

## เทคนิคการเขียนโปรแกรมบนไมโครซอฟต์วินโดวส์

### ความเป็นมา

ในปี ค.ศ. 1983 บริษัทไมโครซอฟต์คอร์ปอเรชันได้เปิดตัวซอฟต์แวร์ไมโครซอฟต์วินโดวส์เป็นครั้งแรก ซึ่งเป็นระบบการทำงานในสภาพแวดล้อมแบบกราฟิกทำหน้าที่เป็นตัวเชื่อมต่อระหว่างผู้ใช้งานกับอุปกรณ์ต่างๆของคอมพิวเตอร์ทำให้เกิดความสะดวกในการใช้งาน โดยเริ่มวางจำหน่ายรุ่น 1.01 ในอีก 2 ปีถัดมา หลังจากนั้นบริษัทได้ทำการพัฒนาขีดความสามารถเพิ่มขึ้น ให้สามารถรองรับกับอุปกรณ์ต่างๆได้หลากหลายชนิดและออกรุ่น 2.0 ในปี ค.ศ.1987 ซึ่งกล่าวได้ว่ารุ่นนี้ได้มีการเปลี่ยนแปลงไปจากรุ่น 1.01 ค่อนข้างมาก โดยวินโดวส์ที่เปิดขึ้นมาสามารถซ้อนทับกันได้หลายๆชั้น รวมทั้งยังสนับสนุนการใช้แผงแป้นอักขระร่วมกับเมาส์อีกด้วย

ภายหลังจากออกรุ่น 2.0 ไม่นานนัก บริษัทก็ได้ออกไมโครซอฟต์วินโดวส์รุ่น/386 เพื่อใช้กับเครื่องไมโครคอมพิวเตอร์ที่ใช้ซีพียู 80386 โดยตรง ซึ่งในรุ่นนี้ บริษัทได้เพิ่มขีดความสามารถของไมโครซอฟต์วินโดวส์ขึ้นมา กล่าวคือได้พัฒนาให้ใช้ความสามารถพิเศษของซีพียู 80386 คือการทำงานในภาวะเสมือน-86 (virtual-86 mode) ทำให้สามารถนำโปรแกรมที่ทำงานภายใต้ระบบปฏิบัติการดอสโดยตรง มาทำงานบนไมโครซอฟต์วินโดวส์ได้พร้อมๆกันหลายๆโปรแกรม และได้เปลี่ยนชื่อของไมโครซอฟต์วินโดวส์รุ่น 2.0 เป็นไมโครซอฟต์วินโดวส์/286 เพื่อให้เกิดความสอดคล้องกัน

ในปี ค.ศ. 1990 บริษัทก็ได้ออกไมโครซอฟต์วินโดวส์รุ่น 3.0 ซึ่งได้มีการเปลี่ยนแปลงพัฒนาไปจากรุ่นเดิมมาก โดยได้รวมไมโครซอฟต์วินโดวส์ทั้งสองรุ่นเข้าด้วยกัน ปรับปรุงการทำงานของเปลือกรูท(shell)ของไมโครซอฟต์วินโดวส์ให้ดีขึ้น สนับสนุนการทำงานกับระบบเครือข่าย(network)และรูปแบบของแผนที่บิตชนิดใหม่ที่ไมขึ้นกับอุปกรณ์ต่างๆ(Device Independent Bitmap : DIB) นอกจากนี้ส่วนที่มีการเปลี่ยนแปลงที่สำคัญที่สุดคือการสนับสนุนการใช้งานหน่วยความจำแบบขยาย (extended memory) โดยสามารถอ้างอิงหน่วยความจำได้ถึง 16 เมกะไบต์ รวมทั้งถ้าหากทำงานบนซีพียู 80386 หรือสูงกว่าในภาวะ 386-Enhanced แล้วไมโครซอฟต์วินโดวส์จะใช้การจัดการหน่วยความจำแบบเสมือนของซีพียู ซึ่งจะทำให้สามารถใช้หน่วยความจำได้มากกว่าที่มีอยู่จริงถึง 4 เท่า

อย่างไรก็ตามในรุ่น 3.0 นี้ก็ยังมีข้อจำกัดในเรื่องการแสดงผลของตัวอักษร เพราะตัวอักษรที่ใช้เป็นแบบแผนที่บิตนั้น ทำให้การแสดงผลตัวอักษรขนาดใหญ่ไม่สวยงามเนื่องจากการนำแผนที่บิตมาขยาย จึงได้

ร่วมกับบริษัทแอปเปิลคอมพิวเตอร์พัฒนารูปแบบตัวอักษรใหม่ขึ้นในรุ่น 3.1 ในปี ค.ศ. 1992 เรียกว่ารูปแบบอักษร True Type Font) ซึ่งเป็นตัวอักษรแบบโครงร่าง ทำให้สามารถย่อ-ขยายตัวอักษรได้ตามความต้องการ นอกจากนี้ก็ยังได้พัฒนาขีดความสามารถการแลกเปลี่ยนข้อมูลที่เรียกว่า OLE (Object Linked and Embedded) สนับสนุนระบบสื่อหลายแบบ (Multimedia) รวมทั้งปรับปรุงโปรแกรมต่างๆ ให้มีประสิทธิภาพดียิ่งขึ้น และในรุ่นนี้บริษัทก็ได้ตัดการสนับสนุนการใช้งานไมโครซอฟต์วินโดวส์บนเครื่องคอมพิวเตอร์ที่มีซีพียู 8088 หรือที่เรียกว่าภาวะจริง (real mode)

### คุณลักษณะเด่นของไมโครซอฟต์วินโดวส์

คุณลักษณะเด่นของไมโครซอฟต์วินโดวส์ มีดังนี้

1. ระบบเชื่อมโยงกับผู้ใช้แบบกราฟิก (Graphic User Interface : GUI)
 

เป็นการติดต่อกับผู้ใช้ผ่านทางรูปภาพกราฟิกที่สื่อความหมายถึงการทำงานต่างๆ เช่น สัญลักษณ์(icon) แท่งเลื่อน(scroll bar) ทำให้เกิดความสวยงาม มีความน่าใช้ มีความสะดวกและง่ายต่อการเรียนรู้ นอกจากนี้ยังมีความเป็นมาตรฐาน โดยโปรแกรมต่างๆจะมีระบบตัวประสานกับผู้ใช้เหมือนกัน
2. การทำงานแบบหลายภารกิจ (Multitasking)
 

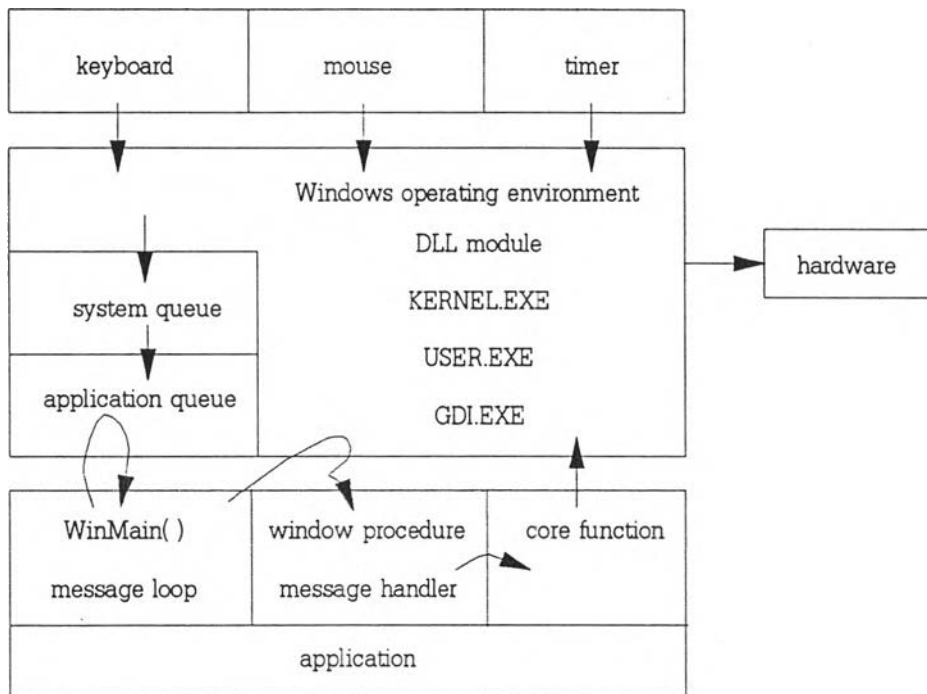
สภาพแวดล้อมของไมโครซอฟต์วินโดวส์ยินยอมให้ผู้ใช้เรียกใช้โปรแกรมมาทำงานได้หลายๆโปรแกรม หรือโปรแกรมเดียวกันพร้อมๆกัน โดยไมโครซอฟต์วินโดวส์จะเป็นผู้จัดสรรหน่วยความจำให้แต่ละโปรแกรมอย่างพอเพียงรวมทั้งคอยจัดการการใช้ทรัพยากรต่างๆร่วมกัน
3. ความเป็นอิสระต่ออุปกรณ์
 

เนื่องจากไมโครซอฟต์วินโดวส์เป็นสภาพแวดล้อมที่เป็นตัวกลางระหว่างโปรแกรมที่เขียนขึ้นกับอุปกรณ์ต่างๆ โปรแกรมจึงไม่ต้องคอยจัดการกับอุปกรณ์ต่างๆโดยตรง เพียงแต่เรียกใช้ฟังก์ชันต่างๆที่ไมโครซอฟต์วินโดวส์จัดเตรียมไว้ให้ถูกต้องเท่านั้น ทำให้ผู้ใช้สามารถเปลี่ยนอุปกรณ์ในระบบได้เลย โดยไม่จำเป็นต้องเปลี่ยนแปลงโปรแกรมที่มีอยู่
4. การแลกเปลี่ยนข้อมูลระหว่างโปรแกรม
 

ไมโครซอฟต์วินโดวส์ได้สนับสนุนการโอนย้าย แลกเปลี่ยนข้อมูลระหว่างโปรแกรมได้ง่าย ด้วยการคัดลอกข้อมูลจากโปรแกรมหนึ่งเข้าไปยังคลิปบอร์ด (clipboard) แล้วจึงคัดลอกจากคลิปบอร์ดไปยังอีกโปรแกรมหนึ่งได้ นอกจากนี้ยังมีขีดความสามารถในการเชื่อมโยงเพิ่มข้อมูลเข้าด้วยกันแบบ DDE (Dynamic Data Exchange) และ OLE (Object Linked and Embedded)

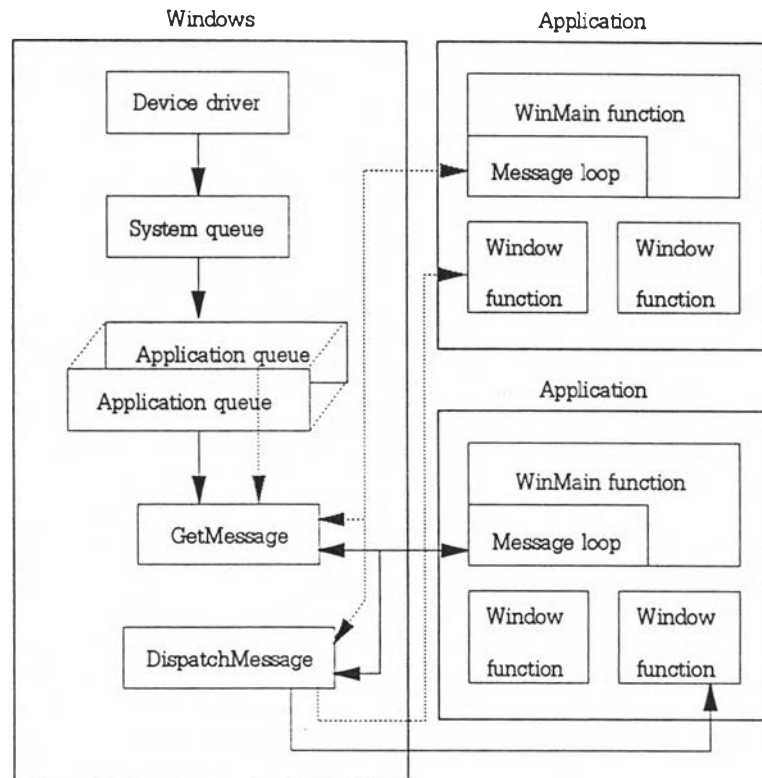
## การทำงานของไมโครซอฟต์วินโดวส์

ดังที่กล่าวแล้วว่าซอฟต์แวร์ไมโครซอฟต์วินโดวส์เป็นซอฟต์แวร์ที่ช่วยให้การใช้งานเครื่องคอมพิวเตอร์เป็นไปอย่างมีประสิทธิภาพโดยไมโครซอฟต์วินโดวส์จะเป็นตัวเชื่อมต่อระหว่างโปรแกรมประยุกต์กับอุปกรณ์ฮาร์ดแวร์ ซึ่งจะมีฟังก์ชันการติดต่อให้โปรแกรมประยุกต์ต่างๆเรียกใช้ (Application Programming Interface) ดังรูปที่ 3.1



รูปที่ 3.1 แสดงการทำงานของไมโครซอฟต์วินโดวส์และโปรแกรมประยุกต์

ไมโครซอฟต์วินโดวส์จะเป็นตัวคอยจับสัญญาณจากอุปกรณ์นำเข้าข้อมูล (Input Device) อาทิเช่น แปงแป้นอักขระ เม้าส์ นาฬิกา เมื่ออุปกรณ์เหล่านี้ให้กำเนิดสัญญาณมา วินโดวส์ก็จะสร้างข้อความ (message) ที่เหมาะสมกับสัญญาณนั้นๆเก็บไว้ในแถวคอยของระบบ (system queue) ตามหลักเข้าก่อนออกก่อน (first-in-first-out : FIFO) จากนั้นวินโดวส์จะพิจารณาว่าข้อความต่างๆนั้นเป็นของโปรแกรมประยุกต์ใด แล้วจึงส่งข้อความนั้นไปยังแถวคอยของโปรแกรมประยุกต์นั้น โดยที่แต่ละโปรแกรมประยุกต์ก็จะมีแถวคอยของโปรแกรมประยุกต์ (application queue) ของตนเองคอยรับข้อความที่ส่งมา จากนั้นโปรแกรมประยุกต์จึงนำข้อความที่ได้รับไปประมวลผลอีกทีหนึ่ง



รูปที่ 3.2 แสดงการไหลของข้อความระหว่างไมโครซอฟต์วินโดวส์กับโปรแกรมประยุกต์

จากรูปที่ 3.2 เมื่อเกิดข้อความ เช่น ผู้ใช้กดแป้นอักขระ วินโดวส์จะเก็บข้อความไว้ที่แถวคอยของระบบและพิจารณาว่าเป็นข้อความของโปรแกรมประยุกต์ใด แล้วส่งไปเก็บไว้ที่แถวคอยของโปรแกรมประยุกต์นั้น จากนั้นโปรแกรมประยุกต์จะรับข้อความมาตรวจสอบว่าเป็นข้อความการหยุดโปรแกรมหรือไม่ ถ้าใช่ก็จะหยุดการทำงาน of โปรแกรมนั้น แต่ถ้าไม่ใช่ก็จะจัดส่งกลับไปยังวินโดวส์ เพื่อให้วินโดวส์จัดส่งไปยังวินโดว์ฟังก์ชันของโปรแกรมประยุกต์ที่เหมาะสมเพื่อประมวลผลข้อความนั้นตามต้องการต่อไป

ลักษณะการทำงานของวินโดวส์โดยใช้ข้อความนี้เอง ทำให้วินโดวส์สามารถทำงานได้หลายงานพร้อมๆ กัน โดยการสลับการทำงานของแต่ละโปรแกรมประยุกต์จะขึ้นอยู่กับข้อความ กล่าวคือถ้าโปรแกรมประยุกต์ที่กำลังทำงานอยู่ไม่มีข้อความในแถวคอย แต่มีอีกโปรแกรมประยุกต์ที่มีข้อความรออยู่ในแถวคอย วินโดวส์ก็จะสลับการทำงานไปยังโปรแกรมประยุกต์หลังแทน ซึ่งการสลับการทำงานแบบนี้เรียกว่า *jumpymultitasking* หรือ *nonpreemptive multitasking* (Petzold, 1990)

### ลักษณะของโปรแกรม

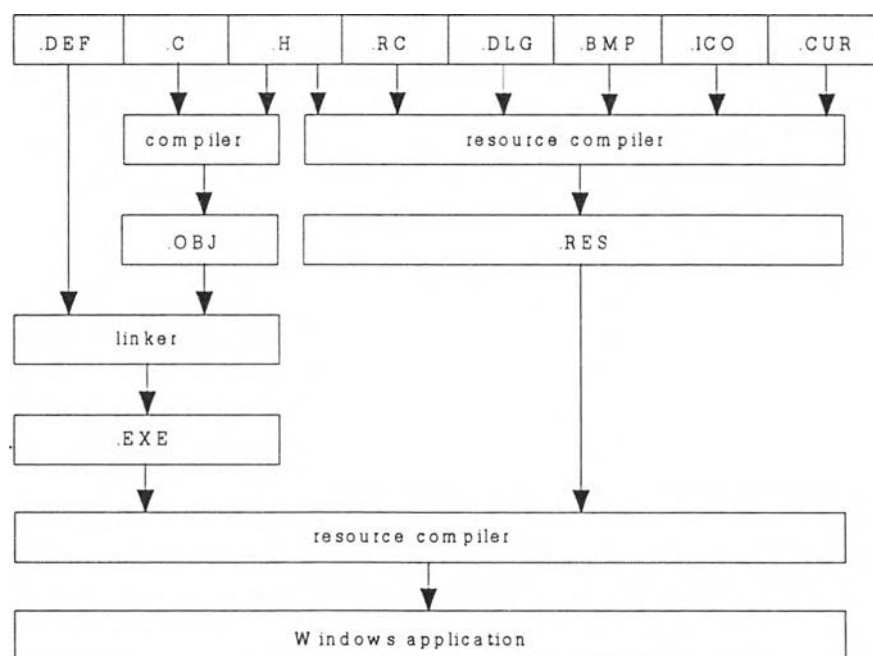
โปรแกรมที่เขียนขึ้นสำหรับการทำงานบนไมโครซอฟต์วินโดวส์นั้น มี 2 ประเภท ดังนี้

1. Executable Program (.EXE files) เป็นโปรแกรมที่สามารถทำงานได้ด้วยตนเองเช่นเดียวกับโปรแกรมต่างๆที่เขียนขึ้นบนระบบปฏิบัติการ DOS ทั่วๆไป

2. Dynamic Link Libraries (.DLL files) เป็นโปรแกรมชนิดหนึ่งที่ไม่สามารถทำงานได้ด้วยตนเอง กล่าวคือ โปรแกรมประเภทนี้จะเป็นการรวบรวมฟังก์ชันการทำงานต่างๆไว้ให้โปรแกรมประเภทเพิ่ม .EXE เรียกใช้ ความแตกต่างที่สำคัญของโปรแกรมประเภทนี้ก็คือ หน่วยความจำที่ใช้ เนื่องจากโปรแกรมที่ทำงานบนไมโครซอฟต์ วินโดวส์สามารถทำงานได้พร้อมๆกัน วินโดวส์จะจัดสรรหน่วยความจำให้ทุกครั้งที่มีการเรียกใช้โปรแกรมประเภทเพิ่ม .EXE นั่นคือแต่ละโปรแกรมก็จะมีหน่วยความจำของตนเอง ในขณะที่โปรแกรมประเภทเพิ่ม .DLL นี้ วินโดวส์ จะจัดสรรหน่วยความจำให้เพียงชุดเดียวและถูกเรียกใช้จากโปรแกรมหลายๆโปรแกรมได้ ทำให้สามารถประหยัด หน่วยความจำ และขนาดของโปรแกรมประเภทเพิ่ม .EXE นั้นก็จะมีขนาดเล็กลง อีกทั้งการแก้ไขฟังก์ชันการทำงาน ต่างๆก็ทำกับโปรแกรมประเภทเพิ่ม .DLL นี้เพียงอย่างเดียว ไม่ต้องไปแก้ไขโปรแกรมประเภทเพิ่ม .EXE อีก

### เทคนิคการเขียนโปรแกรม

หลักการเขียนโปรแกรมเพื่อทำงานบนไมโครซอฟต์วินโดวส์อยู่บนพื้นฐานของการเขียนโปรแกรมแบบมัลติโมดูล (multimodule programming) กล่าวคือจะประกอบด้วยเพิ่มประเภทต่างๆที่ถูกแปล และนำมาเชื่อมต่อกัน เป็นโปรแกรมประเภทเพิ่ม .EXE ในที่นี้จะยกตัวอย่างภายใต้การเขียนโปรแกรมด้วยภาษาซี ซึ่งเป็นภาษาคอมพิวเตอร์ ที่ใช้ในการพัฒนาโดยเพิ่มต่างๆมีความสัมพันธ์กันดังแสดงในรูปที่ 3.3



รูปที่ 3.3 แสดงส่วนประกอบและขั้นตอนการสร้างโปรแกรม

1. เพิ่มต้นฉบับ (Source File - .C file) เป็นเพิ่มโปรแกรมที่ประกอบด้วยคำสั่งต่างๆที่เขียนขึ้นโดยใช้ภาษาคอมพิวเตอร์ เช่น ซี ปาสคาล ฯลฯ ภายในเพิ่มจะประกอบด้วยฟังก์ชันต่างๆตามขั้นตอนการทำงานที่กำหนด โดยโครงสร้างของเพิ่มจะต้องประกอบด้วยส่วนสำคัญดังนี้

1.1 การ include เพิ่มประเภท .H เช่น windows.h ซึ่งเป็นเพิ่มข้อมูลส่วนหัวที่ต้องมีเสมอ เนื่องจากเป็นเพิ่มที่เก็บการประกาศฟังก์ชันต่างๆที่วินโดวส์เตรียมไว้ให้ใช้ ได้แก่ ประเภทข้อมูล ค่าคงที่และโครงสร้างข้อมูลต่างๆ

1.2 ฟังก์ชัน WinMain เป็นฟังก์ชันที่เป็นจุดเริ่มต้นของการทำงานคล้ายกับฟังก์ชัน main ของโปรแกรมภาษาซีทั่วไป โดยจะต้องใช้ชื่อนี้เท่านั้นและการทำงานในฟังก์ชันนี้จะต้องประกอบด้วย

1.2.1 การลงทะเบียนวินโดวส์คลาส (register window class) เป็นการกำหนดค่าให้กับตัวแปรโครงสร้างชนิด WNDCLASS เพื่อกำหนดรายละเอียดลักษณะพื้นฐานของวินโดวส์ที่ต้องการสร้างในโปรแกรม เช่น ชื่อของ window procedure ลัญจรูปและตัวชี้ตำแหน่งของวินโดวส์นั้นๆ การลงทะเบียนนี้จะกระทำเพียงครั้งแรกที่โปรแกรมถูกเรียกขึ้นมาทำงาน

1.2.2 การสร้างวินโดวส์ (creating window) ในขั้นตอนนี้จะเป็นการสร้างวินโดวส์หลักของโปรแกรม โดยจะมีการกำหนดชนิดของวินโดวส์ที่ต้องการสร้าง ขนาดและตำแหน่งเริ่มต้นของวินโดวส์ รายการเลือกการสร้างวินโดวส์นี้จะเป็นการสร้างขึ้นภายในตัวไมโครซอฟต์วินโดวส์เท่านั้น ยังไม่ถูกแสดงผลบนจอภาพจะต้องเรียกฟังก์ชัน ShowWindow และ UpdateWindow โดยฟังก์ชัน ShowWindow จะทำหน้าที่สร้างวินโดวส์ตามลักษณะที่ต้องการบนหน้าจอและฟังก์ชัน UpdateWindow จะทำหน้าที่ส่งข้อความ WM\_PAINT ไปให้กับ window procedure เพื่อให้พื้นที่ภายในวินโดวส์ถูกวาดตามที่โปรแกรมกำหนด

1.2.4 วัฏวนข้อความ (message loop) เป็นส่วนที่โปรแกรมจะเชื่อมต่อกับวินโดวส์ โดยจะเป็นการทำงานในลักษณะวนไปเรื่อยๆจนกว่าต้องการจะจบการทำงานของโปรแกรม ภายในวัฏวนจะมีฟังก์ชันการทำงานสองฟังก์ชัน ได้แก่ TranslateMessage ที่ทำหน้าที่แปลงข้อความเกี่ยวกับแป้นอักขระให้มีค่าที่สามารถนำไปใช้งานได้ และ DispatchMessage ที่ทำหน้าที่ส่งข้อความกลับไปยังวินโดวส์ เพื่อให้วินโดวส์จัดการส่งไปยัง window function ต่อไป

1.2.5 Window procedure ในส่วนนี้ถือว่าเป็นส่วนสำคัญของการเขียนโปรแกรมเพื่อทำงานบนไมโครซอฟต์วินโดวส์ เพราะว่าเป็นส่วนที่ทำหน้าที่ประมวลผลข้อความต่างๆที่ถูกส่งมา โดยชื่อของฟังก์ชันจะต้องกำหนดไว้ในขั้นตอนการลงทะเบียนวินโดวส์คลาส เพื่อให้วินโดวส์รู้ว่าต้องส่งข้อความไปประมวลผลที่ฟังก์ชันใด โดยทั่วไปรูปแบบภายในของฟังก์ชันจะอยู่ในรูปของการใช้คำสั่ง switch - case statement เพื่อเลือกเอาข้อความไปประมวลผลเฉพาะตามต้องการ ส่วนข้อความที่ไม่ต้องการจะถูกส่งไปที่ DefWindowProc

ซึ่งเป็นฟังก์ชันภายในวินโดวส์ที่ถูกกำหนดให้ทำงานกับข้อความต่างๆที่ไม่ได้ถูกประมวลผล ซึ่งค่าที่ส่งกลับจาก DefWindowProc นี้จะถูกส่งกลับคืนสู่วินโดวส์อีกทีหนึ่ง

ตัวอย่างของเพิ่มเติมฉบับที่ทำการลงทะเบียนวินโดวส์คลาส การสร้างวินโดวส์ ว่างวนข้อความของ ฟังก์ชัน และ Window procedure แสดงได้ดังนี้

```

/* Example .C file */
#include <windows.h>

long FAR PASCAL WndProc(HWND, WORD, WORD, WORD);

int PASCAL WinMain(HANDLE hInstance, HANDLE hPrevInstance,
                  LPSTR lpszCmdParam, int nCmdShow)
{
    HWND      hWnd;
    MSG       msg;
    WNDCLASS  wndclass;

    if (!hPrevInstance)
    {
        wndclass.style          = CS_HREDRAW | CS_VREDRAW;
        wndclass.lpfnWndProc    = WndProc;
        wndclass.cbClsExtra     = 0;
        wndclass.cbWndExtra     = 0;
        wndclass.hInstance      = hInstance;
        wndclass.hIcon          = LoadIcon(NULL, IDI_APPLICATION);
        wndclass.hCursor        = LoadCursor(NULL, IDC_ARROW);
        wndclass.hbrBackground  = GetStockObject(WHITE_BRUSH);
        wndclass.lpszMenuName   = NULL;
        wndclass.lpszClassName  = "Example";

        RegisterClass(&wndclass);
    }
    hWnd = CreateWindow("Example", "Example source file",
                      WS_OVERLAPPEDWINDOW, CW_USEDEFAULT, CW_USEDEFAULT,
                      CW_USEDEFAULT, CW_USEDEFAULT,
                      NULL, NULL, hInstance, NULL);

    ShowWindow(hWnd, nCmdShow);
    UpdateWindow(hWnd);
    while (GetMessage(&msg, NULL, 0, 0))
    {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
    return msg.wParam;
}

long FAR PASCAL WndProc(HWND hWnd, WORD message, WORD wParam, LONG lParam)
{
    HDC      hDC;
    PAINTSTRUCT ps;

```

```

switch (message)
{
    case WM_PAINT:
        hDC = BeginPaint(hWnd, &ps);
        EndPaint(hWnd, &ps);
        return 0;
}
return DefWindowProc(hWnd, message, wParam, lParam);
}

```

2. เพิ่มข้อมูลกำหนดโมดูล (Module Definition File - .DEF file) เป็นแฟ้มที่ช่วยในขั้นตอนการเชื่อมต่อโปรแกรมโดยประกอบด้วยข้อมูลทางเทคนิคของโปรแกรมที่สร้างขึ้น เช่น ขนาดของสแตค ขนาดของหน่วยความจำขณะเริ่มต้นโปรแกรม ชื่อของโปรแกรมที่จะถูกเรียกให้ทำงาน คุณสมบัติของ code segment และ data segment รวมทั้งชื่อของ window procedure ทั้งหมด ตัวอย่างของโมดูล .DEF file แสดงได้ดังนี้

```

/* example of .DEF file */
NAME        EXAMPLE
DESCRIPTION  'example of module definition file'
EXETYPE     WINDOWS
STUB        'WINSTUB.EXE'
CODE        PRELOAD MOVEABLE DISCARDABLE
DATA        PRELOAD MOVEABLE MULTIPLE
HEAPSIZE    1024
STACKSIZE   8192
EXPORTS     WndProc

```

3. เพิ่มข้อมูลส่วนหัว (Header File - .H file) เป็นแฟ้มข้อมูลส่วนหัวเช่นเดียวกับการเขียนโปรแกรมภาษาซีทั่วไป

4. เพิ่มข้อมูลทรัพยากร (Resource Script File - .RC file) เป็นแฟ้มข้อมูลที่ใช้กำหนดการใช้ทรัพยากรต่างๆของโปรแกรมประกอบด้วย การบรรยายการสร้างรายการเลือกและเป็นลัด ชื่อความชื่อและชนิดของแฟ้มข้อมูลแบบแผนที่บิต แฟ้มข้อมูลสัญรูป แฟ้มข้อมูลตัวชี้ตำแหน่ง รวมทั้งแฟ้มกล่องคำโต้ตอบ (dialog box) ต่างๆ ตัวอย่างของแฟ้มข้อมูลแสดงได้ดังนี้

```

/* example of .RC file */
#include <windows.h>

curl        CURSOR    cursor.cur
icol        ICON      icon.ICO
bmp1        BITMAP    bmp.BMP

```

Example MENU

```

{
POPUP "&File"
{
    MENUITEM "&New",    IDM_NEW
    MENUITEM "&Save",   IDM_SAVE
}
}

```



```

    MENUITEM "&Quit",    IDM_QUIT
}
}

```

5. เพิ่มข้อมูลกล่องคำโต้ตอบ (Dialog File - .DLG file) เป็นเพิ่มข้อมูลที่ใช้ในการบรรยายการสร้างกล่องคำโต้ตอบต่างๆ

6. เพิ่มข้อมูลแบบแผนที่บิต (Bitmap File - .BMP file) เป็นเพิ่มข้อมูลรูปภาพแบบแผนที่บิตที่ใช้ในโปรแกรม

7. เพิ่มข้อมูลสัญรูป (Icon File - .ICO file) เป็นเพิ่มข้อมูลรูปภาพสัญรูปที่ใช้ในโปรแกรม

8. เพิ่มข้อมูลตัวชี้ตำแหน่ง (Cursor File - .CUR file) เป็นเพิ่มข้อมูลรูปภาพตัวชี้ตำแหน่งที่ใช้ในโปรแกรม

เพิ่มข้อมูลทั้งหมดนี้จะถูกแปลและนำมาเชื่อมต่อกัน จนกลายเป็นโปรแกรมที่สามารถทำงานบนไมโครซอฟต์วินโดวส์ได้ สำหรับรายละเอียดปลีกย่อยอื่นๆตลอดจนเทคนิคขั้นสูง สามารถศึกษาเพิ่มเติมได้จากคู่มือสำหรับการเขียนโปรแกรมเพื่อใช้งานบนไมโครซอฟต์วินโดวส์ (Petzold, 1990; Microsoft Staff, 1990b, 1990c, 1990d; Norton and Yao, 1990; Rector, 1992; Richter, 1991)