เมื่อโดยคลิกเลือก False เมื่อไม่ได้เลือก

Check Box

เป็น Control ที่ใช้สำหรับให้ผู้ใช้เลือกเหมือนกับ OptionButton แต่จะสามารถเลือกได้หลายทางเลือก Properties ที่น่าสนใจมีดังนี้

ชื่อ Properties	ค่าของ Properties
-----------------	-------------------

Caption	ใช้กำหนดข้อความ เพื่อให้ผู้ใช้ได้ทราบ
Style	ใช้เลือกรูปแบบการแสดงผล ซึ่งมี 2 แบบคือ 0 - Standard เป็นลักษณะปุ่มทั่ว ๆ ไป 1 - Graphic สามารถกำหนดสีพื้นหลัง และ ใส่รูปภาพได้และจะมีลักษณะเหมือน CommandButton แต่โดยส่วนใหญ่ไม่แนะนำให้ใช้นะ มันไม่เป็นมาตรฐาน
Value	เป็น Properties ที่ใช้ทดสอบว่าผู้ใช้เลือกหรือยัง ซึ่งมีอยู่ 3 ค่า คือ 0 - UnChecked เมื่อไม่ได้เลือก คือ ช่องสี่เหลี่ยมจะไม่มีเครื่องหมายถูก 1 - Checked เมื่อผู้ใช้เลือก 2 - Gray จะเป็นลักษณะ Disable ไม่อนุญาตให้คลิกเลือกได้

Combo Box

เป็น Control ที่จะสามารถบรรจุรายการเพื่อให้ผู้ใช้เลือกรายการที่บรรจุอยู่ใน Combo Box ได้ ซึ่งจะสามารถแสดงรายการได้เพียงหนึ่งบรรทัดเท่านั้น ซึ่งจะมีลักษณะเหมือน Text Box ในบางส่วน ในกระบวนการออกแบบนั้น จะสามารถขยายได้เพียงด้านกว้างเท่านั้น ด้านสูงจะไม่สามารถขยายได้ Properties ที่น่าสนใจมีดังนี้

ชื่อ Properties	ค่าของ Properties
items	ใช้บรรจุรายการที่ต้องการให้ผู้ใช้เลือก ซึ่งลักษณะการบรรจุรายการนั้นก็สามารถคลิกที่ Properties List จากนั้นก็จะมีกรอบเล็ก ๆ เพื่อที่จะให้เราพิมพ์รายการที่ต้องการบรรจุเข้าไป ซึ่งจะมองเป็น 1 บรรทัด / 1 รายการ
Lock	ไม่อนุญาตให้ผู้ใช้คลิกเลือกรายการได้
Sort	อนุญาตให้มีการเรียงลำดับรายการหรือไม่ ซึ่งจะเรียงจากน้อยไปมากเสมอ
Text	ใช้แสดงข้อความ เมื่อทำการเปิดฟอร์มและ Control พร้อมทั้งจะทำงาน

List Box

เป็น Control ที่มีจุดประสงค์เดียวกับกับ Combo Box เพียงแต่ต่างกันแค่ รูปแบบการแสดงผลต่อผู้ใช้ ซึ่ง List Box จะสามารถแสดงได้หลายบรรทัด แต่จะสามารถเลือกได้เพียง 1 รายการเช่นกัน ซึ่งถ้าหากมีรายการข้อความที่มีความยาวหรือจำนวนมาก List Box นี้จะสร้าง Scroll Bar มาให้โดยอัตโนมัติ Properties ที่น่าสนใจมีดังนี้

ชื่อ Properties	ค่าของ Properties
Columns	ใช้กำหนดจำนวน Column เพื่อแสดงใน List Box โดยถ้าหากเป็น Column เดียวก็ จะกำหนดตัวเลขให้มีค่า 0 ถ้าต้องการ 2 Column ก็กำหนดเป็นเลข 1 เช่นนี้ไปเรื่อย ๆ
items	ใช้บรรจุรายการที่ต้องการให้ผู้ใช้เลือก ซึ่งลักษณะการบรรจุรายการนั้นก็สามารถ

	คลิกที่ Properties List จากนั้นก็จะมีกรอบเล็ก ๆ เพื่อที่จะให้เราพิมพ์รายการที่ต้องการบรรจุเข้าไป ซึ่งจะมองเป็น 1 บรรทัด / 1 รายการ
Style	ใช้เลือกรูปแบบการแสดงผล ซึ่งมี 2 แบบคือ 0 - Standard รายการที่บรรจุอยู่จะแสดงเป็นบรรทัดธรรมดา สามารถเลือกรายการได้เพียง 1 รายการ 1 - Checkbox จะแสดงแต่ละรายการเหมือนกับ Check Box และสามารถเลือกรายการได้มากกว่า 1 รายการ

Properties ร่วม

สำหรับหัวข้อนี้ได้แยกมา เนื่องจากว่ามี Properties บางจำพวก ที่มีในทุก ๆ Control ผู้แต่งก็เลยเอามาอธิบายไว้ส่วนท้ายเลย เพื่อให้ดูง่ายมากขึ้นนั่นเองครับ ดังนี้ครับผม

ชื่อ Properties	ค่าของ Properties
Name	ใช้กำหนดชื่อให้กับ Control ซึ่งเป็นส่วนสำคัญและจำเป็นอย่างยิ่ง เหมือนกับการประกาศตัวแปรนั่นแหละ ซึ่งจะต้องเป็นไปตามกฎการตั้งชื่อ และมีความยาวได้ไม่เกิน 255 ตัวอักษร หลักสำคัญ ต้องตั้งชื่อ Control ให้ตรงกับงาน เนื่องจากว่า ชื่อ Control เหล่านี้ เราจะนำไปสู่การเขียนโปรแกรม
Appearance	เป็นโหมดของการแสดงผลหน้าต่างของ Control ตัวนั้น ซึ่งแต่ละ Control ก็จะมีไม่เท่ากัน ให้ทดสอบดูได้ เนื่องจากว่าเราจะเห็นผลการเปลี่ยนแปลงในขณะที่เราเปลี่ยนค่าในหน้าต่าง Properties เลย
BackColor	ใช้กำหนดสีพื้นหลัง โดยก็ขึ้นอยู่กับชนิดของ Control นั้น ๆ
DataField, DataMember, DataSource, DataFormat	เป็นกลุ่มของ Properties ที่ใช้เชื่อมกับฐานข้อมูล ซึ่งจะมีรายละเอียดใน หลักสูตร Visual Basic with Database (มีในบาง Control เท่านั้นนะครับ)
Enabled	ใช้กำหนดให้ Control นั้นสามารถใช้งานได้หรือไม่ True - ใช้งานตามปกติ False - ใช้งานไม่ได้ แต่แสดงที่ฟอร์ม
Font	ใช้กำหนดชนิดและขนาดของตัวอักษร
ForeColor	ใช้กำหนดสีของตัวอักษร
Height , Width	กำหนดความกว้างและความสูงของ Control ตัวนั้น ซึ่งตัวเลขจะเปลี่ยนตามที่เราวาด Control ลงบนฟอร์ม หรือ สามารถระบุตัวเลขได้เลย
Mouselcon	คือการแสดงรูปเมาส์ เมื่อเลื่อนเมาส์ไปส่วนใดส่วนหนึ่งของฟอร์ม โดยจะใช้คู่กันกับ Properties MousePointer
MousePointer	คือการแสดงรูปเมาส์ ที่ Visual Basic จัดมาให้ โดยจะมีอยู่ 16 ค่า (0-15) และ ค่า99

	หากผู้ใช้ต้องการแสดงรูปอื่น ๆ ที่นอกเหนือจากที่ Visual Basic กำหนดมาให้ โดยเมื่อกำหนดค่าเป็น 99 - Custom แล้ว ก็สามารถไปเลือกรูปที่ Properties MouseIcon ได้ โดยไฟล์จะต้องเป็นนามสกุล .ico หรือ .cur เท่านั้น
Index	ใช้สำหรับ Control Array (รายละเอียดในหัวข้อต่อไป)
TabIndex	เป็นลำดับของ Control ที่วางลงบนฟอร์ม ทั้งนี้หากเขียนโปรแกรมเพื่อการกดแป้น Enter ลำดับการเลื่อนของ Cursor จะไปตาม Properties ตัวนี้แหละครับ เพราะฉะนั้น หากต้องการลำดับที่เป็นขั้นตอน ก็สามารถเปลี่ยนค่าได้ โดยจะเริ่มต้นที่ค่า 0
ToolTipText	เป็นการแสดงข้อความ เมื่อเอาเมาส์ไปชี้บน Control ตัวนั้น ๆ
Visible	กำหนดให้ซ่อนหรือแสดง Control

บทที่ 2

การเขียนคำสั่งภาษา VB.NET

โครงสร้างของคำสั่งภาษา VB.NET 2010

กฎของการตั้งชื่อ

การตั้งชื่อในภาษา VB ใช้สำหรับการตั้งชื่อใด เช่น ชื่อตัวแปร ชื่อคลาส ชื่อ Method โดยจะต้องมีกฎเกณฑ์ดังต่อไปนี้

1. ไม่ขึ้นต้นด้วยตัวเลขหรืออักขระพิเศษ
2. ไม่มีอักขระพิเศษ
3. ไม่มีช่องว่าง
4. สามารถใช้ตัวอักษรดังต่อไปนี้ (a – z, A – Z , 0-9 _ (ขีดล่าง)) เท่านั้น
5. อักษรตัวพิมพ์เล็ก และ ตัวพิมพ์ใหญ่ VB ถือว่าเป็นตัวเดียวกัน เช่น a, A
6. ไม่ซ้ำกับคำสงวน (Reserve word)ของภาษา VB
7. สามารถใช้อักขระได้ 255 ตัวอักษรต่อการตั้งชื่อ 1 ชื่อ

ตัวอย่างที่ถูกต้อง

Hello, Test, Exam, Total, Net, SPC, V1, V3, Balance_score, MidtermScore

ตัวอย่างที่ไม่ถูกต้อง

1Hello, Test%, E*xam, Balance score, MidtermScore”]

ตัวอย่างคำสงวนของ VB

abstract	else	interface	super
boolean	extends	long	switch
break	false	native	synchronized
byte	final	new	this
case	finally	null	throw
catch	float	package	throws
char	for	private	transient
class	goto	protected	true
const	if	public	try
continue	implements	return	void
default	import	short	volatile
do	instanceof	static	while
double	int	strictfp	

คำเหล่านี้จะไม่สามารถนำไปตั้งชื่อได้

ตัวแปร (Variable)

ตัวแปรคือพื้นที่ในหน่วยความจำหลักที่ผู้เขียนโปรแกรมสามารถจองไว้สำหรับเก็บผลลัพธ์ที่เกิดจากโปรแกรม ทุกภาษาที่ใช้เขียนโปรแกรมจะต้องมีการใช้ตัวแปรนะครับ ซึ่งใน VB มีรูปแบบการประกาศตัวแปรดังนี้

ตัวแปรตัวเดียว

```
Dim ชื่อตัวแปร As ชนิดตัวแปร
```

ตัวแปรหลายตัวที่เป็นชนิดเดียวกัน

```
Dim ชื่อตัวแปร1, ชื่อตัวแปร2, ชื่อตัวแปร3, ... As ชนิดตัวแปร
```

เมื่อเราได้กำหนดระบบปฏิบัติการจะทำการเก็บพื้นที่ในหน่วยความจำหลักไว้สำหรับตัวแปรที่เรากำหนดขึ้น ไม่มีใครที่จะสามารถเข้าไปใช้งานพื้นที่หน่วยความจำในส่วนของเราได้ ซึ่งระบบปฏิบัติการจะจองพื้นที่หน่วยความจำไว้เท่าใดนั้นก็ขึ้นอยู่กับชนิดของตัวแปรนั้น โดยชนิดตัวแปรและขนาดของชนิดนั้นมีดังนี้

1. ตัวเลขจำนวนเต็ม ได้แก่ byte, short, int ,long
2. เลขจำนวนจริง ได้แก่ float , double
3. ตัวอักษร ได้แก่ char, String
4. ค่าตรรกะ ได้แก่ boolean

สำหรับชนิดต่าง ๆ นั้นจะมีขนาดของข้อมูลที่แตกต่างกันดังรูป

Type Name	Kind of Value	Memory Used	Size Range
byte	integer	1 byte	-128 to 127
short	integer	2 bytes	-32768 to 32767
int	integer	4 bytes	-2,147,483,648 to 2,147,483,647
long	integer	8 bytes	-9,223,372,036,854,775,808 to 9,223,374,036,854,775,808
float	floating point	4 bytes	+/- 3.4028... x 10 ⁺³⁸ to +/- 1.4023... x 0 ⁻⁴⁵
double	floating point	8 bytes	+/- 1.767... x 10 ⁺³⁰⁸ to +/- 4.940... x 0 ⁻³²⁴
char	single character (Unicode)	2 bytes	all Unicode characters
boolean	true or false	1 bit	not applicable

การใส่หมายเหตุ

บางครั้งการเขียนโปรแกรม ผู้เขียนเองก็ต้องใส่หมายเหตุไปด้วย ซึ่งหมายเหตุนี้จะไม่ใช่คำสั่งของภาษา และ ตัว Compiler เองก็จะไม่นำไป Compile มีรูปแบบดังนี้

‘บรรทัดหมายเหตุ

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Dim s As Char
    'บรรทัดนี้สำหรับใส่หมายเหตุ
End Sub
```

การเขียนโปรแกรม

การเขียนโปรแกรมภาษา Visual Basic นับได้ว่าเป็นภาษาที่มีความง่ายที่สุดในบรรดาทุกภาษาที่บรรจุอยู่ใน Visual Studio เนื่องจากว่ามีรูปแบบที่ไม่ยาก ไม่มีข้อกำหนดหรือกฎเกณฑ์อะไรมากมายนัก และมีเส้นแบ่งระหว่าง Event เพื่อให้เห็นง่ายอีกด้วย

สำหรับการเขียนโปรแกรม (Code) ให้กับโปรเจกต์ของ Visual Basic สามารถแบ่งได้ 2 ประเภทใหญ่ ๆ คือ 1. การเขียนโปรแกรมกับ Control และ 2. การเขียนโปรแกรมด้วยคำสั่งต่าง ๆ ของ Visual Basic ซึ่งก่อนอื่นก็ต้องมาทำความเข้าใจกับหน้าต่าง Code ก่อนนะครับ

วิธีการเรียก Code มีหลายวิธีด้วยกันครับ แต่จะขอแนะนำ 2 วิธีดังนี้

1. Double Click ที่ Control ใด ๆ ก็ได้รวมทั้งฟอร์มด้วย
2. คลิกขวาเลือกชื่อฟอร์มที่ Project Explorer แล้วเลือกเมนู View Code



1. ชื่อ Control หมายถึง เป็น Combo Box ที่รวมรายชื่อ Control ทุกตัวที่เราวางลงบนฟอร์ม โดยจะเอา ชื่อที่มาแสดงจาก Properties Name ของทุก Control ยกเว้น ฟอร์ม อันเดียวเท่านั้นที่ยังคงใช้คำว่า Form
2. ชื่อ Event ก่อนอื่น ต้องมาทำความเข้าใจคำว่า “Event” ก่อนนะครับ
Event ถ้าแปลเป็นภาษาไทยก็แปลว่า “เหตุการณ์” คือเหตุการณ์หนึ่ง ๆ ที่เกิดขึ้นกับ Control ซึ่งทุก Control จะต้องมีเหตุการณ์ เช่น เหตุการณ์ Click ของ Command Button หมายถึง เมื่อผู้ใช้คลิกที่ปุ่ม หรือ เหตุการณ์ Load ของ Form หมายถึง เมื่อฟอร์มถูกโหลดขึ้นมา เป็นต้น ซึ่งแต่

ละ Control จะมีเหตุการณ์ไม่เหมือนกัน เพราะฉะนั้น นักศึกษาต้องหมั่นฝึกหัดเองนะ โดยวิธีไม่ยากครับ เพียงเขียน Code เข้าไปใน Event นั้น ๆ แล้วลองรันโปรแกรมดู ก็จะพบกับสิ่งที่เกิดขึ้น

3. ส่วนแสดง Code เป็นส่วนที่แสดงให้เราเห็นว่าเราเขียน Code อะไรไว้ ของ Control พุดง่าย ๆ ว่า Event ใคร Event มัน ว่าจั้นเถอะ
4. Tab เป็นส่วนที่แสดงหน้า code และส่วนที่ออกแบบ ไว้คนละส่วนกัน ซึ่งผู้ใช้ก็สามารถคลิกสลับไปมาได้ โดยไม่ต้อง double click ที่ control ทุกครั้งหากต้องการไปแสดงหน้าต่าง code

การเขียนโปรแกรมควบคุม Control

หมายถึงการเขียนโปรแกรมเพื่อควบคุมคุณสมบัติของ Control ให้มีคุณสมบัติเป็นแบบไม่คงที่ได้ในขณะรันโปรแกรม ซึ่งโดยปกติ Control หนึ่งๆ จะมีคุณสมบัติเบื้องต้นและในส่วนที่เรากำหนดในหน้าต่าง Properties window

รูปแบบของคำสั่ง

ชื่อ Control.Property = ค่าที่ต้องการกำหนดให้กับ Control

หมายเหตุ

ต้องรู้จักชื่อ Control ซึ่ง VB จะหาชื่อ Control ให้เรา เมื่อเราพิมพ์คำสั่งลงไป และเมื่อได้ชื่อ Control แล้วเราก็ต้องพิมพ์ตามด้วย จุด (.) แล้วหน้าต่าง Properties ก็จะมาปรากฏขึ้นมาให้อัตโนมัติ ดังตัวอย่างต่อไปนี้

ตัวอย่าง

จาก Form1 มี Control Text Box และ Command button อย่างละ 1 อัน จากนั้นเราจะเขียนโปรแกรมเพื่อกำหนดสีพื้นหลังของ Text Box จากสีขาว ให้เป็นสี น้ำเงิน และให้มีข้อความว่า “สวัสดี” โดยเหตุการณ์ทั้งหมดจะเกิดขึ้นเมื่อเราคลิกที่ปุ่ม สามารถทำได้ดังนี้ครับ

The screenshot shows the development environment with the following annotations:

- (1) ค้นหาชื่อ Control: Points to the 'textBox1' property in the Properties window.
- (2) ค้นหาจุด: Points to the 'Click' event in the 'Events' section of the Properties window.
- (3) เลือก Property: Points to the 'BackColor' property in the Properties window.
- (4) กำหนดค่า: Points to the value 'Color.Blue' being assigned to the BackColor property in the Code window.

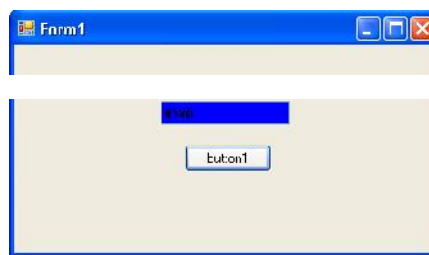
คุณสมบัติส่วนอื่นก็สามารถกำหนดได้เช่นเดียวกันครับ ซึ่งสุดท้ายแล้วเราก็จะได้ตัวอย่างโค้ดโปรแกรมดังรูป

```

Form1.vb* Form1.vb [Design]* Start Page
Form1
(Declarations)
Public Class Form1
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        'บรรทัดนี้หมายเหตุนะ
        Dim net As Integer
        'บรรทัดนี้ก็หมายเหตุเช่นกัน
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        TextBox1.BackColor = Color.Blue
        TextBox1.Text = "สวัสดี"
    End Sub
End Class
    
```

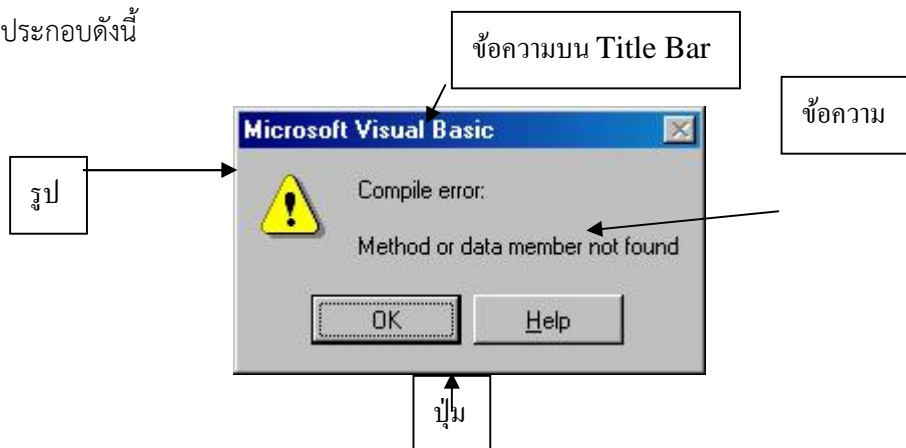
ผลลัพธ์เมื่อรันโปรแกรม



ส่วนการเขียนโปรแกรมเพื่อควบคุมคุณสมบัติของ Control ตัวอื่นๆ ก็สามารถทำได้โดยใช้หลักการเดียวกันนี้ครับ

การเรียกใช้ Message Box

MessageBox เป็นรูปแบบที่สำคัญของการเขียนโปรแกรมบน Windows เนื่องจากเป็นตัวบรรจุข่าวสารจากโปรแกรมแจ้งไปยังผู้ใช้งาน ไม่ว่าจะเป็น ข่าวสารทั่วไป หรือ ข้อความเตือนอื่นๆ มีส่วนประกอบดังนี้



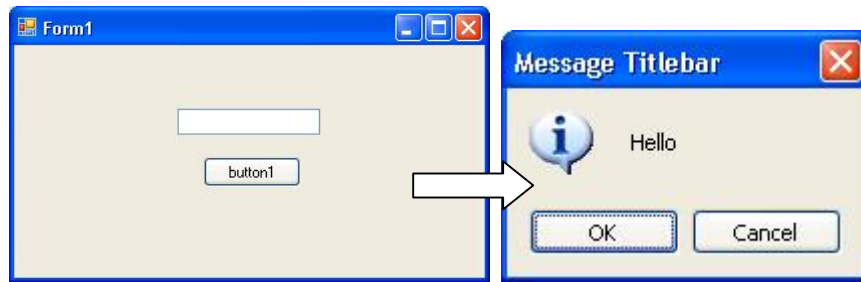
รูปแบบคำสั่ง

```

MessageBox.Show("ข้อความ", "ข้อความบน Titlebar", MessageBoxIcon.รูป, MessageBoxButtons.ปุ่ม);
    
```

ตัวอย่าง

เมื่อคลิกที่ปุ่ม ก็จะทำให้แสดง Message Box ขึ้นมาดังรูป



ซึ่งโค้ดโปรแกรมก็แสดงได้ดังนี้

```
MessageBox.Show("Hello","MessageTitlebar", MessageBoxButtons.OKCancel, MessageBoxIcon.Information);
```

การเขียนโปรแกรมด้วยคำสั่งตัดสินใจ

ใน VB มีหลายคำสั่ง แต่จะยกมาเพียง 2 คำสั่งได้แก่

- if...else
- select...case

คำสั่ง if...else

รูปแบบคำสั่ง

รูปแบบที่ 1 (เงื่อนไขเดียว)

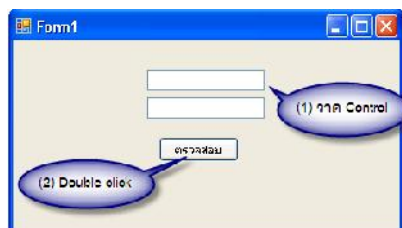
```
if เงื่อนไขที่ใช้ในการทดสอบ then
    คำสั่ง.....(ทำงานเมื่อเงื่อนไขเป็นจริง)
```

รูปแบบที่ 2 (2 เงื่อนไข)

```
if เงื่อนไขที่ใช้ในการทดสอบ then
    คำสั่ง..... (ทำงานเมื่อเงื่อนไขเป็นจริง)
else
    คำสั่ง.....(ทำงานเมื่อเงื่อนไขเป็นเท็จ)
```

ตัวอย่าง

เป็นโปรแกรมที่ใช้ตรวจสอบค่าตัวเลข 2 ค่าที่ป้อนเข้าไปใน Textbox โดยใช้เงื่อนไขเพื่อทดสอบว่าค่าไหนที่มากหรือน้อยกว่ากัน ดังนี้



เขียนโค้ดดังนี้

```

3 Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
    Dim var1, var2 As Integer
    var1 = Integer.Parse(TextBox1.Text)
    var2 = Integer.Parse(TextBox2.Text)
    If var1 > var2 Then
        MsgBox("ค่าที่ 1 มากกว่าค่าที่ 2", MsgBoxStyle.OkOnly, "VB")
    Else
        MsgBox("ค่าที่ 2 มากกว่าค่าที่ 1", MsgBoxStyle.OkOnly, "VB")
    End If
End Sub

```

-End Class

จะสังเกตเห็นว่าตัวอย่างคำสั่ง ใช้ตัวแปร var1 และ var2 รับค่าที่ผู้ใช้ระบุผ่านเข้ามาทาง textbox ซึ่งมีการใช้ function ในการแปลงค่าก่อนเพื่อให้ค่าที่อยู่ใน textbox นั้นสามารถมาเก็บไว้ในตัวแปรได้ โดยจะขออธิบายหลักการดังต่อไปนี้

1. โปรแกรมจะมองทุกอย่างที่อยู่ใน textbox เป็นข้อมูลชนิดตัวอักษรทั้งหมด
2. หากต้องการเก็บค่าที่อยู่ใน textbox มาเก็บไว้ในตัวแปรก็ต้องใช้ function ในการแปลง ทั้งนี้จะใช้ function ไหนก็ต้องดูชนิดตัวแปรที่ประกาศเอาไว้แต่เบื้องต้นด้วย เช่น Var1 และ var2 เป็นชนิดข้อมูลแบบ integer ดังนั้นจึงใช้ function ในการแปลงคือ Integer.parse(ชื่อ textbox) เป็นต้น และเป็นตัวอย่าง
3. หากต้องการนำข้อมูลที่เก็บอยู่ในตัวแปรอื่นใดที่ไม่ใช่ตัวอักษร เมื่อนำไปแสดงที่ textbox ต้องต่อท้ายด้วย function .toString ด้วยเสมอ

คำสั่ง switch...case

ใช้สำหรับการทำงานที่มีหลายเงื่อนไข ซึ่งจริงๆ แล้วก็ใช้ if...else ก็ได้ แต่ถ้าหากมีหลายเงื่อนไขก็ใช้คำสั่งนี้จะสะดวกกว่าเยอะครับ

รูปแบบคำสั่ง

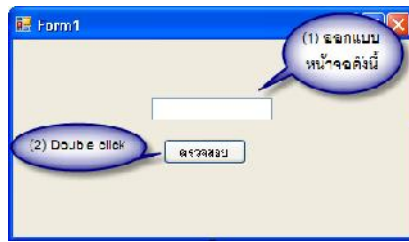
```

switch (เงื่อนไขที่ใช้ในการทดสอบ)
{
    case (เงื่อนไขที่ 1) :
        // ทำงานที่นี่
        break;
    case (เงื่อนไขที่ 2) :
        // ทำงานที่นี่
        break;
    ....
    default :
        //ถ้าไม่ตรงกับเงื่อนไขใด ก็ทำตรงนี้
        break;
}

```

ตัวอย่าง

เป็นโปรแกรมทดสอบการพิมพ์ตัวอักษรลงใน TextBox ซึ่งกำหนดให้พิมพ์ a,b,c เท่านั้น ซึ่งเมื่อพิมพ์เสร็จและคลิกที่ปุ่ม โปรแกรมก็จะแสดง MessageBox มาให้ทราบว่าเราพิมพ์ตัวอะไรลงไป



หมายเหตุ:
ถ้าเงื่อนไขตรงนี้เป็น string ในคำสั่ง case ต้องมีเครื่องหมายคำพูดตรงค่า ดังตัวอย่าง แต่ถ้าเป็นตัวเลขก็ไม่ต้องนะครับ

```

1 Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    Dim st As String
    st = TextBox1.Text
    Select Case st
        Case "a"
            MsgBox("คุณพิมพ์ a", MsgBoxStyle.OkOnly, "VB")
        Case "b"
            MsgBox("คุณพิมพ์ a", MsgBoxStyle.OkOnly, "VB")
        Case "c"
            MsgBox("คุณพิมพ์ a", MsgBoxStyle.OkOnly, "VB")
        Case Else
            MsgBox("คุณพิมพ์ไม่ถูกเงื่อนไข", MsgBoxStyle.OkOnly, "VB")
    End Select
End Sub
- End Class
    
```

การเขียนโปรแกรมด้วยคำสั่งวนรอบ

คำสั่ง for

เป็นคำสั่งวนรอบที่ใช้สำหรับการสั่งให้โปรแกรมทำงานเดิมๆ เป็นจำนวนครั้งที่ตามที่ต้องการ เช่น ต้องการให้พิมพ์ชื่อตนเองมา 10 ครั้ง เป็นต้น ประโยชน์ของคำสั่งนี้ก็คือ ไม่ต้องพิมพ์หลายคำสั่งหากต้องการให้ทำงานเดิมๆ ซ้ำๆ

รูปแบบคำสั่ง

```

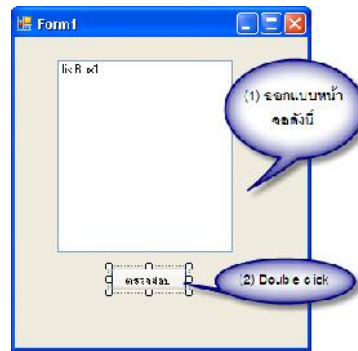
for ตัวแปร(integer)=ค่าเริ่มต้น to ค่าสุดท้าย
    คำสั่ง.....
Next ตัวแปร
    
```

อธิบายเพิ่มเติม

คำสั่ง for เป็นคำสั่งที่สั่งให้โปรแกรมทำงานเดิมๆ ในจำนวนครั้งที่เราสามารถกำหนดได้ว่าให้ทำงานกี่รอบ โดยจะต้องกำหนดค่าเริ่มต้นและค่าสุดท้ายของรอบที่ต้องการทำงาน โดยค่าที่จะสามารถกำหนดได้นั้นต้องใช้ตัวแปรซึ่งจะต้องเป็นชนิดเลขจำนวนเต็ม ซึ่งจะสังเกตได้จากหลังคำสั่ง for นั้นเอง

ตัวอย่าง

เป็นโปรแกรมที่ต้องการให้พิมพ์คำว่า “ประเทศไทย ” ออกมา 10 รอบ ใน List Box เมื่อผู้ใช้คลิกที่ปุ่ม



```

1 Private Sub Button2_Click(ByVal sender As System.Object, E
    Dim i As Integer
    For i = 1 To 10
        ListBox1.Items.Add("ประเทศไทย")
    Next i
- End Sub
-End Class

```

อธิบายคำสั่ง

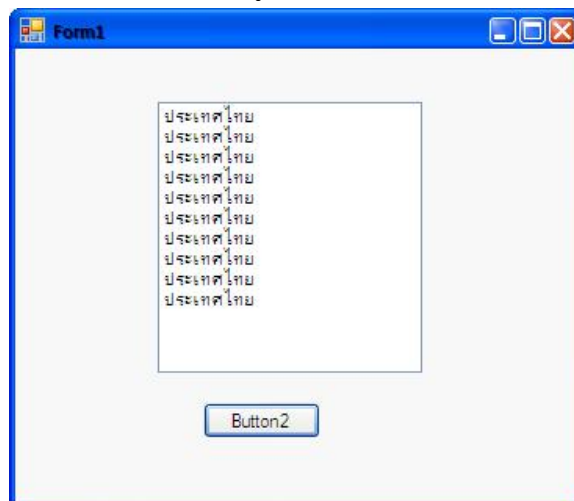
Dim i As Integer เป็นการประกาศตัวแปร i เพื่อใช้เป็นตัวนับรอบ

For i=1 to 10 ใช้กำหนดจำนวนรอบ 10 รอบ

ListBox1.Items.Add(“ประเทศไทย”) เป็นคำสั่งที่ให้เพิ่มคำว่าประเทศไทยเข้าไปใน Listbox

Next i ใช้คู่กับ for โดยต้องเป็นตัวแปรตัวเดียวกัน

เมื่อเขียนโปรแกรมเสร็จก็ลองรันโปรแกรมดูนะครับ ซึ่งก็จะได้ผลดังนี้



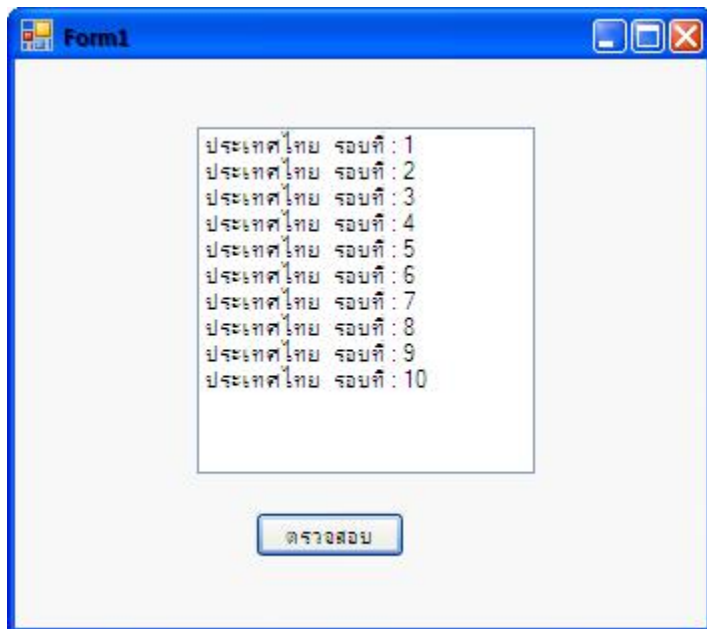
ตัวอย่างที่ 2 (คำสั่ง for)

คล้ายกับตัวอย่างแรกนั่นแหละครับ แต่ครั้งนี้จะให้แสดงรอบออกมาด้วย ซึ่งนั่นก็คือแสดงค่าของตัวแปร i นั้นเอง โดยให้ออกแบบหน้าจอโปรแกรมเหมือนเดิม เพียงแต่เปลี่ยนโค้ดให้เป็นอย่างนี้ครับ

```
1 Private Sub Button2_Click(ByVal sender As System.Object,
    Dim i As Integer
    For i = 1 To 10
        ListBox1.Items.Add("ประเทศไทย รอบที่ : " & i)
    Next i
End Sub
End Class
```

มีหนึ่งบรรทัดคำสั่งที่เปลี่ยนแปลง คือ `ListBox1.Items.Add("ประเทศไทย รอบที่ : " & i)`

ซึ่งผลรันโปรแกรมก็จะได้อันตัวอย่าง

**คำสั่ง while**

เป็นคำสั่งวนรอบอีกคำสั่งหนึ่ง ที่ไม่ได้กำหนดรอบไว้ตั้งแต่เริ่มต้น แต่เป็นการกำหนดเงื่อนไขที่ทำให้ทำคำสั่งเดิมซ้ำๆ トラบเท่าที่เงื่อนไขเป็นจริงๆ ซึ่งโปรแกรมจะทำการตรวจสอบเงื่อนไขก่อนที่จะเข้าไปทำงานใน block คำสั่ง หากโปรแกรมพบว่าเงื่อนไขเป็นเท็จก็จะออกจากการทำงานแบบวนรอบนี้

รูปแบบคำสั่ง

```
while เงื่อนไข
    คำสั่ง.....
    การเพิ่มค่า...
End while
```

ตัวอย่าง

เป็นลักษณะเหมือนกับคำสั่ง for นะครับ โดยจะทำการตรวจสอบเงื่อนไขก่อนที่จะทำงาน ดังนั้น ลองสร้างหน้าจอโปรแกรมดังตัวอย่างคำสั่ง for และเขียนโค้ดดังตัวอย่าง

```

] Private Sub Button2_Click(ByVal sender As System.Object, ByVal
  Dim i As Integer
  While i < 11
    ListBox1.Items.Add("ประเทศไทย รอบที่ : " & i)
    i = i + 1
  End While
End Sub
-End Class

```

อธิบายคำสั่ง

Dim i As Integer เป็นการประกาศตัวแปร i เพื่อใช้เป็นตัวนับรอบ

While i < 11 เมื่อ i มีค่าน้อยกว่า 11 ก็ให้ทำงาน

ListBox1.Items.Add("ประเทศไทย รอบที่ : " & i) เป็นคำสั่งที่ให้เพิ่มคำว่าประเทศไทยเข้าไปใน Listbox โดยให้แสดงรอบด้วย

i=i+1 เพิ่มค่า i เพื่อให้จำนวนรอบเพิ่มขึ้นทีละ 1

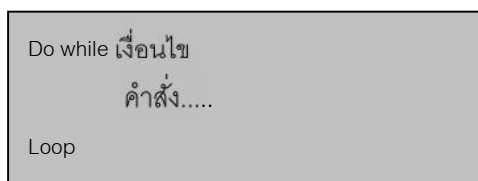
End while จบคำสั่ง while

สำหรับคำสั่ง while นี้อาจจะประยุกต์ใช้กับฐานข้อมูลได้เช่นกัน โดยอาจจะใช้สำหรับการแสดงข้อมูลในฐานข้อมูล เช่น ให้อ่านข้อมูลจนกว่าจะหมด เป็นต้น

คำสั่ง do...while

เป็นคำสั่งวนรอบอีกคำสั่งหนึ่ง ที่ไม่ได้กำหนดรอบไว้ตั้งแต่เริ่มต้น แต่เป็นการกำหนดเงื่อนไขที่ทำให้คำสั่งเดิมซ้ำๆ トラบเท่าที่เงื่อนไขเป็นจริงๆ

รูปแบบคำสั่ง



ลองเขียนคำสั่งดังตัวอย่างดังนี้ แล้วลองรันโปรแกรมดูนะครับ

```

Dim i As Integer
Do While i < 11
  ListBox1.Items.Add("ประเทศไทย รอบที่ : " & i)
  i = i + 1
Loop

```

บทที่ 3

การโปรแกรมเชิงวัตถุ

Object Oriented Programming

เป็นแนวการเขียนโปรแกรมที่เป็นการแก้ปัญหาแบบมองเป็นวัตถุ ซึ่งวัตถุเองก็จะประกอบด้วยส่วนหลักๆ 2 ส่วนได้แก่ คุณสมบัติและหน้าที่ และในโปรแกรมหนึ่งๆ มีหลายวัตถุก็จะสามารถเรียกใช้ สืบทอดหรือเข้าถึงวัตถุใดๆ ได้ตามต้องการ (ขึ้นกับข้อกำหนดและวิธีการ)

Class

เป็นกลุ่มของ Object ที่ประกอบด้วย 2 ส่วนคือ

- คุณสมบัติ (Attribute)
- หน้าที่ (Method)

ตัวอย่าง Class

Class คน

Attribute

- ส่วนสูง
- น้ำหนัก
- แขน
- ขา
- สีผิว
- อายุ

Method

- กิน
- เดิน
- นอนหลับ
- ฯลฯ

Class คอมพิวเตอร์

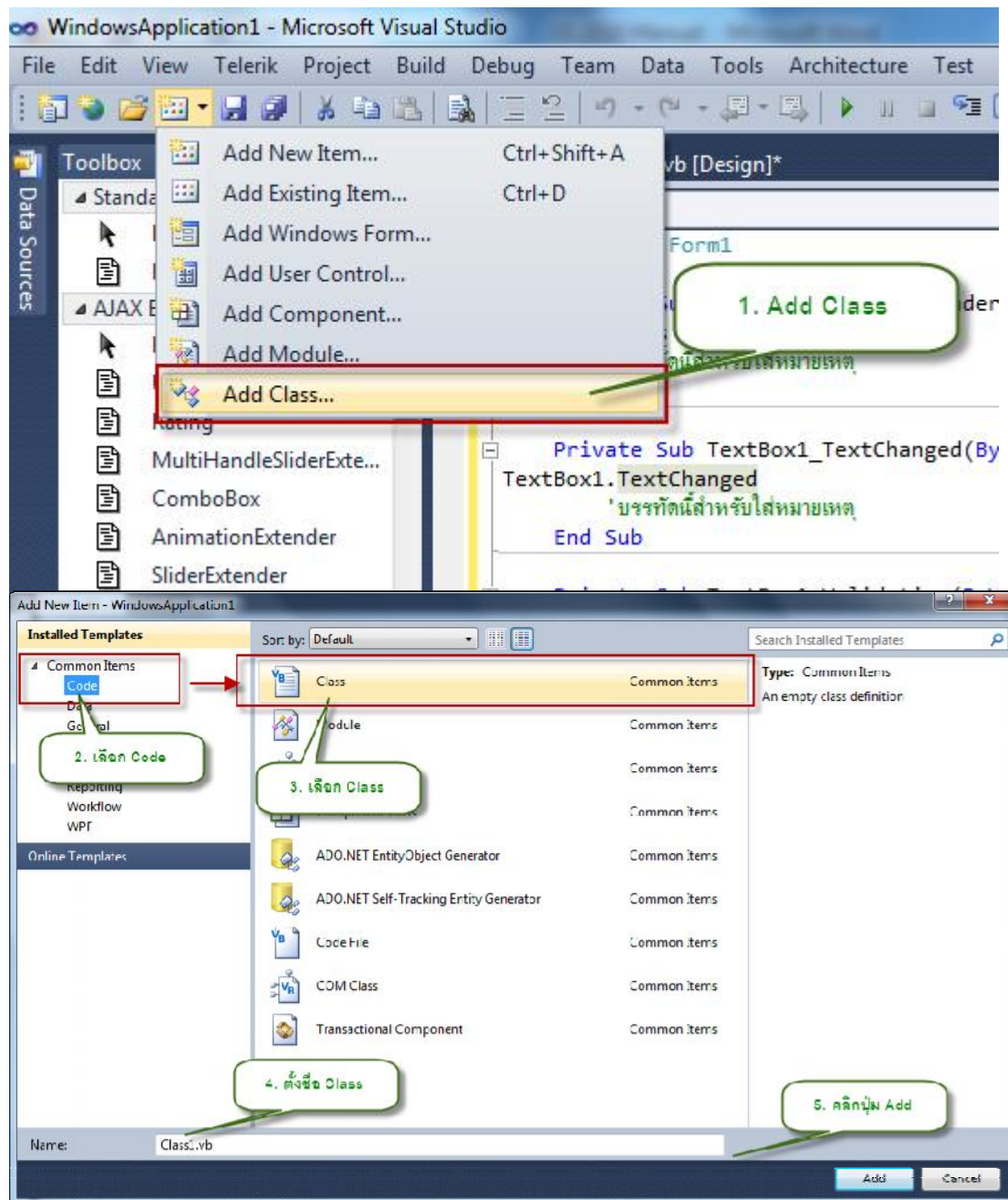
Attribute

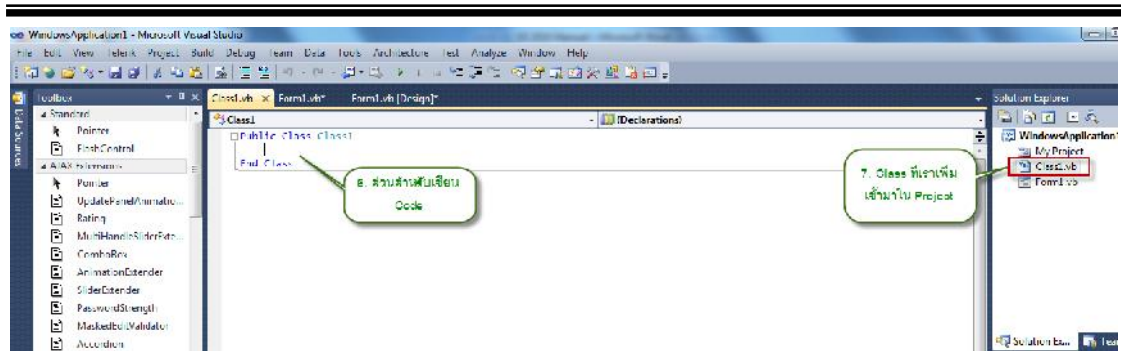
- CPU
- RAM
- VGA

- ฯลฯ
- Method
 - คำนวณเลขได้
 - เก็บข้อมูล
 - แสดงผล
 - ฯลฯ

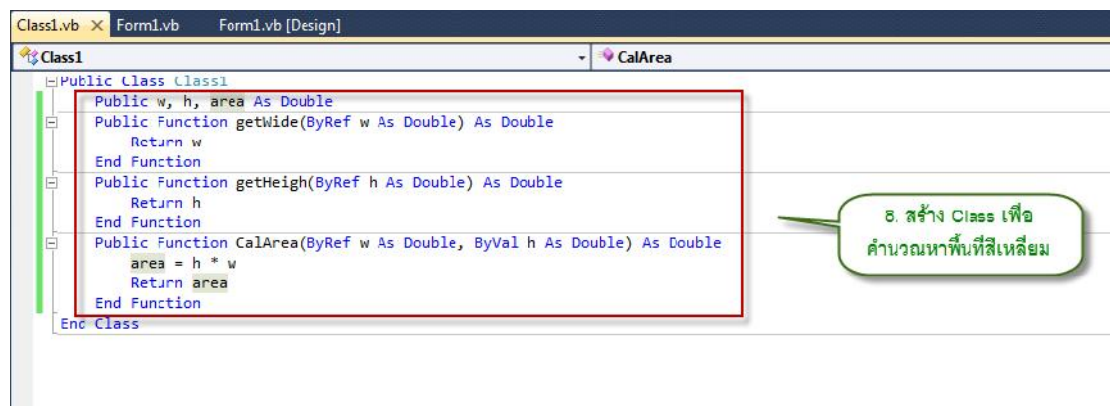
ในทางโปรแกรมไม่สามารถดำเนินการกับ Class ตรงๆ ได้ต้องดำเนินการผ่าน Object

การเขียน Class โดยใช้ VB





ต่อไปเราก็ต้องเขียนโค้ดเพื่อสร้าง Class ซึ่งจะขอยกตัวอย่างเป็น Class รูปสี่เหลี่ยมมุมมนๆ ซึ่งประกอบด้วย Attribute (กว้าง ยาว พื้นที่) และมี Method (รับค่าความกว้าง รับค่าความยาว คำนวณพื้นที่) ซึ่งจะได้โค้ดดังนี้

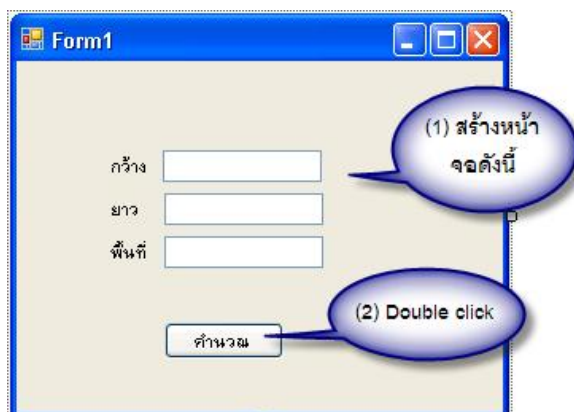


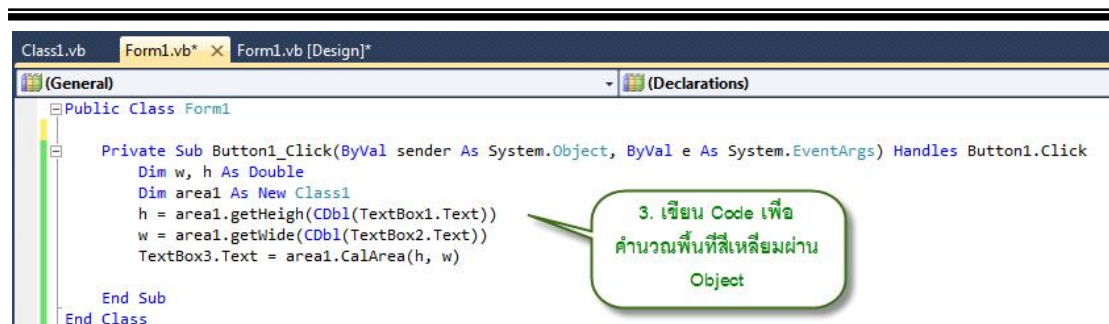
Object

เป็นตัวแทนของ Class ซึ่งมีรูปแบบการสร้าง ดังนี้

```
Dim ชื่อ Object As New ชื่อ Class
```

การสร้าง Object และการเรียกใช้ด้วย VB





Visibility

หมายถึงขอบเขตของการเข้าถึง ไม่ว่าจะเป็นตัวแปร หรือ Method ซึ่ง Visibility นี้จะเป็นสิ่งสำคัญมากในการเขียนโปรแกรมแบบเชิงวัตถุ เพราะคุณสมบัติที่เรียกว่า Encapsulation นั้นเอง โดยในเรื่องของ Visibility สามารถแบ่งออกเป็น

1. Private หมายถึง ถ้าประกาศ Visibility นี้ไว้หน้าตัวแปรหรือ Method ก็จะมีเพียงใน Class เดียวกันเท่านั้นที่สามารถเข้าถึง ตัวแปร และ Method นี้ได้
2. Protected หมายถึง ถ้าประกาศ Visibility นี้ไว้หน้าตัวแปรหรือ Method ก็จะมีเพียง Class เดียวกันและ Class ที่สืบทอดต่อไป ที่จะสามารถเข้าถึงได้
3. Public หมายถึง ถ้าประกาศ Visibility นี้ไว้หน้าตัวแปรหรือ Method ไม่ว่า Class ใดๆ ก็สามารถเข้าถึงได้

ตัวอย่าง 1

Private name As String หมายถึง ตัวแปร name เป็นชนิด String ที่สามารถใช้ได้เฉพาะใน Class ที่มันอยู่เท่านั้น

หมายเหตุ แต่โดยพื้นฐานแล้ว ใน Visual VB.NET ไม่จำเป็นต้องประกาศ Visibility เป็น Private เนื่องจากโปรแกรมจัดเป็นค่าเบื้องต้นให้อัตโนมัติ เช่น

```
Dim name As String
```

คำสั่งนี้จะมีค่าเดียวกันกับตัวอย่างด้านบน

ตัวอย่าง 2

```
Private void Add(int x, int y)
```

คำสั่ง.....

คำสั่งนี้เป็น Function ไม่ใช่ตัวแปร แต่ทว่าสิทธิการเข้าถึงก็เช่นเดียวกับตัวอย่างด้านบน

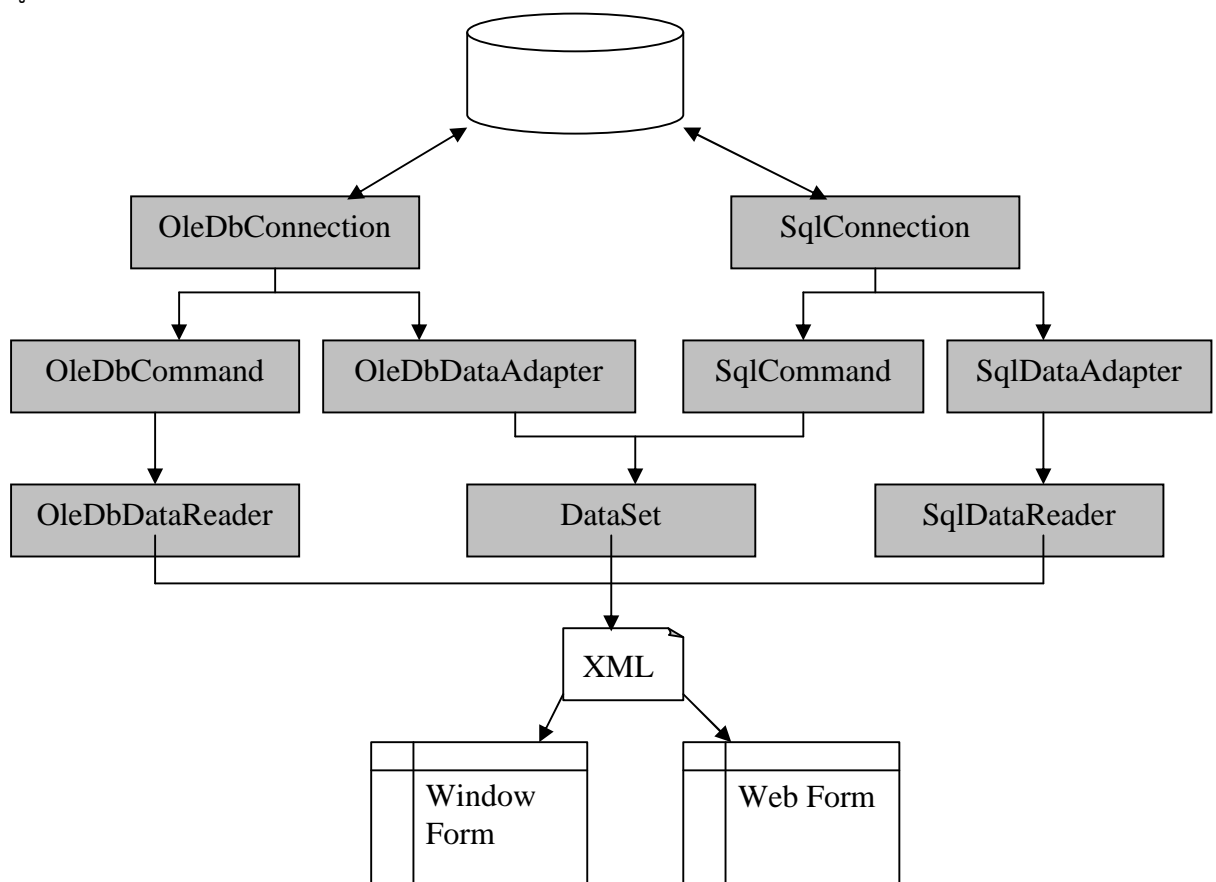
สำหรับ Visibility แบบอื่นๆ ก็เขียนคำสั่งคล้ายๆ กัน เพียงแต่เปลี่ยน Visibility เท่านั้นเอง

บทที่ 4

การเขียนโปรแกรมกับฐานข้อมูล

Database Programming

การเขียนโปรแกรมกับฐานข้อมูลใน Visual Studio .NET 2010 จะเรียกว่า ADO.NET มีโครงสร้างดังรูป



สำหรับการสร้างโปรแกรมเพื่อจัดการฐานข้อมูลสามารถทำได้ 2 แบบใหญ่ๆ ได้แก่

1. เขียนคำสั่งควบคุม
2. ใช้เครื่องมือสำหรับการควบคุมและจัดการฐานข้อมูล

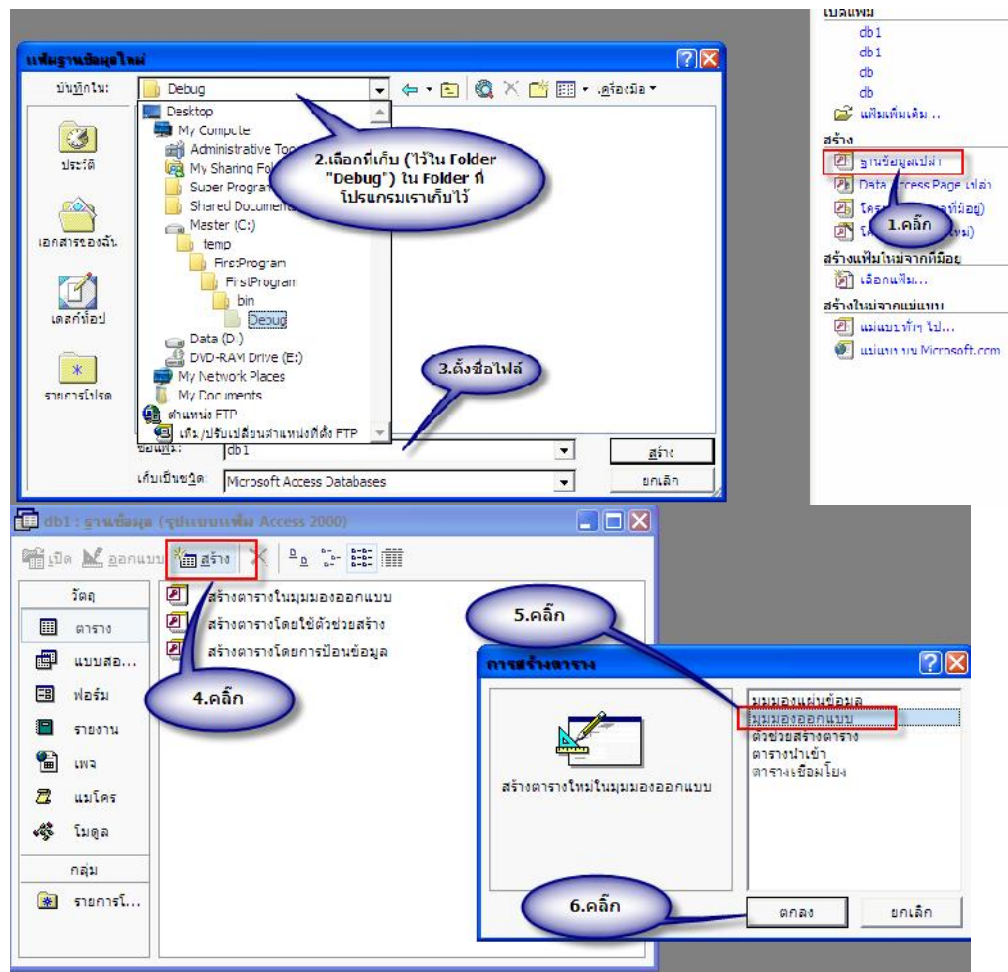
เขียนคำสั่งควบคุม

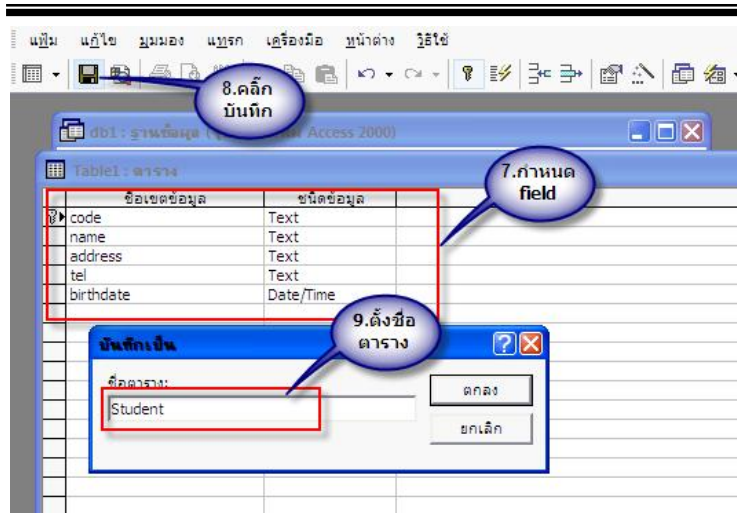
หลักการสร้างโปรแกรม

1. สร้างฐานข้อมูล (จากโปรแกรม Database อย่างใดอย่างหนึ่ง)
2. สร้าง Interface เพื่อติดต่อกับฐานข้อมูล
3. เขียนคำสั่ง

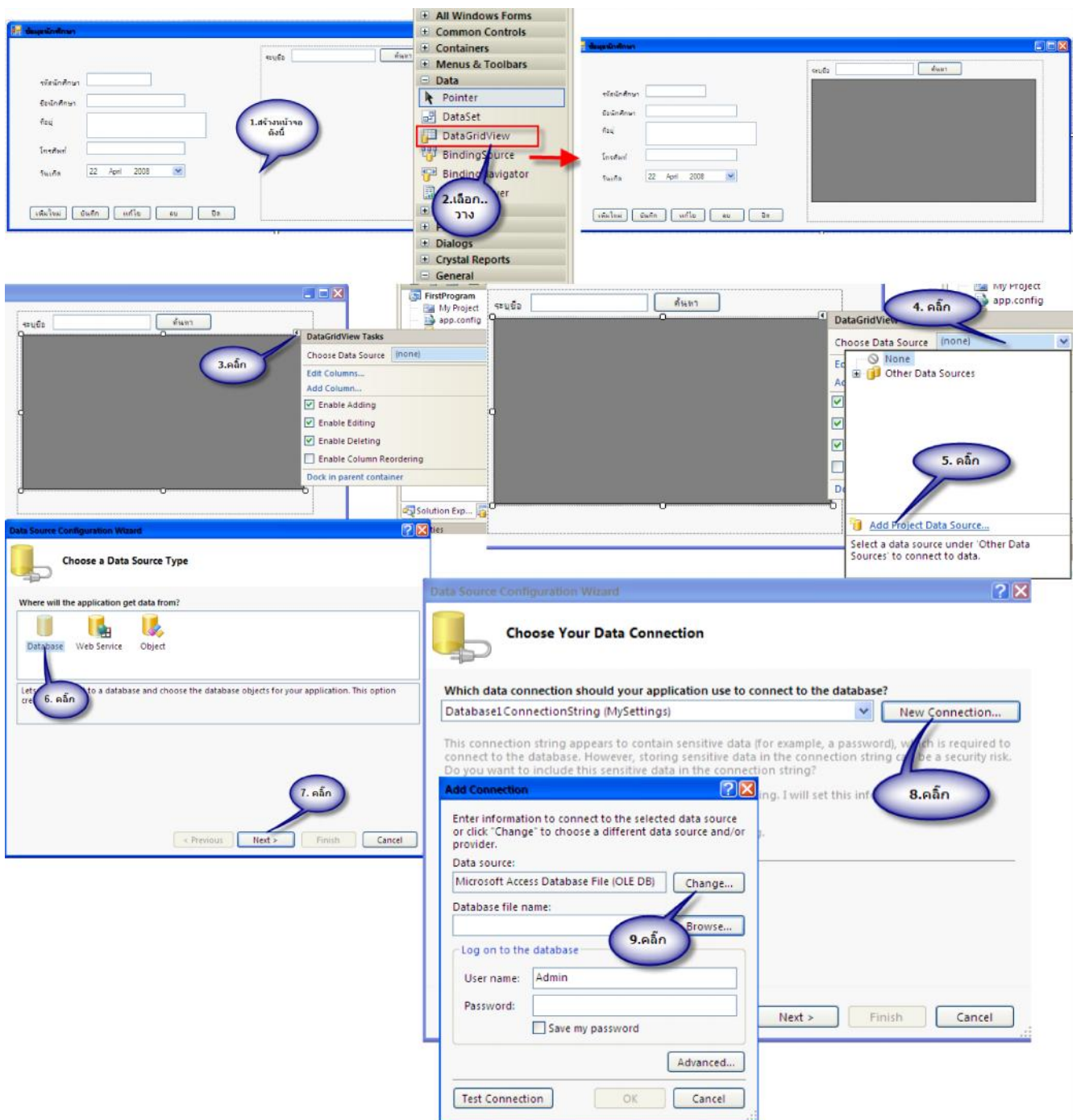
การสร้างฐานข้อมูล (MS Access)

เริ่มต้นด้วยการเปิดโปรแกรม Access ขึ้นมาแล้วสร้างตารางดังตัวอย่าง (ที่แสดงเป็น Access version 2003)





การสร้าง Interface เพื่อติดต่อกับฐานข้อมูล



10.คลิกเลือก

11.คลิก

12.คลิก

13.เลือก path และไฟล์

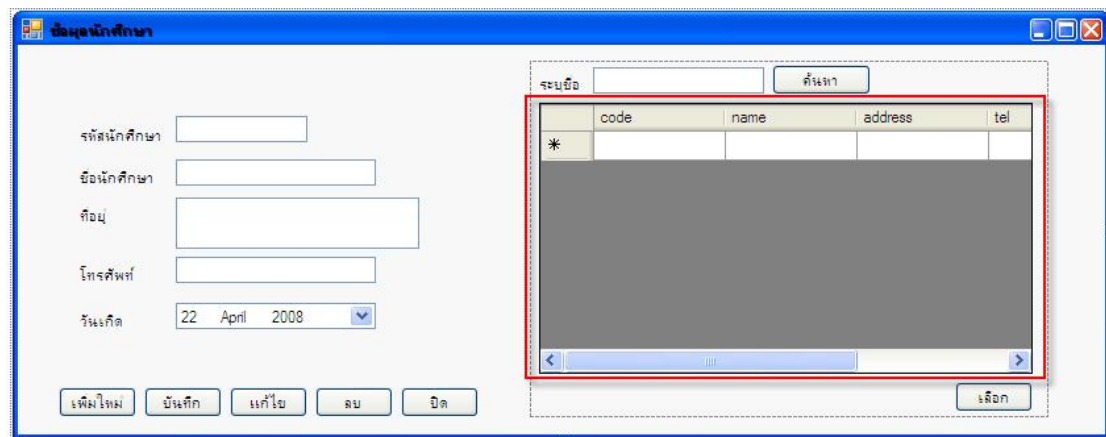
14.คลิก

15.คลิก

16.คลิก

17.คลิก

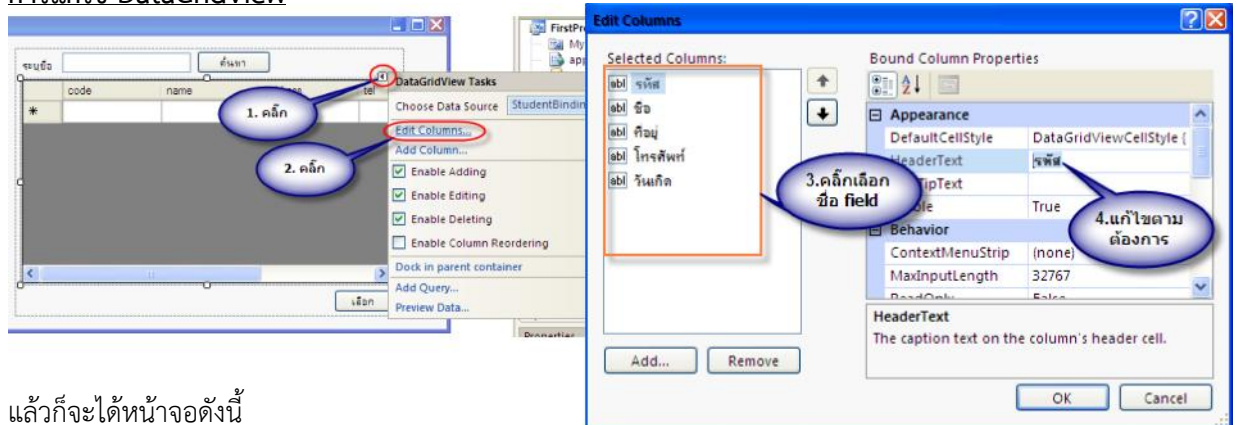
สุดท้ายแล้วก็จะได้นี้จอตงนี้ครับ



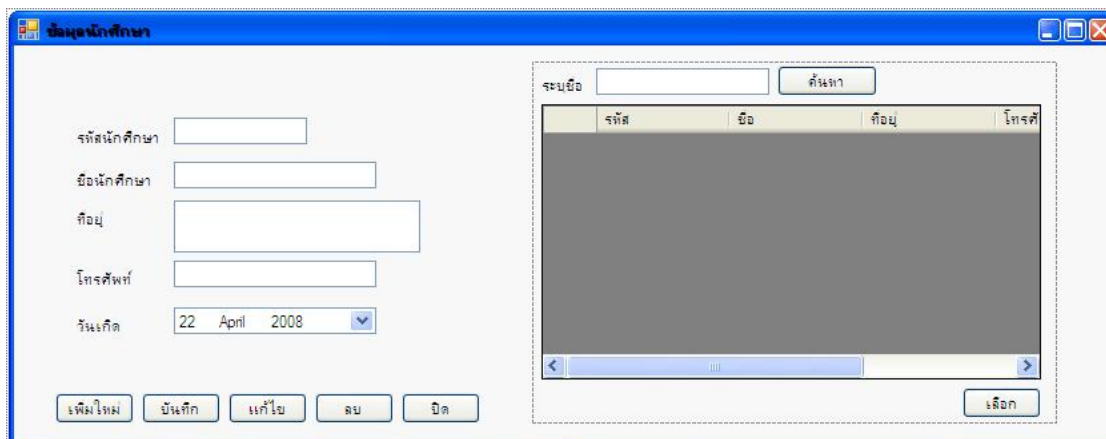
จะพบว่าชื่อคอลัมน์ของ DataGridView จะเป็นชื่อ Field ที่เราได้สร้างในฐานข้อมูลซึ่งเราก็สามารถแก้ไขให้เป็นภาษาไทยได้นะครับ ดังนี้

Data GridView

การแก้ไข DataGridView



แล้วก็จะได้น่าจดังนี้



การเขียนคำสั่งควบคุม

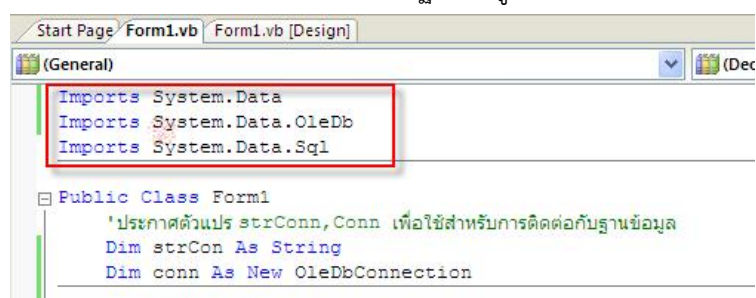
เมื่อได้ดังนี้แล้ว ต่อไปจะเป็นการเขียนคำสั่งเพื่อควบคุมการทำงานทั้งหมด โดยจากตัวอย่างในคู่มือชุดนี้จะมีการทำงานเป็นลำดับสำคัญๆ ดังนี้

- ปุ่ม “เพิ่มใหม่” ทำหน้าที่ล้างข้อความที่อยู่ใน textbox ทั้งหมดให้เป็นช่องว่างและให้ cursor ไปกระพริบอยู่ที่ TextBox ตัวแรก (รหัสนักศึกษา) เพื่อรอรับข้อมูลจากผู้ใช้งาน
- ปุ่ม “บันทึก” ทำหน้าที่บันทึกข้อมูลใหม่ลงไปในฐานะข้อมูล โดยเมื่อบันทึกข้อมูลเสร็จแล้วก็จะเรียกใช้คำสั่งการทำงานของปุ่ม “เพิ่มใหม่” เพื่อทำให้ TextBox เป็นช่องว่าง และจะปรับปรุงข้อมูลใน DataGridView เป็นข้อมูลที่เป็นปัจจุบัน
- ปุ่ม “แก้ไข” ทำหน้าที่บันทึกการแก้ไขข้อมูลเดิม โดยการแก้ไขนั้นหมายถึงการปรับปรุงข้อมูลที่มีอยู่เดิมให้มีการเปลี่ยนแปลงและจะต้องทำการค้นหาข้อมูลก่อน ซึ่งโดยตามหลักเกณฑ์แล้วไม่ควรแก้ไข field ที่เป็น primary key
- ปุ่ม “ลบ” ทำหน้าที่ลบข้อมูลออกจากฐานข้อมูล โดยจะต้องทำการค้นหาข้อมูลก่อน ซึ่งการลบก็ให้มี MessageBox ถามเพื่อยืนยันความแน่ใจในการลบ
- ปุ่ม “ค้นหา” ทำหน้าที่ค้นหาข้อมูล โดยผู้ใช้งานจะระบุชื่อนักศึกษาเข้าไปใน TextBox โดยโปรแกรมนั้นให้ระบุตัวอักษรตัวใดตัวหนึ่งก็ได้
- ปุ่ม “เลือก” ทำหน้าที่เอาข้อมูลที่อยู่ใน DataGridView ที่ผู้ใช้เลือกไปแสดงใน TextBox
- ปุ่ม “ปิด” ทำหน้าที่ในการปิดฟอร์ม

เขียนโค้ดได้ดังตัวอย่าง ตามลำดับดังนี้ครับ

Step 1

- เพิ่มคำสั่งที่ใช้สำหรับการติดต่อฐานข้อมูล ดังตัวอย่าง



ต้องเพิ่มคำสั่งดังกล่าวไว้ด้านบนสุดของหน้าต่างโค้ดเท่านั้นครับ

Step 2

- ประกาศตัวแปรที่จะใช้สำหรับการติดต่อกับฐานข้อมูล

```

Start Page Form1.vb Form1.vb [Design]
(General) (Dec)
Imports System.Data
Imports System.Data.OleDb
Imports System.Data.Sql

Public Class Form1
    'ประกาศตัวแปร strConn, Conn เพื่อใช้สำหรับการติดต่อกับฐานข้อมูล
    Dim strCon As String
    Dim conn As New OleDbConnection

```

Step 3

- เขียนคำสั่งเพื่อติดต่อกับฐานข้อมูล โดยกำหนดไว้ที่ Form_Load หมายถึงเมื่อ Form นั้นถูกเปิดขึ้นมาก็ให้ติดต่อกับฐานข้อมูล

```

Public Class Form1
    'ประกาศตัวแปร strConn, Conn เพื่อใช้สำหรับการติดต่อกับฐานข้อมูล
    Dim strCon As String
    Dim conn As New OleDbConnection

    Private Sub Form1_Load (ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Lo
        'สร้างการติดต่อกับฐานข้อมูลที่เรียกว่า Connection String
        strCon = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=|DataDirectory|\data\db1.mdb"
        'เชื่อมต่อกับฐานข้อมูล
        conn.ConnectionString = strCon
        'ตรวจสอบว่าเป็นอย่างไรยัง ถ้าเป็นแล้วให้ปิด แล้วค่อยเปิดใหม่
        If conn.State = ConnectionState.Open Then
            conn.Close()
        End If
        conn.Open()
        callGrid()
    End Sub

```

Step 4

- เขียนคำสั่งเพื่อดำเนินการกับปุ่มต่างๆ

ปุ่ม บันทึก

```

Private Sub Button2_Click (ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
    'ประกาศตัวแปร เก็บค่าวันที่เลือกคือวันที่เลือก
    Dim sqlFind, sqlAdd As String
    Dim todate As Date
    'ค้นหาว่าวันที่ระบุเข้าไปนั้นอยู่ในฐานข้อมูลหรือไม่
    sqlFind = "select * from student where code=" & TextBox1.Text & " 'ค้นหาสิ่งคำสั่ง sql
    Dim cmdFind As New OleDbCommand(sqlFind, conn)
    'กำหนดตัวแปร reader เพื่อให้อ่านข้อมูลจาก cmdFind
    Dim reader As OleDbDataReader = cmdFind.ExecuteReader()
    'ถ้ามีข้อมูลใน reader
    If reader.HasRows Then 'ถ้าพบข้อมูล
        MsgBox("รหัสยังติดซ้ำอยู่ใน", MsgBoxStyle.Exclamation, "VB")
    Else
        'คำนวณ todate เก็บวันที่
        todate = DateTimePicker1.Value
        'คำสั่งที่ส่งมาให้เป็นคำสั่ง SQL
        sqlAdd = "insert into student (code, name, address, sex, birthdate) values (" & TextBox1.Text & " , " &
        TextBox2.Text & " , " & TextBox3.Text & " , " & TextBox4.Text & " , " & todate & " )"
        'ประกาศตัวแปรที่เป็นคำสั่ง sqlCommand เพื่อประมวลผลคำสั่ง
        Dim cmdAdd As New OleDbCommand(sqlAdd, conn)
        'ประมวลผลคำสั่ง
        cmdAdd.ExecuteNonQuery()
        ClearTextBox()
        CallGrid()
    End If
End Sub

```

Function clrtxt()

```

End Sub
'===== function clrtxt เพื่อทำให้เป็นช่องว่าง =====
Private Sub clrtxt ()
    'กำหนดให้ Textbox เป็นช่องว่าง
    TextBox1.Text = ""
    TextBox2.Text = ""
    TextBox3.Text = ""
    TextBox4.Text = ""
    'กำหนดให้ Cursor ไปกระพริบอยู่ ณ TextBox
    TextBox1.Focus ()
End Sub
    
```

Function callGrid()

```

'===== function calGrid เพื่อปรับปรุงข้อมูลใน DataGridView ให้เป็นปัจจุบัน =====
Private Sub callGrid()
    Dim sqlC As String
    'ตัวแปร sqlC เก็บคำสั่ง เพื่อเรียกข้อมูลทุก record ทุก field ที่อยู่ในตาราง student
    sqlC = "select * from student"
    'ตัวแปร da ปรนมาผลคำสั่งด้วย DataAdapter
    Dim da As New OleDbDataAdapter(sqlC, conn)
    'ตัวแปร ds เป็น DataSet ที่เก็บการประมวลผลไว้
    Dim ds As New DataSet ()
    da.Fill (ds, "St")
    'กำหนดให้ข้อมูลให้กับ DataGridView
    DataGridView1.DataSource = ds
    DataGridView1.DataMember = "St"
End Sub
    
```

ปุ่ม “เพิ่มใหม่”

```

'===== ปุ่ม "เพิ่มใหม่" =====
Private Sub Button5_Click (ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button5.Click
    'เรียกใช้ function clrtxt เพื่อล้างข้อมูลใน textbox
    clrtxt ()
End Sub
    
```

ปุ่ม “ปิด”

```

'===== ปุ่ม "ปิด" =====
Private Sub Button4_Click (ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
    'ในคำถามว่าต้องการปิดหรือไม่ โดยถ้าคลิกที่ปุ่ม "Yes" ก็ปิดโปรแกรม
    If MsgBox ("Close ?", MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then
        Close ()
    End If
End Sub
    
```

ปุ่ม “แก้ไข”

```

'===== ปุ่ม "แก้ไข" =====
Private Sub Button1_Click (ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
    'ถ้า TextBox1 ไม่เป็นช่องว่าง
    If TextBox1.Text <> "" Then
        'ประกาศตัวแปรเพื่อเก็บคำสั่งในนามกรในโปรแกรม
        Dim sqlEdit As String
        sqlEdit = "update student set name=" & TextBox2.Text & ", address=" & TextBox2.Text & " & _
            " , del=" & TextBox3.Text & ", birthdate=" & DateTimePicker1.Value & " & _
            " where code=" & TextBox1.Text & ""
        Dim cmdEdit As New OleDbCommand (sqlEdit, conn)
        cmdEdit.ExecuteNonQuery ()
        clrtxt () 'เรียกใช้ function
        callGrid () 'เรียกใช้ Function
    End If
End Sub
    
```

ปุ่ม “ลบ”

```

'===== ปุ่ม "ลบ" =====
Private Sub Button3_Click (ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
    'ถ้า TextBox1 ไม่เป็นช่องว่าง
    If TextBox1.Text <> "" Then
        If MsgBox ("Delete ?" & TextBox2.Text, MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then
            Dim sqlDel As String
            sqlDel = "delete from student where code=" & TextBox1.Text & ""
            Dim cmdDel As New OleDbCommand (sqlDel, conn)
            cmdDel.ExecuteNonQuery ()
            clrtxt ()
            callGrid ()
        End If
    End If
End Sub
    
```

ปุ่ม “ค้นหา”

```

----- ปุ่ม "ค้นหา" -----
Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button6.Click
    Dim sqlF As String
    sqlF = "select * from student where name like " & "%" & TextBox5.Text & "%" & ""
    Dim cmdF As New OleDbCommand(sqlF, conn)
    Dim reader As OleDbDataReader = cmdF.ExecuteReader()
    If reader.HasRows Then
        Dim da As New OleDbDataAdapter(sqlF, conn)
        Dim ds As New DataSet()
        da.Fill(ds, "StF")
        DataGridView1.DataSource = ds
        DataGridView1.Parameters = "StF"
    Else
        MsgBox("ไม่พบข้อมูล", MsgBoxStyle.Exclamation, "VB")
    End If
End Sub
Public Shared Const Explorer As Microsoft.VisualBasic MsgBoxStyle = 48

```

ปุ่ม “เลือก”

```

===== ปุ่ม "เลือก" =====
Private Sub Button7_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button7.Click
    'กำหนดให้ ข้อมูลใน Field "code" มาแสดงที่ TextBox1
    TextBox1.Text = DataGridView1.CurrentCell.Value.ToString
End Sub

```

Event “TextBox1_Change “

```

----- เปลี่ยนข้อมูลใน TextBox1 ให้เปลี่ยนแปลงให้ทันทุกขณะแสดงข้อมูล -----
Private Sub TextBox1_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles TextBox1.TextChanged
    Dim sqlF As String
    'ค้นหารหัสศึกษาจาก TextBox1
    sqlF = "select * from student where code = " & TextBox1.Text & ""
    Dim cmdF As New OleDbCommand(sqlF, conn)
    Dim reader As OleDbDataReader = cmdF.ExecuteReader()
    If reader.HasRows Then 'ถ้าพบข้อมูล
        While reader.Read() 'อ่านทีละแถว
            TextBox2.Text = reader("name").ToString 'ให้พำข้อมูลที่อยู่ในฐานข้อมูลมาแสดงที่ TextBox
            TextBox3.Text = reader("address").ToString
            TextBox4.Text = reader("tbl").ToString
            DateTimePicker1.Value = reader("dateofdate")
        End While
    End If
End Sub

```

*** การเขียนโปรแกรมเพื่อจัดการฐานข้อมูลใน Visual Studio 2010 มีมากมายหลายวิธี และในคู่มือฉบับนี้ก็เป็นเพียงแค่วิธีหนึ่งเท่านั้น และคงเป็นรูปแบบที่ง่ายต่อการทำความเข้าใจ มีรูปแบบที่ง่าย ลองศึกษาและลองปฏิบัติดูแล้วกันนะครับ

*** คำสั่งที่เกี่ยวข้องกับภาษา SQL ให้ไปดูรายละเอียดในหัวข้อเรื่อง ภาษา SQL ในส่วนท้ายของคู่มือเล่มนี้นะครับ

บทที่ 5

การสร้าง Report, Menu bar และ Tool bar

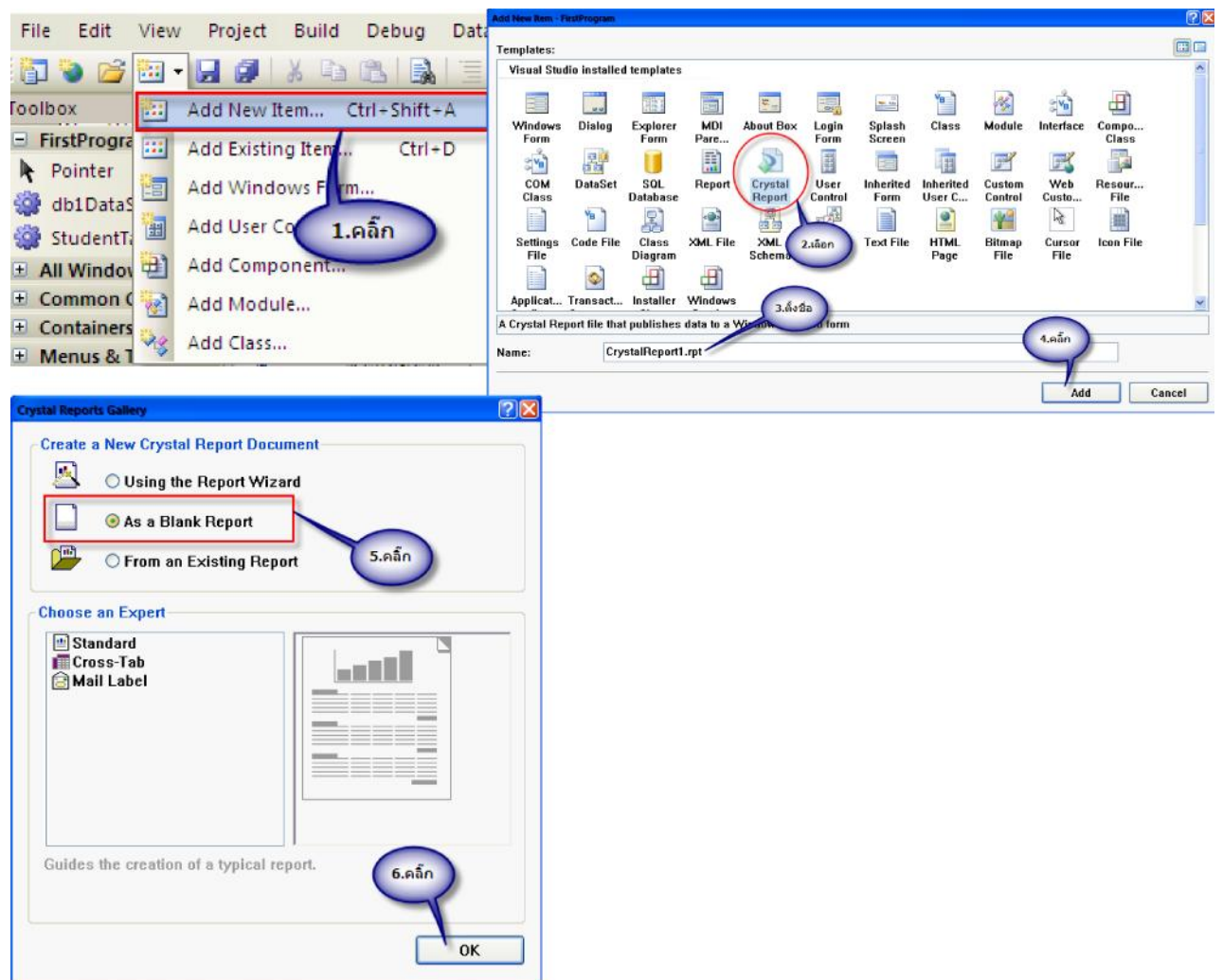
การทำรายงานด้วย Crystal Report

การทำรายงานเป็นส่วนสำคัญในการทำโปรแกรม เพราะว่าข้อมูลทั้งหมดที่อยู่ในโปรแกรม ที่เกิดจากการประมวลผลต่างๆ ในโปรแกรม จะนำมาใช้ได้หรือไม่ได้ก็อยู่ที่รายงานนี้แหละครับ สำหรับ Visual Studio 2010 จะบรรจุ Crystal Report มาด้วยอยู่แล้ว ดังนั้นเมื่อลงโปรแกรม Visual Studio 2010 ก็จะได้ Crystal Report มาด้วย

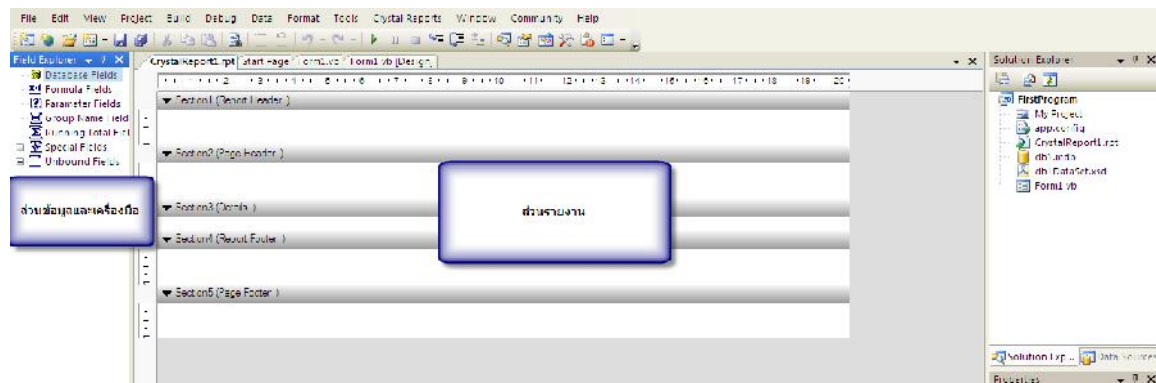
Crystal Report เป็นโปรแกรมที่ใช้จัดทำรายงานที่มีประสิทธิภาพมากและมีรายละเอียดค่อนข้างมาก สำหรับในคู่มือฉบับนี้จะแนะนำเพียงการสร้างรายงานเบื้องต้นเท่านั้น ลองศึกษาดูนะครับ

Step 1

- สร้างไฟล์รายงาน



จากนั้นจะมี Crystal Report ปรากฏขึ้นมาในโปรแกรมเราดังนี้



สำหรับส่วนรายงานมีส่วนประกอบดังนี้

Report Header = ใช้กำหนดหัวรายงาน จะแสดงในหน้าแรกของรายงานเท่านั้น

Page Header = ใช้กำหนด หัวกระดาษ ซึ่งจะมีในทุกหน้า

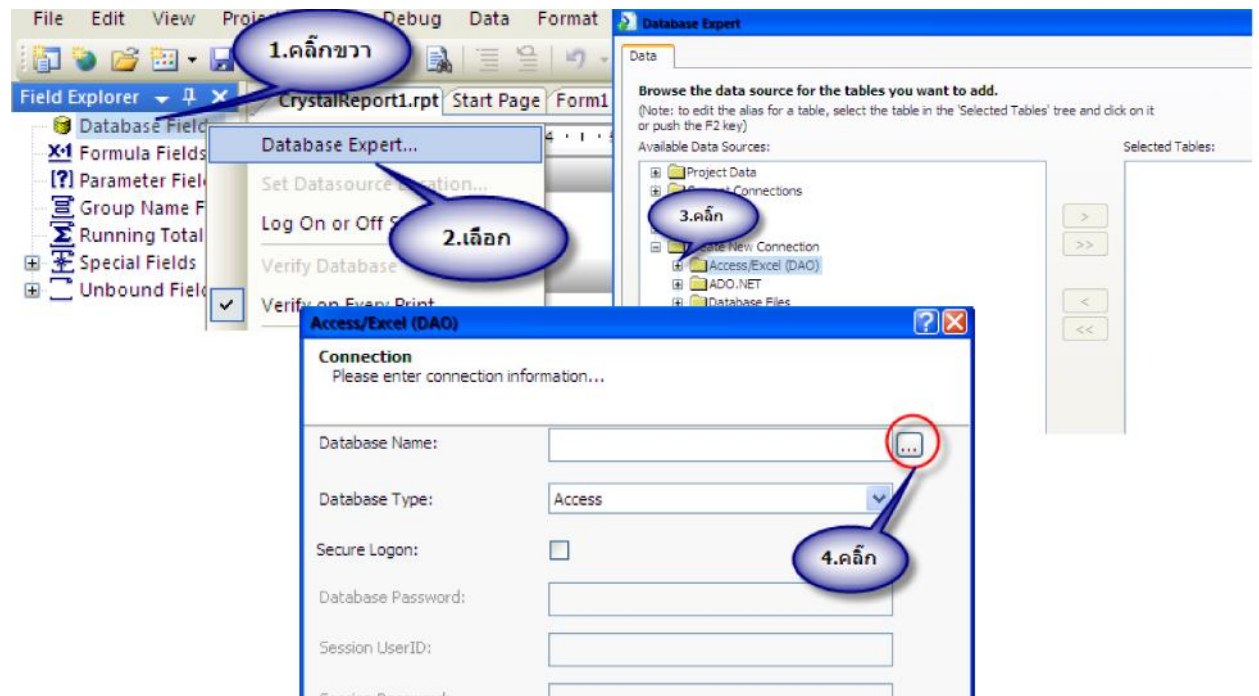
Details = ใช้กำหนดส่วนแสดงรายละเอียดของรายงาน หรือส่วนแสดงข้อมูลนั่นเอง

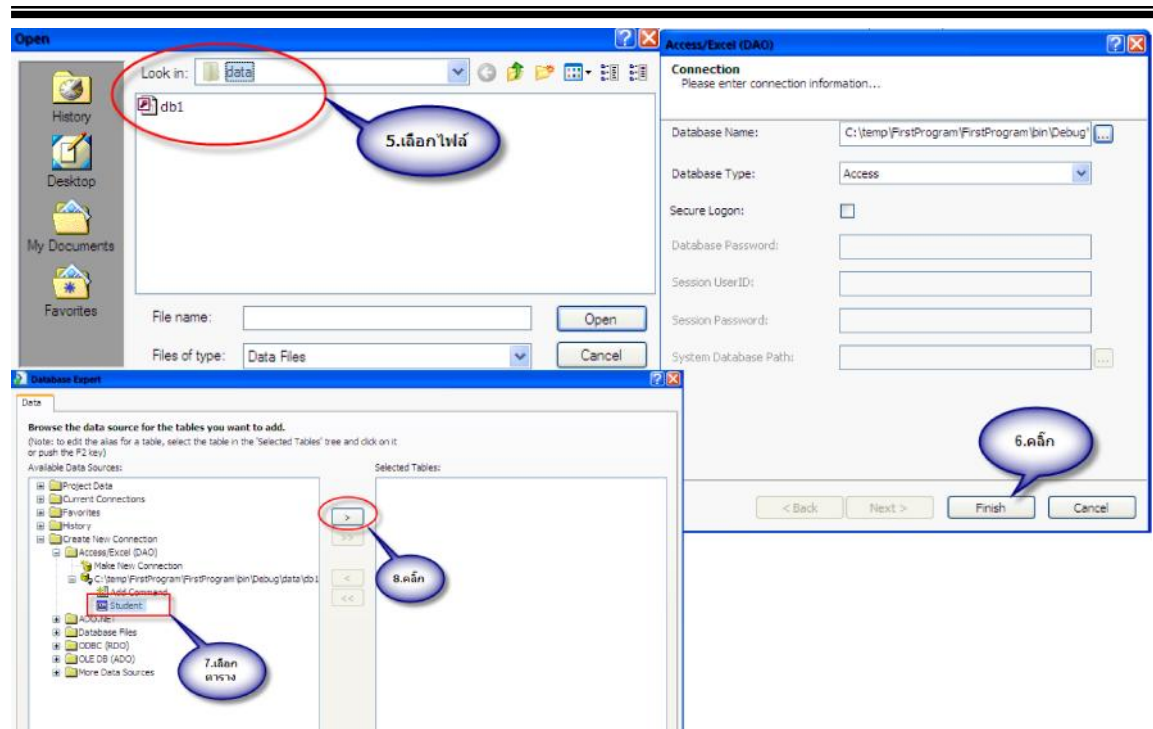
Report Footer = ใช้กำหนดส่วนท้ายกระดาษ

Page Footer = ใช้กำหนดส่วนท้ายรายงาน จะปรากฏเป็นส่วนสุดท้ายเมื่อแสดงรายงานเสร็จสิ้นแล้ว

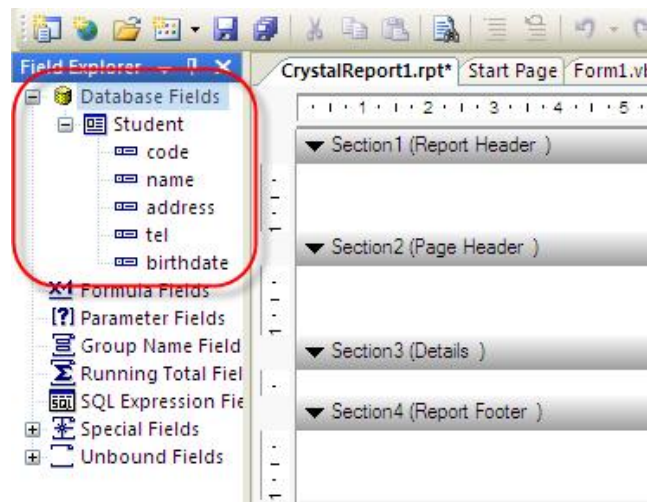
Step 2

- ติดต่อฐานข้อมูล



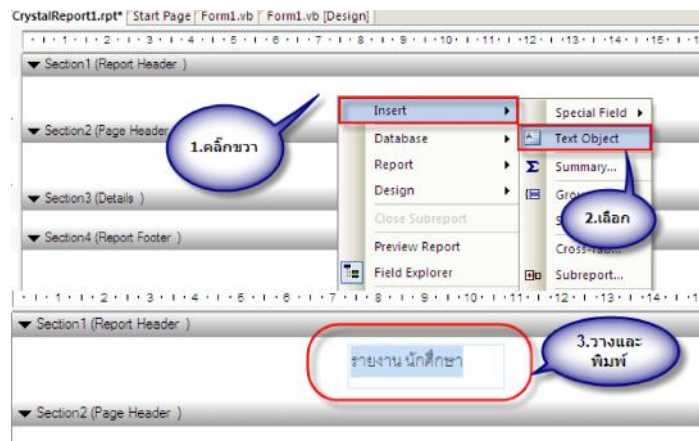


หลังจากนั้นโปรแกรมจะกลับมาที่หน้าจอออกแบบ ซึ่งเราก็จะพบว่าได้ติดต่อฐานข้อมูลเรียบร้อยแล้ว โดยสังเกตุได้ดังรูป

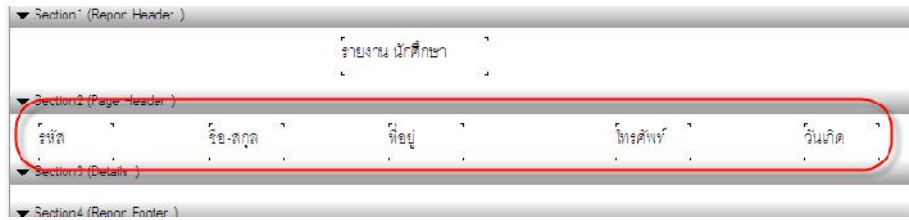


Step 3

- ออกแบบหัวรายงาน

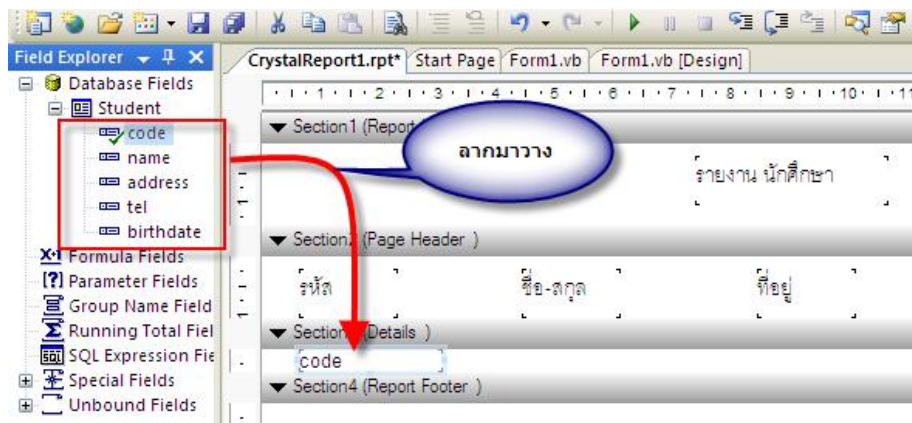


- ออกแบบหัวคอลัมน์รายงาน ให้ทำตามขั้นตอนเดิม หรือ คัดลอก Text Object จากตัวเดิมก็ได้ โดยนำมาวางไว้ในส่วนของ Page Header สุดท้ายก็จะได้ตามตัวอย่างดังนี้

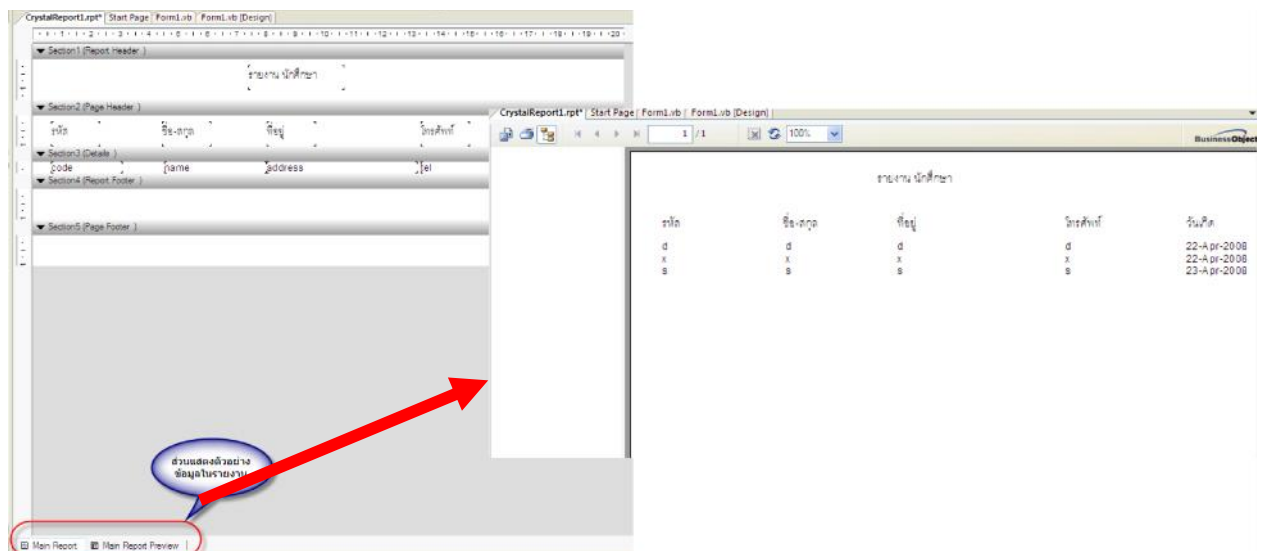


Step 4

- ใส่ field ข้อมูล เป็นส่วนที่สำคัญ เพราะข้อมูลจะแสดงหรือไม่แสดงก็อยู่ที่ส่วนนี้ หลักการง่ายๆ ก็เพียงแค่ลาก field มาวางไว้ในส่วน Details ในรายงานแค่นั้นเอง

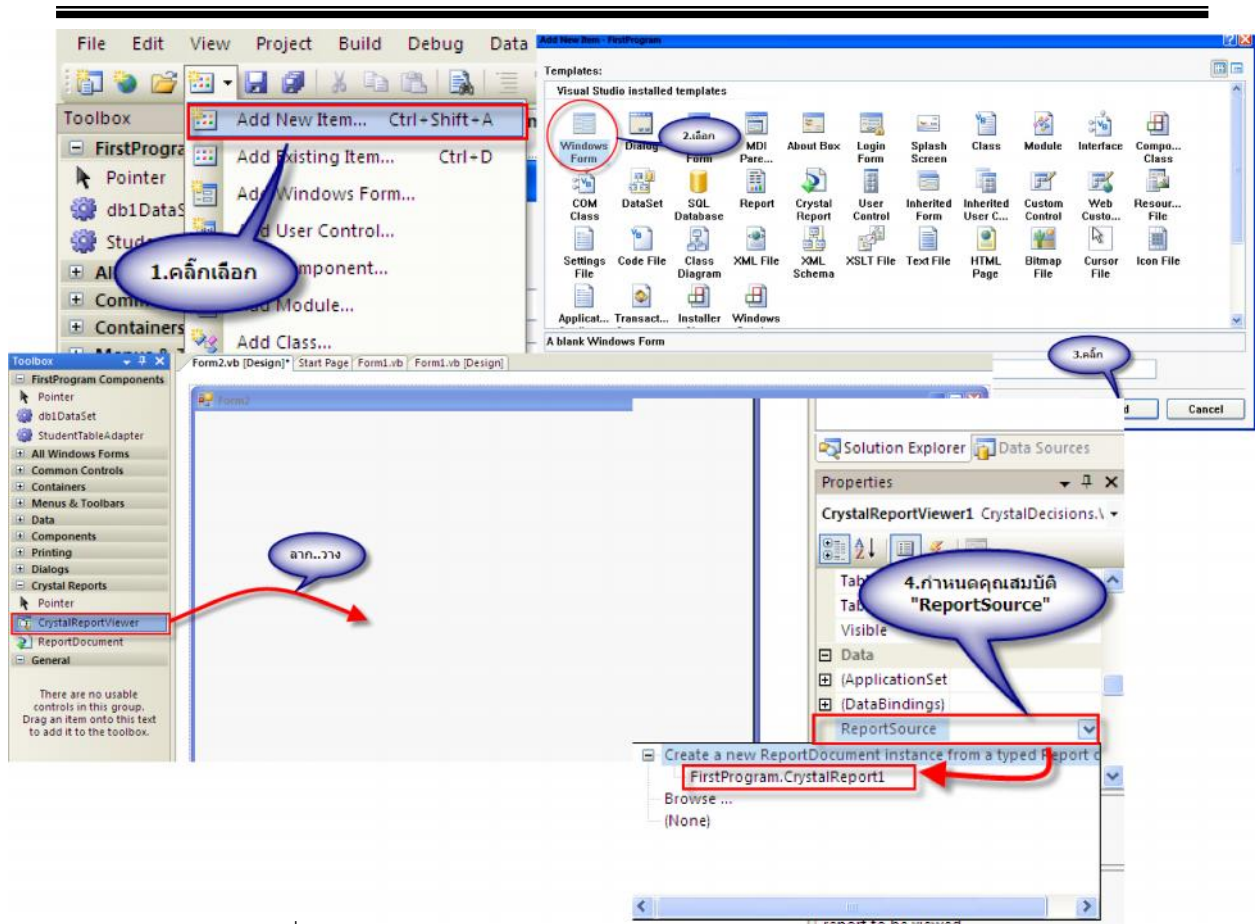


สุดท้ายที่ผ่านการวาง field และตกแต่งเรียบร้อยแล้ว ซึ่งก็จะสามารถแสดงตัวอย่างข้อมูลได้ ดังตัวอย่างหน้าจอ

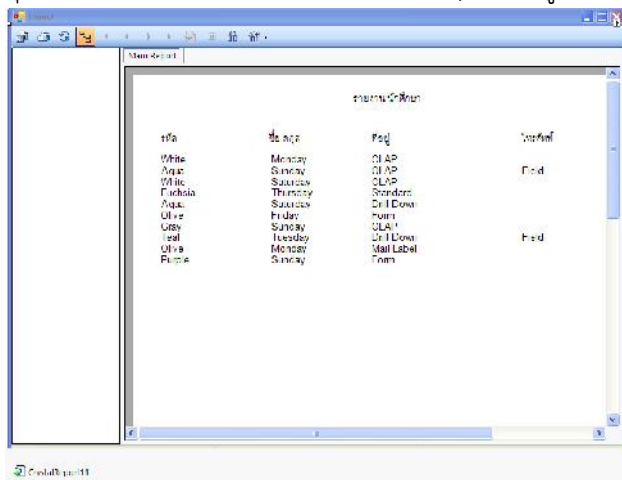


Step 5

- การนำรายงานเข้ามาใช้บน Form



สุดท้ายก็จะได้ Form ที่จะใช้สำหรับเป็น Report ดังรูป



นี่เป็นตัวอย่างการสร้างรายงานด้วย Crystal Report อย่างง่าย ผู้ที่สนใจก็สามารถศึกษาเพิ่มเติมได้นะครับ
โดย Crystal Report มีความสามารถมากมายและมีรายละเอียดเยอะ ลองใช้ดูครับแล้วจะชอบ

การทำ Menu bar และ Tool bar

เป็นมาตรฐานของการเขียนโปรแกรมแบบ Windows Application ซึ่งจะต้องมีทั้งเมนูและทูลบาร์ ลองศึกษาดูนะครับ

Step 1

- การทำฟอร์มแม่ (MDI Form)

1.คลิกเลือก

2.กำหนดคุณสมบัติ

3.กำหนดคุณสมบัติ

Step 2

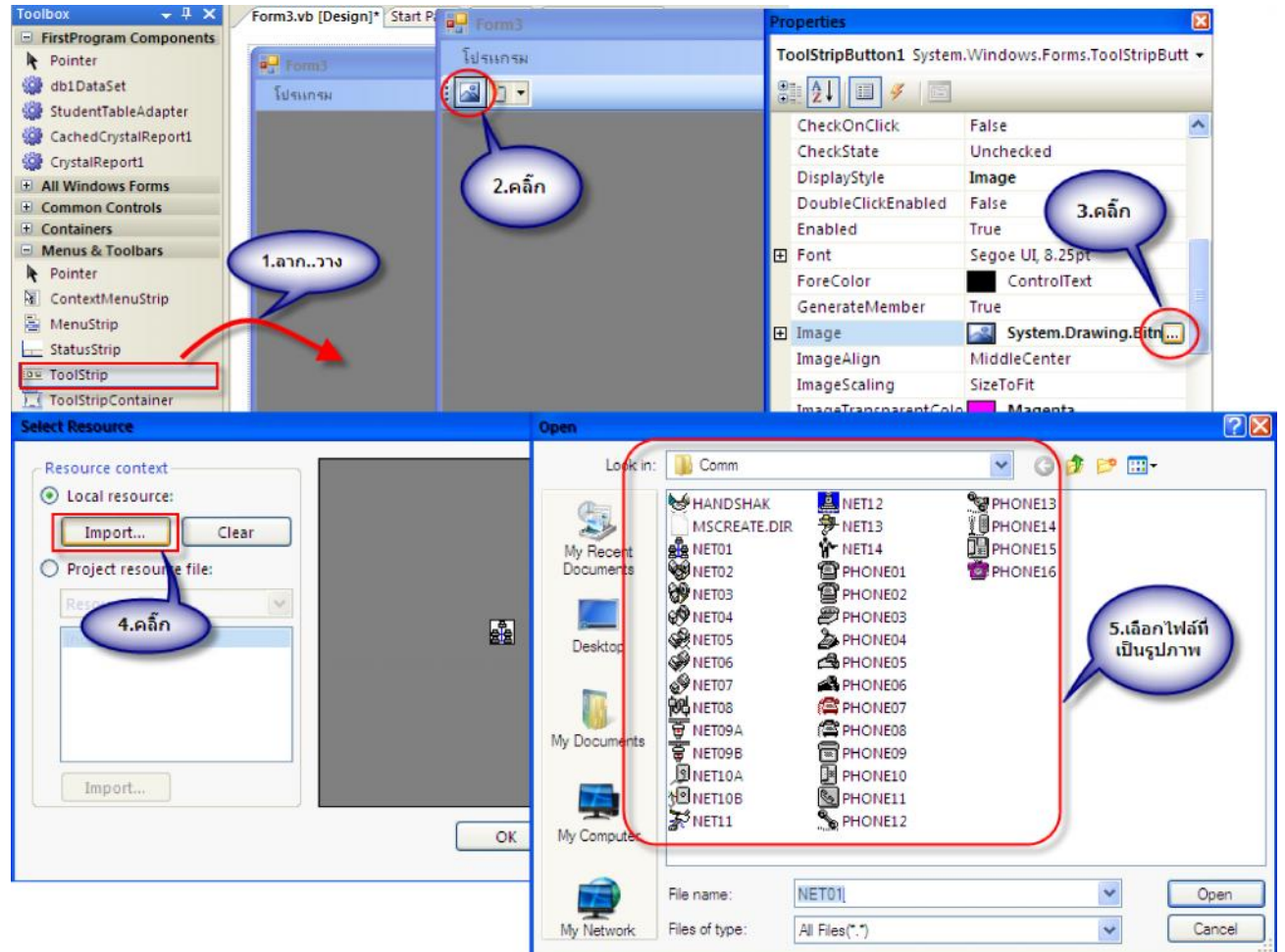
- การทำเมนู

1.ลากวาง

2.คลิกแล้วพิมพ์

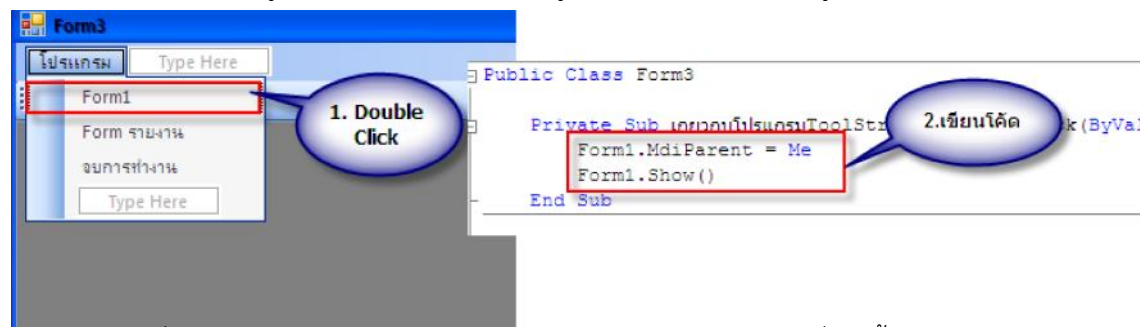
Step 3

- การทำทูลบาร์



Step 4

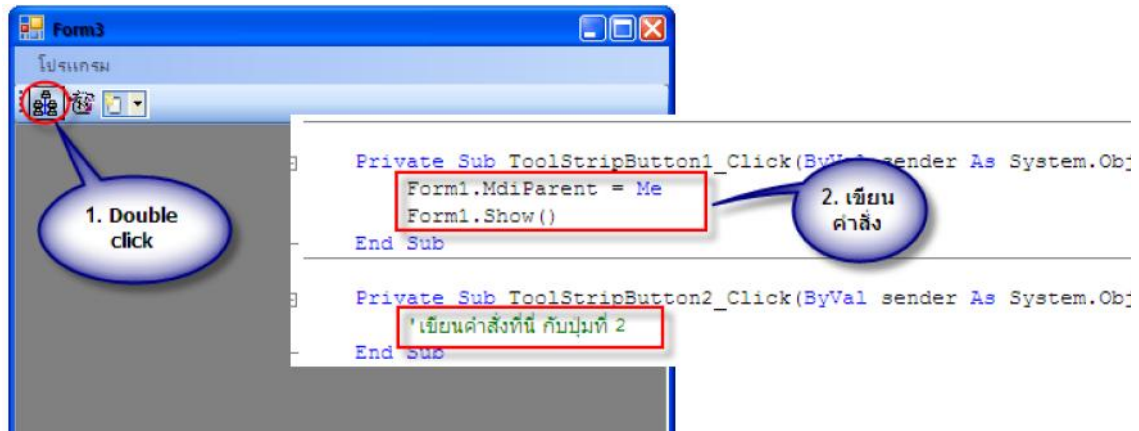
- เขียนโค้ดกับเมนูบาร์ ต้องการเขียนกับเมนูใดก็ double click ที่เมนูนั้น



สำหรับเมนูอื่นๆ ก็ทำเช่นเดียวกัน อยากรให้เมนูไหนทำงานอะไร ก็เขียนโค้ดที่เมื่อนั้นครับ

Step 5

- เขียนโค้ดกับทูลบาร์ คล้ายกับเมนูบาร์ หากต้องการเขียนโค้ดกับปุ่มไหนก็ double click ที่ปุ่มนั้นได้เลย จากนั้นก็เขียนคำสั่งควบคุมได้ตามต้องการ



```
Private Sub ToolStripButton1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ToolStripButton1.Click
    Form1.MdiParent = Me
    Form1.Show()
End Sub

Private Sub ToolStripButton2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ToolStripButton2.Click
    'เขียนคำสั่งที่นี่ กับปุ่มที่ 2
End Sub
```


บทที่ 6

ภาษา SQL

ภาษา SQL

ภาษา SQL เป็นภาษาที่ใช้สำหรับจัดการฐานข้อมูล โดยภาษาที่ใช้เขียนโปรแกรมที่มีในปัจจุบันจะใช้ภาษา SQL นี้เพื่อจัดการข้อมูลทั้งนั้น โดยสามารถนำภาษา SQL ใช้ร่วมกับ Compiler ได้ทุกภาษา ดังนั้นหากมีความรู้ความเข้าใจในภาษา SQL แล้วก็จะสามารถพัฒนาโปรแกรมที่เกี่ยวกับฐานข้อมูลได้อย่างมีประสิทธิภาพ

สำหรับเนื้อหาในส่วนนี้ ได้นำเสนอคำสั่งพื้นฐานของภาษา SQL ที่จะสามารถนำไปใช้ได้ในการเขียนโปรแกรม ไม่ได้แนะนำเสนอทั้งหมด ลองศึกษาดูกันนะครับ

คำสั่งสำหรับเรียกข้อมูล

มีรูปแบบดังนี้

รูปแบบที่ 1 `Select field1,field2,fieldn.. from Table`

เป็นคำสั่งสำหรับเรียกข้อมูลจากฐานข้อมูล โดย Select คือคำสั่ง

Field นั้น เป็นชื่อ Field ที่เรามีในฐานข้อมูล โดยสามารถเลือก Field เพื่อแสดงข้อมูลได้ตามที่ฐานข้อมูลมี

From เป็นคำสั่ง ที่บ่งบอกว่า เราเลือก Field จากตารางข้อมูลไหน

Table คือ ชื่อตารางฐานข้อมูลที่เราต้องการ

ตัวอย่าง

สมมติว่าในตารางฐานข้อมูลชื่อ Employee ที่เก็บข้อมูลเกี่ยวกับลูกจ้างในบริษัท โดยมีตารางข้อมูลดังนี้

ID	NAME	ADDRESS	TEL	SALARY
AC001	น.ส.อนงค์ ใจดี	123 ถ.ทหาร อ.เมือง จ.อุดรธานี	042-254615	7,500
MK001	นายถาวร วรดี	145 ถ.อุดร-สกล อ.เมือง จ.อุดรธานี	09-7821546	8,200
MK002	น.ส.ลักษณ อาทร	321/38 อ.เมือง จ.อุดรธานี	042-236457	8,500
BC001	นายมงคล ไปได้	145 หมู่ 2 ต.บ้านเลื่อม อ.เมือง จ.อุดรธานี	01-7121288	8,600
BC002	นายจงรัก รักดี	145 ถ.พัฒนาการ อ.เมือง จ.หนองคาย	042-248665	8,900
BC003	น.ส.ศิริ ศิริลักษณ์	146 ถ.ทหาร อ.เมือง จ.อุดรธานี	042-246658	11,000

Select id,name from employee ซึ่งก็จะได้รายงานดังนี้

ID	NAME
AC001	น.ส.อนงค์ ใจดี
MK001	นายถาวร วรดี
MK002	น.ส.ลักษณ อาทร
BC001	นายมงคล ไปได้
BC002	นายจรงค์ รักดี
BC003	น.ส.ศิริ ศิริลักษณ

รูปแบบที่ 2

Select * from Table

- หมายความว่า เลือกเอาทุก Field ในตารางฐานข้อมูล

ตัวอย่าง

Select * from employee ก็จะได้รายงานดังนี้

ID	NAME	ADDRESS	TEL	SALARY
AC001	น.ส.อนงค์ ใจดี	123 ถ.ทหาร อ.เมือง จ.อุดรธานี	042-254615	7,500
MK001	นายถาวร วรดี	145 ถ.อุดร-สกล อ.เมือง จ.อุดรธานี	09-7821546	8,200
MK002	น.ส.ลักษณ อาทร	321/38 อ.เมือง จ.อุดรธานี	042-236457	8,500
BC001	นายมงคล ไปได้	145 หมู่ 2 ต.บ้านเลื่อม อ.เมือง จ.อุดรธานี	01-7121288	8,600
BC002	นายจรงค์ รักดี	145 ถ.พัฒนาการ อ.เมือง จ.หนองคาย	042-248665	8,900
BC003	น.ส.ศิริ ศิริลักษณ	146 ถ.ทหาร อ.เมือง จ.อุดรธานี	042-246658	11,000

Where

เป็น Clauses ซึ่งใช้กำหนดเงื่อนไขในการแสดงข้อมูล โดยจะต้องใช้ร่วมกับคำสั่ง Select เท่านั้น มีรูปแบบดังนี้

Select field1,field2,fieldn.. from Table where field ที่ต้องกำหนดเป็นเงื่อนไข เงื่อนไข

เงื่อนไข หมายถึง เงื่อนไขในสิ่งที่เราต้องการแสดงผล โดยทั้งนี้จะต้องคำนึงถึงชนิดของข้อมูลในฐานข้อมูลด้วย

ตัวอย่าง

จากตัวอย่างฐานข้อมูล Employee หากต้องการแสดงข้อมูลของพนักงานที่มีเงินเดือนที่มากกว่า 8,500 ก็จะสามารถเขียนคำสั่งได้ดังนี้

Select * from employee where salary > 8,500

ซึ่งก็จะได้รายงานดังนี้

ID	NAME	ADDRESS	TEL	SALARY
BC001	นายมงคล ไปได้	145 หมู่ 2 ต.บ้านเลื่อม อ.เมือง จ.อุดรธานี	01-7121288	8,600
BC002	นายจรงค์ รักดี	145 ถ.พัฒนาการ อ.เมือง จ.หนองคาย	042-248665	8,900
BC003	น.ส.ศิริ ศิริลักษณ์	146 ถ.ทหาร อ.เมือง จ.อุดรธานี	042-246658	11,000

แต่ถ้าหากในกรณี Field ที่ต้องการกำหนดเป็นเงื่อนไข มีชนิดข้อมูลเป็น Text นั้นจะต้องใส่เครื่องหมาย ‘ ด้วย

Operator Boolean

- AND ใช้ Boolean 2 ตัว ในแบบฟอร์มดังนี้ A And B เป็น Argument และประเมินผลเป็นจริง ถ้าทั้งสองเป็นจริง
- OR ใช้ Boolean 2 ตัว ในแบบฟอร์มดังนี้ A Or B เป็น Argument และประเมินผลเป็นจริง ถ้าตัวใดตัวหนึ่งเป็นจริง
- NOT ใช้ Boolean 1 ตัว ในแบบฟอร์มดังนี้ Not A เป็น Argument และเปลี่ยนค่าของ Argument นั้นจากเท็จเป็นจริงหรือ จากจริงเป็นเท็จ

โดยจะมีรูปแบบดังนี้

Select *field1,field2,fieldn..* from *Table* where *field* ที่ต้องกำหนดเป็นเงื่อนไข1 เงื่อนไข1
Operator Boolean *field* ที่ต้องกำหนดเป็นเงื่อนไข2 เงื่อนไข2

IN

IN จะกำหนดเซต ซึ่งอาจจะรวมหรือไม่รวมค่าที่กำหนดให้ได้อย่างชัดเจน ซึ่งเป็นการรวมที่ง่ายกว่า AND และ OR เช่น

Select * from employee where salary IN (8500,11000)

ก็จะได้ข้อมูลดังนี้

ID	NAME	ADDRESS	TEL	SALARY
MK002	น.ส.ลักษณ์ ออาหาร	321/38 อ.เมือง จ.อุดรธานี	042-236457	8,500
BC003	น.ส.ศิริ ศิริลักษณ์	146 ถ.ทหาร อ.เมือง จ.อุดรธานี	042-246658	11,000

BETWEEN

คล้ายกับ IN นอกจากจะกำหนดเซตอย่าง IN แล้ว BETWEEN ยังกำหนดย่านของค่าที่จะทำให้เป็นจริง อีกนัยหนึ่งคือจะเป็นคำสั่งที่กำหนดช่วงของการแสดงผลได้ โดยเฉพาะข้อมูลที่เป็นช่วงวันที่ โดยใช้ร่วมกับ WHERE

ตัวอย่าง

Select * from employee where salary between 8500 and 10000 ซึ่งก็จะได้รายงานดังนี้

ID	NAME	ADDRESS	TEL	SALARY
MK002	น.ส.ลักษณ อาทร	321/38 อ.เมือง จ.อุตรธานี	042-236457	8,500
BC001	นายมงคล ไปได้	145 หมู่ 2 ต.บ้านเลื่อม อ.เมือง จ.อุตรธานี	01-7121288	8,600
BC002	นายจรงค์ รักดี	145 ถ.พัฒนาการ อ.เมือง จ.หนองคาย	042-248665	8,900

LIKE

ใช้ได้กับชนิดข้อมูลที่เป็น Text เท่านั้น เมื่อใช้ในการหาแถว ข้อความย่อย หรือกล่าวอีกนัยหนึ่งได้ว่า ใช้ค้นหาใน Field ที่เป็นตัวอักษรเพื่อดูว่ามีส่วใดของ Field นั้นตรงกับแถวข้อความใดบ้าง โดยจะมี ตัวอักษรพิเศษที่ใช้ร่วมกับ LIKE คือ

- `_` หมายถึงตัวอักษรใด ๆ เพียงตัวเดียว เช่น 'b_t' จะตรงกับ 'bat' หรือ 'bit' แต่จะไม่ตรงกับ 'brat' เป็นต้น
- `%` หมายถึงลำดับของตัวอักษรจำนวนเท่าใดก็ได้ เช่น '%p%' จะตรงกับ 'put', 'poist' หรือ 'opt' แต่จะไม่ตรงกับ 'spite'
- `*` หมายถึงตัวอักษรใด ๆ ทุกตัวอักษรที่อยู่หลังตัวอักษรที่เรากำหนด เช่น 'p*' จะตรงกับ 'parent', 'present' หรืออื่น ๆ ที่ขึ้นต้นด้วย P

ตัวอย่าง

select * from employee where id like 'mk*' ซึ่งจะได้ข้อมูลดังนี้

ID	NAME	ADDRESS	TEL	SALARY
MK001	นายถาวร วรดี	145 ถ.อุตร-สกล อ.เมือง จ.อุตรธานี	09-7821546	8,200
MK002	น.ส.ลักษณ อาทร	321/38 อ.เมือง จ.อุตรธานี	042-236457	8,500

Select * from employee where name like 'นาย*'

ID	NAME	ADDRESS	TEL	SALARY
MK001	นายถาวร วรดี	145 ถ.อุตร-สกล อ.เมือง จ.อุตรธานี	09-7821546	8,200
BC001	นายมงคล ไปได้	145 หมู่ 2 ต.บ้านเลื่อม อ.เมือง จ.อุตรธานี	01-7121288	8,600
BC002	นายจรงค์ รักดี	145 ถ.พัฒนาการ อ.เมือง จ.หนองคาย	042-248665	8,900

Function ที่ใช้ในการรวมค่า

การสอบถามข้อมูลสามารถทำให้เกิดหลักเกณฑ์ในการรวมกลุ่มค่าต่าง ๆ โดยผ่านทางฟังก์ชันในการรวม ฟังก์ชันในการรวมจะให้ค่าเพียงค่าเดียวสำหรับค่าของทั้งกลุ่มที่ป้อนไว้ในตาราง ได้แก่

- COUNT จะให้จำนวนแถวหรือค่าต่าง ๆ ใน Field ที่ไม่มีค่าว่าง ที่การสอบถามข้อมูลนั้นเลือกไว้
- SUM จะให้ผลบวกทางคณิตศาสตร์ของค่าที่เลือกทั้งหมดของ Field ที่กำหนด
- AVG จะให้ค่าเฉลี่ย ของค่าที่เลือกทั้งหมดของ Field ที่กำหนด
- MAX จะให้ค่ามากที่สุดของค่าที่เลือกทั้งหมดใน Field ที่กำหนดให้
- MIN จะให้ค่าน้อยที่สุดของค่าที่เลือกทั้งหมดใน Field ที่กำหนดให้

รูปแบบการใช้ฟังก์ชัน

select FUNCTION(field) from Table

เช่น หากต้องการดูรายงานของพนักงานที่มีเงินเดือนสูงที่สุดในบริษัท ก็สามารถใช้คำสั่งได้ดังนี้

select *,max(salary) from employee จะได้รายงานดังนี้

ID	NAME	ADDRESS	TEL	SALARY
BC003	น.ส.ศิริ ศิริลักษณ์	146 ถ.ทหาร อ.เมือง จ.อุตรธานี	042-246658	11,000

โดยการใช้ฟังก์ชันอื่น ๆ ก็สามารถทำได้เช่นกัน

Group By

group by จะยอมให้กำหนดเซตย่อยของค่าต่าง ๆ ใน Field ใด Field หนึ่ง โดยเฉพาะในรูปของอีก Field หนึ่ง แล้วใช้ฟังก์ชันในการรวมกับเซตย่อยนั้น ดังนี้แล้วทำให้สามารถรวม Field ต่าง ๆ กับฟังก์ชันในการรวมเข้าด้วยกันในคำสั่ง Select คำสั่งเดียว

การเรียงลำดับ Output ด้วย Field

ในตารางฐานข้อมูลที่ได้สร้างไว้ นั้น โดยส่วนใหญ่จะไม่มีการเรียงข้อมูล หรือการปรับปรุงข้อมูลให้อยู่ในรูปแบบของการเรียงข้อมูล โดยส่วนใหญ่ข้อมูลก็จะมีกรเข้าออกในตารางฐานข้อมูลอยู่ตลอดเวลา เมื่อต้องการรายงานและเพื่อให้รายงานนั้นมีความสวยงามและดูง่าย ก็จะต้องมีการเรียงข้อมูล โดยในภาษา SQL นั้น ก็จะมีการเรียงข้อมูลโดยเลือก Field ที่มีค่าที่ต้องการเรียงได้ตามความต้องการ ซึ่งการเรียงข้อมูลนั้นก็จะมีอยู่ 2 ประเภท คือ จากน้อยไปมาก และจากมากไปน้อย ในภาษา SQL มีรูปแบบดังนี้

select FIELDS from TABLE < CONDITION > ORDER BY FIELD (DESC,ASC)

เป็นคำสั่งที่ต่อท้ายจากคำสั่งต่าง ๆ ที่ได้กล่าวมาแล้ว โดยรูปแบบก็จะเป็นดังข้อความที่ขีดเส้นใต้ ซึ่ง DESC หมายถึง เรียงจากมากไปน้อย และ ASC คือเรียงข้อมูลจากน้อยไปมาก ส่วน CONDITION นั้นจะมีหรือไม่มีก็ได้ สำหรับ SQL นั้น สามารถเรียงข้อมูลได้กับข้อมูลทุกประเภทไม่ว่าจะเป็นข้อมูลตัวอักษร ตัวเลข หรือวันที่ก็ตาม ซึ่งก็ขึ้นอยู่กับความต้องการของผู้ใช้ เช่น หากต้องการเรียงข้อมูลพนักงานจากแฟ้ม employee โดยให้เรียงตามเงินเดือน จากมากไปน้อย ก็สามารถใช้คำสั่งได้ดังนี้

select * from employee order by salary asc ซึ่งจะได้รายงานดังนี้

ID	NAME	ADDRESS	TEL	SALARY
BC003	น.ส.ศิริ ศิริลักษณ์	146 ถ.ทหาร อ.เมือง จ.อุดรธานี	042-246658	11,000
BC002	นายจรงค์ รักษ์ดี	145 ถ.พัฒนาการ อ.เมือง จ.หนองคาย	042-248665	8,900
BC001	นายมงคล ไปดี	145 หมู่ 2 ต.บ้านเลื่อม อ.เมือง จ.อุดรธานี	01-7121288	8,600
MK002	น.ส.ลักษณ์ อาท	321/38 อ.เมือง จ.อุดรธานี	042-236457	8,500

การประมวลผลข้อมูลพร้อมกันหลายตารางด้วยภาษา SQL

การเชื่อมโยงของข้อมูลระหว่าง Table ที่สัมพันธ์กันภายในภาษา SQL ใช้คำสั่ง INNER JOIN ซึ่งมีรูปแบบดังนี้

INNER JOIN <TABLE> ON <RELATED ATTRIBUTES >

โดยที่ TABLE หมายถึง Table ที่จะนำมา join กับ table ที่ระบุไว้ในส่วน FROM

RELATED ATTRIBUTES หมายถึง รายชื่อ Field ที่สัมพันธ์กันระหว่าง 2 Table

เช่น มีตารางฐานข้อมูลอีกตารางหนึ่ง ชื่อ sale โดยมีข้อมูลดังนี้

ID	CODE_PRODUCT	AMOUNT	DATE_SALE
MK001	HD-40SG	5	22/7/45
MK001	CD-52SN	10	22/7/45
MK002	FDD	25	15/6/45
MK002	MB-P4ASUS	13	18/7/45

หากต้องการดูรายละเอียดของการขายสินค้าของพนักงานโดยมีรายละเอียดของพนักงานบางส่วนด้วย ก็สามารถใช้คำสั่งดังนี้

select id,name from employee inner join sale on sale.id = employee.id

ซึ่งก็จะได้รายงานดังนี้

ID	NAME	CODE_PRODUCT	AMOUNT	DATE_SALE
MK001	นายถาวร วรดี	HD-40SG	5	22/7/45
MK001	นายถาวร วรดี	CD-52SN	10	22/7/45
MK002	น.ส.ลักษณ์ อาท	FDD	25	15/6/45
MK002	น.ส.ลักษณ์ อาท	MB-P4ASUS	13	18/7/45

สำหรับการเขียนโปรแกรมในคู่มือฉบับนี้ ที่มีคำสั่งเกี่ยวข้องกับภาษา SQL นั้น สรุปให้เห็นดังต่อไปนี้

การเพิ่มข้อมูล

รูปแบบคำสั่ง

Insert into ชื่อตาราง (field1,field2,...)values(ค่าที่1,ค่าที่2,...)

เช่น

Insert into student(code,name)values("001","นายใจดี")

การแก้ไขข้อมูล

รูปแบบคำสั่ง

Update ชื่อตาราง set field1=ค่าใหม่,field2=ค่าใหม่,... Where field (ที่เป็น key)=ค่าที่มี

เช่น

Update student set name="นายดีใจ" where code="001"

การลบข้อมูล

รูปแบบคำสั่ง

Delete from ชื่อตาราง Where field (ที่เป็น key)=ค่าที่มี

เช่น

Delete from student Where code="001"

การใช้ภาษา SQL ร่วมกับ VB

สำหรับการเขียนคำสั่งที่จะให้โปรแกรมทำงานกับฐานข้อมูลนั้น นอกจากจะเข้าใจโครงสร้างของภาษา SQL แล้วต้องเข้าใจโครงสร้างของชนิดข้อมูลและตัวแปรที่มีใน VB ด้วย ซึ่งการเขียนโปรแกรมจริงๆ จะต้องมีการรับค่าพารามิเตอร์ เพื่อทำการส่งค่าที่ต้องการผ่านไปมาในระหว่างโปรแกรม ดังนั้น พารามิเตอร์ต่างๆ ที่จะส่งค่านั้นจะต้องมีรูปแบบที่จะเขียนได้นั้นต้องขึ้นอยู่กับชนิดข้อมูลด้วย สามารถสรุปได้ดังนี้

ชนิดข้อมูลหลักๆ แบ่งได้ 3 ประเภทดังนี้

1. ชนิดตัวอักษร
2. ชนิดตัวเลข (จำนวนเต็ม,จำนวนจริง)
3. วันที่

***** ทั้งนี้จะเขียนแบบใดต้องพิจารณาชนิดข้อมูลที่กำหนดในฐานข้อมูลเสมอ**

การเขียนภาษา SQL ร่วมกับชนิดข้อมูลที่เป็นตัวอักษรผ่าน VB

พารามิเตอร์ที่จะส่งค่านั้นต้องอยู่ในเครื่องหมาย

ข้อมูลชนิดตัวอักษร

"& ชื่อตัวแปรหรือ Control &"

ข้อมูลชนิดตัวเลข

" & ชื่อตัวแปรหรือ Control &"

ข้อมูลชนิดวันที่

#& ชื่อตัวแปรหรือ Control &#

ตัวอย่าง

```
where code='" + TextBox1.Text + "'  
name='" + TextBox2.Text + "',address='" + TextBox3.Text + "'  
birthdate=#" + DateTimePicker1.Value + "#"
```


บทที่ 7

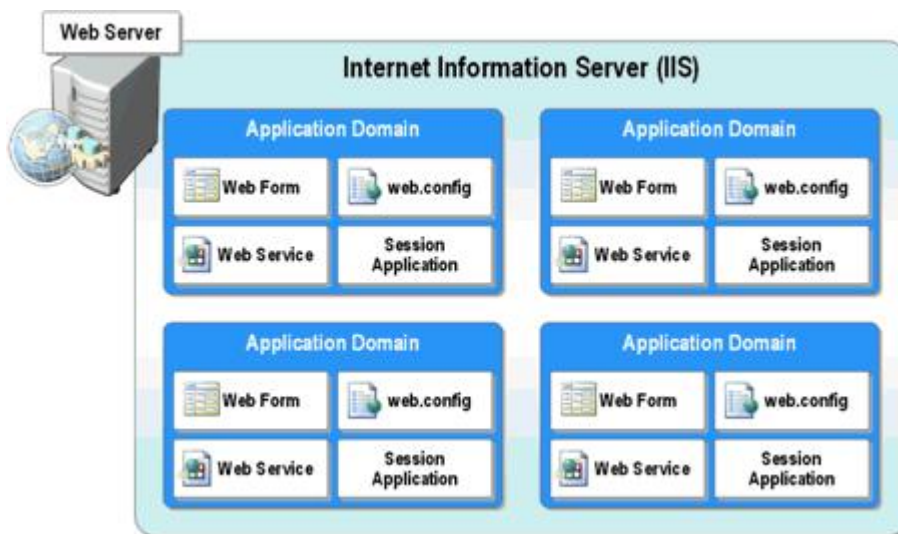
ASP.NET

ASP.NET

ส่วนประกอบของเว็บไซต์ใน ASP.NET 2.0 มีด้วยกัน 3 ส่วนใหญ่ๆ ดังนี้

1. แอปพลิเคชันโดเมน ในเว็บไซต์แต่ละเว็บไซต์ ไม่ว่าจะเป็เว็บไซต์ที่รันบน ASP .NET 1.x หรือ 2.0 จะประกอบด้วยไฟล์หลายๆ ไฟล์ ที่ทำงานภายใต้หน่วยความจำกลุ่มเดียวกัน และทำงานภายใต้ค่าคอนฟิก (Configuration Setting) เดียวกันที่เรียกว่า แอปพลิเคชันโดเมน
















ในแต่ละเว็บไซต์ จะมีแอปพลิเคชันโดเมนเป็นของตัวเอง แต่ละแอปพลิเคชันโดเมนจะไม่สามารถมองเห็นข้อมูลของแอปพลิเคชันโดเมนอื่นได้ เช่น ค่าของตัวแปรเซสชัน หรือตัวแปรแอปพลิเคชัน แม้ว่าจะมีหลายๆ แอปพลิเคชันโดเมนทำงานอยู่ในเซิร์ฟเวอร์ตัวเดียวกันก็ตาม ซึ่งตรงนี้ทำให้เว็บไซต์มีความปลอดภัยในตัวเอง แต่ถ้เกิดมีเว็บไซต์บางตัวเกิดตายไป ก็จะไม่ส่งผลกระทบต่อเว็บไซต์อื่นที่อยู่คนละแอปพลิเคชันโดเมน แสดงภาพแอปพลิเคชันโดเมนบนเว็บเซิร์ฟเวอร์










เพิ่มเติม ในเว็บไซต์ 1 เว็บไซต์ เราสามารถนำไฟล์ที่สร้างจาก คลาสสิก ASP (*.asp หรือ global.asa) มาบรรจุไว้ในเว็บไซต์ของ ASP .NET เพื่อใช้งานได้ แต่การทำเช่นนี้ จะทำให้เว็บไซต์เราประกอบไปด้วยแอปพลิเคชันโดเมน 2 ตัว คือ แอปพลิเคชันโดเมนของ คลาสสิก ASP กับ แอปพลิเคชันโดเมนของ ASP .NET ซึ่งทั้งสองโดเมนนั้น ไม่สามารถที่จะแลกเปลี่ยนข้อมูลระหว่างกันได้ เนื่องจากอยู่คนละแอปพลิเคชันโดเมนนั่นเอง

2. ไฟล์ในเว็บไซต์ ASP .NET 2.0 เว็บไซต์ที่สร้างบน ASP .NET 2.0 สามารถรองรับไฟล์ที่มีอยู่ใน ASP .NET 1.x ได้ทั้งหมด นอกจากนี้ยังได้เพิ่มไฟล์ชนิดใหม่ ที่ทำให้การพัฒนาเว็บไซต์เป็นไปได้อย่างสะดวกมากขึ้นดังนี้

ไฟล์ชนิดเดิมที่มีในเว็บไซต์ ASP .NET 1.x มีดังนี้

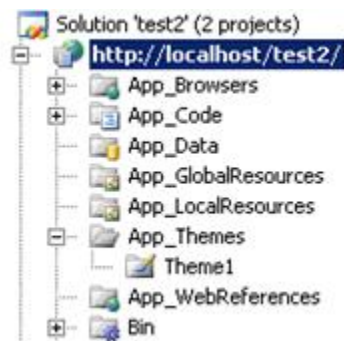
ไอคอน	ชนิดไฟล์	นามสกุล	รายละเอียด
	เว็บฟอร์ม	*.aspx	ใช้ออกแบบหน้าเว็บเพจ และติดต่อกับ Code Behind ได้
	เว็บยูสเซอร์คอนโทรล	*.ascx	ใช้สร้างคอนโทรลที่ใช้สำหรับเว็บฟอร์ม
	เว็บเซอร์วิส	*.asmx	ใช้สร้างเว็บเซอร์วิส
	HTML	*.html	หน้า HTML ที่สามารถบรรจุไคเอนสคริปต์ได้
	สไตล์ ชีต	*.css	ใช้กำหนดสไตล์ของหน้า HTML
	คลาส	*.cs	คลาสเปล่าที่ให้นำไปเขียนโค้ดต่อ
	ไฟล์ Global Application	*.asax	ไฟล์สำหรับจัดการกับเหตุการณ์ (Event) ต่างๆ ของเว็บไซต์ ทำหน้าที่คล้ายกับไฟล์ global.asa ของคลาสสิค ASP
	ไฟล์คอนฟิกูเรชัน	*.config	ไฟล์สำหรับกำหนดค่าต่างๆ ในเว็บไซต์
	XML	*.xml	ไฟล์ XML เปล่า
	Schema ของ XML	*.xsd	ไฟล์ Schema ที่ใช้สำหรับตรวจสอบความถูกต้องของ XML
	เท็กซ์	*.txt	ไฟล์ข้อความเปล่า
	รีซอร์ส	*.resx	ไฟล์ที่ใช้เก็บข้อมูลของ .NET
	Dataset	*.xsd	ใช้สร้าง Schema จากคลาส Dataset
	คริสตรัล รีพอร์ต	*.rpt	ไฟล์ที่ใช้สร้างรายงานสำหรับแอปพลิเคชันเว็บ และวินโดวส์
	XSLT	*.xslt	ไฟล์ที่ใช้แปลงข้อมูล XML ให้เป็นรูปแบบต่างๆ

ไฟล์ชนิดใหม่ ที่เพิ่มเติมเข้ามาในเว็บไซต์ ASP .NET 2.0 มีดังนี้










ไอคอน	ชนิดไฟล์	นามสกุล	รายละเอียด
	ฐานข้อมูล SQL	*.mdf	ฐานข้อมูล SQL เปล่า ที่ใช้กับ SQL Server
	Generic Handler	*.ashx	ไฟล์ที่ใช้จัดการกับร้องขอไฟล์ต่างๆ ภายในเว็บไซต์
	แผนผังเว็บไซต์	*.sitemap	ใช้เก็บแผนที่เว็บไซต์ซึ่งอยู่ในรูปของ XML
	รายงาน	*.rdlc	ใช้สร้างรายงาน โดยใช้เทคโนโลยีการสร้างรายงานของ ไมโครซอฟต์
	Skin	*.skin	ใช้กำหนดสีสัน (Theme) ภายในเว็บไซต์
	Browser	*.browser	ไฟล์ที่ใช้กำหนดความหมายของเบราว์เซอร์
	คลาสไลออะแกรม	*.cd	คลาสไลออะแกรม

3. โฟลเดอร์ในเว็บไซต์ ASP .NET 2.0

ภายในเว็บไซต์ที่สร้างบน ASP .NET 2.0 เราสามารถเพิ่มไฟล์ต่างๆ เช่น Code Behind เว็บฟอร์ม หรือ User Control ได้เหมือนกับเว็บไซต์ที่สร้างบน ASP .NET 1.x แต่ส่วนที่เพิ่มเติมเข้ามาคือโฟลเดอร์ 7 โฟลเดอร์ดังภาพ

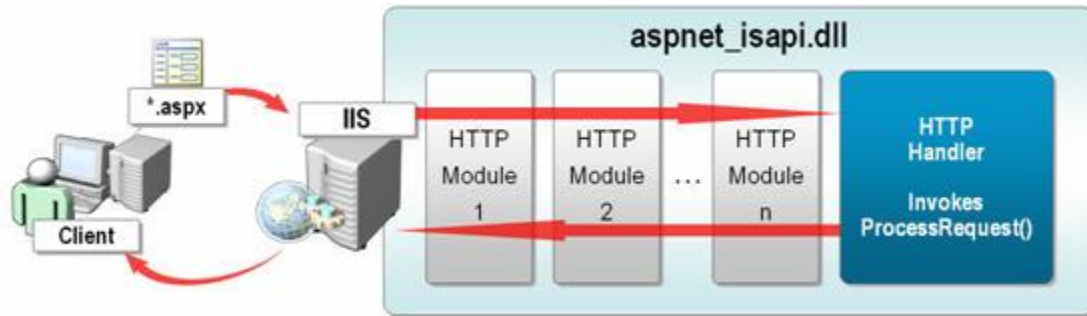


เราสามารถเพิ่มโฟลเดอร์เหล่านี้ โดยคลิกขวาที่โปรเจกต์ แล้วเลือก Add ASP .NET Folder จะมีโฟลเดอร์ 7 โฟลเดอร์มาให้เราเลือก ซึ่งโฟลเดอร์เหล่านี้จะมีหน้าที่ต่างๆ กัน ดังตารางด้านล่าง

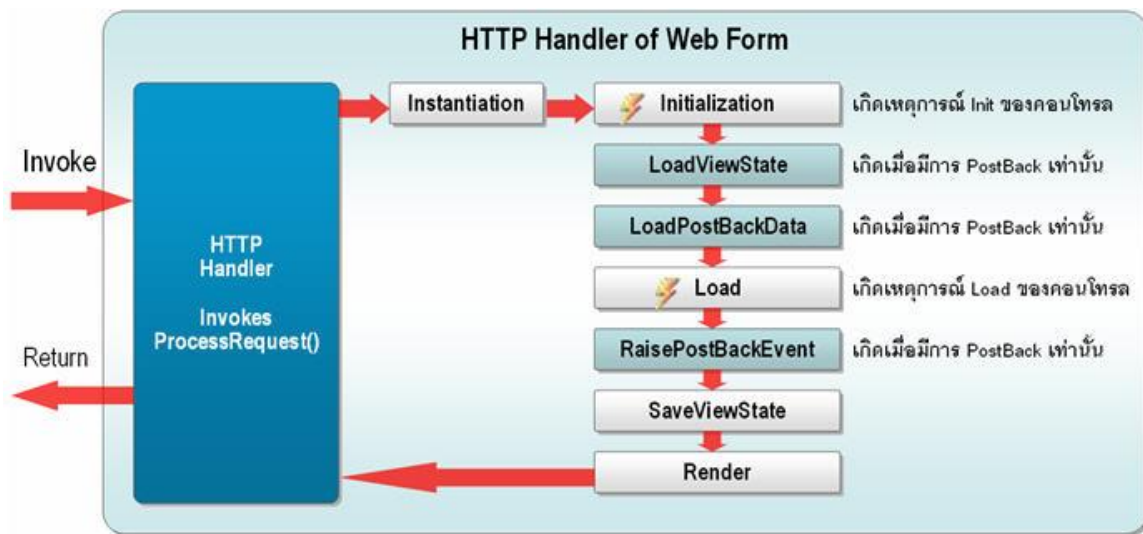
ไอคอน	ชนิดโฟลเดอร์	รายละเอียด
	Bin	บรรจุแอสเซมบลีที่ได้ถูกคอมไพล์ไว้แล้วเพื่อใช้กับเว็บไซต์
	App_Browsers	เป็นไฟล์ XML ที่อนุญาตให้เราสามารถเขียนขึ้นมาได้เองเพื่อบอกถึงคุณสมบัติในด้านต่างๆ ทางฝั่งไคลเอนต์ของเบราเซอร์แต่ละชนิด เช่น ไคลเอนต์สคริปต์ การรองรับเฟรม การรองรับคูกี้ เราสามารถเขียนตรวจสอบเบราเซอร์ขณะติดต่อกับเว็บเซิร์ฟเวอร์ได้ว่าตรงกับ เบราเซอร์ตัวใดที่เรากำหนด เพื่อที่จะได้คืนผลลัพธ์ให้เหมาะสมให้กับไคลเอนต์
	App_Code	บรรจุไฟล์คลาสต่างๆ ที่อยู่ในรูปของ Source Code (*.vb หรือ *.cs) โดยไฟล์คลาสที่ถูกเขียนขึ้น แล้วนำไปวางในโฟลเดอร์นี้ จะถูกคอมไพล์โดยอัตโนมัติ ดังนั้นจึงเหมาะสำหรับการเก็บคอมโพเนนต์ หรือไลบรารีต่างๆ ที่สามารถถูกเรียกใช้ได้จากเว็บฟอร์มทุกตัวในเว็บไซต์ หลังจากที่มีการคอมไพล์ไฟล์คลาสในโฟลเดอร์นี้แบบอัตโนมัติแล้ว จะได้แอสเซมบลีชุดหนึ่ง ซึ่งจะถูกเก็บไว้ในโฟลเดอร์ชั่วคราว ที่ไม่ได้เก็บในโฟลเดอร์ Bin
	App_Data	บรรจุไฟล์ข้อมูลต่าง รวมทั้งไฟล์ฐานข้อมูล SQL (*.mdf) หรือไฟล์ข้อมูล XML แต่เราไม่ควรจำเป็นต้องเก็บไฟล์ข้อมูลไว้ในที่นี้ เป็นเพียงแค่อัดแน่นเท่านั้น
	App_GlobalResources	บรรจุไฟล์รีซอร์สสำหรับ .NET ซึ่งสามารถถูกดึงข้อมูลไปใช้จากทุกเว็บฟอร์มต่างๆ ภายในเว็บไซต์
	App_LocalResources	คล้ายกับโฟลเดอร์ App_GlobalResources แต่จะสามารถถูกดึงข้อมูลไปใช้กับเฉพาะเว็บฟอร์มเท่านั้น
	App_Themes	บรรจุ Theme ที่ใช้กับเว็บไซต์
	Theme	บรรจุรูปภาพ และสีสันทันที่ใช้กับเว็บไซต์
	App_WebReferences	บรรจุการอ้างอิงไปถึงเว็บเซอร์วิสอื่นๆ ที่สามารถถูกเรียกใช้ได้ในเว็บไซต์ ซึ่งภายในจะบรรจุข้อมูล WSDL และ Discovery ของเว็บเซอร์วิส

Tip: ไฟล์ที่เรานำไปวางไว้ได้โฟลเดอร์ Bin และโฟลเดอร์ที่ขึ้นต้นด้วย App ยกเว้น App_Themes นี้ จะไม่สามารถถูกเรียกได้จากไคลเอนต์ เนื่องจากเป็นโฟลเดอร์ที่ถูกบล็อกไว้เพื่อความปลอดภัย มีเฉพาะเว็บไซต์ที่ทำงานบน ASP .NET 2.0 ด้วยเหตุนี้ เราสามารถใช้ความสามารถนี้ในการป้องกันไฟล์สำคัญๆ ได้ เช่น ถ้าเรามีไฟล์ข้อมูลที่เป็น XML ที่เก็บข้อมูลสำคัญ แล้วไม่ยอมให้ไคลเอนต์สามารถเข้าถึงได้ เราสามารถก๊อปปี้ไปวางไว้ในโฟลเดอร์ App_Data ได้

เมื่อไคลเอนต์มีการร้องขอไฟล์แบบไดนามิกไปที่ IIS การเรียกนี้จะถูกส่งต่อไปให้กับ aspnet_isapi.dll จากนั้นจะส่งผ่านเป็นทอดๆ ให้กับ HTTP Module และสุดท้ายก็จะมาทำงานที่ HTTP Handler ที่เป็นตัวรันเว็บฟอร์ม ดังภาพ



เมื่อการร้องขอไฟล์เว็บฟอร์มมาถึง HTTP Handler ของเว็บฟอร์ม และจะมีการเรียกเมธอด ProcessRequest และเข้าสู่วงจรชีวิตของเว็บฟอร์มดังภาพ



จากภาพ เราจะแบ่งขั้นตอนการทำงานย่อยของเมธอด ProcessRequest ออกได้เป็น 8 ขั้นตอนดังนี้

ขั้นตอนที่ 1 Instantiation

ในขั้นตอนนี้ ASP .NET จะมีการสร้างอ็อบเจกต์ของเว็บฟอร์มขึ้นมาโดยอัตโนมัติ ไม่ว่าเว็บฟอร์มของเราจะมีโค้ดแบบ Inline หรือ Code Behind ก็ตาม โดย ASP.NET จะเข้าไปทำการอ่านแท็กต่างๆ ที่อยู่ในมุมมองของ Source ในไฟล์ .aspx แล้วทำการสร้างอ็อบเจกต์เว็บฟอร์มขึ้นมาก่อน จากนั้น จึงทำการสร้าง อ็อบเจกต์ของคอนโทรลแต่ละตัวที่อยู่ในเว็บฟอร์ม

ขั้นตอนที่ 2 Initialization

หลังจากที่มีการสร้างลำดับชั้นของคอนโทรลเสร็จแล้ว จะเข้ามาในส่วนของเหตุการณ์เริ่มต้น (Initialization) ของทั้งเว็บฟอร์ม และของคอนโทรลภายในเว็บฟอร์ม เหตุการณ์นี้ตรงกับเมธอด OnInit ของทั้งเว็บฟอร์ม และคอนโทรล โดยเมธอด OnInit ของคอนโทรลจะถูกเรียกก่อนจนครบทุกตัว จากนั้นจึงทำการเรียกเมธอด OnInit ของเว็บฟอร์ม

สำหรับเมธอด OnInit ของเว็บฟอร์ม เราสามารถเห็นเมธอดนี้ได้ ในตอนที่เพิ่มเว็บฟอร์มเข้ามาใหม่ในโปรเจกต์ของ Visual Studio .NET 2003 เราจะสังเกตเห็นส่วนของ Web Form Designer generated code ซึ่งถ้าคลิกเข้าไปดูภายในจะพบว่ามีเมธอด OnInit อยู่ภายใน และเราสามารถเข้าไปเขียนโค้ดเพิ่มเติมในส่วนนี้ได้ แต่สำหรับ Visual Studio 2010 ส่วนนี้จะถูกซ่อนไว้ ถ้าหากเราต้องการเขียนโค้ดเพิ่มในส่วนนี้เราต้องเพิ่มเมธอด OnInit นี้เข้าไปเองดังโค้ดตัวอย่าง

```
protected override void OnInit(EventArgs e){
    base.OnInit(e);
    //Add your custom OnInit here
    ...}
```

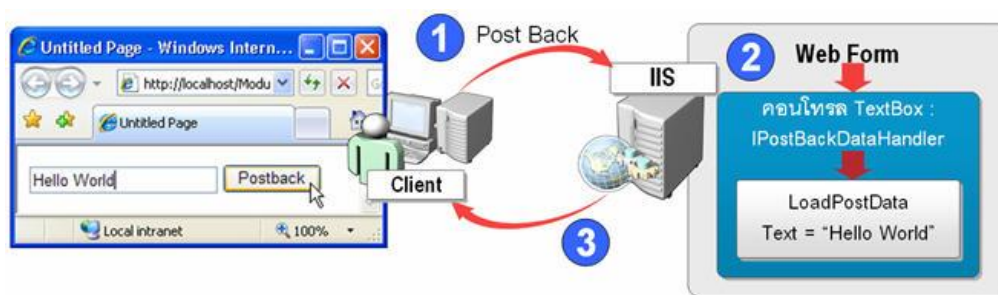
ขั้นตอนที่ 3 โหลด View State

การโหลดค่าจาก View State จะทำเฉพาะตอนที่เว็บฟอร์มมีการ Post back เท่านั้น ในขั้นตอนนี้ ASP .NET จะทำการโหลดค่าจาก View State แล้วนำค่านี้ไปกำหนดให้กับคอนโทรลในเว็บฟอร์มตามลำดับชั้นของอ็อบ

เจ็ทเว็บฟอร์ม และคอนโทรลในเว็บฟอร์ม สำหรับค่าใน View State นี้ บางครั้งอาจถูกแก้ไขโดยแฮกเกอร์ได้ เพื่อมีเจตนาที่ไม่ดีบางอย่าง ในบทความนี้จะได้เรียนรู้ถึงวิธีการป้องกัน การแก้ไขข้อมูลใน View State

ขั้นตอนที่ 4 โหลด Post Back Data

จะเกิดเมื่อเว็บฟอร์มมีการ Post back เท่านั้น ขั้นตอนนี้เว็บฟอร์ม จะทำการโหลดค่าจาก HTTP POST Headers แล้วส่งต่อค่าที่เหมาะสมให้กับเซิร์ฟเวอร์คอนโทรลแต่ละตัวที่ทำการอิมพลิเมนต์อินเตอร์เฟซ IPostBackDataHandler หลังจากนั้นเซิร์ฟเวอร์คอนโทรลจะทำการเรียกเมธอด LoadPostData เพื่อดึงค่าที่เว็บฟอร์มส่งให้ไปกำหนดหรือเพอต์ Text ให้ตัวเอง ตัวอย่างของการโหลดค่าในขั้นตอนนี้ดังภาพ



ขั้นตอนการโหลดค่าจากการ Post Back ดังนี้

1. ไคลเอนต์ร้องขอไฟล์เว็บฟอร์ม ซึ่งบรรจุคอนโทรล Textbox เพื่อให้กรอกข้อความ และคอนโทรล Button ซึ่งเป็นปุ่มที่ทำให้เกิดการ Post Back ไปที่เซิร์ฟเวอร์ โดยไม่มีส่วนของโค้ดที่แก้ไขค่าในคอนโทรล TextBox เลย
2. เมื่อการ Post Back มาถึงที่เว็บเซิร์ฟเวอร์ เว็บฟอร์มจะถูกรันขึ้นมา และเมื่อถึงขั้นตอนการโหลด Post Back Data เว็บฟอร์มตรวจดูว่ามีคอนโทรลตัวใดที่ทำการอิมพลิเมนต์อินเตอร์เฟซ IPostBackDataHandler บ้าง ซึ่งในที่นี้มีเพียงคอนโทรล TextBox เท่านั้น จากนั้นเว็บฟอร์มจะทำการอ่านจาก HTTP POST Headers เฉพาะที่เป็นของ TextBox แล้วนำค่าที่ได้ไปกำหนดให้กับพรอบเพอต์ Text ซึ่งในที่นี้ก็คือคำว่า “Hello World”
3. เมื่อรันเว็บฟอร์มครบทุกขั้นตอนแล้ว ก็จะส่งผลลัพธ์คืนกลับไปไคลเอนต์ ที่ไคลเอนต์จะเห็นว่าค่าใน TextBox นี้ จะพบคำว่า “Hello World” เหมือนกับสถานะตอนที่มีการ Post Back ไปที่เว็บเซิร์ฟเวอร์

เพิ่มเติม สำหรับคอนโทรลที่มีการอิมพลิเมนต์อินเตอร์เฟซ IPostBackDataHandler มักจะเป็นคอนโทรลที่เราสามารถกรอกค่า หรือแก้ไขค่าได้จากฝั่งไคลเอนต์ เช่น คอนโทรล TextBox คอนโทรล DropDownList สำหรับคอนโทรลเหล่านี้ แม้ว่าเราจะกำหนดพรอบเพอต์ EnableViewState ให้เป็นเท็จแล้วก็ตาม คอนโทรลเหล่านี้ก็

ยังสามารถจำค่าที่กำหนดจากทางฝั่งไคลเอนต์ได้ เพราะคอนโทรลเหล่านี้โหลดค่าจาก Post Back มาใช้

ขั้นตอนที่ 5 Load

เมธอดนี้เป็นของเว็บฟอร์ม ซึ่งนักพัฒนาเว็บจะรู้จักเป็นอย่างดี เหตุการณ์นี้ตรงกับเหตุการณ์ในโพธิ์เซอร์ Page_Load ขั้นตอนนี้จะถูกเรียกทั้งตอนที่มีการ Post Back และไม่มีการ Post Back

ขั้นตอนที่ 6 Raise Post Back Event

เป็นเหตุการณ์ที่เกิดกับคอนโทรลที่มีความสามารถในการ Post Back ได้เช่น เหตุการณ์ Click ของ Button เหตุการณ์ TextChanged ของคอนโทรล TextBox คอนโทรลที่สามารถสร้างการ Post Back ได้นั้น ต้องมีการอิมพลิเมนต์อินเตอร์เฟซ IPostBackDataHandler ที่เราได้รู้จักกันไปแล้ว

ขั้นตอนที่ 7 Save View State

ขั้นตอนนี้เว็บฟอร์มจะเรียกเมธอด SaveViewState ของคอนโทรลแต่ละตัวที่อยู่ภายใต้เว็บฟอร์มตามลำดับชั้น เพื่อทำการบันทึกสถานะของคอนโทรลล่าสุดที่อาจมีการเปลี่ยนแปลงไป เนื่องจากขั้นตอนก่อนหน้านี้ ผลของการบันทึกนี้จะถูกเก็บอยู่ในรูปตัวหนังสือที่เข้ารหัสแบบ Base-64 แล้วเก็บใน Hidden Field ที่จะถูกสร้างในขั้นตอนถัดไป

ขั้นตอนที่ 8 Render

ขั้นตอนนี้ เว็บฟอร์มจะทำการแปลงข้อมูลในขั้นตอนที่ผ่านมาออกมาในรูปแบบภาษา HTML เพื่อส่งต่อให้กับไคลเอนต์ ในขั้นตอนนี้เว็บฟอร์มจะเรียกเมธอด RenderControl ของคอนโทรลแต่ละตัวที่อยู่ภายใต้เว็บฟอร์มตามลำดับชั้นสำหรับ ขั้นตอนทั้ง 8 ที่กล่าวมานี้ เป็นขั้นตอนที่สำคัญที่มีความเกี่ยวข้องกับ View State ซึ่งได้ละขั้นตอน PreInit ขั้นตอน PreRender และขั้นตอน Unload ไว้

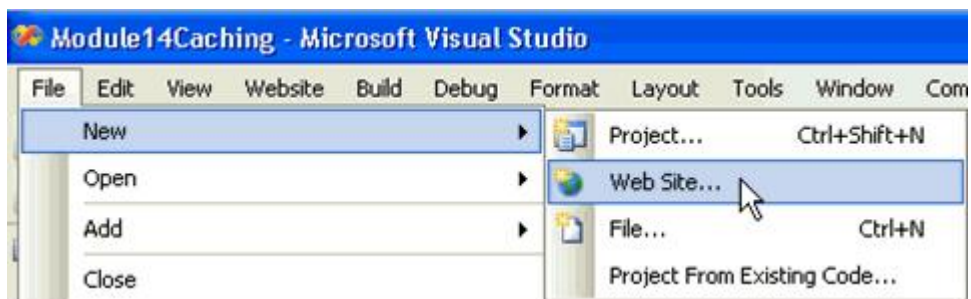
อันที่จริงแล้ว เนื่องจากเบื้องหลังของ ASP.NET นั้น ก็คือโปรแกรมที่ทำงานอยู่บน .NET Framework คุณจึงไม่ได้ถูกจำกัดว่า จะต้องเขียนโค้ด ASP.NET ด้วยภาษาใด ภาษาหนึ่งเท่านั้น ซึ่ง VB ก็เป็นหนึ่งในภาษาที่ .NET นั้นรองรับ และอาจจะเรียกได้ว่า เป็นภาษาอย่างเป็นทางการของแพลตฟอร์ม .NET แต่ถ้าหากว่าคุณเป็นผู้ที่เคยพัฒนาเว็บไซต์ด้วย ASP มาก่อน ภาษาที่มีพื้นฐานมาจากภาษา Visual Basic เดิม อย่าง Visual Basic.NET ก็อาจจะเป็นตัวเลือกที่คุณคุ้นเคยมากกว่า แต่ไม่ว่าคุณจะพัฒนาด้วยภาษาอะไรก็ตาม ในทางทฤษฎีแล้ว จะไม่มีผล

ใด ๆ กับประสิทธิภาพโดยรวมของโปรแกรมแต่อย่างใด และทุกภาษา สามารถเข้าถึงพีเจอร์ต่าง ๆ ของ .NET ได้ อย่างเท่าเทียมกัน

อย่างไรก็ดี ถ้าหากว่าคุณเคยมีพื้นฐานจาก Java หรือว่า C++ มาก่อน หรือต้องการจะเรียนรู้ “ภาษาทางการ” ของแพลตฟอร์ม .NET ภาษา VB ก็เป็นน่าจะเหมาะสมที่สุดสำหรับคุณ ซึ่งภายในบทนี้ ผู้เขียนจะแนะนำถึง โครงสร้างพื้นฐานและการใช้งานภาษา VB พร้อมกับ VB.NET อย่างคร่าว ๆ สำหรับผู้ที่เคยมี

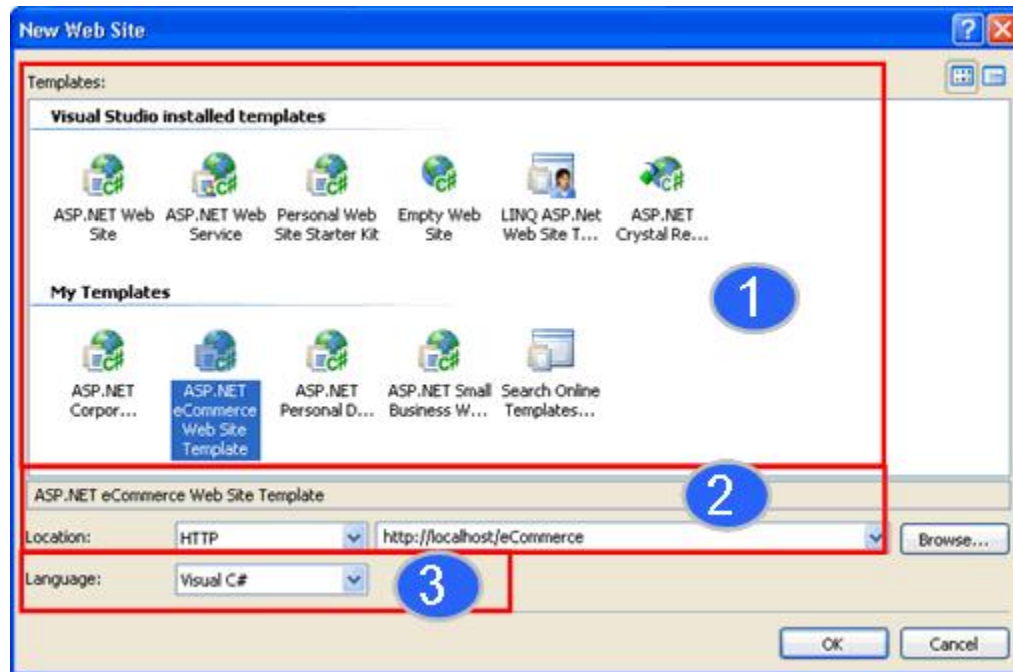
การสร้างเว็บโปรเจคใน Visual Studio 2010

โปรเจคจะมีทั้งวินโดวส์ และเว็บ โดยโปรเจคเว็บจะถูกตั้งชื่อเป็นเว็บไซต์ (Web Site) ที่เราสนใจในที่นี้ จะถูก แยกออกมาดังภาพ



สำคัญมาก โปรเจคประเภท Web Application ที่เคยมีใน Visual Studio .NET 2003 จะหายไป ใน Visual Studio 2010 แต่จะมีโปรเจคเว็บไซต์เข้ามาแทน แต่หลังจากที่ Visual Studio 2010 ออกมาได้ไม่นาน ไมโครซอฟต์ถูกบ่นมาจากนักพัฒนาชินกับโปรเจค Web Application ดังนั้นทางไมโครซอฟต์จึงได้ทำตัวติดตั้งโปรเจค Web Application เพิ่มเติมให้นักพัฒนาดาวน์โหลดเพิ่มฟรีในตอนหลัง ซึ่งคุณสามารถเข้าไปอ่านรายละเอียดหรือโปรเจคนี้ได้ที่ <http://msdn2.microsoft.com/en-us/asp.net/aa336618.aspx> แต่อย่างไรก็ตามโปรเจค Web Application นี้ได้ถูกรวมอยู่ใน Visual Studio 2010 Service Pack 1 เรียบร้อยแล้ว ดังนั้นเราไม่จำเป็นต้องติดตั้งโปรเจค Web Application ซ้ำอีก หลังจากติดตั้ง Service Pack 1 ไปแล้ว

เมื่อเลือกสร้างเว็บไซต์ใหม่แล้ว จะสามารถเลือกชนิดของโปรเจคเว็บดังภาพ



ส่วนประกอบที่จำเป็นต้องเลือกในการสร้างโปรเจกมีอยู่ด้วยกัน 3 ส่วนดังนี้

ส่วนที่ 1 เทมเพลต

จะเป็นส่วนที่ระบุว่ารูปร่างหน้าตาของโปรเจกจะประกอบด้วยไฟล์ และมีการอ้างอิงกับแอสเซมบลีตัวใดบ้าง เทมเพลตที่มี จะคล้ายกับ Visual Studio .NET 2003 คือจะมีเทมเพลตที่เป็นทั้งเว็บไซต์ปกติ (ASP .NET Website) และเทมเพลตที่เป็นเว็บเซอร์วิส (ASP .NET Web Service)

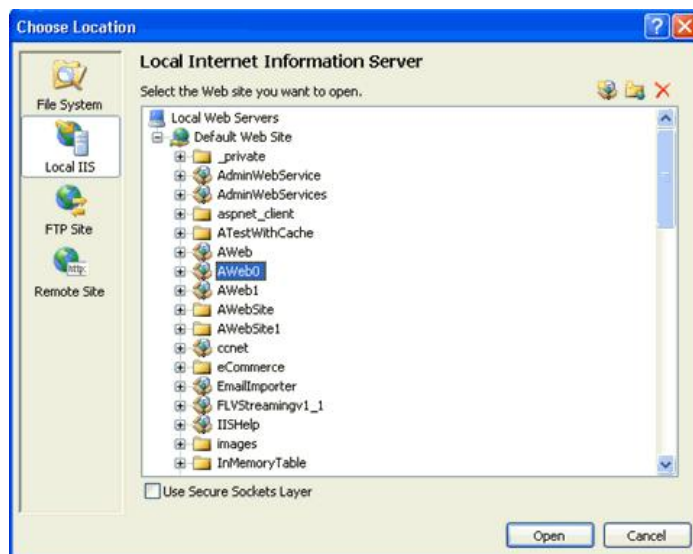
ที่เพิ่มพิเศษเข้ามาคือ เทมเพลตเว็บเซอร์วิสที่เป็น Crystal Report และที่หายไปคือ เทมเพลตที่ใช้สร้าง Web Server Control ซึ่งต้องไปใช้วิธีเพิ่มเข้ามาในโปรเจกที่หลังเอง

เพิ่มเติม เราสามารถดาวน์โหลดเทมเพลตเพิ่มเติมเข้ามาในส่วน My Template ได้ โดยสามารถดาวน์โหลดได้ที่ <http://msdn.microsoft.com/asp.net/reference/design/templates/default.aspx> ที่ลิงค์นี้จะมีเทมเพลตทั้งของภาษา VB .NET และ VB หลายชนิดให้เลือกโดยแบ่งตามจุดประสงค์ของเว็บไซต์ เช่น เว็บอีคอมเมิร์ซ เว็บส่วนตัว หรือเว็บบริษัท

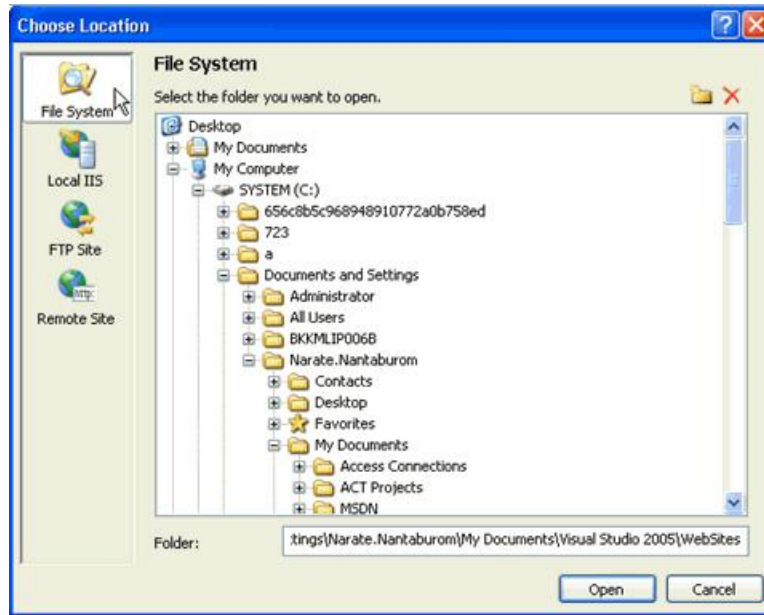
ส่วนที่ 2 Location

จะเป็นตัวที่ระบุว่าเว็บไซต์ของเราจะถูกเก็บไว้ที่ใด ซึ่งเราสามารถระบบได้ทั้งเก็บในเครื่องเราเอง หรือเก็บไว้ในเครื่องอื่นที่อยู่ภายในเน็ตเวิร์ค หรืออาจจะเก็บไว้ใน FTP Server ก็ได้ ในการเก็บเว็บไซต์แต่ละแบบต้องการค่าพารามิเตอร์ที่ต่างกัน ขึ้นอยู่กับ Location ที่เราเลือกดังนี้

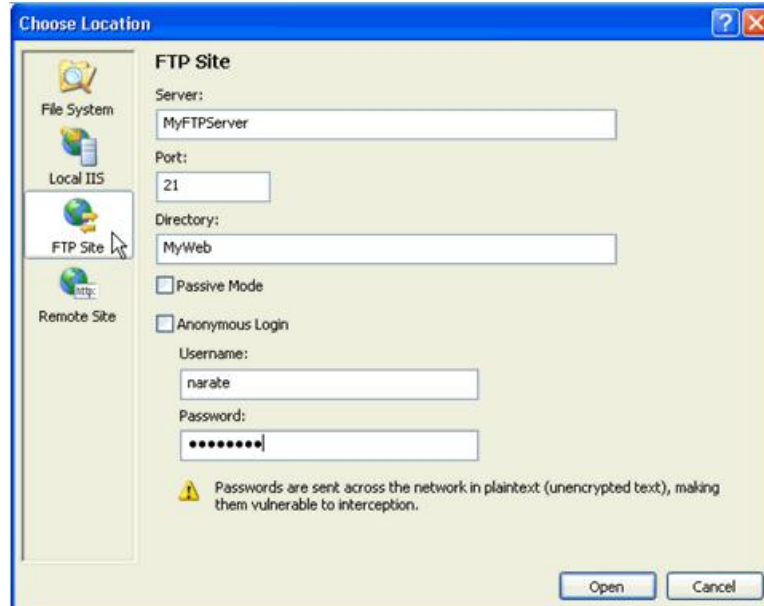
- HTTP เก็บเว็บไซต์ไว้ที่ Virtual Directory ของ IIS เราสามารถใส่ชื่อของเว็บเซิร์ฟเวอร์ และชื่อเว็บไซต์ที่ต้องการลงไปได้ หรืออาจกดปุ่ม Browse



- File System เก็บเว็บไซต์ไว้ที่โฟลเดอร์ของเครื่องเราเอง หรือเครื่องอื่นในเน็ตเวิร์ค เราสามารถตั้ง Path ที่ต้องการเก็บให้เป็นเครื่องของเราเอง หรือเป็น Path ที่อยู่บนเน็ตเวิร์คก็ได้ หรืออาจกดปุ่ม Browse จะปรากฏไดอะล็อกให้เราเลือกดังภาพ



- FTP เก็บเว็บไซต์ไว้ใน FTP เซิร์ฟเวอร์ ซึ่งการที่จะเข้าถึงได้นั้นเราต้องใส่ข้อมูลชื่อเซิร์ฟเวอร์ และไดเรกทอรี เป็นอย่างน้อย สำหรับ FTP เซิร์ฟเวอร์ที่ต้องการ Authentication เราต้องใส่ข้อมูลนี้เพิ่มเข้าไปด้วย หรือจากคปุม Browse จะปรากฏไดอะล็อกให้เราเลือกดังภาพ

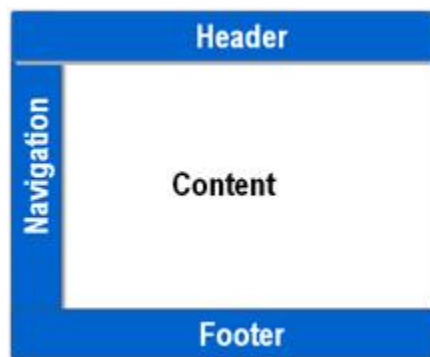


ส่วนที่ 3 ภาษาที่ใช้

เราสามารถเลือกภาษาได้ตามความถนัด ไม่ว่าจะเป็น VB, VB .NET หรือ J# นอกจากนี้ ในโปรเจคเดียวกัน เราสามารถเขียนโค้ดหลายภาษารวมกันได้ เช่น ในหนึ่งโปรเจค เราสามารถสร้างเว็บฟอร์มที่หนึ่งเขียนด้วยภาษา VB ในขณะที่เว็บฟอร์มที่สองเขียนด้วยภาษา VB .NET ได้ แต่ภายในเว็บฟอร์มเดียวกัน เราจะไม่สามารถเขียนโค้ดแบบ Inline (เขียนโค้ดลงบนไฟล์ .aspx) เป็นคนละภาษากับโค้ดที่เป็นเขียนบน Code Behind (เขียนโค้ดลงบนไฟล์ .cs หรือ .vb) ได้

Master Page

สำหรับเว็บไซต์ที่มีขนาดกลาง ไปถึงขนาดใหญ่ มักจะมีส่วนหลักๆ ที่เหมือนกันในแทบทุกหน้า ปกติแล้วส่วนที่เหมือนกันนี้จะป็นรูปแบบเกือกม้า (Horse shoe) ซึ่งจะประกอบด้วยส่วน Header ด้านบน ส่วน Navigation ด้านซ้าย และส่วน Footer ที่อยู่ด้านล่าง ดังภาพ



จากภาพ เราจะเห็นว่าส่วนของ Header ส่วนของ Navigation และส่วนของ Footer ประกอบกันเป็นรูปตัว C หรือมีลักษณะคล้ายเกือกม้า ในส่วนที่เป็นเกือกม้านี้ จะเป็นส่วนที่เหมือนกันแทบทุกหน้าในเว็บไซต์ เราจึงมักสร้างส่วนที่เหมือนกันนี้แยกออกจากส่วนที่เป็นเนื้อหา (Content) ในบางครั้งเราจะเรียกส่วนที่เหมือนกันในรูปเกือกม้านี้ว่าเทมเพลต จากที่เรารอคอยเทมเพลตของ ASP.NET เป็นเวลานานตั้งแต่ปี ค.ศ. 2000 จนในปัจจุบัน ASP.NET 2.0 ได้ออกมาพร้อมกับเทมเพลตที่ชื่อว่า Master Page ในปลาย ค.ศ. 2010

ความยากลำบากในการสร้างเทมเพลต ก่อนที่จะมี Master Page

เทคโนโลยีที่ใช้สำหรับสร้างเว็บไซต์ของไมโครซอฟท์ จะเริ่มตั้งแต่สมัยภาษาคาสสิก ASP ในตอนนั้น เรามักจะใช้การ Include file ที่เป็น ส่วน Header ส่วน Navigation และส่วน Footer เข้ามาในไฟล์ *.asp ดังภาพ



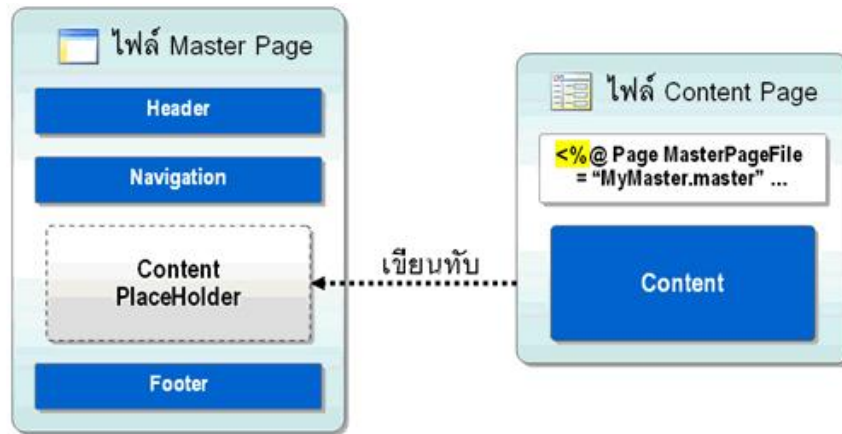
แต่ข้อเสียของวิธีนี้คือ เราต้องจัดการกับแท็ก HTML เองในการเปิด และปิดแท็กที่รอบตัวเนื้อหา (Content) ให้พอดี ซึ่งมักจะใช้ตาราง และสร้างเซลล์ที่บรรจุเนื้อหาเอาไว้ นอกจากนี้เราจะไม่สามารถมองเห็นมุมมอง Design ที่สมบูรณ์ หรือไม่สามารถเห็นทั้ง include file และเนื้อหาพร้อมกันในมุมมอง Design ในตอนออกแบบของ Visual InterDev ได้จริงๆ แล้วในยุคของ ASP บางท่านได้ใช้เฟรม (HTML frame) แทนการใช้ include file แต่ก็เจอปัญหาของ Scroll bar แต่แยกกันของแต่ละเฟรม และเฟรมยังมีปัญหาเกี่ยวกับ Search engine หลายๆ ตัวที่ไม่รู้จักแท็กเฟรม

ถัดจากยุคของ ASP จะเป็นยุคของ ASP.NET 1.0 เราสามารถสร้างยูสเซอร์คอนโทรลที่บรรจุแท็ก HTML ต่างๆ ไปได้ และสามารถนำมาใช้กับเว็บฟอร์มได้อย่างง่ายๆ ด้วยการ Drag and Drop แต่ตัวของยูสเซอร์คอนโทรลนี้จะมีไม่สามารถแสดงในมุมมอง Design ของ Visual Studio รุ่น 2000 2002 หรือ 2003 ได้ เราจะเห็นเพียงแค่กล่องสี่เทา ที่แสดงถึงตัวยูสเซอร์คอนโทรลที่เราลากมาวางในเว็บฟอร์มนี้เท่านั้น และเรายังต้องจัดการกับแท็กปิด และเปิด ที่ล้อมรอบตัวเนื้อหาอยู่เหมือนเดิม

สำหรับปัจจุบัน Master Page ได้แก้ปัญหาที่กล่าวมา โดยการแยก Master Page ออกมาต่างหากจากส่วนเนื้อหา เราสามารถออกแบบ Master Page ได้จากมุมมอง Design และสามารถกำหนดตำแหน่งของเนื้อหาที่เราต้องการวางได้อย่างอิสระ และสำหรับส่วนไฟล์เนื้อหา ซึ่งในที่นี้คือไฟล์เว็บฟอร์ม เราจะเรียกว่าไฟล์ Content Page ซึ่งเราสามารถมองเห็นหน้าตาของทั้งไฟล์ Master Page และส่วนเนื้อหาที่อยู่ในไฟล์ Content Page พร้อมกันได้ ในมุมมอง Design นอกจากนี้เราสามารถเขียนโปรแกรมเพื่อกำหนด Master Page แบบไดนามิก และนำ Master Page หลายๆ ตัวมาซ้อนกันเพื่อสร้าง Master Page ตัวใหม่ที่มีความยืดหยุ่นได้

การทำงานร่วมกันระหว่างไฟล์ Master Page กับไฟล์ Content Page

การทำงานของไฟล์ Master Page กับไฟล์ Content Page แสดงดังภาพ

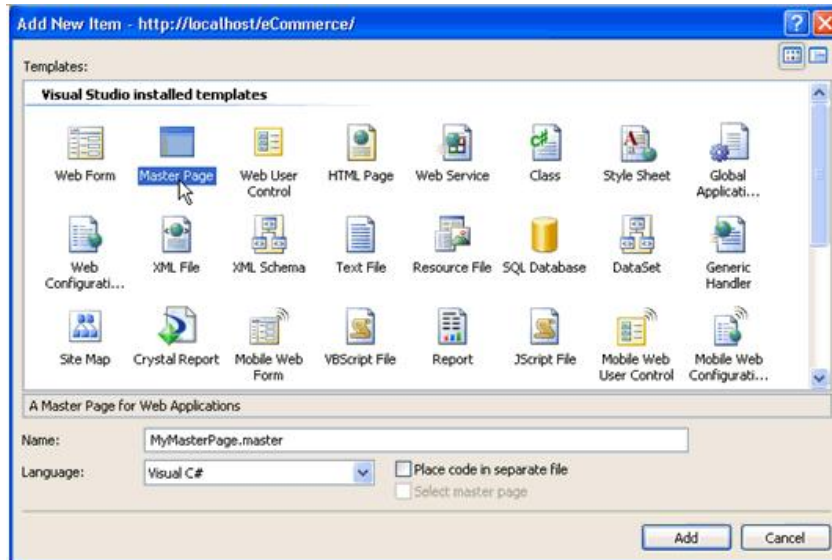


จากภาพ เรามีไฟล์สองไฟล์ ไฟล์แรกคือ Master Page ที่บรรจุเทมเพลตของ Header เทมเพลต Navigation และเทมเพลต Footer และได้เว้นว่างในส่วนของเนื้อหา (Content Placeholder) เอาไว้ ส่วนไฟล์ที่สองคือ Content Page ที่บรรจุเฉพาะเนื้อหาของเว็บฟอร์มเท่านั้น

ในการทำงานจริง จะต้องสร้างไฟล์ Master Page ขึ้นมาก่อน จากนั้นจึงสร้างไฟล์ Content Page ที่ทำการสืบทอดรูปร่างหน้าตา (Visual Inheritance) ซึ่งก็คือส่วน Header ส่วน Navigation และส่วน Footer จากไฟล์ Master Page ที่สร้างไว้ก่อนหน้าผ่านแอตทริบิวต์ MasterPageFile ที่อยู่ในไคเรคทีฟ Page

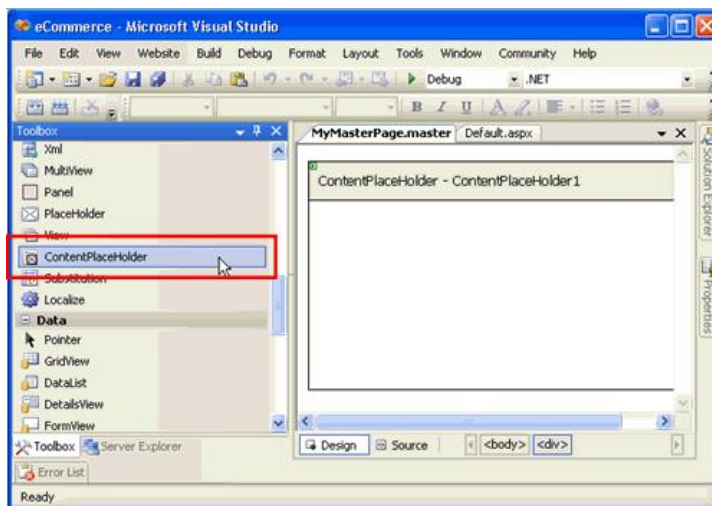
หลังจากที่ไฟล์ Content Page ทำการสืบทอดหน้าตาจากไฟล์ Master Page เรียบร้อยแล้ว ไฟล์ Content Page จะทำการเติมเต็มเนื้อหาของมันเองลงไป ใน Content Placeholder ของไฟล์ Master Page เมื่อเราทำการรันไฟล์ Content Page ในเบราว์เซอร์ เราจะพบว่าเนื้อหาที่ได้จะเป็นผลรวมกันของส่วน Header ส่วน Navigation และส่วน Footer จากไฟล์ Master Page และเนื้อหาจากไฟล์ Content Page

การสร้างไฟล์ Master Page ทำได้คล้ายกับการเพิ่มเว็บฟอร์ม โดยการคลิกขวาที่โปรเจค --> Add New Item... --> เลือกที่ไฟล์ประเภท Master Page ดังภาพ



จากภาพ จะพบว่าไฟล์ Master Page นี้ จะมีนามสกุลเป็น master และมีคุณสมบัติคล้ายกับเว็บฟอร์ม คือเราสามารถเลือกภาษาที่จะเขียนได้ (VB หรือ VB) และเลือกที่จะวางโค้ดไว้ในไฟล์ Code Behind หรือไม่ได้

เมื่อเราเปิดไฟล์ Master Page ที่สร้างในมุมมอง Design เราจะพบว่า เราสามารถมองเห็นคอนโทรลตัวใหม่ที่ชื่อว่า ContentPlaceHolder เพิ่มเข้ามาในทูลบ็อกซ์ ซึ่งเราสามารถลากมาวางในหน้า Design เพื่อใช้งานได้ คอนโทรล ContentPlaceHolder จะมีเฉพาะกับไฟล์ Master Page เท่านั้น จะไม่สามารถมองเห็นได้ในไฟล์เว็บฟอร์ม ดังภาพ



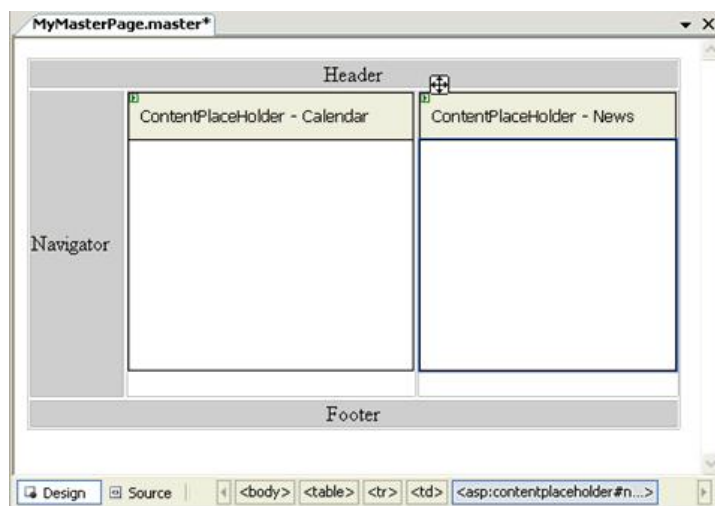
คอนโทรล ContentPlaceHolder ถ้ามองในมุมมอง Source จะเห็นเป็นแท็ก `asp:contentplaceholder` ซึ่งเราสามารถกำหนด id ให้กับมันได้ ดังโค้ดตัวอย่าง

```

<form id="form1" runat="server">
<div>
  <asp:contentplaceholder id="ContentPlaceHolder1" runat="server">
  </asp:contentplaceholder>
</div>
</form>

```

ในการทำงานจริง ContentPlaceHolder จะถูกล้อมด้วยส่วน Header ส่วน Navigation และส่วน Footer ดังภาพ



ในทางปฏิบัติ เรามักไม่ใช่แท็ก div ของภาษา HTML เพื่อทำการแบ่ง Master Page ออกเป็นส่วนๆ แต่เรามักใช้แท็ก table ของภาษา HTML แทน ดังโค้ดตัวอย่างที่ 9-32 ที่แสดงมุมมอง Source ของภาพ

```

<form id="form1" runat="server">
  <table border=0 width="500">
    <tr><td colspan="3" bgcolor="#CCCCCC">Header</td></tr>
    <tr>
      <td bgcolor="#CCCCCC">Navigator</td>
      <td><asp:contentplaceholder id="Calendar"
        runat="server"></asp:contentplaceholder></td>
      <td><asp:contentplaceholder id="News"
        runat="server"></asp:contentplaceholder></td>
    </tr>

```

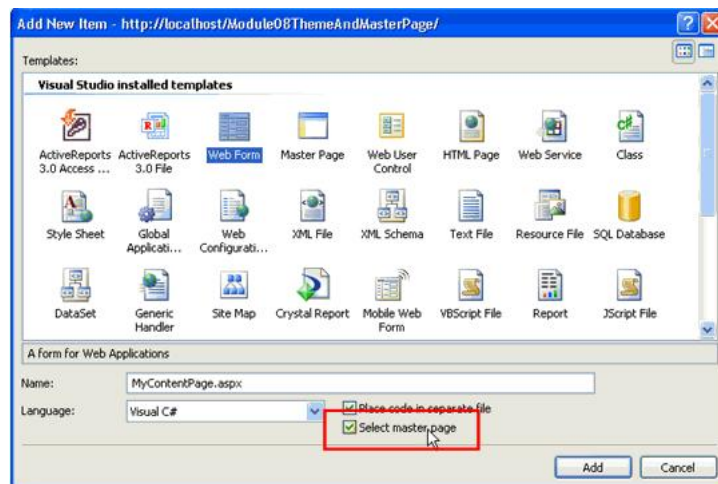
```

<tr><td colspan="3" bgcolor="#CCCCCC">Footer</td></tr>
</table>
</form>

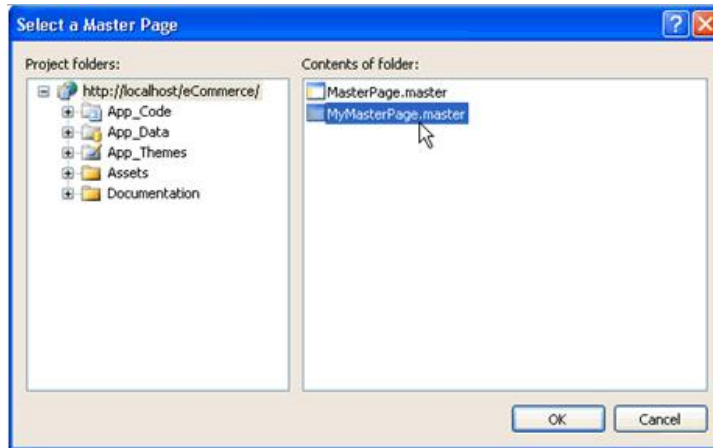
```

จากโค้ดตัวอย่างจะเป็นว่าในไฟล์ Master Page นี้ มีคอนโทรล ContentPlaceHolder อยู่สองตัว โดยตัวแรกจะแสดงปฏิทิน (id="Calendar") และตัวที่สองแสดงข่าว (id="News") ซึ่งคอนโทรลทั้งสองจะไม่มีรายละเอียดของทั้งปฏิทิน และข่าวอยู่เลย เพราะต้องรอให้ไฟล์ Content Page มาเติมรายละเอียดในส่วนนี้ ซึ่งจะกล่าวถึงในหัวข้อถัดไป

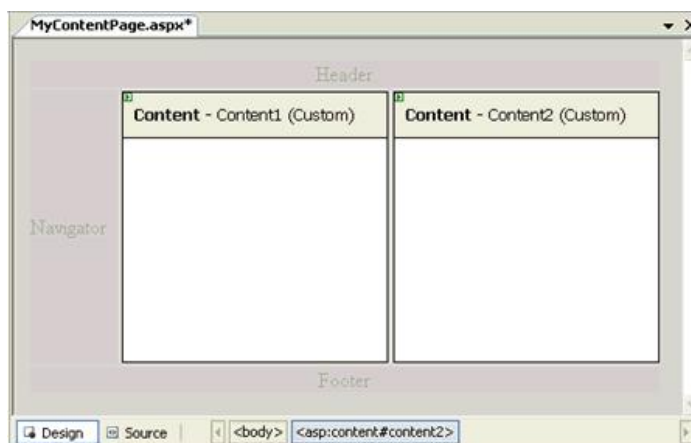
ไฟล์ Content Page จะเป็นไฟล์เว็บฟอร์มตามปกติ แต่ในตอนที่เราสร้างมันขึ้นมา เราต้องเลือกว่าเว็บฟอร์มนี้ต้องการใช้ Master Page ดังภาพ



จากภาพ หลังจากเลือกตัวเลือก Select master page แล้ว กดปุ่ม Add ที่ด้านล่างขวา จะพบหน้าต่างให้เลือกไฟล์ Master Page ดังภาพ



จากภาพ จะเป็นหน้าจอที่ให้เราเลือก Master Page เราสามารถเบร่าซีไปยังไฟล์ Master Page ไฟล์ต่างๆ ที่อยู่ในเว็บไซต์ของเรา หลังจากนั้นกดปุ่ม OK เพื่อจบการสร้าง Content Page เมื่อเราเปิดไฟล์ Content Page ที่เพิ่งสร้างเสร็จในมุมมอง Design จะพบหน้าจอดังภาพ



จากภาพ จะพบว่าในส่วนที่สืบทอดรูปร่างหน้าตามาจากไฟล์ Master Page นั้น ซึ่งก็คือส่วน Header ส่วน Navigation และส่วน Footer ที่เราสามารถมองเห็นได้ แต่จะไม่สามารถแก้ไขได้ เพราะมันจะถูกล็อคเอาไว้ หากต้องการแก้ไข เราจะต้องไปแก้ไขที่ไฟล์ Master Page โดยตรง สำหรับส่วนที่แก้ไขได้ จะอยู่ในคอนโทรล Content เท่านั้น เมื่อเราเปิดไฟล์ Content Page ในมุมมอง Source จะพบดังโค้ดตัวอย่าง

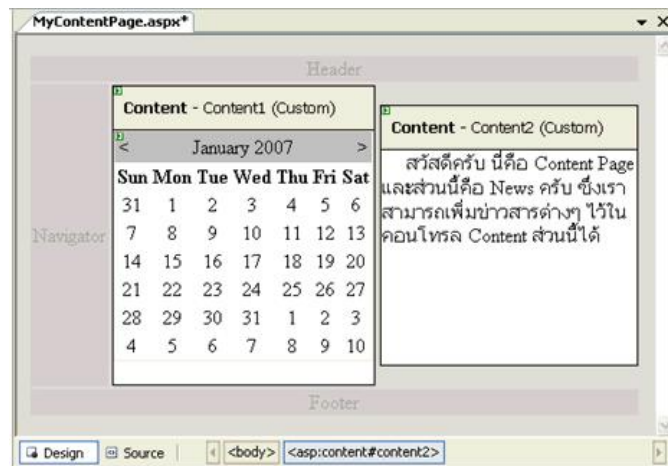
```
<%@ Page Language=... %>
```

```
<asp:Content ID="Content1" ContentPlaceHolderID="Calendar" Runat="Server">
</asp:Content>
```

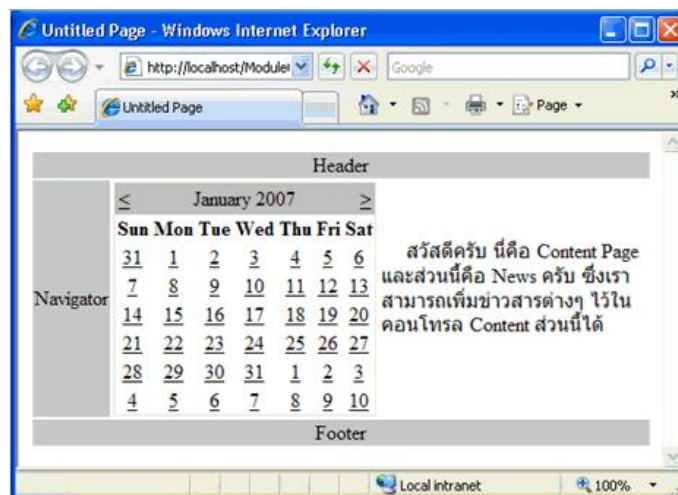
```
<asp:Content ID="Content2" ContentPlaceHolderID="News" Runat="Server">
```

```
</asp:Content>
```

จากโค้ดตัวอย่าง เราจะพบแอตทริบิวต์ MasterPageFile อยู่ในไต่แรกที่ Page ด้านบนสุด และจะมีคอนโทรล Content อยู่สองตัวที่สอดคล้องกับคอนโทรล ContentPlaceHolder ของไฟล์ Master Page โดยคอนโทรลตัวแรกจะแสดงเนื้อหาเกี่ยวกับปฏิทิน ส่วนตัวที่สองแสดงเนื้อหาข่าว เป็นที่น่าสังเกตว่าภายในไฟล์ Content Page จะไม่มีแท็ก HTML แท็ก Body แท็ก Head หรือแท็ก Form เหมือนกับเว็บฟอร์มปกติ เพราะแท็กเหล่านี้จะถูกเขียนไว้ภายในไฟล์ Master Page เรียบร้อยแล้ว หน้าที่ไฟล์ Content Page จึงเหลือเพียงเติมเนื้อหาที่ขาดไปให้กับไฟล์ Master Page เท่านั้น ในการเพิ่มเติมเนื้อหาทั้งสองส่วน เราสามารถใช้มุมมอง Design หรือ Source ในการเพิ่มเติมเนื้อหาได้ตามสะดวก ตัวอย่างหลังจากเพิ่มเติมเนื้อหาในไฟล์ Content Page แสดงดังภาพ

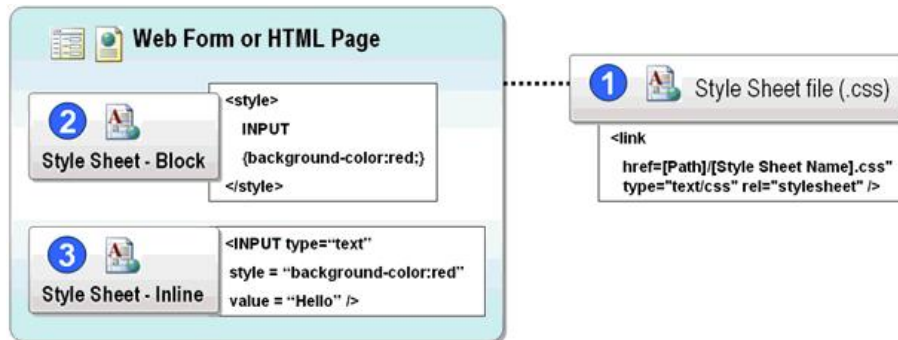


เมื่อทำการเปิดไฟล์ Content Page ในเบราว์เซอร์ จะได้หน้าจอดังภาพ



การสร้างสไตล์ให้กับเว็บฟอร์ม

การสร้างสไตล์ให้กับเว็บฟอร์ม หรือเอกสาร HTML จะทำได้โดยการกำหนดสไตล์ชีต โดยทั่วไปสไตล์ชีตมี 3 แบบดังภาพ



จากภาพ จะเห็นว่ามีการกำหนดสไตล์ชีต 3 แบบดังนี้

1. **สไตล์ชีตแบบไฟล์** เป็นการอ้างอิงถึงไฟล์สไตล์ชีตผ่าน URL โดยใช้แท็ก link วิธีนี้ เบราเซอร์ต้องทำการดาวน์โหลดไฟล์นี้เพิ่มเติมจากเว็บฟอร์ม เพื่อนำสไตล์ที่อยู่ในไฟล์สไตล์ชีตมากำหนดให้กับเว็บฟอร์มอีกที

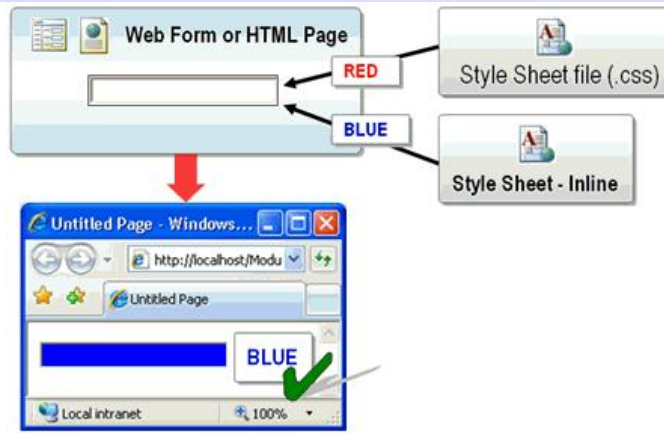
2. **สไตล์ชีตแบบบล็อก (Block)** เป็นการกำหนดสไตล์ชีตลงบนเว็บฟอร์มโดยตรง โดยค่าของสไตล์จะอยู่ระหว่างแท็ก <style> และ </style> ค่าของสไตล์ที่อยู่ในสไตล์ชีตแบบบล็อกนี้ จะถูกนำไปกำหนดให้กับแท็กทุกตัวที่ตรงกับสไตล์ที่กำหนด เช่น จากภาพที่ 9-1 จะทำให้แท็กที่ขึ้นต้นด้วย INPUT ทุกตัวมีสีของพื้นหลังเป็นสีแดง เพราะมีการกำหนด background-color: red ให้เป็นสีแดง

ในบางครั้งเราอาจสร้างสไตล์แบบคลาสด้วยการเพิ่มสัญลักษณ์ “.” ที่ด้านหน้าของชื่อคลาส เช่น .MyClass {text-decoration: underline} เป็นการกำหนดว่าสไตล์ที่ชื่อว่า MyClass จะมีการขีดเส้นใต้

3. **สไตล์ชีตแบบอินไลน์ (In-line)** เป็นการกำหนดสไตล์ลงบนเว็บฟอร์มโดยตรงเหมือนกับแบบบล็อก แต่แบบอินไลน์จะกำหนดสไตล์ลงบนแท็กโดยตรงผ่านแอตทริบิวต์ style ซึ่งผลของสไตล์แบบนี้จะมีผลกับแท็กที่เรากำหนดเท่านั้น จะไม่มีผลกับแท็กตัวอื่นๆ นอกจากนี้ เราสามารถกำหนดสไตล์แบบคลาสให้กับแท็กได้ โดยการเพิ่มแอตทริบิวต์ class เช่น <INPUT type="text" class="MyClass" /> ถ้ามีการกำหนดสไตล์ชีตแบบอินไลน์ โดยผ่านทั้งแอตทริบิวต์ style และ class ทั้งคู่แล้ว ค่าที่อยู่ในแอตทริบิวต์ style จะเขียนทับแอตทริบิวต์ class เสมอ

สำคัญมาก ในกรณีที่มีการกำหนดสไตล์ชีตทั้ง 3 แบบให้กับแท็กใดแท็กหนึ่ง เช่น สไตล์ชีตแบบไฟล์ กำหนดให้

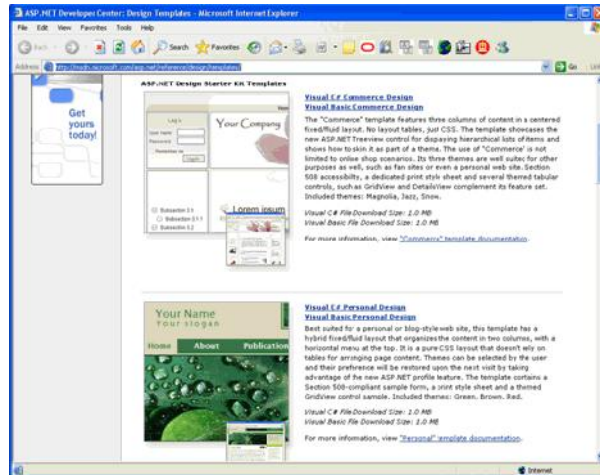
แท็ก INPUT มีสีแดง แต่สไตล์ชีตแบบออนไลน์กำหนดให้แท็ก INPUT มีสีน้ำเงิน ผลลัพธ์ที่ได้คือ แท็ก INPUT จะมีสีน้ำเงินตามสไตล์ชีตแบบออนไลน์ หรือกล่าวได้ว่าสไตล์ชีตแบบออนไลน์จะเขียนทับสไตล์แบบไฟล์ หรือแบบบล็อกเสมอ นั่นเอง ดังภาพ



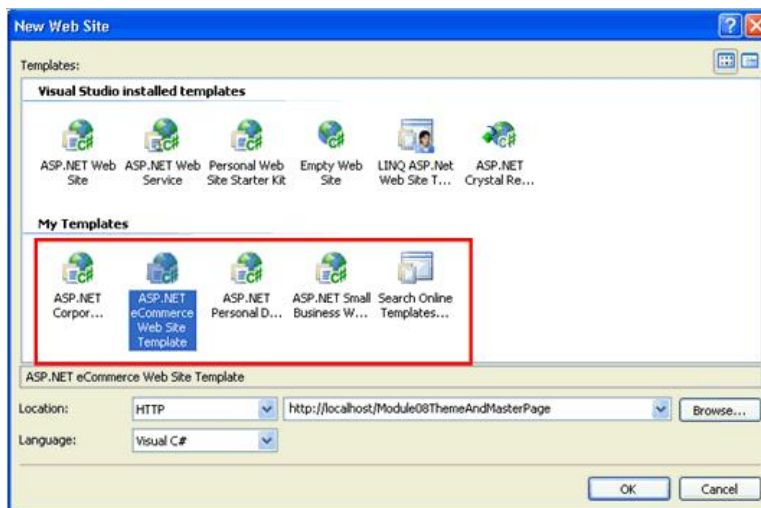
ในกรณีที่มีการเขียนทับกันเองของสไตล์ชีตแบบไฟล์ และแบบบล็อกนั้น ไม่ได้มีการกำหนดตายตัวว่าสไตล์ชีตแบบใดจะมีการเขียนทับตัวใด เพราะทั้งสองแบบมีความสำคัญเท่ากัน แต่มีหลักอยู่ว่า สไตล์ชีตที่วางไว้หลังสุด (เรียงจากบนลงล่างของเว็บฟอร์ม) จะมีการเขียนทับสไตล์ชีตที่อยู่ด้านบนเสมอ ที่เป็นเช่นนี้เพราะในตอนที่เราเซอร์ทำการสร้างสไตล์ เบราเซอร์จะทำการอ่านสไตล์ชีตที่พบในเว็บฟอร์มทั้งหมดเข้าไปไว้ในหน่วยความจำของเครื่อง โดยเริ่มอ่านจากบนลงล่าง และสไตล์ชีตที่อ่านเข้าไปตอนหลัง จะเขียนทับสไตล์ที่มีอยู่เดิมเสมอ นั่นเอง สำหรับความรู้เรื่องสไตล์ชีตนี้ จะเป็นจุดเริ่มต้นที่ดีในการเรียนรู้เรื่อง Theme ที่เป็นของใหม่ใน ASP.NET 2.0 เพราะสุดท้ายแล้วผลลัพธ์จากการทำงานของ Theme ก็จะเป็นสร้างเป็นสไตล์ชีตออกมา เพื่อกำหนดหน้าตา และสีสันท่างๆ ให้กับเว็บฟอร์ม

โครงสร้างของ Theme

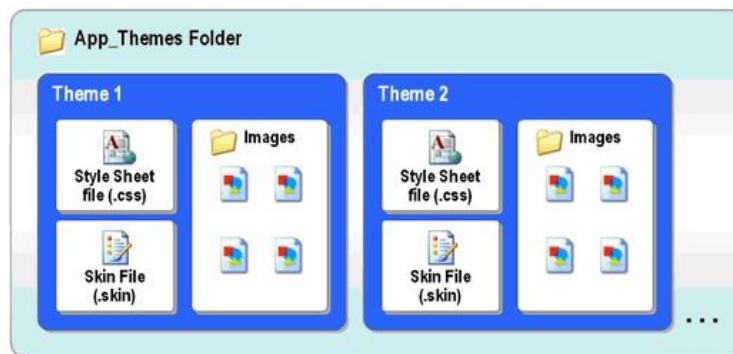
ไฟล์ที่จะทำงานกับ Theme ได้ จะเป็นเว็บฟอร์ม ที่มีการทำงานที่ฝั่งเซิร์ฟเวอร์เท่านั้น จะไม่สามารถใช้กับไฟล์ยูสเซอร์คอนโทรล (User Control) ได้ เพื่อความรวดเร็วในการสร้าง Theme ขอแนะนำให้เข้าไปเลือกดาวน์โหลดตัวอย่างเทมเพลตของไมโครซอฟท์ที่ <http://msdn2.microsoft.com/en-us/asp.net/aa336613.aspx> ดังภาพ



หลังจากที่ดาวน์โหลดตัวอย่างเทมเพลตมาแล้ว เราจะได้ไฟล์ติดตั้งที่มีนามสกุล vsi ให้เราทำการติดตั้งลงบนเครื่อง เมื่อติดตั้งเสร็จแล้วให้เปิด Visual Studio 2010 ขึ้นมาจะพบเทมเพลตดังกล่าว



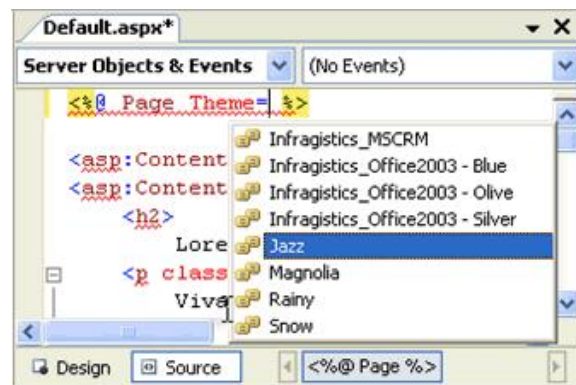
เมื่อสร้างเว็บไซต์จากเทมเพลตแล้ว ให้เราสำรวจโฟลเดอร์ App_Themes ซึ่งเป็นโฟลเดอร์ที่ใช้เก็บ Theme หลายๆ แบบ ใน ASP.NET 2.0 โครงสร้างของโฟลเดอร์ App_Themes แสดงดังภาพที่ 9-5



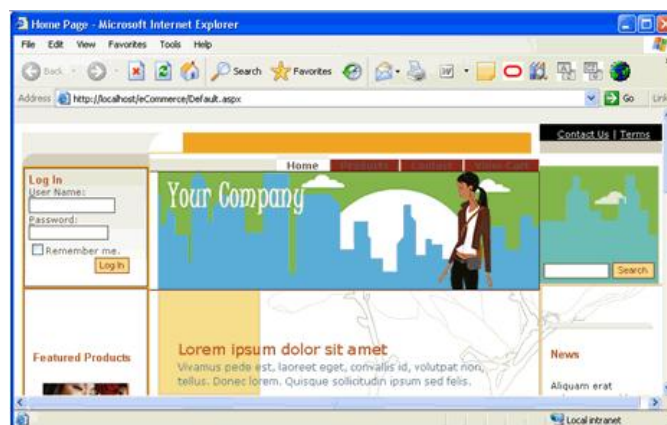
จากภาพ ภายในโฟลเดอร์ App_Themes จะประกอบไปด้วย Theme ได้หลายๆ แบบ จากภาพเราจะเห็นว่า มี Theme1 และ Theme2 อยู่ใน และในโฟลเดอร์ Theme ก็มีไฟล์สไตล์ชีต โฟลเดอร์ที่เก็บรูปภาพ และไฟล์ Skin ซึ่งเป็นไฟล์หลักที่ใช้ในการกำหนดสไตล์ของเว็บฟอรัม

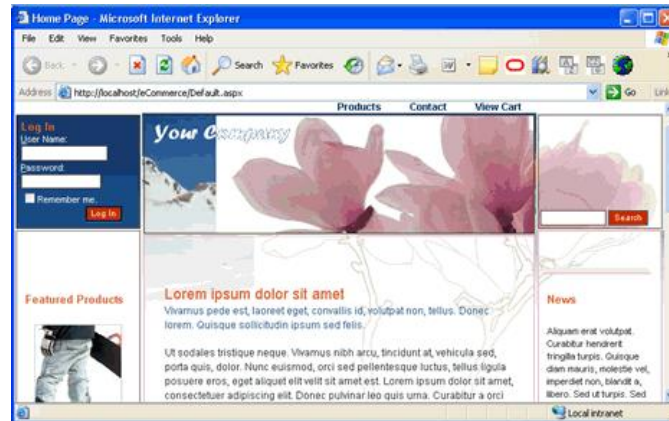
การกำหนด Theme ให้กับเว็บฟอรัม

เราสามารถกำหนด Theme ให้กับเว็บฟอรัมได้ง่ายๆ โดยเข้าไปกำหนดที่ Page Directive ในมุมมอง Source ของเว็บฟอรัม ดังภาพ



จากภาพ จะเห็นว่าเราสามารถเพิ่มแอตทริบิวต์ Theme ให้กับไต่เรคทีฟ Page ซึ่งจะระบุชนิดของ Theme ที่เราต้องการกำหนดให้กับเว็บฟอรัม ในกรณีที่มี Theme อยู่ในเว็บไซต์ ตัว Intelligence ของ Visual Studio 2010 จะตรวจพบโดยอัตโนมัติ พร้อมแสดงรายการ Theme ทั้งหมดที่เราสามารถกำหนดได้ขึ้นมา จะแสดงผลการรันเว็บฟอรัมเดียวกัน แต่ต่าง Theme กัน





จากภาพตัวอย่างของ Theme ทั้งสองแบบ จะเห็นว่าผลของ Theme จะทำให้สีสัน และหน้าตาของ ตัวอักษรมีรูปแบบต่างกัน ที่น่าสนใจมากกว่านั้น คือรูปภาพที่แสดงในแต่ละ Theme จะแตกต่างกันด้วย ซึ่งตรง นี้เทคโนโลยีของสไลด์ซีตเดิมทำได้เหมือนกันผ่านสไลด์ที่ชื่อ background-image แต่จะไม่ยืดหยุ่นเท่า Theme ซึ่งสามารถกำหนดรูปภาพให้กับคอนโทรลที่ต้องการโดยตรง ในการสร้างเว็บไซต์ขึ้นมา เราสามารถใช้คอนโทรล พื้นฐานที่มีมาให้แล้วในทูลบ็อกส์ได้ แต่ในบางครั้งเราอาจไม่พอเพียงสำหรับงานที่ซับซ้อน เราสามารถสร้าง คอนโทรลของเราขึ้นมาเองได้ หรือในบางครั้ง เพื่อให้ได้ตามความต้องการ ก็ต้องเขียนโปรแกรมในเว็บฟอร์มที่ ซับซ้อนมากเพื่อตอบสนองความต้องการนี้

หากเรามองเห็นว่าหลายๆ เว็บฟอร์ม มีการใช้กลุ่มคอนโทรลในลักษณะที่เหมือนๆ กัน เราสามารถนำกลุ่ม คอนโทรลที่เหมือนๆ กันนี้ ไปสร้างเป็นคอนโทรลตัวใหม่ขึ้นมา 1 ตัว แล้วทำการแชร์ให้กับเว็บฟอร์มที่ต้องการ แทน วิธีการนี้ทำให้เราเขียนโค้ดในเว็บฟอร์มสั้นลงได้มาก และเมื่อมีการสร้างเว็บฟอร์มใหม่ที่ต้องการคอนโทรลนี้ อีก เราก็สามารถนำคอนโทรลตัวใหม่นี้มาใช้ได้ หรือเป็นการเพิ่ม Reusability อีกแบบหนึ่ง ในบริษัท ซอร์ฟแวร์ขนาดกลาง ไปถึงขนาดใหญ่ นิยมสร้างคอนโทรลเฉพาะของตัวเองขึ้นมา เพราะประหยัดเวลาในการ เขียนโปรแกรมได้มาก ลดจำนวนโค้ดที่เขียน ลดโอกาสผิดพลาดที่จะเกิดกับเว็บฟอร์มที่ซับซ้อน และ ปัญหาใน การควบคุมมาตรฐานของการเขียนโปรแกรมเมอร์ลดลง บ่อยครั้งที่เราเห็นบริษัทซอร์ฟแวร์ซื้อคอนโทรลจาก บริษัท Third Party เช่น Infragistics Dundas ChartFX ComponentOne หรือ Exceed มาใช้งาน

Control

User Control

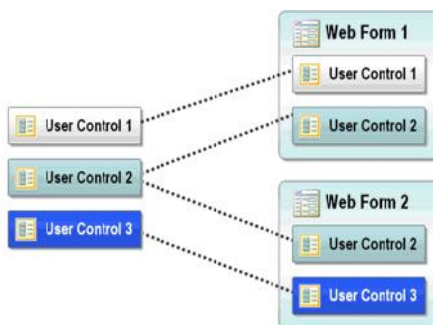
เป็นคอนโทรลที่เกิดจาก คอนโทรลพื้นฐานได้หลายๆ ตัว ในตอนสร้าง เราสามารถลากคอนโทรลจาก ทูลบ็อกส์มาวางได้เหมือนกับเว็บฟอร์ม เราสามารถจัดตำแหน่ง และคุณสมบัติได้จากหน้าจอออกแบบของ Visual Studio เลย นอกจากนี้เราสามารถเขียนโปรแกรมใน Code Behind ได้เหมือนกับเว็บฟอร์ม และเป็น การเขียนโปรแกรมในลักษณะ Event Procedure ก็เช่นเดียวกัน

Custom Server Control

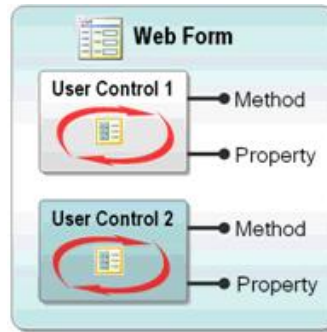
เป็นคอนโทรลที่ไม่มีหน้าออกแบบให้ ทำให้เราต้องเขียนโปรแกรมเพื่อจัดวาง และกำหนดคุณสมบัติต่างๆ เอง ข้อดีของคอนโทรลประเภทนี้คือสามารถคอมไพล์กับคีย์ไฟล์ แล้วลงทะเบียนกับ GAC (Global Assembly Cache) เพื่อทำให้คอนโทรลของเราใช้งานได้กับหลายๆ .NET แอปพลิเคชัน ในการสร้างคอนโทรล Third Party เพื่อขาย ก็จะสร้างคอนโทรลประเภทนี้

รู้จักกับ User Control

User Control เป็นคอนโทรลที่ประกอบด้วยคอนโทรลพื้นฐานที่มีในทูลบ็อกซ์หลายๆ ตัวได้ นอกจากนี้พฤติกรรมของมันยังเหมือนกับเว็บฟอร์มมาก คือมีหน้าจอในการออกแบบที่มุมมอง Design และมุมมอง Source เหมือนกัน เราสามารถลากคอนโทรลหลายๆ ตัวจากทูลบ็อกซ์มาวางที่มุมมอง Design ได้ นอกจากนี้มันยังมี Event ในตัวมันเองได้เช่น Page_Load และ Page_Init และสามารถเรียกใช้อ็อบเจกต์ของเว็บไซต์ได้เช่น Session และ Application อาจจะบอกได้ว่าเว็บฟอร์มทำอะไรได้ User Control นี้ก็ทำได้แทบไม่ต่างกัน แต่เรามักใช้ User Control เพื่อจุดประสงค์ในการเพิ่ม Reusability ให้กับเว็บไซต์ของเราดังภาพ

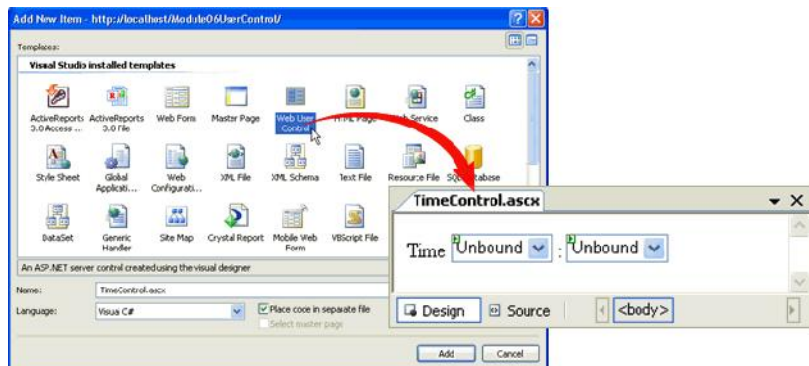


จากภาพ จะเห็นว่า User Control หนึ่งตัวสามารถนำไปใช้กับเว็บฟอร์มหลายๆ ตัวได้ ถ้าหากเราในตอนที่เราออกแบบเว็บฟอร์ม แล้วเรามองเห็นว่าหลายๆ เว็บฟอร์มมีส่วนของหน้าจอที่เหมือนกัน เราสามารถส่วนที่เหมือนกันนี้มาสร้างเป็น User Control ได้ User Control สามารถมี Event หลายๆ อย่างเกิดในตัวมันเองได้ ซึ่งปกติแล้ว เหตุการณ์ในคอนโทรลจะไม่สามารถส่งผ่านไปยังเว็บฟอร์ม หรือคอนโทรลตัวอื่นได้ ยกเว้นว่าเราจะตั้งใจส่งเหตุการณ์ออกไปให้เว็บฟอร์มรับรู้ ซึ่งจะได้พูดถึงในส่วนถัดไป นอกจากนี้มันยังสามารถมีอินเตอร์เฟซเช่น พร็อพเพอร์ตี้ และเมธอดในตัวมันเองเพื่อให้สามารถติดต่อกับเว็บฟอร์ม หรือคอนโทรลตัวอื่นได้ดังภาพ



การสร้าง User Control

ในการสร้าง User Control จะขอยกตัวอย่างการสร้าง TimeControl ซึ่งเป็นคอนโทรลที่ใช้สำหรับบันทึกเวลาของการนัดหมาย หรือการประชุมต่างๆ เหมือนใน Outlook เริ่มต้นหลังจากที่เราเปิดโปรเจกต์เว็บไซต์ขึ้นมาแล้ว ให้คลิกขวาที่โปรเจกต์ แล้วเลือก Add New Item ? เลือก Web User Control จากนั้นตั้งชื่อคอนโทรลว่า TimeControl จะได้หน้า Design ของ User Control เราสามารถลากคอนโทรล DropDownList และเพิ่มเติมคำพูดจนมีหน้าตาภาพ



ค่าที่อยู่ใน DropDownList ตัวแรกชื่อ ddlHour ทำการเก็บค่าชั่วโมง และตัวที่สองชื่อ ddlMinute ทำการเก็บค่านาที เราสามารถกำหนดค่าใน DropDownList ทั้งสองตัวได้ใน Code Behind (TimeControl.ascx.cs) สำหรับชั่วโมงควรกำหนดให้มีค่าตั้งแต่ 0 ถึง 23 และนาทีควรกำหนดให้มีค่าตั้งแต่ 0 ถึง 59 การกำหนดค่านี้เป็นค่าเริ่มต้นของคอนโทรล ดังนั้นเราต้องกำหนดในส่วนของโพรซีเตอร์ OnInit ซึ่งจะเกิดก่อนเหตุการณ์ Page_Load

ADO.NET

ADO.NET นั่นก็คือกลุ่มของอ็อบเจกต์ภายใต้เนมสเปซ System.Data ซึ่งจะทำหน้าที่เป็นสื่อกลางระหว่าง โปรแกรมที่พัฒนาด้วยสถาปัตยกรรม .NET Framework กับ “แหล่งข้อมูล” ซึ่งในที่นี้ อาจจะหมายถึง

ไฟล์ฐานข้อมูลของ Access ไฟล์ Excel ก็ และยังหมายรวมถึง ระบบจัดการฐานข้อมูลโดยเฉพาะอย่างเช่น Microsoft SQL Server หรือว่า Oracle ได้อีกด้วย

ADO.NET นั้น ได้รับการปรับปรุงจาก ADO เวอร์ชันก่อน ให้สนับสนุนการทำงานทั้งแบบ Connected และ Disconnected ซึ่งแต่เดิมนั้น จะสนับสนุนเพียงการทำงานแบบ Connected หรือ แบบเชื่อมต่อกับแหล่งข้อมูลตลอดเวลาเท่านั้น ด้วยการทำงานแบบ Disconnected คุณจึงสามารถที่จะปิดการติดต่อกับแหล่งข้อมูล ในขณะที่ทำการเรียกดู หรือทำการแก้ไขข้อมูล และยังสามารถทำการจัดเรียง (Sort) ข้อมูล หรือแม้กระทั่ง Filter ข้อมูลเพื่อการแสดงผลได้ เป็นการช่วยประหยัดทรัพยากรระบบ โดยเฉพาะอย่างยิ่ง เมื่อเป็นการทำงานร่วมกับระบบจัดการฐานข้อมูลอย่าง Microsoft SQL Server หรือ Oracle ที่เป็นการติดต่อกันระหว่างโปรเซสของ ASP และโปรเซสของตัวระบบจัดการฐานข้อมูล หรือการติดต่อกันระหว่างเครื่องเซิร์ฟเวอร์ 2 เครื่อง

โครงสร้างหลักของ ADO.NET

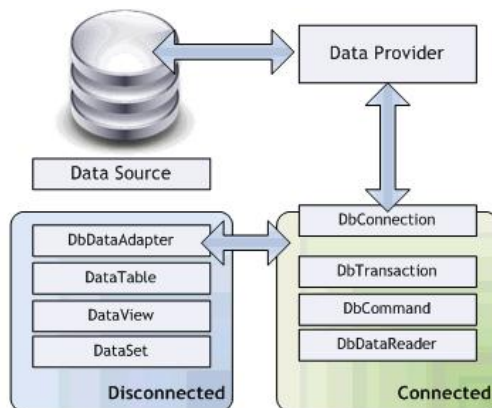
หลังจากเราได้ทำความรู้จักกับ ADO.NET อย่างคร่าวๆ แล้ว เพื่อความเข้าใจในโครงสร้างและการทำงานของ ADO.NET ให้ชัดเจนขึ้น เราจะมาดูโครงสร้างหลักๆ ของ ADO.NET กัน ซึ่งใน ADO.NET นั้นจะประกอบไปด้วย คลาสจำนวนมาก แต่จะสามารถแบ่งออกได้เป็น 2 กลุ่ม คือ

· กลุ่มที่ทำงานกับข้อมูลบนหน่วยความจำ หรือทำงานแบบ Disconnected

คลาสในกลุ่มนี้อาจจะเรียกได้ว่า เป็นคลาสในกลุ่มที่ได้รับการพัฒนาขึ้นมาใหม่ และเป็นเอกลักษณ์ของ ADO.NET เลยก็ว่าได้ ซึ่งคลาสกลุ่มนี้ จะเป็นกลุ่มที่ใช้ในการจำลองโครงสร้างของข้อมูล ให้มีลักษณะคล้ายคลึงกับข้อมูลจริงที่ถูกเก็บอยู่ในฐานข้อมูลให้มากที่สุด โดยการจำลอง ตาราง (Table) 필ด์ (Field) และเรคคอร์ด (Record) หรือแม้กระทั่งความสัมพันธ์ระหว่างตาราง (Data Relation) จากแหล่งข้อมูล มาไว้ในหน่วยความจำหลัก ซึ่งผู้พัฒนาจะสามารถเรียกใช้ และแก้ไขข้อมูลที่อยู่ในคลาสนี้ได้ เสมือนว่ากำลังทำงานกับฐานข้อมูลจริง ก่อนที่จะทำการ Update ข้อมูลจากคลาสนี้ กลับไปยังแหล่งข้อมูล

· กลุ่มที่ทำงานกับแหล่งข้อมูลโดยตรง หรือทำงานแบบ Connected

คลาสในกลุ่มนี้จะเป็นคลาสที่ทำการสร้าง Connection กับแหล่งข้อมูลโดยตรง ไม่ว่าจะเป็นไฟล์ หรือระบบจัดการฐานข้อมูลก็ตาม และทำการอ่าน หรือแก้ไขข้อมูลไปยังแหล่งข้อมูลนั้น ซึ่งในคลาสนี้จะเกี่ยวข้องกับการเปิด Connection ไปยังระบบจัดการฐานข้อมูล และการใช้คำสั่ง SQL เพื่อทำการเรียกดู หรือว่าแก้ไขข้อมูล โดยคลาสนี้จะมีชื่อเรียกเป็นทางการว่า .NET Data Provider



ทำความรู้จักกับ .NET Data Provider

อย่างที่เรทราบกันดีแล้วว่า “แหล่งข้อมูล” ในปัจจุบันนี้มีอยู่เป็นจำนวนมาก แม้ว่าผู้ที่พัฒนาด้วยสถาปัตยกรรม .NET Framework มักจะนิยมใช้งาน Microsoft SQL Server ก็ตาม แต่ก็อาจมีกรณีที่คุณต้องการสร้างโปรแกรมเพื่อติดต่อกับฐานข้อมูลอื่น ๆ เช่นกัน แต่ไม่ว่า “แหล่งข้อมูล” เบื้องหลังนั้นจะเป็นระบบใด ผู้พัฒนาก็ยังสามารถทำงานกับแหล่งข้อมูลเหล่านั้น ผ่าน ADO.NET ได้เหมือนเดิม นั่นก็เพราะว่า ADO.NET ได้มีการกำหนดโครงสร้างที่เรียกว่า .NET Data Provider เอาไว้นั่นเอง การที่ ADO.NET นั้น เลือกใช้การกำหนดโครงสร้างโปรแกรมที่เรียกว่า API ที่สามารถทำงานได้กับฐานข้อมูลทุกประเภทอย่างที่เราเคยพบใน OleDb หรือ ODBC นั้นก็เป็นเพราะเหตุผลทางด้านประสิทธิภาพในการทำงาน ถึงแม้ว่าระบบจัดการฐานข้อมูล แม้จะสามารถติดต่อกับโปรแกรมผ่านทางอินเทอร์เน็ตเฟสกลางอย่าง OleDb หรือ ODBC ได้ แต่เบื้องหลังแล้ว การทำงาน หรือแม้กระทั่งวิธีที่ระบบจัดการฐานข้อมูลนั้นใช้ในการเก็บข้อมูลย่อมแตกต่างกันโดยสิ้นเชิง ซึ่งแม้แต่วิธีที่ ODBC นั้นใช้ในการเก็บข้อมูล ก็อาจจะแตกต่างกับตัวภาษาที่นำ ODBC ไปใช้งานอีกด้วย ทำให้เกิดการแปลงข้อมูลไป-มา ระหว่างตัวโปรแกรมที่เรียกใช้ กับตัว ODBC เอง และยังอาจมีการแปลงข้อมูลระหว่างตัว ODBC กับระบบจัดการฐานข้อมูลอีกชั้นหนึ่ง และการที่มีอินเทอร์เน็ตเฟสกลางนั้น ก็จะเป็นการจำกัดให้ผู้พัฒนา ไม่สามารถใช้ ความสามารถพิเศษ หรือแม้แต่วิธีการเก็บข้อมูล ที่มีเฉพาะในระบบจัดการฐานข้อมูลนั้นๆ ได้ Provider จึงเป็นทางออกในการแก้ปัญหาความไม่ลงรอยกันระหว่างตัวภาษา ตัว API (เช่น ODBC) และระบบจัดการฐานข้อมูลได้เป็นอย่างดี เนื่องจาก .NET Data Provider นั้น เป็นที่แน่นอนว่าย่อมต้องได้รับการพัฒนาด้วยภาษา และ Data Type ที่ CLR (Common Language Runtime) รองรับ จึงตัดปัญหาความไม่เข้ากันของข้อมูลระหว่างตัว API และภาษาที่เรียกใช้ได้ และเจ้าของโปรแกรมเอง ก็สามารถพัฒนาอินเทอร์เน็ตเฟสการติดต่อของตนได้อย่างเต็มที่ เพื่อให้ผู้พัฒนาสามารถใช้งานฟีเจอร์พิเศษของระบบจัดการฐานข้อมูลได้ (อย่างเช่น Fetch Size ใน Oracle Data Provider) รวมไปถึงการสร้าง Data Type ใหม่ขึ้นมาเพื่อรองรับรูปแบบของข้อมูลเฉพาะ ในระบบจัดการฐานข้อมูลของตนได้อีกด้วย

Provider มาตรฐานใน ADO.NET สำหรับ .NET Framework 2.0

ใน .NET Framework 2.0 นั้น ทางทีมพัฒนา ได้มีการสร้างโพรไวเดอร์มาตรฐานไว้ดังนี้

แหล่งข้อมูล	เนมสเปซของ Provider
Microsoft SQL Server 7.0 ขึ้นไป	System.Data.SqlClient
Oracle 8.1.6 ขึ้นไป	System.Data.OracleClient
SqlXml ใน SQL Server	System.Data.SqlXml
ODBC DataSource	System.Data.ODBC
OleDb	System.Data.OleDb

จะเห็นได้ว่า ODBC และ OleDb นั้น ก็ยังคงได้รับการพัฒนาให้เป็น Data Provider ใน .NET Framework อยู่ นั่นก็เพื่อที่ว่า สถาปัตยกรรม .NET จะได้สามารถทำงานได้กับระบบจัดการฐานข้อมูลที่รองรับ ODBC และไฟล์ฐานข้อมูลของ Access และ Excel ผ่านทาง OleDb ได้ เช่นเดียวกับ ADO Classic แน่นอนว่าคุณสามารถเลือกที่จะติดต่อกับ SQL Server หรือ Oracle ผ่านทาง ODBC ได้เหมือนเคย แต่อย่าลืมว่า คุณจะต้องประสบกับข้อจำกัดของ API ที่สามารถทำงานได้กับฐานข้อมูลทุกรูปแบบ อย่างที่ผู้เขียนได้กล่าวถึงไปแล้ว โดยสรุปอีกครั้งหนึ่งนั่นก็คือ

- ประสิทธิภาพในการทำงานที่ลดลง จาก Overhead ในการแปลงข้อมูล ไปมา
- ความไม่เข้ากันระหว่างรูปแบบการจัดเก็บข้อมูล ซึ่งใน .NET จะทำให้เกิดการ Boxing-Unboxing รวมไปถึงความสามารถในการใช้งาน Type เฉพาะสำหรับระบบการจัดการฐานข้อมูลนั้นๆ
- และ ท้ายที่สุดคือ การสูญเสียความสามารถในการเรียกใช้พีเจอร์เฉพาะบางอย่าง ที่คุณอาจจะไม่สามารถเรียกใช้ได้ผ่านทาง ODBC ภายใน Visual Studio 2010 โดยเฉพาะอย่างยิ่ง ในส่วนของ Web Development นั้น ได้รับการพัฒนาในด้านการเชื่อมต่อ และทำงานกับฐานข้อมูลไปมาก จนคุณสามารถสร้าง Website ด้วย ASP.NET ที่มีการแสดงผล แก๊ซ หรือค้นหาข้อมูลได้ โดยแทบจะไม่จำเป็นต้องเขียนโค้ดเลยแม้แต่บรรทัดเดียว ซึ่งในความเห็นของผู้เขียนแล้ว นับว่าเป็นสิ่งที่อันตรายมาก สำหรับผู้ที่ไม่ได้ผ่านการใช้งาน ASP.NET ในเวอร์ชันเก่า หรือการเขียนโปรแกรมด้วย .NET ในโปรเจกต์อื่นมาก่อน เพราะความสะดวก และง่ายของ Wizard และ Tool นี้ จะเป็นการซ่อนขั้นตอนพื้นฐานในการใช้งาน ADO.NET จริงๆ ไว้ทั้งหมด นั่นก็หมายความว่า คุณจะไม่สามารถใช้ความรู้ความชำนาญจากการพัฒนา Web Application ด้วย ASP.NET ที่ติดต่อกับฐานข้อมูลไปใช้กับโปรเจกต์อื่นๆ ได้เลย พื้นฐานการติดต่อกับฐานข้อมูลนั้น อาจจะเป็นเรื่องที่ยากและไกลตัว แต่อันที่จริงแล้ว มีอ็อบเจกต์ที่เกี่ยวข้องเพียงแค่ 4 ชนิดเท่านั้น และมีขั้นตอนการทำงานที่ตายตัว ไม่ว่าคุณจะใช้โพรไวเดอร์ตัวใดก็ตาม ซึ่งอ็อบเจกต์ทั้ง 4 ชนิดนั้น ประกอบไปด้วย

· **Data Connection** เป็นอ็อบเจกต์ที่ทำหน้าที่สร้างการเชื่อมต่อกับระบบจัดการฐานข้อมูล โดยคุณสามารถระบุพารามิเตอร์ในการใช้งานระบบจัดการฐานข้อมูลได้ ผ่านทาง **Connection String** ที่ภายในจะประกอบไปด้วย ชื่อของเครื่องที่เป็นเซิร์ฟเวอร์ฐานข้อมูล ชื่อฐานข้อมูล ชื่อผู้ใช้ และพาสเวิร์ด แต่ทั้งนี้ สิ่งที่คุณจะสามารถกำหนดได้ใน Connection String ก็จะไม่แตกต่างกันไปในแต่ละโปรแกรม

· **Command** เมื่อคุณสามารถสร้างการเชื่อมต่อกับระบบจัดการฐานข้อมูลได้เรียบร้อยแล้ว ในขั้นตอนต่อไป คุณก็จะเริ่มสั่งงานให้ระบบจัดการฐานข้อมูลทำงานด้วยคำสั่งในภาษา SQL โดยการใช้อ็อบเจกต์ประเภท Command ถ้าหากว่าคำสั่ง เป็นประเภท INSERT UPDATE DELETE การทำงานกับระบบฐานข้อมูลก็จะสิ้นสุดที่การเรียกให้ Command สั่งงานไปยังฐานข้อมูล และปิดการเชื่อมต่อ แต่ถ้าหากว่าเป็นการเรียกดูข้อมูลจากฐานข้อมูล ด้วยคำสั่ง SELECT หรือการใช้ Stored Procedure ก็จะต้องมีขั้นตอนในการอ่านข้อมูลที่เป็นผลลัพธ์เพิ่มขึ้นอีก

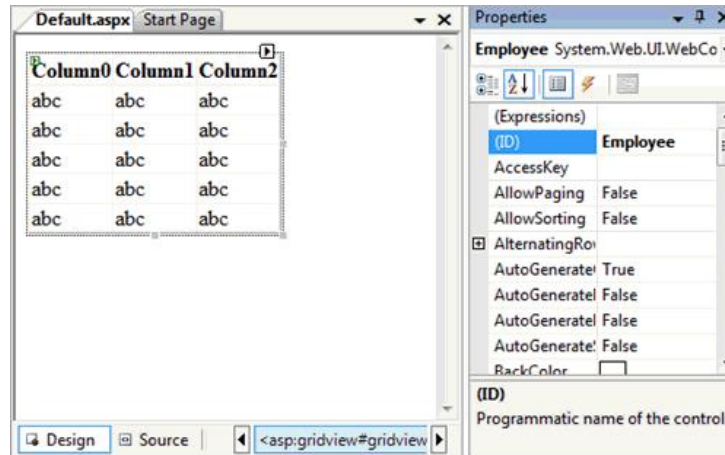
· **DataReader** จะเป็นอ็อบเจกต์ที่ใช้ในการอ่านข้อมูลที่เป็นผลลัพธ์จากคำสั่ง SELECT หรือ Stored Procedure ในลักษณะครั้งละเรคคอร์ด จนหมด และไม่สามารถย้อนกลับไปอ่านเรคคอร์ดที่ผ่านมาได้ (Forward-Only) จึงเหมาะกับการใช้งานกับผลลัพธ์ที่จะไม่มีการ Sort หรือ Filter ข้อมูลทางฝั่ง ASP ภายหลัง DataReader จึงมีจุดเด่นในเรื่องประสิทธิภาพ และการใช้งานหน่วยความจำที่น้อยกว่าการนำข้อมูลทั้งหมดขึ้นมาในคราวเดียว การทำงานของ DataReader นั้น จำเป็นจะต้องมีการเชื่อมต่อกับแหล่งข้อมูลตลอดเวลา จนกว่าข้อมูลทั้งหมดจะถูกอ่านขึ้นมาได้ นับเป็นการทำงานกับข้อมูลในแบบ **Connected**

· **Data Adapter** นั้น จะเป็นอ็อบเจกต์ทำการอ่านข้อมูลทั้งหมดที่เป็นผลลัพธ์ของคำสั่ง SELECT หรือ Stored Procedure ขึ้นมาเก็บไว้ในอ็อบเจกต์ **DataSet** แล้วจึงตัดการเชื่อมต่อกับระบบจัดการฐานข้อมูล จึงเหมือนเป็นการจำลองโครงสร้างของตารางที่เป็นผลลัพธ์ไว้ภายในหน่วยความจำ (ลักษณะคล้ายการ Cache) คุณสามารถเรียกดูข้อมูลเรคคอร์ดใด ๆ ก็ได้แม้ว่าการติดต่อกับระบบจัดการฐานข้อมูลจะถูกปิดไปแล้วก็ตาม หรือเป็นการทำงานกับข้อมูลแบบ **Disconnected** นั้นเอง และ นอกจากนี้ คุณยังสามารถใช้พีแอร์ของ ADO.NET ในการ Sort หรือ Filter ข้อมูลจาก DataSet ได้อีกด้วย การใช้ DataAdapter และ DataSet นั้น จึงจะเหมาะสมกว่ากับการใช้งานทั่วไปใน ASP.NET แต่ข้อเสียของการใช้ Data Adapter และ DataSet นั้นก็คือ ปริมาณหน่วยความจำที่ใช้ เนื่องจากข้อมูลทั้งหมดจะต้องถูกนำขึ้นมาเก็บไว้ในหน่วยความจำหลัก และยังมี Overhead จากการทำ Boxing-Unboxing เมื่อมีการเรียกดูข้อมูลอีกด้วย เนื่องจาก DataSet นั้นจะเก็บข้อมูลโดยใช้ตัวแปรประเภท Object

การแสดงผลข้อมูลแบบ Connected

หลังจากที่เราได้ทราบถึงอ็อบเจกต์ที่เกี่ยวข้องกับการติดต่อฐานข้อมูลกันแล้ว เราจะทดลองนำอ็อบเจกต์เหล่านั้น มาใช้ในการเรียกดูข้อมูลจากฐานข้อมูล pubs ที่เราได้ทำการติดตั้งไว้ เพื่อมาแสดงผลใน

GridView สิ่งแรกที่คุณจะต้องทำก็คือ สร้างเว็บไซต์ใหม่ใน Visual Studio 2010 หรือ Visual Web Developer Express จากนั้น ลาก GridView จากทูลบ็อกซ์นำออกมาวางไว้ในเว็บฟอร์ม Default.aspx และตั้งชื่อว่า Author สำหรับ GridView นั้น เราจะมาพูดถึงรายละเอียดการใช้งานอย่างละเอียดในบทถัดไป



จากนั้น เราจะเขียนโค้ดในเหตุการณ์ Page_Load เพื่อเปิดการติดต่อกับฐานข้อมูล ด้วยโค้ดด้านล่างนี้

```
using ( SqlConnection connection = new SqlConnection())
```

```
    connection.ConnectionString =
```

```
        @"Data Source=.\SQLEXPRESS;Initial Catalog=pubs;Integrated Security=True";
```

```
    connection.Open();
```

เนื่องจากการเปิด Connection ไปยัง Microsoft SQL Server Express นั้น มีการเรียกใช้งาน Resource ภายนอก (สังเกตได้จากการที่อ็อบเจกต์นั้นมีคำสั่ง Dispose) ซึ่ง Garbage Collector ของ CLR ไม่สามารถช่วยเราจัดการหน่วยความจำได้ จึงควรจะต้องใช้ using block เพื่อเป็นการกำหนดให้ CLR นั้น เรียกคำสั่ง Dispose ของอ็อบเจกต์ และจัดการล้างหน่วยความจำให้เมื่อการทำงานของโค้ดนั้น ออกจาก block ของ using ไป ซึ่งอันที่จริงแล้ว จะมีผลเหมือนกับการเรียกคำสั่ง Dispose ด้วยตัวเอง แต่การใช้ using block นั้นจะทำให้ Code ดูง่ายขึ้น สามารถมองเห็นได้ชัดเจนว่าอ็อบเจกต์มีการใช้งานตั้งแต่ส่วนไหน จนถึงส่วนไหน และสามารถมั่นใจได้ว่า อ็อบเจกต์นั้นจะถูก Dispose อย่างแน่นอน แม้ว่าจะมี Exception หรือว่าการเปลี่ยน Scope อย่างไม่ตั้งใจก็ตาม หลังจากเปิด Connection แล้ว ขั้นตอนต่อไปคือการสร้าง Command ขึ้นมา เพื่อสั่งให้ Microsoft SQL Server นั้นดึงข้อมูลออกมาจากฐานข้อมูล อ็อบเจกต์ Command นั้นก็มีการเรียกใช้งาน Resource ภายนอก

เช่นเดียวกัน ดังนั้น จึงควรใช้งานอ็อบเจกต์ Command นี้ใน using blockเช่นเดียวกับการใช้อ็อบเจกต์ Connection

```
using ( SqlConnection connection = new SqlConnection())

    connection.ConnectionString =

        @"Data Source=.\SQLEXPRESS;Initial Catalog=pubs; " +

            "Integrated Security=True";

    connection.Open();

    using (SqlCommand command = new SqlCommand())

        command.CommandText = "SELECT * FROM authors";

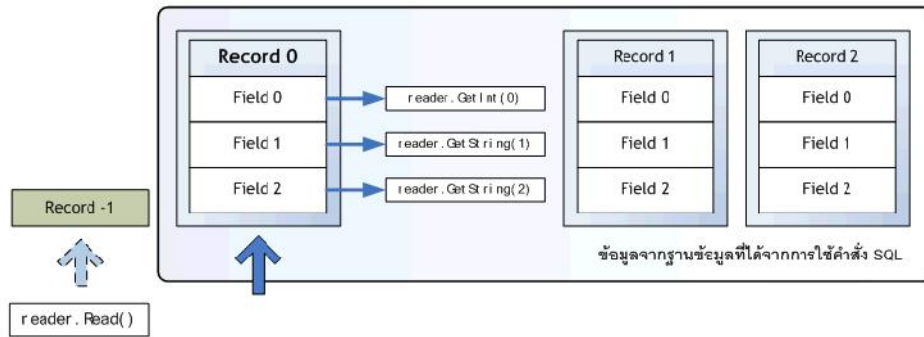
        command.Connection = connection;

        command.ExecuteReader();
```

จากนั้น เมื่อเราทำการเรียกใช้งานคำสั่ง ExecuteReader ก็จะเป็นการสั่งงานให้ Microsoft SQL Server นั้น Execute คำสั่ง SQL ที่กำหนดไว้ และคืนค่าออกมาเป็น SqlDataReader เพื่อใช้สำหรับการอ่านข้อมูล

การใช้งาน Data Reader เพื่ออ่านข้อมูลจากแหล่งข้อมูล

การอ่านข้อมูลของ Data Reader นั้นจะเป็นการอ่านครั้งละฟิลด์ ภายใน 1 เรคคอร์ด คล้ายกับอ็อบเจกต์ Record Set ของคลาสสิก ADO โดยในครั้งแรกนั้น ตัว DataReader จะอยู่ที่ตำแหน่งเรคคอร์ดที่ -1 หรืออาจเรียกว่าเรคคอร์ด ก่อน เรคคอร์ดแรก ซึ่งถือเป็นเรคคอร์ดที่ใช้สำหรับระบุว่า DataReader นั้น ยังไม่ได้ทำการอ่านเรคคอร์ดใดๆ ขึ้นมา คุณสามารถเลื่อนตำแหน่งการอ่านของ DataReader ไปข้างหน้า ครั้งละ 1 เรคคอร์ดได้เรื่อยๆ โดยการเรียกใช้ฟังก์ชัน Read ของ Data Reader ซึ่งเมธอด Read จะคืนค่าเป็น True ถ้าหากว่า DataReader นั้น สามารถอ่านข้อมูลจากเรคคอร์ดตัวถัดไปขึ้นมาได้ หลังจากนั้นคุณสามารถใช้คำสั่ง GetInt หรือ GetString เพื่ออ่านข้อมูลจากฟิลด์ที่ต้องการจากเรคคอร์ดนั้นได้



สำหรับโค้ดตัวอย่าง นี้ จะทำการอ่านข้อมูลจากตาราง authors ขึ้นมาแสดงผลภายในคอนโทรล Literal ซึ่งผลลัพธ์ที่ได้ ก็จะเป็นข้อความแสดงชื่อ และนามสกุลของนักเขียน จากตาราง authors นั้นเอง

```
using ( SqlConnection connection = new SqlConnection() )

    connection.ConnectionString =

        @"Data Source=.\SQLEXPRESS;Initial Catalog=pubs;" +

        "Integrated Security=True";

    connection.Open();

    using ( SqlCommand command = new SqlCommand() )

        command.CommandText = "SELECT * FROM authors";

        command.Connection = connection;

        using ( SqlDataReader reader = command.ExecuteReader() )

            StringBuilder sb = new StringBuilder();
```

```
int record = 0;

while ( reader.Read() )

    sb.AppendFormat( "Record #{0}:<BR />", record );

    sb.AppendFormat( "au_lname: {0}<BR/>", reader.GetString( 2 ) );

    sb.AppendFormat( "au_fname: {0}<BR/>", reader.GetString( 1 ) );

    record += 1;

litOutput.Text = sb.ToString();
```

นอกจากนี้แล้ว คุณยังสามารถนำ DataReader นี้ไปเป็น DataSource ให้กับคอนโทรล GridView ได้ทันที โดยการกำหนด DataReader ให้กับพร็อพเพอร์ตี้ DataSource ของคอนโทรล GridView และเรียกเมธอด DataBind เพื่อให้ GridView เริ่มอ่านข้อมูลจาก DataSource ซึ่ง DataReader นั้นก็ควรจะอยู่ใน using block เช่นเคย

```
using ( SqlConnection connection = new SqlConnection()

    connection.ConnectionString = @"Data Source=.\SQLEXPRESS;Initial Catalog=pubs;Integrated Security=True";

    connection.Open();

    using (SqlCommand command = new SqlCommand()

        command.CommandText = "SELECT * FROM authors";

        command.Connection = connection;

        using (SqlDataReader reader = command.ExecuteReader())

            this.Author.DataSource = reader;
```

```
this.Author.DataBind();
```

การทำงานของโค้ดข้างต้น จะให้ผลลัพธ์ตามรูป

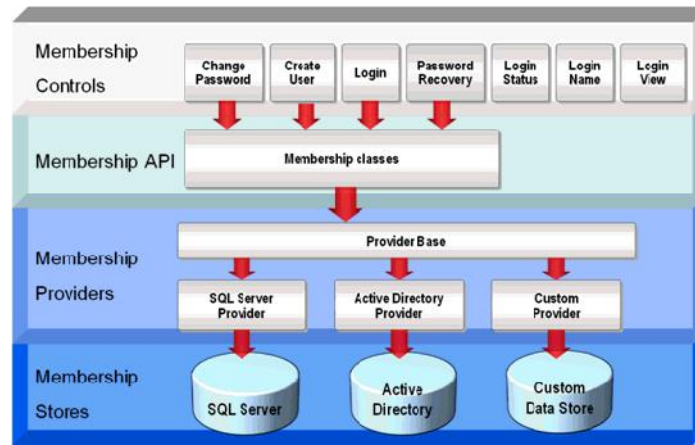
au_id	au_lname	au_fname	phone	address	city	state	zip	contract
172-32-1176	White	Johnson	408 496-7223	10932 Bigge Rd.	Menlo Park	CA	94025	<input checked="" type="checkbox"/>
213-46-8915	Green	Marjorie	415 986-7020	309 63rd St. #411	Oakland	CA	94618	<input checked="" type="checkbox"/>
238-95-7766	Carson	Cheryl	415 548-7723	589 Darwin Ln.	Berkeley	CA	94705	<input checked="" type="checkbox"/>
267-41-2394	O'Leary	Michael	408 286-2428	22 Cleveland Av. #14	San Jose	CA	95128	<input checked="" type="checkbox"/>
274-80-9391	Straight	Dean	415 834-2919	5420 College Av.	Oakland	CA	94609	<input checked="" type="checkbox"/>
341-22-1782	Smith	Meander	913 843-0462	10 Mississippi Dr.	Lawrence	KS	66044	<input type="checkbox"/>
409-56-7008	Bennet	Abraham	415 658-9932	6223 Bateman St.	Berkeley	CA	94705	<input checked="" type="checkbox"/>
427-17-2319	Dull	Ann	415 836-7128	3410 Blonde St.	Palo Alto	CA	94301	<input checked="" type="checkbox"/>
472-27-2349	Gringlesby	Burt	707 938-6445	PO Box 792	Covelo	CA	95428	<input checked="" type="checkbox"/>
486-29-1786	Locksley	Charlene	415 585-4620	18 Broadway Av.	San Francisco	CA	94130	<input checked="" type="checkbox"/>

อย่าลืมว่าถ้าหากว่าคุณไม่ได้ใช้งาน using block คุณจะต้องทำการปิด DataReader และ Connection เองทุกครั้ง หลังจากใช้งานเสร็จ ด้วยเมธอด Close หรือ Dispose เสมอ เพื่อคืนหน่วยความจำที่ ี ้อปเจ็กต์นั้นเรียกใช้งานกลับสู่ระบบ

ส่วนประกอบที่สำคัญ และจำเป็นต้องมีในเว็บไซต์สมัยใหม่คือ ความเป็นสมาชิก หากเรามีความเป็นสมาชิกกับเว็บไซต์แห่งหนึ่งแล้ว เราจะได้รับสิ่งที่ติดตัวเราคือ สิ่งที่แสดงตัวตนของเรา หรือแอคเคาต์ (Account) และสิทธิ์ที่เราจะได้รับจากการเป็นสมาชิก (Authorization) ใน ASP.NET 1.x เราสามารถกำหนดแอคเคาต์ และสิทธิ์ของสมาชิกแบบง่ายๆ ได้ในไฟล์คอนฟิกูเรชัน web.config แต่วิธีนี้จะใช้กับเว็บไซต์ที่มีขนาดเล็ก คือจะมีสมาชิกในเว็บไซต์ไม่มากนัก เช่น 10-20 คน เพราะไม่มีเครื่องมือตัวใดพิเศษที่ช่วยจัดการ แต่ถ้าหากมีสมาชิกมากกว่านั้น และต้องการจัดการกับแอคเคาต์ และสิทธิ์ของสมาชิกแบบที่ซับซ้อน เราต้องเขียนโค้ดที่ติดต่อกับฐานข้อมูลเอง ซึ่งจะต้องเสียเวลาพอสมควร เพื่อจัดการกับความเป็นสมาชิกที่เป็นสิ่งจำเป็นนี้ ASP.NET 2.0 ได้สร้างคอนโทรล API และโพรไวเดอร์ที่ใช้ติดต่อกับฐานข้อมูลสำหรับความเป็นสมาชิกโดยเฉพาะ ซึ่งทำให้เราประหยัดเวลามาก บางทีเราแทบจะไม่ต้องเขียนโค้ดเลยหากใช้ฐานข้อมูล SQL Server แต่ถ้าใช้กับฐานข้อมูลอื่น เราต้องสร้างโพรไวเดอร์ขึ้นมาเองที่สามารถติดต่อกับฐานข้อมูลเอง แต่การเขียนโค้ดที่อยู่เหนือระดับของโพรไวเดอร์นั้นไม่ต้องมีเปลี่ยนแปลงใดๆ

การสร้าง Member ใน ASP.NET 2.0

โครงสร้างของความเป็นสมาชิกได้ถูกแบ่งออกเป็นส่วนๆ เพื่อ่ายในการจัดการ ดังภาพ



รายละเอียดของส่วนย่อยแต่ละส่วนของความเป็นสมาชิกมีดังนี้

- **คอนโทรลความเป็นสมาชิก (Membership Controls)** เป็นคอนโทรลกลุ่ม Login ของทุลบล็อกซ์ ใน Visual Studio 2005 ซึ่งจะเป็นกลุ่มคอนโทรลที่จัดการเกี่ยวกับความเป็นสมาชิกทั้งหมดตั้งแต่ การสร้างสมาชิก การล็อกอิน การเปลี่ยนรหัสผ่าน และแจ้งรหัสผ่านในกรณีลืม เราสามารถลากไปใช้งานบนเว็บฟอร์มได้ทันที ทำให้เราประหยัดเวลาในการเขียนโปรแกรมมาก นอกจากนี้คอนโทรลเหล่านี้ทำงานสัมพันธ์กับคลาส Membership API เพื่อรับ และส่งข้อมูลความสัมพันธ์อยู่แล้ว จึงทำให้เราไม่ต้องเสียเวลาเขียนโปรแกรมเพื่อจัดการกับคอนโทรลในกลุ่มนี้มากนัก
- **API ความเป็นสมาชิก (Membership API)** เป็นกลุ่มของคลาสที่ทำงานร่วมกันเพื่อรับ และส่งข้อมูลระหว่างคอนโทรลความเป็นสมาชิก และโพรไวเดอร์ของฐานข้อมูลแต่ละชนิด
- **โพรไวเดอร์ความเป็นสมาชิก (Membership Providers)** เป็นกลุ่มของคลาสที่ทำหน้าที่ติดต่อกับฐานข้อมูลแต่ละประเภท ใน ASP.NET 2.0 จะมีโพรไวเดอร์ของฐานข้อมูล SQL Server และ Active Directory ติดมาด้วย และสามารถเรียกใช้งานได้เลย หากต้องการใช้ฐานข้อมูลอื่นจะต้องเขียนโพรไวเดอร์เฉพาะขึ้นมาเอง (Custom Providers)
- **ฐานข้อมูลความเป็นสมาชิก (Membership Stores)** เป็นฐานข้อมูลที่เก็บความเป็นสมาชิก ซึ่งปกติภายในฐานข้อมูลนี้ จะประกอบไปด้วยตารางที่เก็บข้อมูลสมาชิก และบทบาทของสมาชิก

การเตรียมใช้งานความเป็นสมาชิก

จากโครงสร้างของความเป็นสมาชิกในหัวข้อที่ผ่านมา ส่วนของคอนโทรล และ API ความเป็นสมาชิกนั้น ถูกสร้างมาเรียบร้อยแล้ว และพร้อมใช้งานได้ทันที แต่ในส่วนของโปรแกรมที่ติดต่อกับฐานข้อมูล และตัวฐานข้อมูลนั้น เราต้องทำการสร้างขึ้นมาเอง ดังนั้นการเตรียมการใช้งานความเป็นสมาชิกจึงมีงานหลักๆ คือการสร้างโปรแกรม และเตรียมพร้อมฐานข้อมูลเป็นส่วนใหญ่ ใน Visual Studio 2010 ได้สร้างโปรแกรมสำหรับฐานข้อมูล SQL Server และ Active Directory ไว้ให้แล้ว ซึ่งในที่นี้จะขอแสดงการใช้งานโปรแกรม และฐานข้อมูล SQL Server ก่อน เพราะมีการใช้งานไม่ซับซ้อนนัก และใช้งานได้ง่าย เมื่อเราเลือกใช้โปรแกรมที่เข้ากับฐานข้อมูล SQL Server แล้ว งานหลักที่เหลือจะเป็นการเตรียมฐานข้อมูล SQL Server และการกำหนดค่าต่างๆ เกี่ยวกับความเป็นสมาชิกในไฟล์คอนฟิกูเรชัน web.config เท่านั้น การเตรียมฐานข้อมูล SQL Server เพื่อใช้เก็บข้อมูลความเป็นสมาชิก เราสามารถเลือกฐานข้อมูลได้ 2 แบบคือ

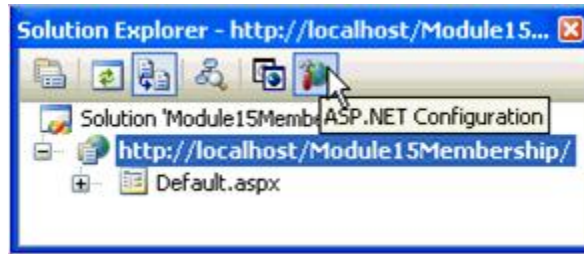
1. **ฐานข้อมูล SQL Server ดีฟอลต์** เป็นฐานข้อมูลที่จะถูกสร้างขึ้นโดยอัตโนมัติ เมื่อคลิกที่แท็บความปลอดภัย (Security) ของเครื่องมือ Website Administration Tool (WAT)
2. **ฐานข้อมูล SQL Server ที่สร้างขึ้นมาเองโดยเฉพาะ** เป็นฐานข้อมูลที่เราต้องสร้างขึ้นมาเองใน SQL Server จากนั้นจึงรันสคริปต์ที่ติดมากับ Visual Studio 2010 เพื่อสร้างตารางความเป็นสมาชิกเอง

การใช้งานฐานข้อมูลทั้งสองแบบ จะกล่าวถึงรายละเอียดในหัวข้อถัดไป

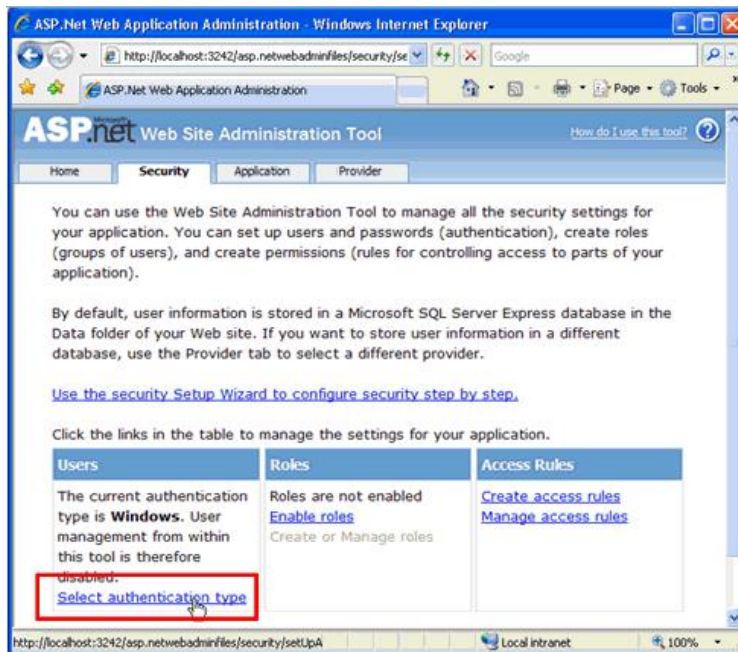
การใช้งานฐานข้อมูล SQL Server ดีฟอลต์

ฐานข้อมูลแบบนี้ สามารถสร้างได้ง่าย เพราะจะถูกสร้างขึ้นโดยอัตโนมัติผ่านเครื่องมือ WAT เหมาะกับการสร้างฐานข้อมูลความเป็นสมาชิกสำหรับเว็บไซต์ขนาดเล็กถึงขนาดกลาง แต่ก็มีข้อเสียคือฐานข้อมูลชนิดนี้ จะถูกเก็บเอาไว้ในเว็บไซด์เอง ซึ่งจะเป็นไฟล์ที่ชื่อ ASPNETDB.MDF ที่เก็บอยู่ในโฟลเดอร์ App_Data บางครั้งไฟล์นี้มีชื่อเรียกว่าเป็น ฐานข้อมูลแบบไฟล์ (File Database) เมื่อมันทำงาน ไฟล์นี้จะถูกนำไปผูกกับฐานข้อมูล (Attach Database) SQL Server Express โดยอัตโนมัติเมื่อถูกเรียกโดยเว็บฟอร์ม และทำงานอยู่ในหน่วยความจำเท่านั้น ดังนั้นเราจึงไม่สามารถเข้าไปจัดการฐานข้อมูลชนิดนี้ผ่านเครื่องมือ SQL Server Management Studio ได้ แต่จะสามารถเข้าไปจัดการผ่าน Server Explorer ได้เท่านั้น นอกจากนี้ยังไม่เหมาะกับเว็บไซต์ที่รันอยู่บนเว็บเซิร์ฟเวอร์หลายๆ ตัว (Web Farm)

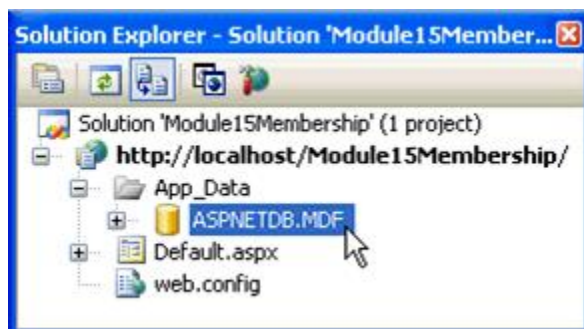
วิธีการสร้างฐานข้อมูลนี้ ให้เราทำการเปิดเครื่องมือ WAT โดยคลิกที่ไอคอน ASP.NET Configuration ใน Solution Explorer ของ Visual Studio 2010 ดังภาพ



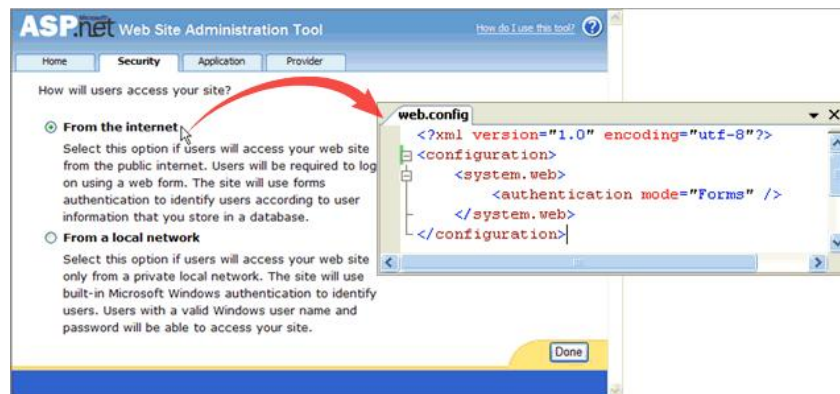
หลังจากที่คลิกที่ไอคอน ASP.NET Configuration แล้ว เครื่องมือ WAT จะถูกเปิดใน Internet Explorer ดังภาพ



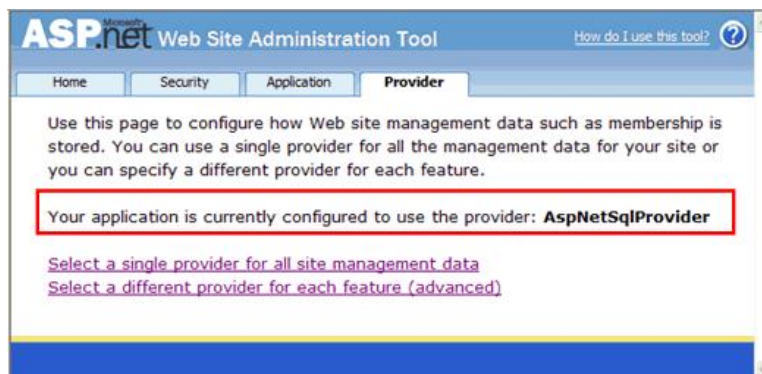
เครื่องมือ WAT สามารถจัดการค่าเกี่ยวกับความปลอดภัย กำหนดค่าเริ่มต้น และกำหนดค่าไพรไวดอร์ ให้กับเว็บไซต์ ในที่นี้จะขอเริ่มต้นจากแท็บแรกคือ Security เมื่อเราเข้าคลิกเข้ามาที่แท็บนี้ Visual Studio 2010 จะทำการสร้างฐานข้อมูล ASPNETDB.MDF ที่ใช้สำหรับเก็บข้อมูลความเป็นสมาชิกในโฟลเดอร์ App_Data ของเว็บไซต์โดยอัตโนมัติ ดังภาพ



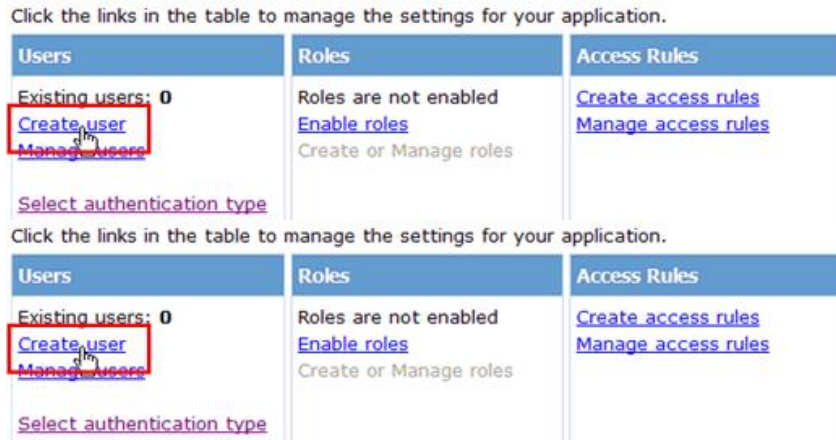
เพื่อให้เราสามารถใช้งานความเป็นสมาชิก ขั้นตอนแรกเราต้องทำการกำหนดวิธีการเข้าสู่ระบบให้เป็นแบบฟอร์ม (Form Authentication) ก่อน เพราะความเป็นสมาชิกใน ASP.NET 2.0 นี้ได้สร้างต่อยอดจากการเข้าสู่ระบบแบบฟอร์มของ ASP.NET 1.x ซึ่งหากการเข้าสู่ระบบด้วยการล็อกอินสำเร็จ เว็บไซต์จะทำการส่งคุกกี้ที่แสดงตัวตนของสมาชิกกลับไปเก็บไว้ที่ไคลเอนต์ และถ้าหากคุกกี้ที่เก็บไว้ที่ไคลเอนต์หมดอายุ ผู้ใช้ระบบจะต้องเข้าไปล็อกอินใหม่อีกที เพื่อที่จะให้มีการเข้าสู่ระบบแบบฟอร์ม ให้คลิกที่ลิงค์ Select authentication type หลังจากคลิกแล้ว จะพบหน้าจอให้เลือกวิธีที่ผู้ใช้ระบบจะสามารถเข้าถึงเว็บไซต์ดังกล่าว



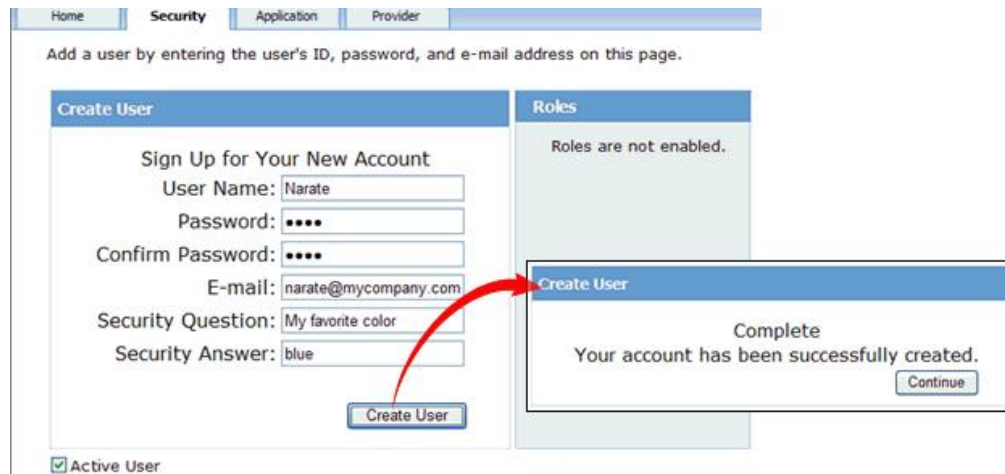
จากภาพ ให้เราเลือกตัวเลือกที่บอกว่าผู้ใช้ระบบจะเข้ามาจากอินเทอร์เน็ต (From the internet) แล้วกดปุ่ม Done ที่ด้านล่างขวา ผลจากการกำหนดค่าในเครื่องมือ WAT นี้ จริงๆ แล้วมันจะไปทำการกำหนดค่าแอตทริบิวต์โหมดของอ็อบเจกต์ authentication ในไฟล์คอนฟิกูเรชัน web.config ต่ออีกที ถึงตอนนี้เราได้เปิดใช้ฐานข้อมูล SQL Server แบบดีฟอลต์เรียบร้อยแล้ว แล้วเราสามารถตรวจสอบได้โดยคลิกที่แท็บไพรไวดเซอร์ จะเห็นหน้าจอที่บอกว่าตอนนี้ไพรไวดเซอร์ AspNetSqlProvider ถูกใช้งานอยู่ดังภาพ



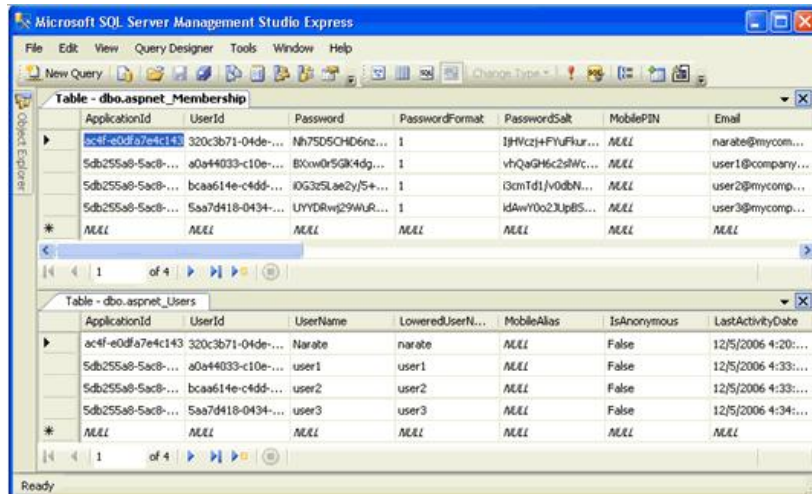
ในตอนนี้ เราพร้อมใช้งานความเป็นสมาชิกของเว็บไซต์แล้ว เราสามารถทดสอบได้โดยการเพิ่มสมาชิกในระบบผ่านเครื่องมือ WAT จากแท็บความปลอดภัย ดังภาพ









เมื่อเราคลิกที่ลิงค์ Create user จะพบหน้าจอสร้างสมาชิกระบบ ให้ทดลองกรอกข้อมูลต่างๆ แล้วกดปุ่มสร้างสมาชิกระบบ (Create User)




หากเราเข้าไปดูในฐานข้อมูล จะพบข้อมูลของสมาชิกที่เราสร้างขึ้นในตาราง aspnet_Membership และ aspnet_Users แยกตามแอปพลิเคชันแต่ละชนิด (โปรดสังเกตคอลัมน์ ApplicationId) ดังภาพ



หากกล่าวถึงเรื่องความเป็นสมาชิกแล้ว จะพบว่าเกือบทุกเว็บไซต์มีส่วนประกอบที่จำเป็นนี้อยู่ แล้วก็มี ความคล้ายคลึงกันมาก เช่น การล็อกอินด้วยชื่อ และรหัสผ่าน การแจ้งเตือนรหัสผ่านในกรณีที่มีลืม การสร้างแอคเคาต์ ให้สมาชิกใหม่ เพื่อให้เราจัดการกับสิ่งที่จำเป็นนี้ได้ง่าย ASP.NET 2.0 ได้สร้างคอนโทรลขึ้นมาใหม่สำหรับ ความ เป็นสมาชิกโดยเฉพาะซึ่งอยู่ถูกวางอยู่ในกลุ่ม Login ของทูลบ็อกซ์ของ Visual Studio 2010 ซึ่งจะมีคอนโทรล อยู่ทั้งหมด 7 ตัวดังตาราง

คอนโทรล	รายละเอียด
 Login	ใช้สำหรับการล็อกอิน เป็นคอนโทรลที่ประกอบไปด้วยคอนโทรลย่อยหลายตัว คือ คอนโทรล TextBox สำหรับกรอกชื่อ และรหัสผ่าน และคอนโทรล Button สำหรับ Submit และ Reset
 LoginName	ใช้แสดงชื่อของผู้ใช้ระบบที่ได้ผ่านการล็อกอินมาแล้ว
 LoginStatus	ใช้แสดงลิงค์ หรือปุ่มออกจากระบบ (Sign Out) ในกรณีที่ผู้ใช้ระบบได้ล็อกอิน แล้ว หรือหากผู้ใช้ระบบยังไม่ล็อกอิน จะแสดงลิงค์ หรือปุ่มที่ลิงค์ไปหน้าล็อกอิน (Sign In)
 LoginView	ใช้แสดงกลุ่มของคอนโทรลใน 2 มุมมอง คือ มุมมองผู้ใช้ระบบที่ยังไม่ได้ล็อกอิน (Anonymous) และมุมมองผู้ใช้ระบบที่ล็อกอินแล้ว (Logged In)
 ChangePassword	ใช้สำหรับเปลี่ยนรหัสผ่าน ผู้ใช้ระบบต้องกรอกรหัสผ่านเก่า รหัสผ่านใหม่ และ ยืนยันรหัสผ่านใหม่
 PasswordRecovery	ใช้สำหรับแจ้งเตือนรหัสผ่านด้วยอีเมลในกรณีที่มีลืม ด้วยค่าดีฟอลต์ ระบบจะทำการถามคำถาม และคำตอบก่อนที่จะส่งรหัสผ่านไปให้ แต่ถ้ากำหนดค่าแอตทริบิวต์ requiresQuestionAndAnswer ให้มีค่าเป็น false ระบบจะถามเพียงชื่อ

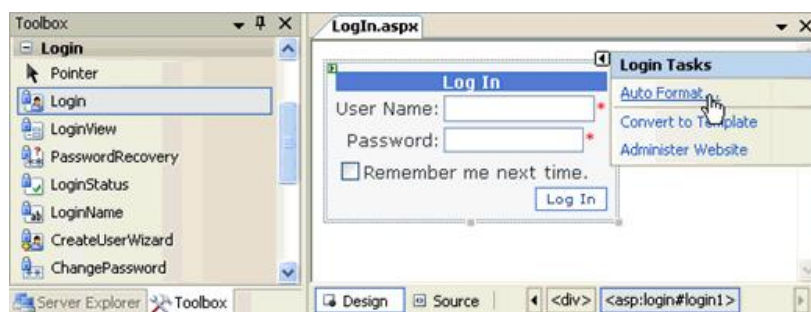
	<p>อย่างเดียว โดยไม่มีการถามคำถาม และคำตอบ ก็จะส่งอีเมลล์ไปให้ทันที</p>
 <p>CreateUserWizard</p>	<p>วิซาร์ดที่ใช้สร้างสมาชิกใหม่ หรือผู้ดูแลระบบใหม่ให้เว็บไซต์ ด้วยค่าดีฟอลต์ ระบบจะบังคับให้กรอกคำถาม และคำตอบเพื่อใช้สร้างสมาชิกใหม่ แต่ถ้ากำหนดค่าแอตทริบิวต์ requiresQuestionAndAnswer ให้มีค่าเป็น false ระบบจะไม่มีส่วนคำถาม และคำตอบมาให้กรอกเพื่อใช้สร้างสมาชิกใหม่</p>

คอนโทรลในกลุ่มนี้ส่วนใหญ่ ซึ่งได้แก่ คอนโทรล ChangePassword คอนโทรล CreateUserWizard คอนโทรล Login และคอนโทรล PasswordRecovery จะทำงานร่วมกับ API และโพรไวเดอร์ความเป็นสมาชิก เพื่อติดต่อกับฐานข้อมูล ดังนั้นคอนโทรลเหล่านี้จึงมีพร็อพเพอร์ตี้ MembershipProvider มาให้เรากำหนดค่าได้ ซึ่งเราสามารถกำหนดค่านี้ได้ในวินโดว์พร็อพเพอร์ตี้ หรือในไฟล์ Code Behind ของเว็บฟอร์ม แต่ถ้าหากเราละค่านี้ไว้ หรือไม่มีการกำหนดค่า คอนโทรลเหล่านี้จะใช้ค่าโพรไวเดอร์ดีฟอลต์ที่ได้กำหนดไว้ในแอตทริบิวต์ defaultProvider ของเซกชัน membership ของไฟล์คอนฟิกูเรชัน web.config แทน นอกจากนี้คอนโทรลเหล่านี้ยังอนุญาตให้เรากำหนดเทมเพลตของเราเองได้ เพื่อความยืดหยุ่นในการใช้งาน

สำหรับคอนโทรลอีกกลุ่มที่ไม่ได้ทำงานร่วมกับ API และโพรไวเดอร์ความเป็นสมาชิกคือ คอนโทรล LoginName คอนโทรล LoginStatus และคอนโทรล LoginView คอนโทรลเหล่านี้จะดึงชื่อผู้ใช้ระบบ สถานะการล็อกอินปัจจุบันมาใช้งาน สำหรับรายละเอียดของคอนโทรลแต่ละตัวจะกล่าวในหัวข้อถัดไป

การใช้งานคอนโทรล Login

คอนโทรล Login เป็นคอนโทรลตัวแรกสำหรับการเข้าสู่ระบบ เมื่อเรลากจากทูลบ็อกซ์มาวางในมุมมอง Design แล้วเราสามารถเปลี่ยนสไตล์แบบง่ายๆ ด้วยการกำหนด Auto Format หากต้องการปรับข้อความต่างๆ ในคอนโทรล เราสามารถปรับค่าทุกค่าได้ในวินโดว์พร็อพเพอร์ตี้ หรือหากต้องการปรับเปลี่ยนแก้ไขหน้าตาให้ตามความต้องการ เราสามารถแปลงคอนโทรล Login ให้เป็นเทมเพลตได้โดยเลือก Convert to Template ใน Login Tasks ได้ดังภาพ



หากเราเลือก Convert to Template จะได้หน้าต่างของคอนโทรล Login ดังภาพ จากภาพจะเห็นว่า คอนโทรลถูกแปลงให้เป็นตารางที่บรรจุคอนโทรลชนิดต่างๆ ในตอนนี้เราสามารถลากคอนโทรลตัวอื่นจาก ทูลบ็อกซ์มาวางในคอนโทรล Login เพิ่มเติมได้ แต่เราไม่สามารถที่จะลบคอนโทรล TextBox ที่ชื่อ UserName และ Password ออกได้ เพราะคอนโทรลทั้งสองจำเป็นต้องถูกใช้งานในคอนโทรล Login ในกรณีที่คอนโทรล Login ได้ถูกแปลงให้เป็นเทมเพลตแล้ว หากต้องการกลับมายังค่าเริ่มต้นใหม่ก่อนถูกแปลงเป็นเทมเพลตให้เลือก Reset ที่ Login Tasks



สำหรับการเข้าสู่ระบบแบบฟอร์มนี้ เว็บไซต์จะต้องมีหน้าล็อกอินอย่างน้อย 1 หน้า เพื่อทำการตรวจสอบผู้ใช้ระบบว่าเป็นสมาชิกของระบบหรือไม่ หากผู้ใช้ระบบสามารถกรอกชื่อ และรหัสผ่านได้ถูกต้อง เว็บไซต์จะทำการเขียนคุกกี้ที่ชื่อ .ASPXAUTH ไปให้กับไคลเอนต์ เพื่อให้สามารถถูกอ้างอิงได้จากทุกๆ เว็บฟอร์มในเว็บไซต์ คุกกี้ที่เขียนไปให้กับไคลเอนต์นี้มี 2 ชนิด ชนิดแรกเป็นคุกกี้แบบชั่วคราว (Temporary Authentication Cookie) จะถูกลบไปเมื่อปิดเบราว์เซอร์ ชนิดที่สองจะเป็นแบบถาวร (Permanent Authentication Cookie) จะไม่ถูกลบเมื่อมีการปิดเบราว์เซอร์ แต่จะหมดอายุไปเองเมื่อครบเวลาตามที่กำหนดไว้ในไฟล์คอนฟิกูเรชัน web.config คุกกี้แบบถาวรนี้สามารถกำหนดได้ในคอนโทรล Login โดยการคลิกเลือกที่คอนโทรล CheckBox ที่มีข้อความว่า Remember me next time

เพื่อที่จะทำการทดสอบคอนโทรล Login เราจะสร้างเว็บไซต์ที่อนุญาตเฉพาะผู้ที่ได้ผ่านการล็อกอินแล้วเท่านั้น หากผู้ใช้ระบบที่ไม่ได้ล็อกอินพยายามที่จะเข้าไปเปิดเว็บฟอร์มใดเว็บฟอร์มหนึ่งในเว็บไซต์ ผู้ใช้ระบบคนนั้นจะถูกบังคับให้ไปหน้าล็อกอินเพื่อให้ทำการล็อกอินก่อน การกำหนดให้เว็บไซต์อนุญาตให้ผู้ใช้ระบบที่ได้ล็อกอินแล้วเท่านั้นเข้ามาใช้งานได้ดังโค้ดตัวอย่าง

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
```

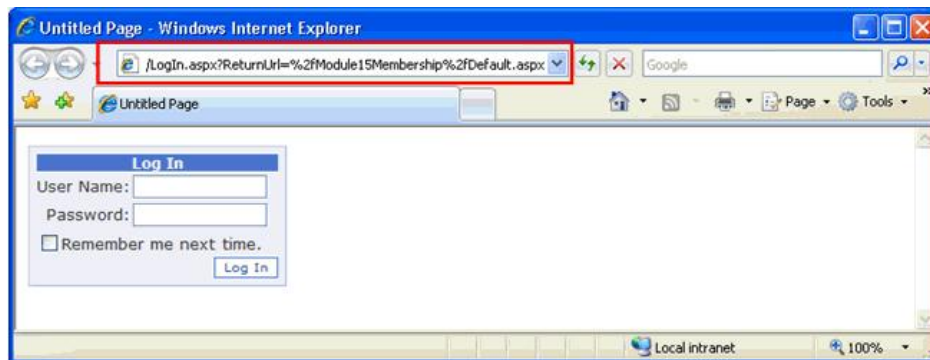
```
<system.web>
  <authorization>
    <deny users="?" />
  </authorization>
  ...
</system.web>
</configuration>
```

จากโค้ดตัวอย่าง เราสามารถเพิ่มอิลเมนต์ authorization ได้เซกชัน system.web เพื่อกำหนดให้ปฏิเสธผู้ใช้ระบบที่ไม่ได้ล็อกอินเข้ามาใช้เว็บไซต์ อิลเมนต์ deny จะบอกว่าให้ทำการปฏิเสธผู้ใช้ระบบที่ไม่รู้จัก (?) หรือ Anonymous แต่ถ้าผู้ใช้ได้ผ่านการล็อกอินแล้ว ผู้ใช้คนนั้นจะเป็นผู้ใช้ระบบที่รู้จักแล้ว หรือเป็นสมาชิกของเว็บไซต์จริงก็จะเข้ามาเรียกใช้เว็บฟอร์มต่างๆ ในเว็บไซต์ได้ ถ้าผู้ใช้ระบบไม่ได้ล็อกอิน แต่พยายามจะเรียกใช้เว็บฟอร์ม การเรียกนั้นจะถูกปฏิเสธ และจะถูกบังคับให้ไปหน้าล็อกอินที่เราสามารถกำหนดได้ในไฟล์คอนฟิกูเรชัน web.config ดังโค้ดตัวอย่าง

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.web>
    <authentication mode="Forms">
      <forms loginUrl="~/LogIn.aspx" timeout="30" />
    </authentication>
    ...
  </system.web>
</configuration>
```

จากโค้ดตัวอย่าง ผู้ใช้ระบบที่ไม่ได้ล็อกอินจะถูกบังคับให้ไปหน้าล็อกอิน LogIn.aspx ซึ่งปกติแล้วแอตทริบิวต์ loginUrl นี้สามารถถูกละได้หากเว็บฟอร์มล็อกอินมีชื่อว่า LogIn.aspx เพราะเป็นค่าดีฟอลต์ แต่ถ้าหากเป็นชื่ออื่นให้เรากำหนดค่าให้กับแอตทริบิวต์นี้ สำหรับการกำหนดเวลาหมดอายุของคุกกี้แบบถาวร สามารถกำหนดในแอตทริบิวต์ timeout ซึ่งมีค่าเป็นช่วงเวลาในหน่วยนาที

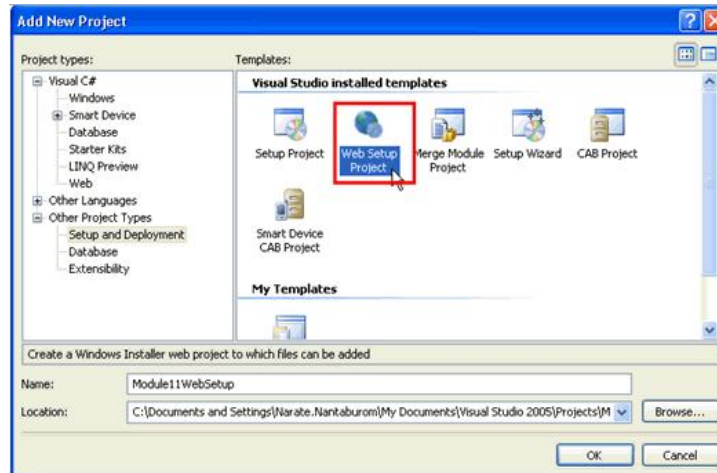
หลังจากสร้างหน้าเว็บฟอร์มที่มีคอนโทรลล็อกอิน และกำหนดค่าในไฟล์คอนฟิกูเรชัน web.config แล้ว ต่อไปเราจะสร้างหน้า Default.aspx ขึ้น เพื่อเป็นหน้าแรกหากผู้ใช้ระบบทำการล็อกอินสำเร็จ เพื่อให้หน้าหน้านี้อาจจะเขียนเพียงข้อความ Welcome เท่านั้น ให้ลองทดสอบเปิดไฟล์ Default.aspx นี้ในเบราว์เซอร์ จะพบว่าเราจะถูกบังคับให้ไปหน้าล็อกอินก่อนโดยมี Query String ที่ชื่อ returnUrl ที่บอกว่าหากล็อกอินสำเร็จแล้ว ให้ไปที่หน้า Default.aspx ดังภาพ



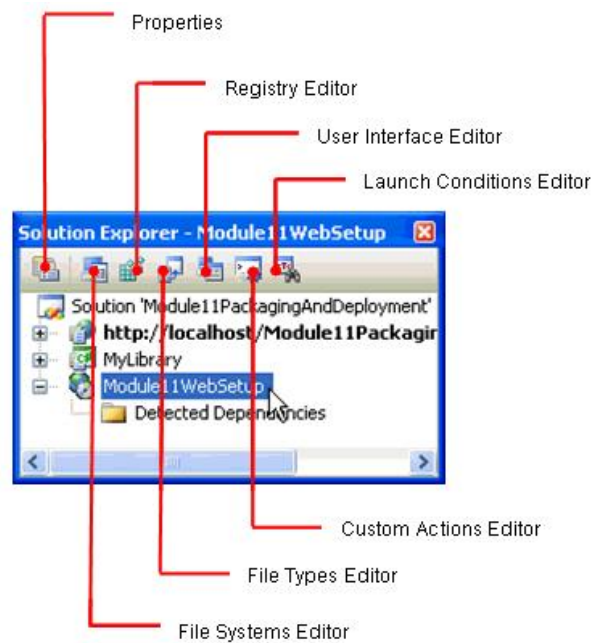
ปกติแล้วคอนโทรล Login จะทำการกับโปรไวเดอร์ความเป็นสมาชิกดีฟอลต์ที่เราได้กำหนดไว้ในไฟล์คอนฟิกูเรชัน web.config (กำหนดผ่านแอตทริบิวต์ defaultProvider) แต่ถ้าเราต้องการใช้โปรไวเดอร์ตัวอื่น เราสามารถกำหนดค่าของพร็อพเพอร์ตี้ MembershipProvider ได้ในวินโดว์พร็อพเพอร์ตี้ของคอนโทรล Login หากเราต้องการทดสอบคอนโทรล Login เราอาจจะใช้แอสเซมบลีที่สร้างจากเครื่องมือ WAT หรืออาจจะใช้คอนโทรล CreateUserWizard ที่จะกล่าวถึงในหัวข้อถัดไป

กร Setup Project

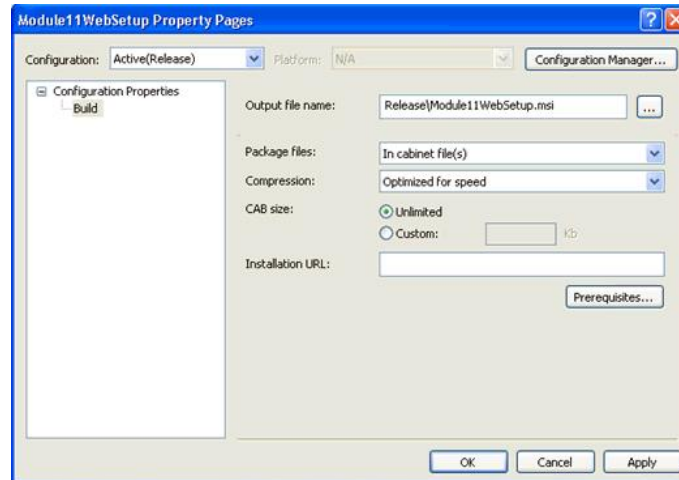
การติดตั้งเว็บไซต์ที่เว็บเซิร์ฟเวอร์ในบางครั้ง เราอาจจะไม่สามารถเข้าถึงเว็บเซิร์ฟเวอร์ได้โดยตรง อาจจะเนื่องจากเรื่องความปลอดภัยในบริษัทใหญ่ หรือหน่วยงานขนาดใหญ่ของราชการ กรณีนี้เราสามารถสร้างแพ็คเกจเพื่อส่งให้กับผู้ดูแลระบบทำการติดตั้งให้ได้ วิธีการสร้างแพ็คเกจนี้ให้เราสร้างโปรเจกต์ Web Setup โดยคลิกขวาที่โซลูชัน แล้วเลือก Add --> New Project... จะปรากฏไดอะล็อก Add New Project ให้เราเลือกชนิดของโปรเจกต์แบบ Other Project Types --> Setup and Deployment จากนั้นคลิกเลือกที่ Web Setup Project ดังภาพ



จากภาพ ให้เราตั้งชื่อโปรเจกต์ Web Setup พร้อมระบุตำแหน่งที่ต้องการวางโปรเจกต์นี้ แล้วกดปุ่ม OK จะได้ดังภาพ



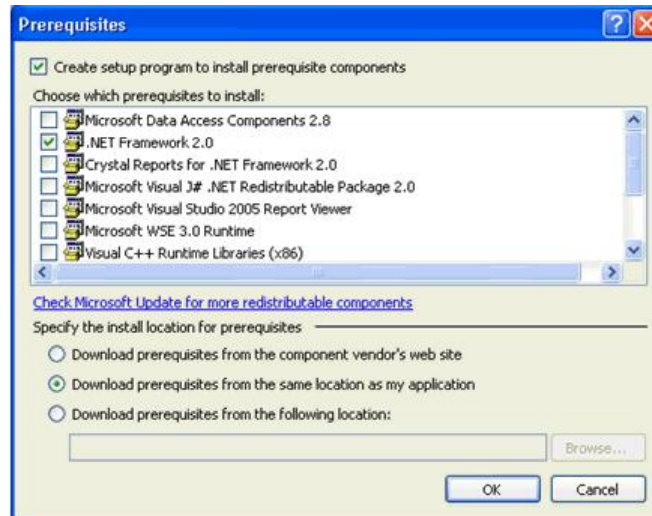
เมื่อเราคลิกเลือกที่โปรเจกต์ Web Setup จะปรากฏไอทีเตอร์ ที่ด้านบนดังภาพ ซึ่งจะมีไอทีเตอร์ 6 ชนิด ซึ่งจะต่างกันในหัวข้อถัดไป การเราคลิกที่ไอคอน Properties จะปรากฏไดอะล็อก Property Page ดังภาพ



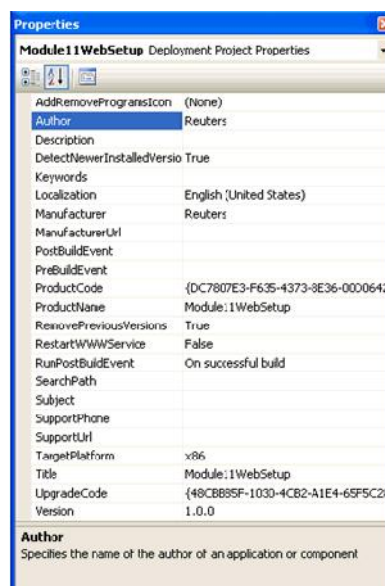
สำหรับการกำหนดค่าต่างๆ ในไดอะล็อก Property Pages ของโปรเจกต์ Web Setup มีดังนี้

- **Output file name** อนุญาตให้เรากำหนดชื่อ และไฟล์เตอร์ของไฟล์ติดตั้งซึ่งจะมีนามสกุลเป็น msi
- **Package files** จะมี 3 ตัวเลือกรูปนี้
 - o **As loose, uncompressed files** จะเป็นการกำหนดให้มีการแยกไฟล์ต่างๆ ที่อยู่ในเว็บไซต์ไว้ต่างหากจากไฟล์ติดตั้ง (ไฟล์นามสกุล msi) และในเวลาติดตั้ง เราต้องทำการส่งทั้งติดตั้ง และไฟล์ที่อยู่ในเว็บไซต์ทั้งหมดไปด้วยกัน ตัวเลือกรูปนี้จึงค่อนข้างยุ่งยากในการจัดการ
 - o **In Setup File** ตัวเลือกรูปนี้จะรวมไฟล์ต่างๆ ไว้ในไฟล์ต่างๆ ในเว็บไซต์ไว้ในไฟล์ติดตั้งเรียบร้อยแล้ว ตัวเลือกรูปนี้จึงง่ายในการจัดการมาก เราสามารถนำไฟล์นี้ไปติดตั้งได้ทันที
 - o **In Cabinet File(s)** ตัวเลือกรูปนี้อนุญาตให้เราแบ่งไฟล์ที่จะติดตั้งออกเป็นหลายๆ ไฟล์ได้ ซึ่งจะประกอบด้วยไฟล์ติดตั้ง และไฟล์นามสกุล CAB ที่บรรจุไฟล์ต่างๆ ในเว็บไซต์
- **Compression** เราสามารถกำหนดให้มีการบีบอัด CAB ไฟล์เพื่อความเร็วในการติดตั้ง (Optimized for speed) หรือเพื่อให้มีขนาดเล็กที่สุด (Optimized for size) หรือไม่อนุญาตให้มีการบีบอัด (None) ได้
- **CAB size** เราสามารถกำหนดขนาดของ CAB ไฟล์ได้ เพื่อให้เหมาะกับขนาดของแผ่น Floppy Disk แผ่นซีดี หรือแผ่นดีวีดี หากขนาดของไฟล์ต่างๆ ในเว็บไซต์มากกว่าขนาดของ CAB ที่กำหนด คอมไพเลอร์จะสร้างไฟล์ CAB ตัวใหม่ขึ้นมาเพื่อบันทึกข้อมูลของเว็บไซต์โดยอัตโนมัติ

- Prerequisites จะเป็นโปรแกรมที่จำเป็นต้องลงก่อนที่จะติดตั้งเว็บไซต์ ซึ่งเราสามารถกำหนดโปรแกรมที่จำเป็นนี้ได้จากไดอะล็อก Prerequisites โดยการคลิกที่ปุ่ม Prerequisites... ที่ด้านล่างขวาจะปรากฏไดอะล็อกดังภาพ



จากภาพ เราสามารถเลือกโปรแกรมที่จำเป็น และเลือกว่าจะติดตั้งโปรแกรมนี้จากที่ไหน โดยตัวเลือกแรกจะดาวน์โหลดจากเว็บไซต์ของไมโครซอฟท์ ตัวเลือกที่สองจะโหลดจากไฟล์เตอร์ของโปรแกรมติดตั้ง และตัวเลือกที่สามจะดาวน์โหลดจากเว็บไซต์ หรือไฟล์เซิร์ฟเวอร์ที่เรากำหนดเอง นอกจากการกำหนดค่าในการติดตั้งจากไดอะล็อก Property Pages แล้ว เราสามารถกำหนดค่าเพิ่มเติมได้จากพร็อพเพอร์ตี้วินโดวส์

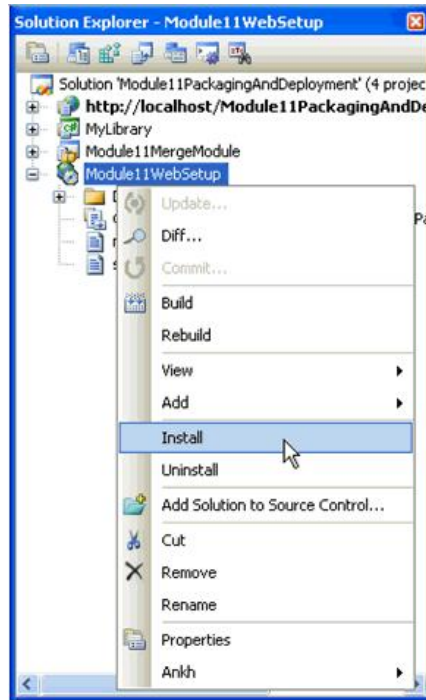


สำหรับพร็อพเพอร์ตี้ที่น่าสนใจของโปรเจค Web Setup มีดังนี้

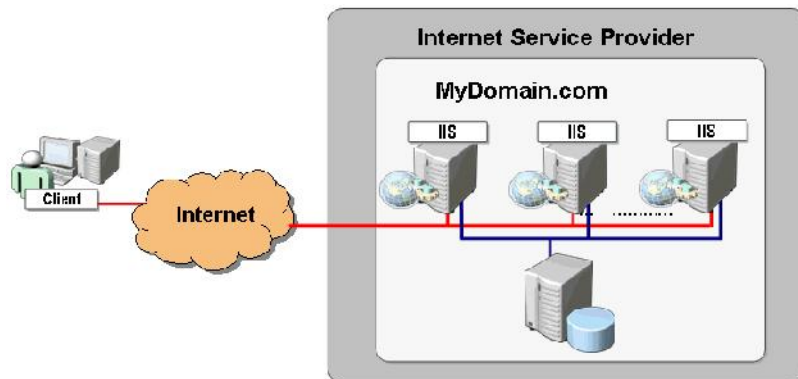
-
- **DetectNewerInstalledVersion** เป็นตัวบอกให้แพ็คเกจตรวจสอบเว็บไซต์ที่ได้ติดตั้งไว้ก่อนหน้า หากเว็บไซต์ที่ติดตั้งไว้ก่อนมีเวอร์ชันใหม่กว่า ตัวแพ็คเกจจะหยุดทำการติดตั้งทันที
 - **PreBuildEvent** บอกให้แพ็คเกจทำการรันคอมมานไลน์ที่กำหนดไว้ก่อนการติดตั้ง
 - **PostBuildEvent** บอกให้แพ็คเกจทำการรันคอมมานไลน์ที่กำหนดหลังการติดตั้ง
 - **RemovePreviousVersions** บอกให้แพ็คเกจทำการลบเวอร์ชันเก่าที่มีอยู่ก่อนการติดตั้ง
 - **RestartWWWService** บอกให้แพ็คเกจทำการรีสตาร์ท IIS หลังการติดตั้งสำเร็จ
 - **RunPostBuildEvent** บอกเหตุการณ์ที่จะรันคอมมานไลน์หลังการติดตั้ง ซึ่งปกติจะรันในเมื่อมีเหตุการณ์ติดตั้งสำเร็จ (On successful build) เท่านั้น หรืออีกเหตุการณ์คือรันทุกครั้งไม่ว่าการติดตั้งจะสำเร็จหรือไม่ (Always)
 - **SearchPath** บอกให้แพ็คเกจทราบพาร์ธที่สามารถใช้ค้นหาไฟล์ แอสเซมบลี หรือ Merge Module

การทดสอบการติดตั้ง และยกเลิกการติดตั้งที่เครื่องนักพัฒนา

การทดสอบการติดตั้ง หรือการยกเลิกการติดตั้งที่เครื่องนักพัฒนาสามารถทำได้ง่ายโดยการคลิกขวาที่โปรเจกต์ Web Setup แล้วเลือก ติดตั้ง (Install) หรือยกเลิกการติดตั้ง (Uninstall) ดังภาพ



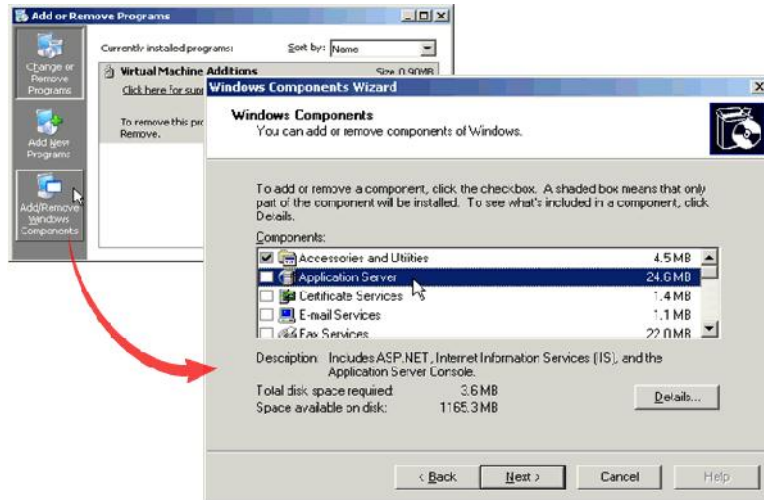
IIS 6.0 ทำหน้าที่ให้บริการต่างๆ ทางด้านอินเทอร์เน็ต โดยบริการที่มีจะประกอบด้วย บริการเว็บ (www) บริการรับส่งอีเมล (SMTP) บริการข่าว (NNTP) และบริการรับส่งไฟล์ (FTP) หากเราต้องการเปิดใช้บริการต่างๆ เหล่านี้บนอินเทอร์เน็ต เราต้องนำเซิร์ฟเวอร์ของเราไปเชื่อมต่อกับอินเทอร์เน็ต โดยปกติเพื่อให้ได้รับความเร็วอินเทอร์เน็ตที่สูงๆ เราต้องตั้งเซิร์ฟเวอร์ของเราไว้กับ Inter Service Provider หรือ ISP ตามภาพ



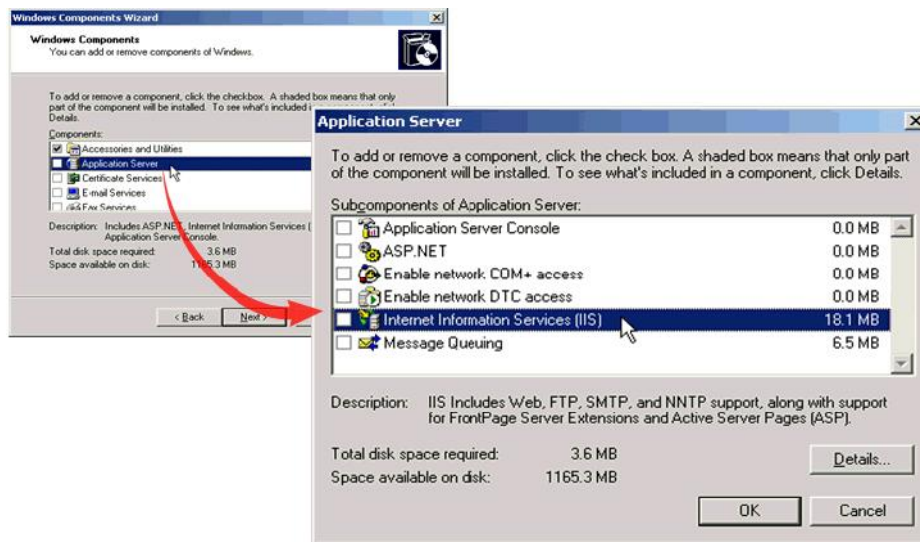
ในแต่ละ ISP จะมีบริษัทต่างๆ เข้ามาตั้งเซิร์ฟเวอร์อยู่มากมายอยู่รวมกัน เราสามารถเช่าเป็นห้องรวมหรือห้องส่วนตัวให้กับเซิร์ฟเวอร์ได้ (ยังกะบ้านคนเลย ต้องเอาใจหน่อย) ในบางครั้งเราอาจจะตั้งเป็นเน็ตเวิร์คย่อยๆ ที่มีทั้ง DNS, เว็บเซิร์ฟเวอร์ และฐานข้อมูลด้วยก็ได้

การติดตั้ง IIS บน Windows Server

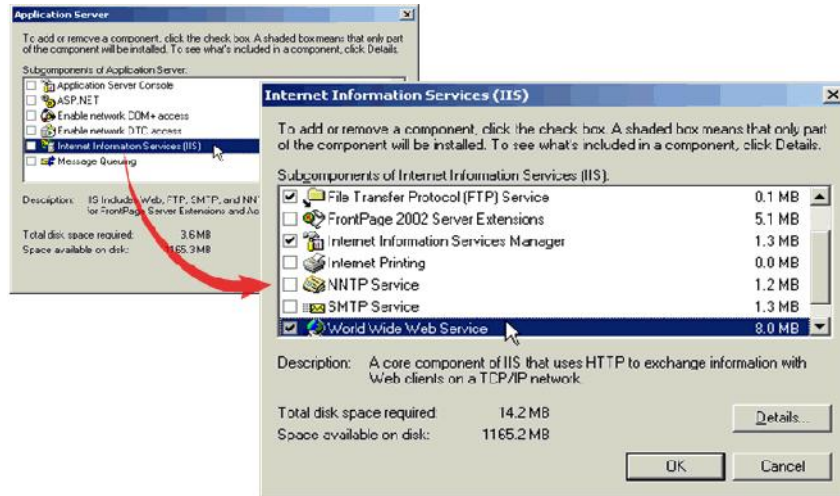
เพื่อที่จะติดตั้งเราต้องเข้าไปที่ Add/Remove Program แล้วคลิกที่ Add/Remove Windows Components ดังภาพ



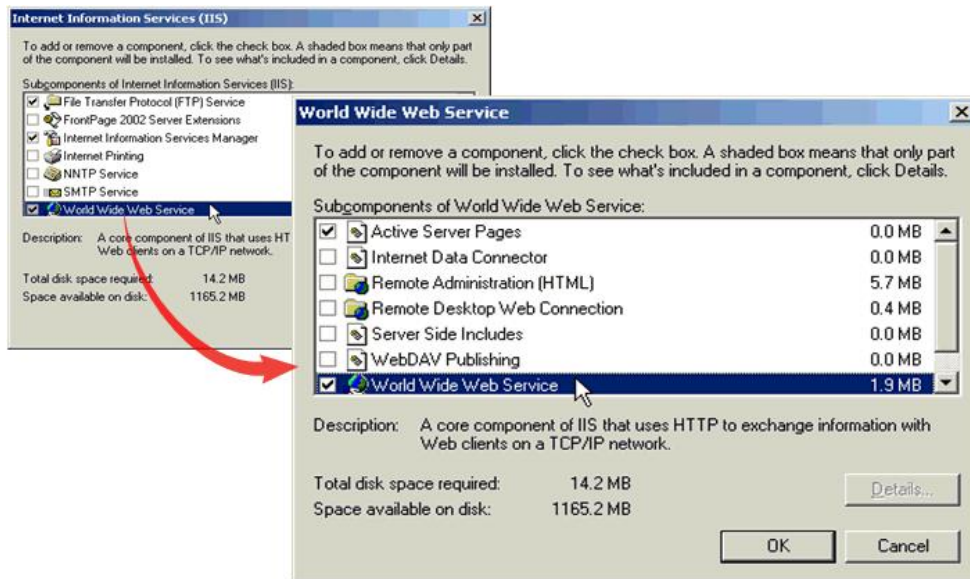
จะพบไดอะล็อกซ์ Windows Component Wizard แล้วคลิกเลือกที่ Application Server แล้วคลิกปุ่ม Detail... ที่ด้านล่างขวาจากพบไดอะล็อกซ์ Application Server ด้านล่าง



จากนั้นคลิกเลือกที่ Internet Information Server (IIS) แล้วคลิกที่ปุ่ม Detail... ที่ด้านล่างขวาอีกครั้ง จะพบไดอะล็อกซ์ด้านล่าง



ให้คลิกเลือกที่บริการที่ต้องการ ซึ่งอาจจะเป็น FTP, NNTP, SMTP หรือ World Wide Web Service ในหน้าจอนี้ให้เลือก Internet Information Services Manager เอาไว้ด้วย เพื่อใช้เป็นหน้าจอ Admin สำหรับที่ World Wide Web Service ยังสามารถเข้าไปเลือก option ต่ออีกด้วยการคลิกที่ปุ่ม Detail... ด้านล่างขวาจะพบไดอะล็อกซ์ด้านล่าง

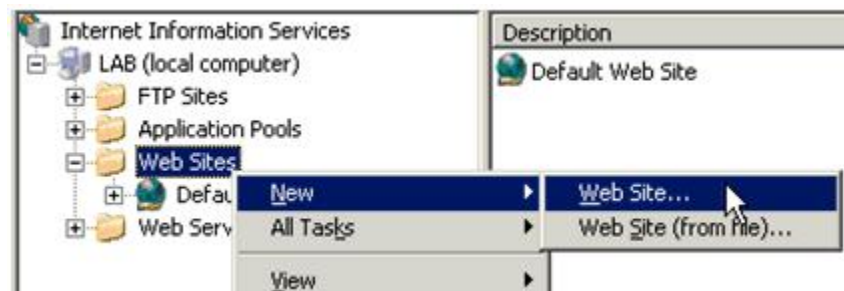


หากต้องการให้รัน ASP ได้ให้คลิกเลือกที่ Active Server Pages ด้วย

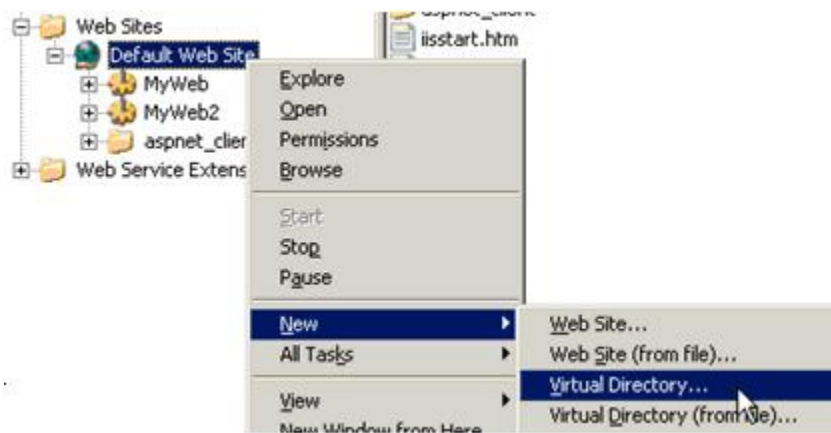
การสร้างเว็บไซต์ และการสร้าง Virtual Directory ในเว็บไซต์

เพื่อให้เว็บแอปพลิเคชัน (หรือเว็บไซต์ใน Visual Studio 2010) สามารถทำงานในโพรเซสของตัวเอง มีค่าของตัวเอง แอปพลิเคชัน ตัวแปร Session และมีค่าคอนฟิกจากไฟล์ web.config ของมันเอง เรามีวิธีการสร้างอยู่ 2 แบบ คือ

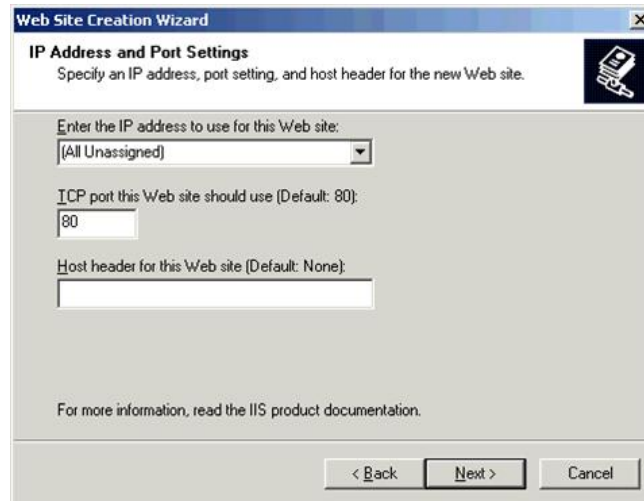
1. สร้าง Web Site ใต้โฟลเดอร์ IIS WebSite ดังภาพ



2. สร้าง Virtual Directory ที่อยู่ใต้ Web Site ต่ออีกที



สำหรับการสร้างเว็บไซต์ จะมีรายละเอียดที่ต้องระบุอยู่ 3 ส่วนที่ต้องระบุเพื่อประกอบขึ้นมาเป็น Web Site คือ IP, Port และ Host Header ดังภาพ



แต่ละ Web Site ที่อยู่ในเว็บเซิร์ฟเวอร์ตัวหนึ่งจะต้องมีค่า 3 ค่านี้ไม่ซ้ำกัน ไม่เช่นนั้นตัวที่ซ้ำจะ Start ไม่ขึ้น โดยปกติแล้ว Web Site ที่ไม่ซับซ้อนมาก เรากำหนดเพียง IP และ Port (IP Port = Socket) ก็เพียงพอ โดยปกติแล้วสำหรับบริการที่เปิดใช้บนอินเทอร์เน็ตจะใช้ Port 80 แต่ถ้าหากเป็น SSL จะใช้พอร์ต 443 แต่ในบางครั้งมีปัญหาว่ามีไอพีไม่พอกับแต่ละเว็บไซต์ เราจึงแก้ปัญหาด้วยการเพิ่ม Host Header เข้าไปเพื่อให้ 1 IP และสามารถรองรับได้หลายเว็บไซต์ ซึ่งค่าที่จะกรอกลงในช่อง Host Header นี้จะเป็น Domain Name เช่น ASPNETTHAI.COM และ WWW.ASPNETTHAI.COM เบื้องหลังของการรันเว็บไซต์

