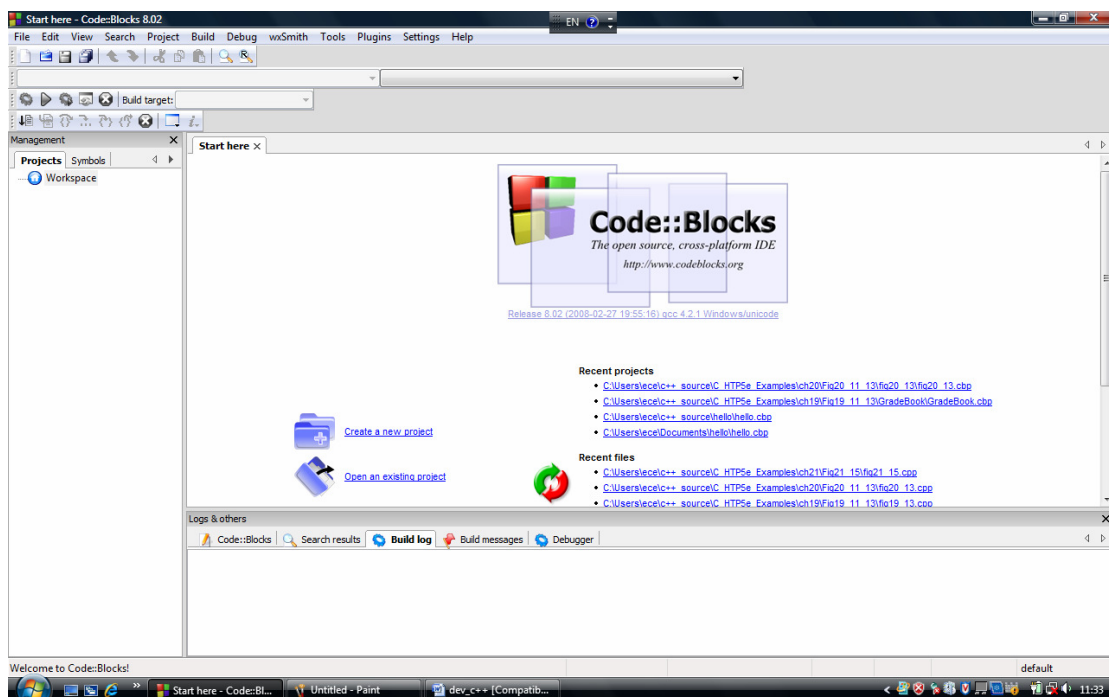


วิธีการใช้โปรแกรม Code::Blocks

ซอฟต์แวร์ Code::Blocks เป็นซอฟต์แวร์ Editor (ส่วนเรียบเรียงโปรแกรม) ที่สามารถใช้ร่วมกับ Compiler (ส่วนแปลภาษา) ของภาษาต่างๆ อันได้แก่ ภาษา C ภาษา Fortran ภาษา Java เป็นต้น ได้อย่างหลากหลาย ซอฟต์แวร์นี้จัดเป็น Free Software อันหนึ่ง ส่วนประกอบสำคัญของซอฟต์แวร์นี้คือ ส่วน Integrated Development Environment (IDE) ซึ่งเป็นสิ่งอำนวยความสะดวกให้กับผู้เขียนโปรแกรม เพื่อให้สามารถแปลภาษาที่ต้องการได้ จำเป็นต้องกำหนด Compiler ที่ใช้งานให้กับ Code::Blocks สำหรับ Code::Blocks ที่ใช้ในการเรียนการสอนวิชานี้ได้กำหนดให้ใช้ GNU C++ Compiler ซึ่งใช้ได้ทั้งในการแปลภาษา C และ C++ รูปที่ 1 แสดงหน้าจอเริ่มต้นของ Code::Blocks



รูปที่ 1 หน้าจอเริ่มต้นของ Code::Blocks

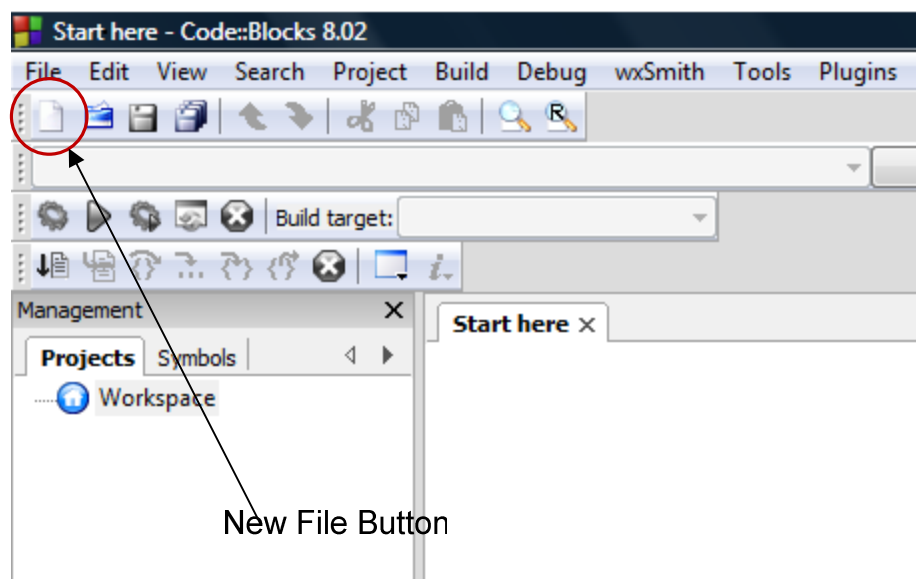
การใช้งานทั่วไปของ Code::Blocks จะเริ่มต้นด้วยการสร้างไฟล์โปรแกรม เรียบเรียง แล้วทำการ Compile เพื่อสร้างเป็นไฟล์ Executable (ไฟล์ที่สั่งงานได้) ในกรณีที่พัฒนาโปรแกรมขนาดใหญ่ ซึ่งอาจจะประกอบด้วยไฟล์จำนวนมากกว่า 1 ไฟล์ ควรจะเรียบเรียงโปรแกรมให้อยู่ในรูป Project ในกรณีนี้ การใช้งาน Code::Blocks จะเริ่มที่การสร้าง Project พร้อมกำหนดค่าตัวเลือกต่างๆของ Project และทำการเพิ่มไฟล์เข้าไปใน Project จากนั้นก็จะเป็นขั้นตอนการแก้ไขไฟล์ และการ Compile เมื่อ Compile สำเร็จแล้ว จึงจะสามารถรันโปรแกรมที่พัฒนาขึ้นได้

1 การสร้างไฟล์ใหม่

โดยทั่วไป สิ่งแรกสำหรับการเขียนโปรแกรมคือ การสร้างไฟล์โปรแกรม (ซึ่งเรียกว่า Source code หรือ Source file) ใน Code::Blocks ให้ทำการคลิกที่ปุ่ม New File ดังแสดงในรูปที่ 2 หรือเลือกจากแถบคำสั่ง File (รูปที่ 3) แล้วเลือก New รูปที่ 4 แสดงตัวเลือกของปุ่ม New File ถ้าต้องการสร้างไฟล์ที่ไม่เจาะจงประเภท ให้เลือก Empty

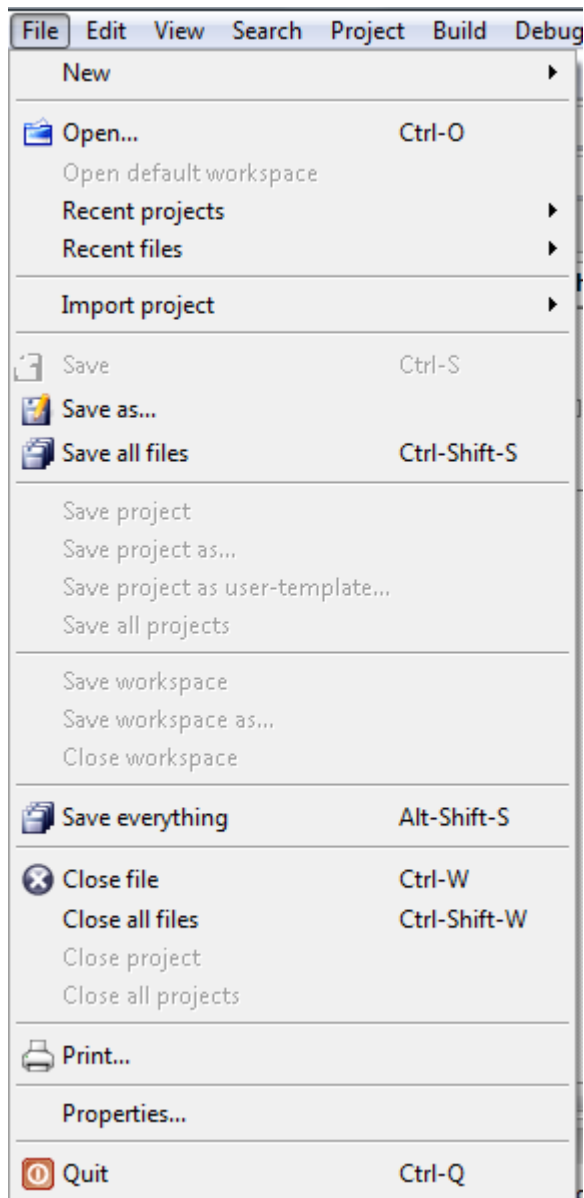
วิธีการใช้โปรแกรม Code::Blocks

file ซึ่งสามารถกำหนดภายหลังได้ แต่ถ้าต้องการสร้างไฟล์โปรแกรมภาษา C (หรือ C++) ให้เลือก File ซึ่งวินโดว์ในรูปที่ 5 จะปรากฏขึ้น และ Code::Blocks จะใช้ template ในการช่วยผู้ใช้ ในที่นี้ เราสามารถเลือกประเภทของไฟล์ที่ต้องการสร้างได้ ถ้าต้องการสร้างไฟล์โปรแกรมภาษา C ให้คลิกที่ C/C++ source แล้วทำตามขั้นตอนที่ Code::Blocks เตรียมไว้ กล่าวคือ เลือก C หรือ C++ และกำหนดชื่อไฟล์ เมื่อเสร็จสิ้นขั้นตอนนี้ หน้าจอจะเป็นดังตัวอย่างในรูปที่ 6 เราก็จะเริ่มการเขียนโปรแกรมได้ หลังจากนั้น เราสามารถทำการเพิ่มข้อความในส่วน Editor แล้วทำการบันทึกการเปลี่ยนแปลงโดยคลิกที่ปุ่ม Save (รูปที่ 7) หรือเลือกจากแถบคำสั่ง File แล้วเลือก Save (รูปที่ 3) หรือกดปุ่ม Ctrl-s ให้สังเกตว่า ไฟล์ที่อยู่ในระหว่างการแก้ไขจะมีเครื่องหมาย * อยู่หน้าชื่อไฟล์ ดังแสดงในรูปที่ 8 อนึ่ง ถ้าต้องการปิดไฟล์ที่กำลังแก้ไขอยู่ ให้คลิกที่เครื่องหมาย X หลังชื่อไฟล์ หรือใช้คำสั่ง Close file จากแถบเมนู File (ดูรูปที่ 3 ประกอบ)

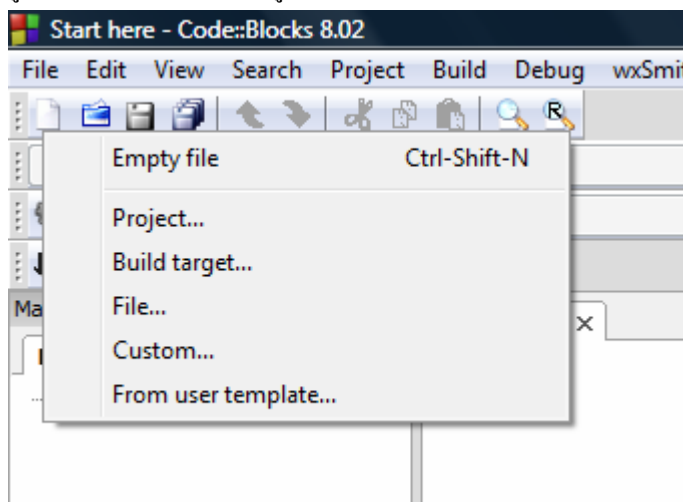


รูปที่ 2 ปุ่ม new file สำหรับสร้างไฟล์ใหม่

วิธีการใช้โปรแกรม Code::Blocks

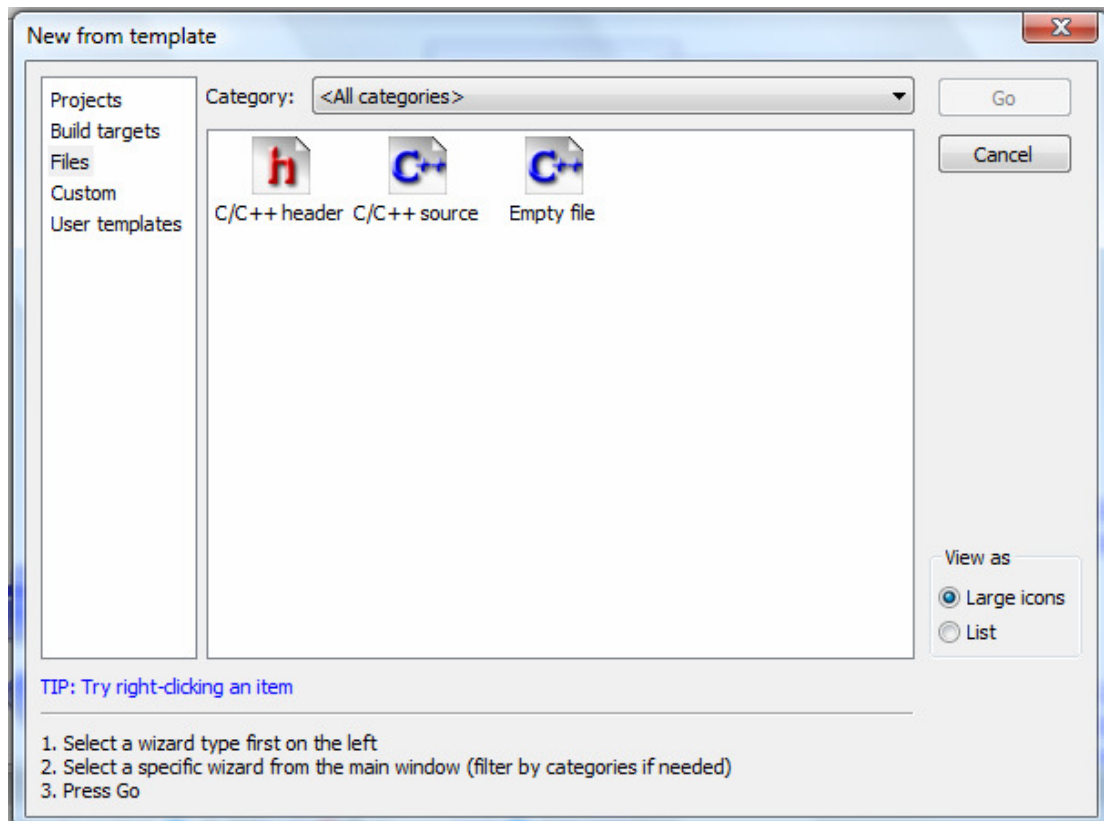


รูปที่ 3 ตัวเลือกของแถบเมนู File

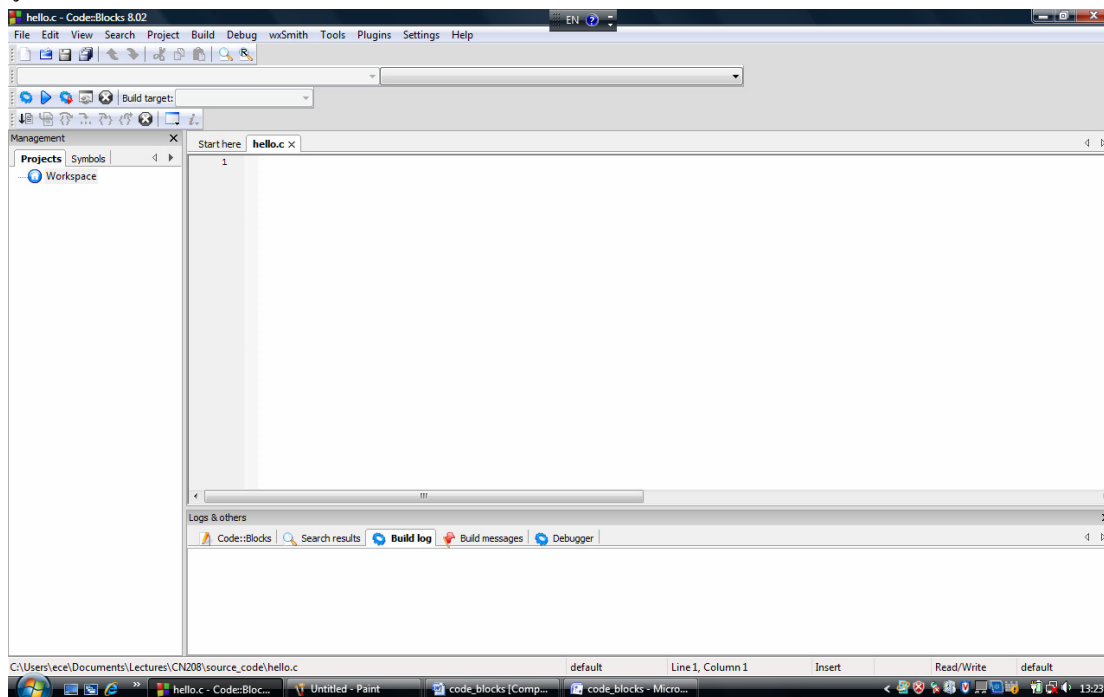


รูปที่ 4 ตัวเลือกของปุ่ม New File

วิธีการใช้โปรแกรม Code::Blocks

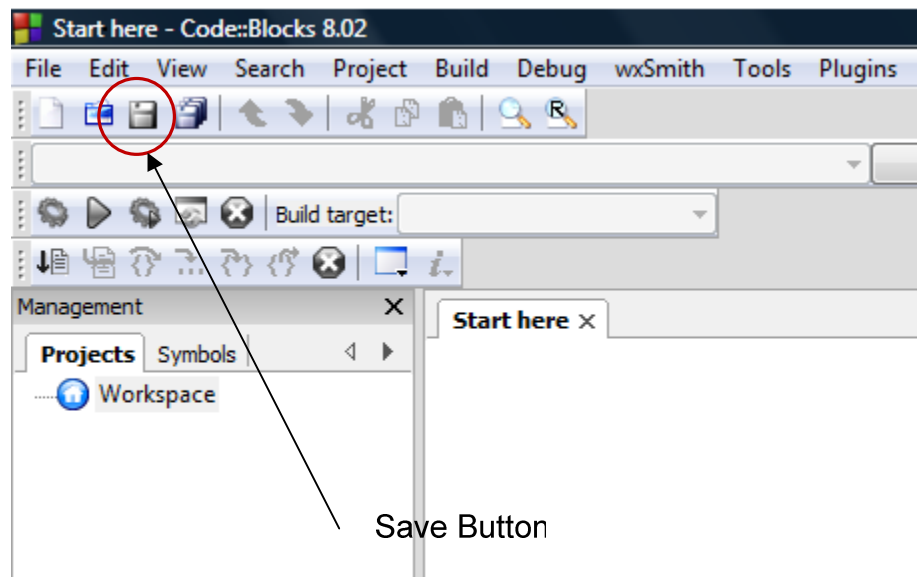


รูปที่ 5 วินโดว์หลังเลือกตัวเลือก File

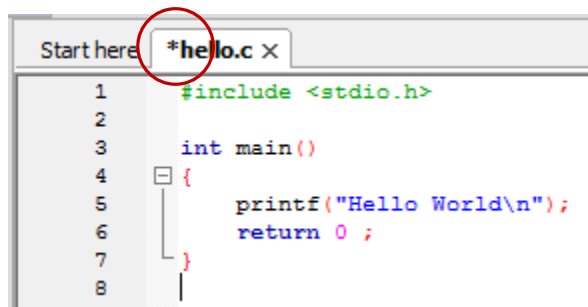


รูปที่ 6 หน้าจอของโปรแกรมหลังกำหนดชื่อไฟล์ที่จะสร้าง ในที่นี้ กำหนดให้ไฟล์ที่จะสร้างมีชื่อว่า hello.c

วิธีการใช้โปรแกรม Code::Blocks



รูปที่ 7 ปุ่ม Save

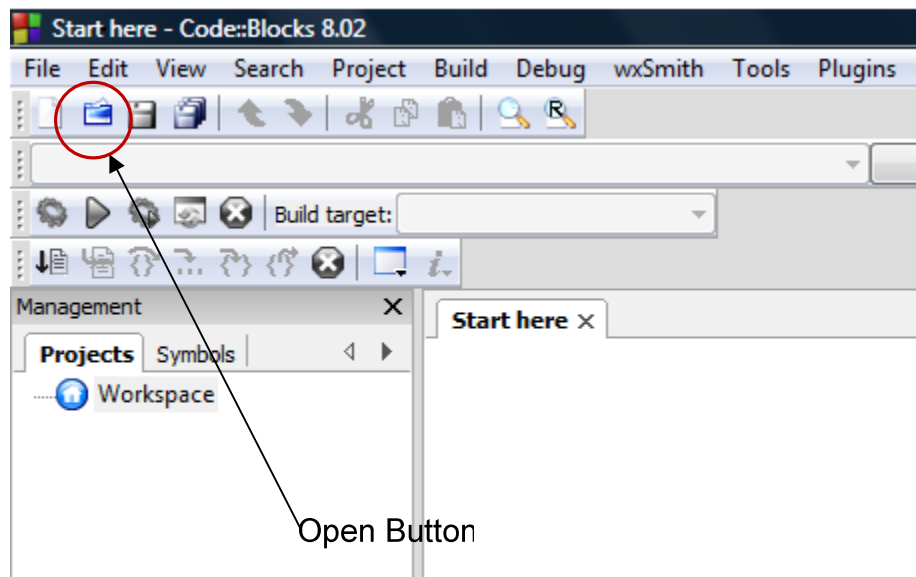


รูปที่ 8 ไฟล์ที่อยู่ระหว่างการแก้ไข

2 การแก้ไขไฟล์เดิม

ถ้าต้องการแก้ไขไฟล์เดิมที่สร้างไว้ก่อนหน้านี้ สามารถทำได้โดยการคลิกที่ปุ่ม Open (รูปที่ 9) หรือเลือกจากแถบคำสั่ง File แล้วเลือก Open จากนั้นให้เลือกชนิดของไฟล์ให้เป็น C/C++ files เมื่อเปิดไฟล์ได้สำเร็จ หน้าจอก็จะคล้ายกับที่แสดงในรูปที่ 6 ซึ่งเราสามารถทำการแก้ไขโปรแกรมได้ในลักษณะเดียวกับกรณีสร้างไฟล์ใหม่ เมื่อแก้ไขเรียบร้อยแล้วให้ทำการบันทึกโดยใช้คำสั่ง Save ดังที่กล่าวในตอนที่แล้ว อนึ่ง คำสั่งที่ใช้ในการเรียงเรียงที่ใช้อยู่บ่อยครั้งได้แก่ cut copy paste delete undo redo เป็นต้น ซึ่งจะคล้ายกับคำสั่งใน Microsoft Word หรือโปรแกรม Editor อื่นๆ อันจะสามารถดูได้จากแถบเมนู Edit

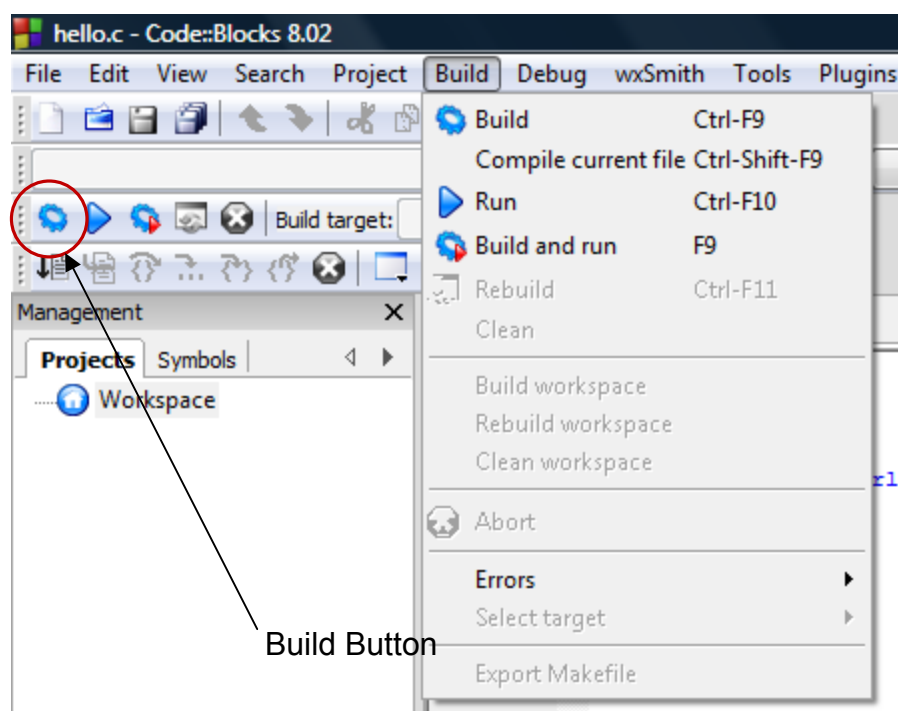
วิธีการใช้โปรแกรม Code::Blocks



รูปที่ 9 ปุ่ม Open

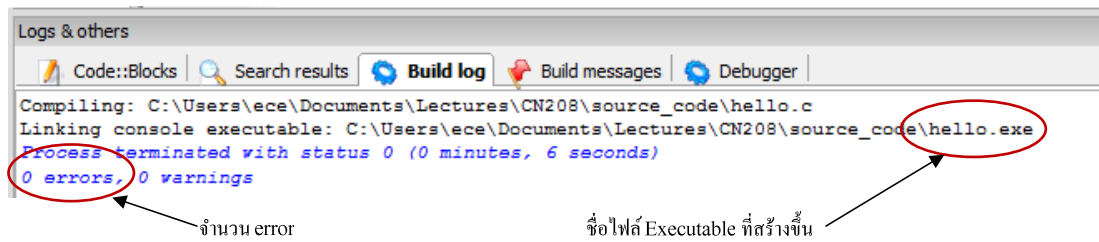
3 การคอมไพล์และรันโปรแกรม

เมื่อทำการแก้ไขโปรแกรมเรียบร้อยแล้ว เราสามารถทำการ Compile โปรแกรมที่เขียนขึ้นได้โดยคลิกที่ปุ่ม Build หรือเลือกจากเมนู Build->Build หรือกดปุ่ม Ctrl-F9 ดังแสดงในรูปที่ 10 หลังจากคอมไพล์แล้ว จะมีข้อความแสดงผลการคอมไพล์ปรากฏขึ้นที่ส่วน Build log ดังรูปที่ 11 ในรูปนี้ จะแสดงให้เห็นว่า โปรแกรมที่ถูกทำการคอมไพล์ปราศจากข้อผิดพลาด กล่าวคือ จำนวน error เท่ากับ 0 และให้สังเกตด้วยว่า ไฟล์ชื่อว่า "hello.exe" ซึ่งเป็นไฟล์ที่สามารถใช้งานได้ (Executable file) จะถูกสร้างขึ้น เราสามารถรันโปรแกรมโดยกดปุ่ม Run (รูปที่ 12) หรือเลือกจากเมนู Build->Run หรือกดปุ่ม Ctrl-F10 รูปที่ 13 เป็นหน้าจอแสดงผลการรันโปรแกรมนี้

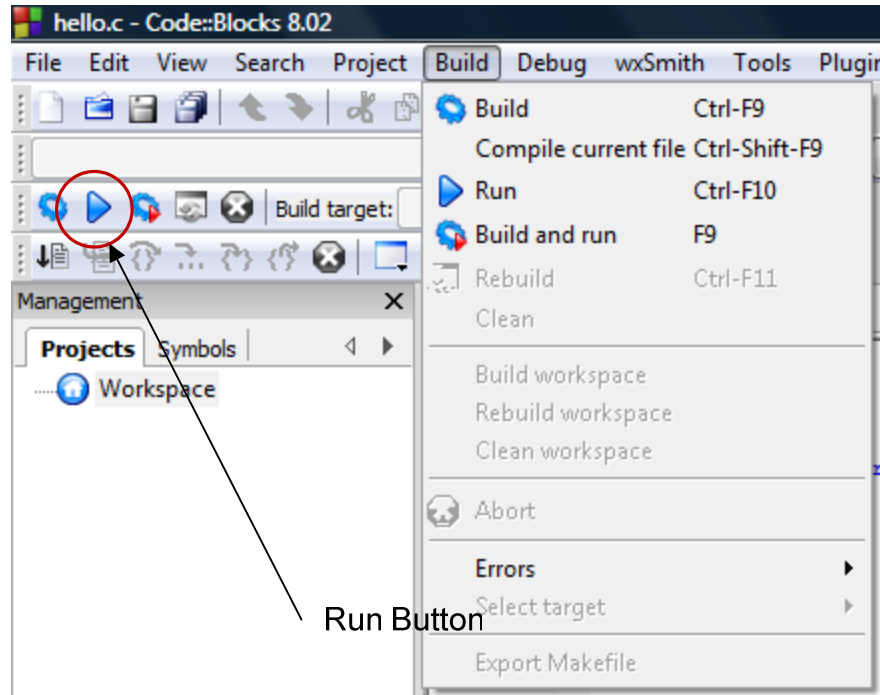


รูปที่ 10 ปุ่ม Build และการเลือกคำสั่ง Build

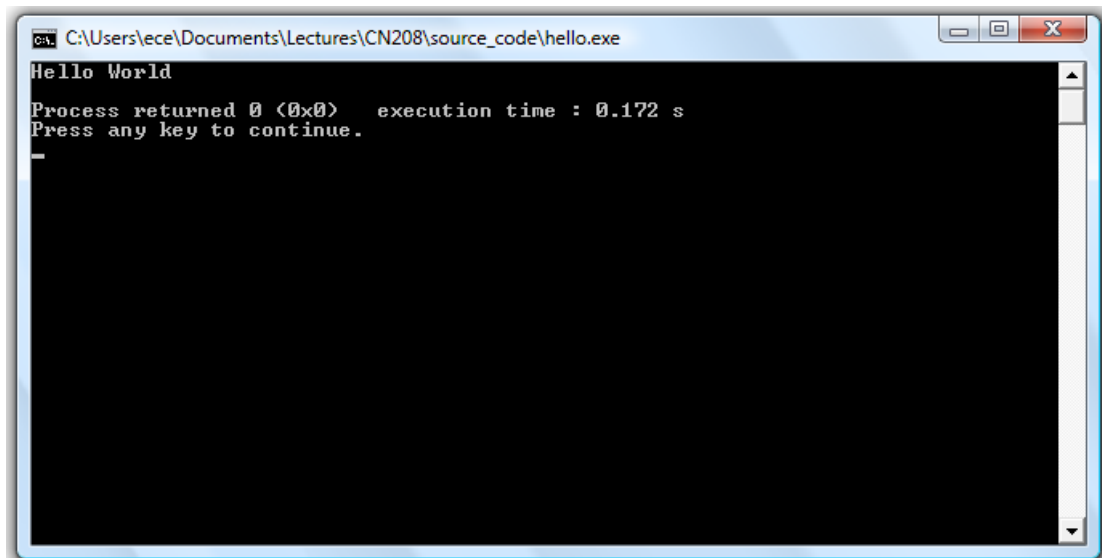
วิธีการใช้โปรแกรม Code::Blocks



รูปที่ 11 หน้าจอแสดง Build log



รูปที่ 12 ปุ่ม Run และการเลือกคำสั่ง run

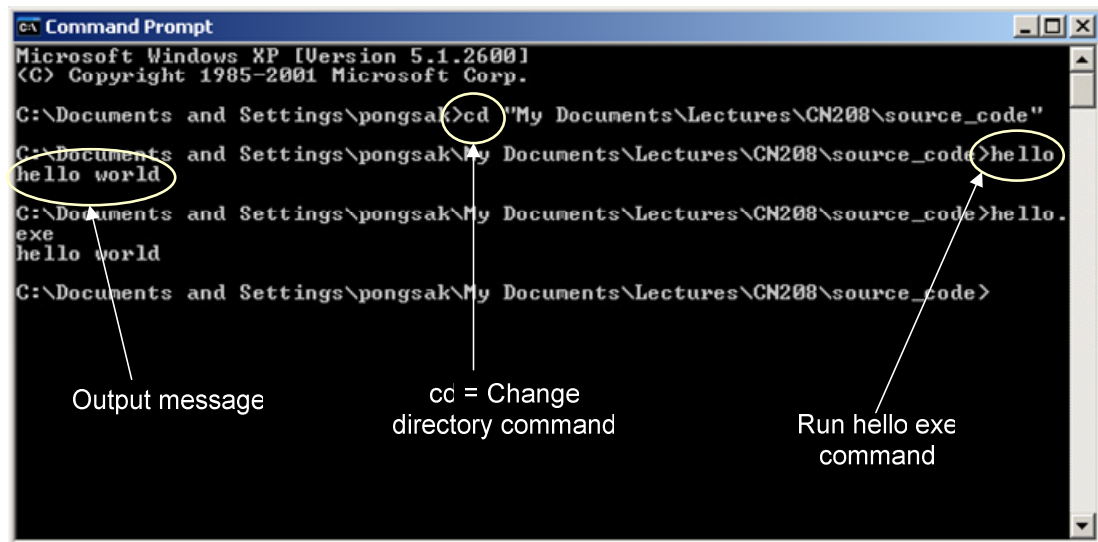


รูปที่ 13 หน้าจอแสดงผลการทำงานของโปรแกรม ในที่นี้ โปรแกรมแสดงข้อความ “Hello World”

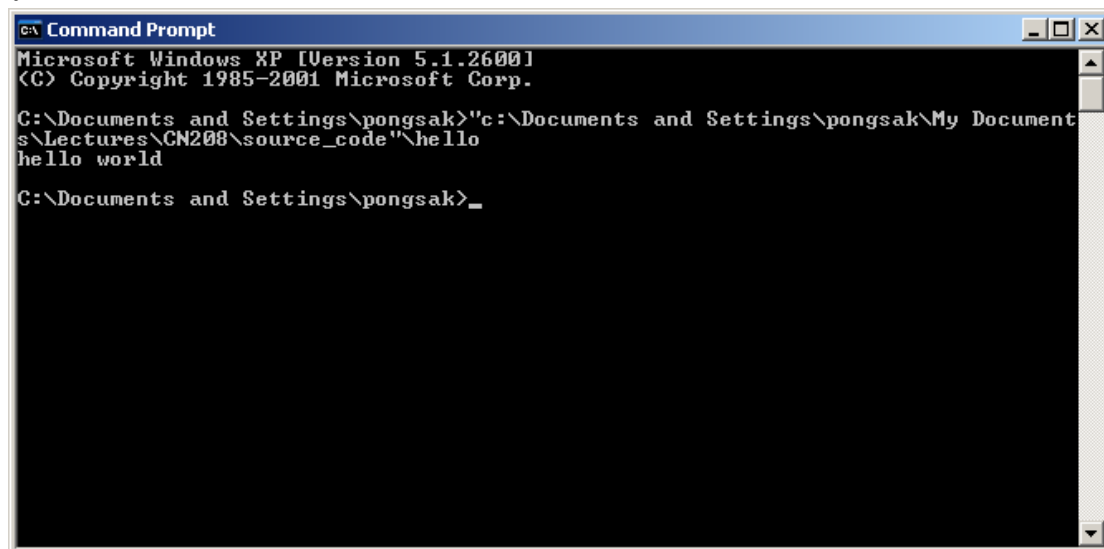
4 การรันโปรแกรมจาก Command Prompt

วิธีการใช้โปรแกรม Code::Blocks

นอกเหนือจากวิธีดังกล่าวข้างต้นแล้ว เราสามารถรันโปรแกรมที่ Command Prompt โดยพิมพ์ชื่อไฟล์ Executable ที่ต้องการเข้าไป ดังแสดงในรูปที่ 14 ในที่นี้ สมมติให้ชื่อไฟล์ Executable เป็น hello.exe สังเกตว่า จะมี .exe หรือไม่ก็ได้ ข้อควรระวังคือ จะต้องอยู่ใน Directory ที่มีไฟล์นั้นอยู่หรือไม่ก็ต้องใส่ชื่อ Path ให้ครบถ้วน (Absolute Path) ดังแสดงในรูปที่ 15 การรันวิธีนี้จะเป็นประโยชน์ในกรณีที่มีการกำหนดให้โปรแกรมรับค่าจาก Command Line โดยใช้ Command-line Argument



รูปที่ 14 หน้าจอแสดงตัวอย่างการรันโปรแกรมจาก Command Prompt



รูปที่ 15 หน้าจอแสดงตัวอย่างการรันโปรแกรมจาก Command Prompt โดยใช้ absolute path

แบบฝึกหัด

1. เขียนโปรแกรมสำหรับแสดงชื่อ เลขประจำตัว และทัศนคติอะไรก็ได้ที่เกี่ยวกับการเขียน โปรแกรม
2. สร้างไฟล์โปรแกรมใหม่แล้วเพิ่มคำสั่งต่อไปนี้ในโปรแกรม

```
int i, j;  
float x, y;  
i = 8;  
j = 5;
```


วิธีการใช้โปรแกรม Code::Blocks

```
x = 8.0;
y = 5.0;
printf("%d * %d = %d\n", i, j, i*j);
printf("%d / %d = %d\n", i, j, i/j);
printf("%f * %f = %f\n", x, y, x*y);
printf("%f / %f = %f\n", x, y, x/y);
```

จากนั้น ทำการคอมไพล์แล้วทำการรันโปรแกรม จงอธิบายการทำงานของโปรแกรม และข้อแตกต่างระหว่างผลลัพธ์ที่ได้จาก i/j และ x/y พร้อมเหตุผลประกอบ

(HINT: สังเกตที่ชนิดของข้อมูลของ i, j, x, y)

5 การสร้าง Project ใหม่

ในกรณีที่ต้องการพัฒนาโปรแกรมขนาดใหญ่ที่ประกอบด้วยไฟล์มากกว่า 1 ไฟล์ หรือต้องการกำหนดตัวเลือกของโปรแกรม เช่น console application¹ หรือ windows application เป็นต้น การจัดโปรแกรมให้อยู่ในรูป Project ทำให้เกิดประสิทธิภาพในการพัฒนาโปรแกรมที่สูงขึ้น เนื่องจาก

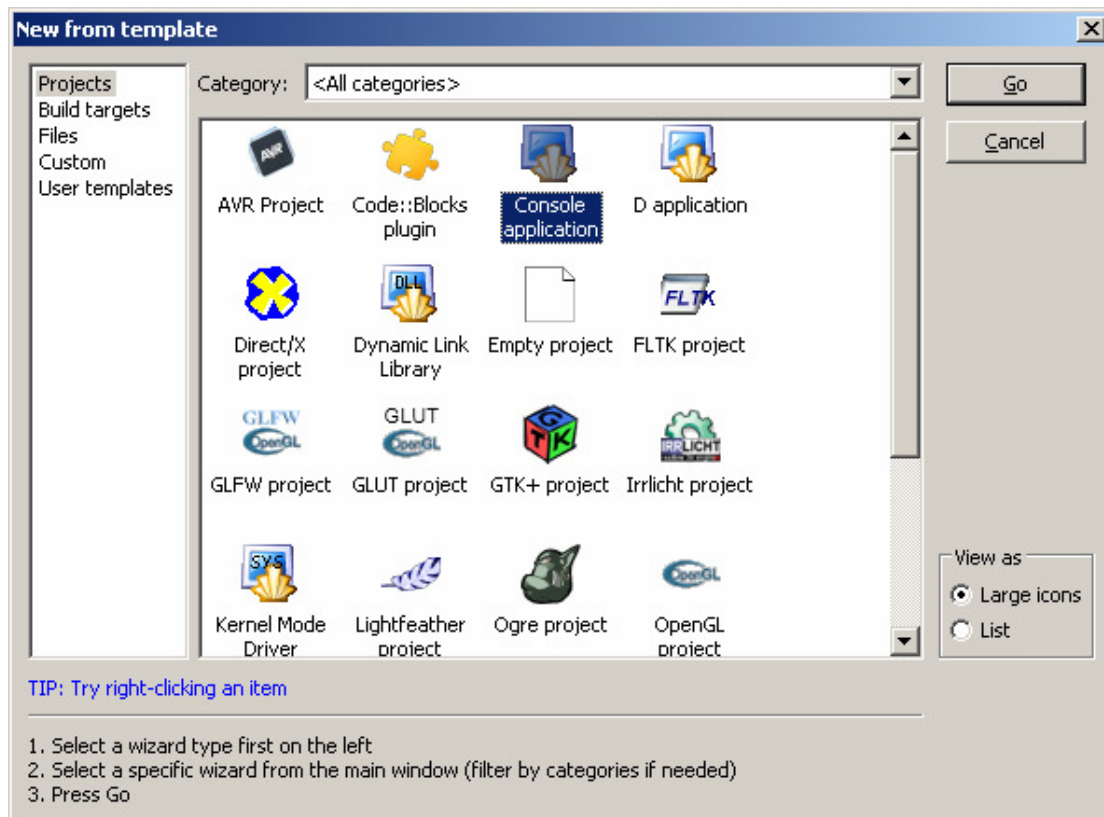
- การแบ่งโปรแกรมให้แต่ละไฟล์เล็กลง ทำให้ง่ายต่อการเขียนและแก้ไขข้อผิดพลาด
- การคอมไพล์ไฟล์ที่สั้นใช้เวลาน้อยกว่าไฟล์ที่ยาว
- ในกรณีที่มีการแก้ไขเฉพาะบางไฟล์ Compiler จะทำการคอมไพล์เฉพาะไฟล์ที่มีการแก้ไขเท่านั้น
- ทำให้สะดวกในการใช้ไฟล์ร่วมกันในหลายโปรแกรม

เราสามารถสร้าง Project ใหม่โดยคลิกที่ปุ่ม New File แล้วเลือกตัวเลือก Project ดังแสดงในรูปที่ 2 หรือเลือกจากแถบคำสั่ง File แล้วเลือก New ต่อด้วย Project (ดูรูปที่ 4 ประกอบ) เมื่อนั้น วินโดว์แสดงตัวเลือกประเภทของ application จะปรากฏขึ้น ดังแสดงในรูปที่ 16 โปรแกรมที่จะทำการพัฒนาในวิชานี้ จะเป็นแบบ Console Application โดยสามารถกำหนดได้โดยคลิกที่ปุ่ม Console Application (ตามรูป) แล้วคลิกปุ่ม Go หลังจากนั้น ควรจะกำหนดภาษาให้เป็น C และตั้งชื่อ Project หลังจากนั้น วินโดว์แสดง configuration ของ console application จะปรากฏขึ้น ดังแสดงในรูปที่ 17 เมื่อคลิกที่ปุ่ม Finish ก็เป็นการสิ้นสุดการสร้าง project ใหม่

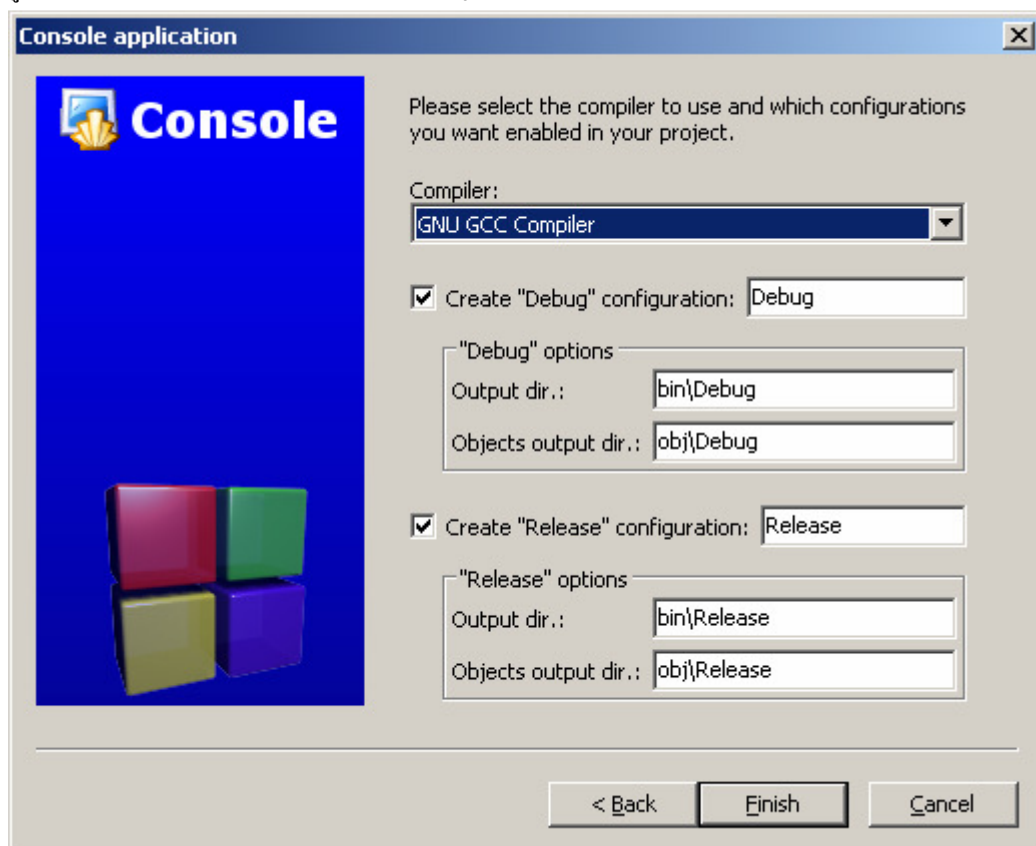
หลังจากนั้น ก็เป็นขั้นตอนการเรียบเรียง project ตัวอย่างหน้าจอหลังการสร้าง project จะเป็นดังรูปที่ 18 โดยด้านซ้ายมือจะเป็นส่วนแสดงรายละเอียดของ project อันได้แก่ sources ซึ่งแสดงไฟล์ที่ประกอบเป็น project โดยทั่วไป Code::Blocks ไม่ทำการแสดงรายละเอียดโดยจะแสดงอยู่ในรูปเครื่องหมาย + ซึ่งหมายถึงว่ามีรายละเอียดเพิ่มเติม เราสามารถสั่งให้ Code::Blocks แสดงรายละเอียดโดยคลิกที่เครื่องหมาย + ดังกล่าวนั้น นอกจากนี้ Code::Blocks จะทำการกำหนดให้ไฟล์ชื่อ main.c เป็นส่วนประกอบ (สามารถแก้ไขภายหลังได้) การแก้ไขไฟล์ใน project ก็เหมือนกับการแก้ไขไฟล์โปรแกรมทั่วไป เมื่อทำการแก้ไขแล้ว ต้องทำการบันทึกโดยใช้คำสั่ง save ถ้าต้องการปิด project ที่เปิดอยู่ ให้ใช้คำสั่ง close project ในแถบเมนู File (ดูรูปที่ 3 ประกอบ)

¹ โดยทั่วไป console หรือ system console หมายถึง อุปกรณ์ที่ใช้ป้อนหรือแสดงข้อมูลแบบ text รูปที่ 13 เป็นตัวอย่างของ console application

วิธีการใช้โปรแกรม Code::Blocks

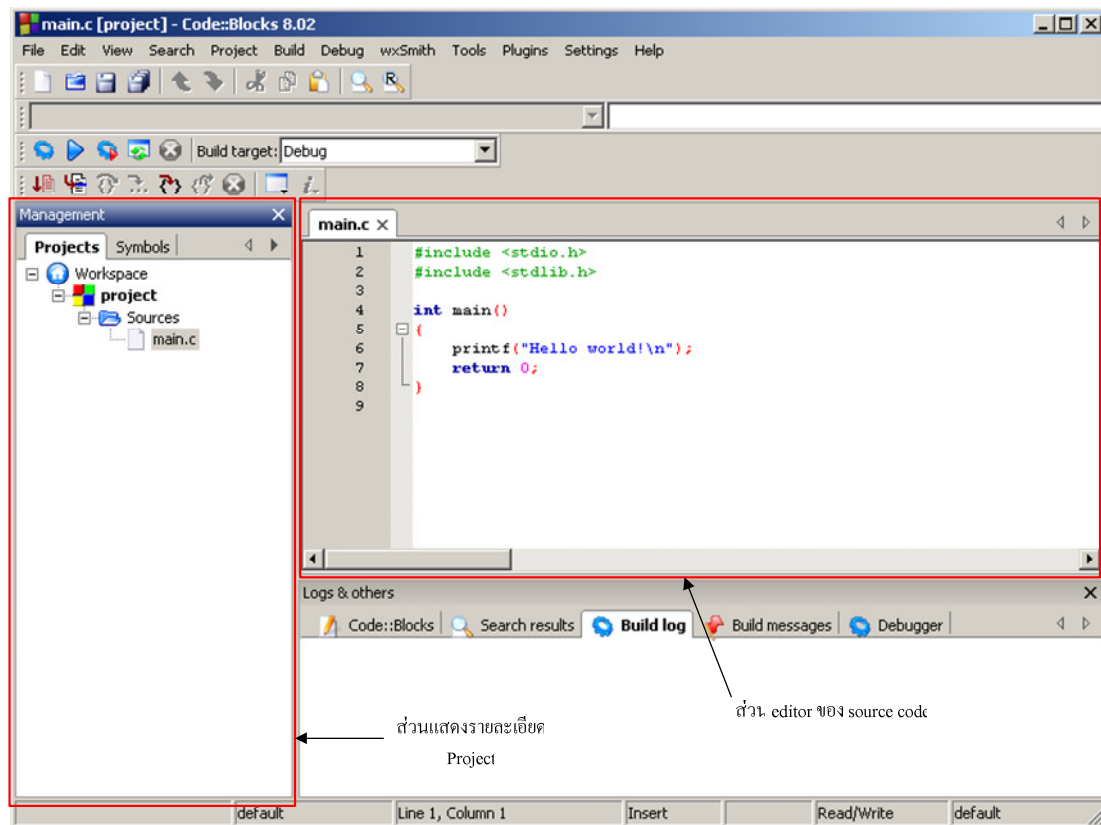


รูปที่ 16 วินโดว์แสดงตัวเลือกสำหรับสร้าง Project ใหม่



รูปที่ 17 วินโดว์แสดง configuration ของ console application

วิธีการใช้โปรแกรม Code::Blocks



รูปที่ 18 หน้าจอหลังสร้าง Project แล้ว

6 การแก้ไข project เดิม

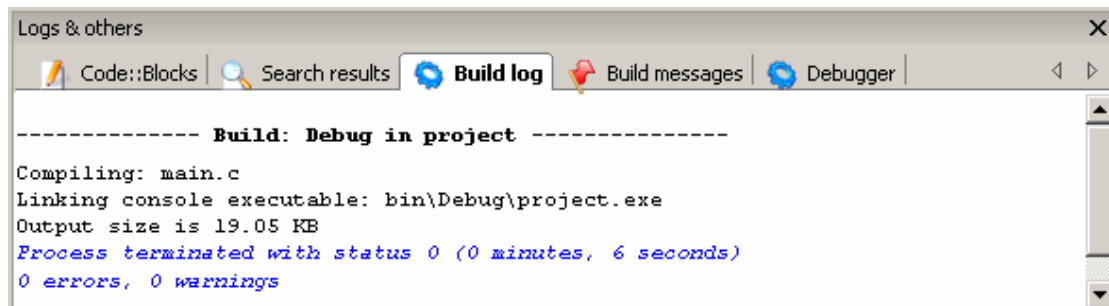
ถ้าต้องการแก้ไข project เดิมที่สร้างไว้ก่อนหน้านี้ สามารถทำได้โดยการคลิกที่ปุ่ม Open (รูปที่ 9) หรือเลือกจากแถบคำสั่ง File แล้วเลือก Open จากนั้นให้เลือกชนิดของไฟล์ให้เป็น code::blocks project files เมื่อเปิด project ได้สำเร็จ หน้าจอก็จะคล้ายกับที่แสดงในรูปที่ 18 ซึ่งเราสามารถทำการแก้ไขโปรแกรมได้ในลักษณะเดียวกับกรณีสร้าง project ใหม่ เมื่อแก้ไขเรียบร้อยแล้วให้ทำการบันทึกโดยใช้คำสั่ง Save ดังที่กล่าวในตอนที่แล้ว อนึ่ง คำสั่งที่ใช้ในการแก้ไข project ได้แก่ เพิ่มไฟล์ใน project (Add files) ลบไฟล์จาก project (Remove files) แก้ไขตัวเลือก (Build options) เป็นต้น ซึ่งได้จากการคลิกที่แถบเมนู Project

7 การคอมไพล์และรันโปรแกรมที่ได้จาก project

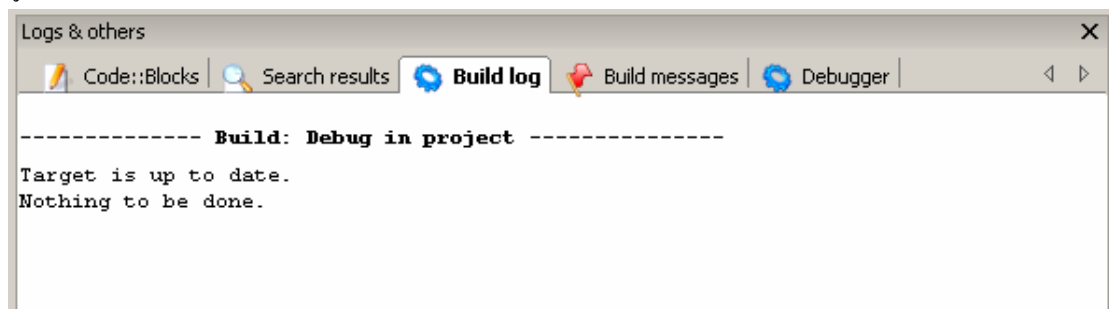
เมื่อทำการแก้ไขโปรแกรมเรียบร้อยแล้ว เราสามารถทำการ Compile โปรแกรมที่เขียนขึ้นได้โดยคลิกที่ปุ่ม Build หรือเลือกจากเมนู Build->Build หรือกดปุ่ม Ctrl-F9 ดังแสดงในรูปที่ 10 หลังจากคอมไพล์แล้ว จะมีข้อความแสดงผลการคอมไพล์ปรากฏขึ้นที่ส่วน Build log ดังรูปที่ 19 ในรูปนี้ จะแสดงให้เห็นว่า โปรแกรมที่ถูกทำการคอมไพล์ปราศจากข้อผิดพลาด กล่าวคือ จำนวน error เท่ากับ 0 และให้สังเกตด้วยว่า ไฟล์ชื่อว่า “project.exe” ซึ่งเป็นไฟล์ที่สามารถใช้งานได้ (Executable file) จะถูกสร้างขึ้น อนึ่ง ชื่อไฟล์นี้จะถูกตั้งขึ้นตามชื่อ project เราสามารถรันโปรแกรมโดยกดปุ่ม Run (รูปที่ 12) หรือเลือกจากเมนู Build->Run หรือกดปุ่ม Ctrl-F10 ผลที่ได้จากโปรแกรมในตัวอย่างจะเหมือนกับผลที่แสดงในรูปที่ 13 อนึ่ง ในกรณีที่ project ไม่มีการเปลี่ยนแปลง ข้อความที่ได้จากการคอมไพล์จะเป็นดังรูปที่ 20 เนื่องจากเมื่อโปรแกรมถูกจัดให้อยู่ในรูป project โปรแกรม compiler จะทำ

วิธีการใช้โปรแกรม Code::Blocks

การคอมไพล์เฉพาะไฟล์ที่มีการเปลี่ยนแปลงเท่านั้น แล้วทำการสร้างไฟล์ Executable ใหม่ เพื่อเป็นการเพิ่มประสิทธิภาพในการพัฒนาโปรแกรม



รูปที่ 19 Build log แสดงรายละเอียดผลการคอมไพล์ project



รูปที่ 20 Build log แสดงรายละเอียดผลการคอมไพล์ project ที่ไม่มีการแก้ไข

8 การ Debug โปรแกรม

การ Debug โปรแกรมเป็นการค้นหาและแก้ไขข้อผิดพลาด การ Debug โปรแกรมเป็นขั้นตอนสำคัญหนึ่งในกระบวนการเขียนโปรแกรม โดยเฉพาะอย่างยิ่ง ในการพัฒนาโปรแกรมที่มีจำนวนคำสั่งและจำนวนขั้นตอนการทำงานค่อนข้างมาก

โดยทั่วไป โปรแกรม Compiler (หรือ Editor) ในปัจจุบันจะมีส่วน Debugger ซึ่งเป็นโปรแกรมที่ช่วยในการ Debug โปรแกรม โดยปกติ Debugger จะช่วยให้ผู้เขียนโปรแกรมสามารถรันทีละ step หรือรันไปจนถึงจุดที่ต้องการ เพื่อช่วยให้การหาข้อผิดพลาดทำได้ง่ายขึ้น

ปุ่มที่ใช้สั่งงานในส่วน Debugger จะถูกรวมอยู่ในแถบคำสั่ง Debugger บริเวณใต้ปุ่มคำสั่ง build, run และอื่นๆ ดังแสดงในรูปที่ 21 อนึ่ง รูปที่ 22 แสดงหน้าที่ของปุ่มต่างๆบนแถบคำสั่งนี้

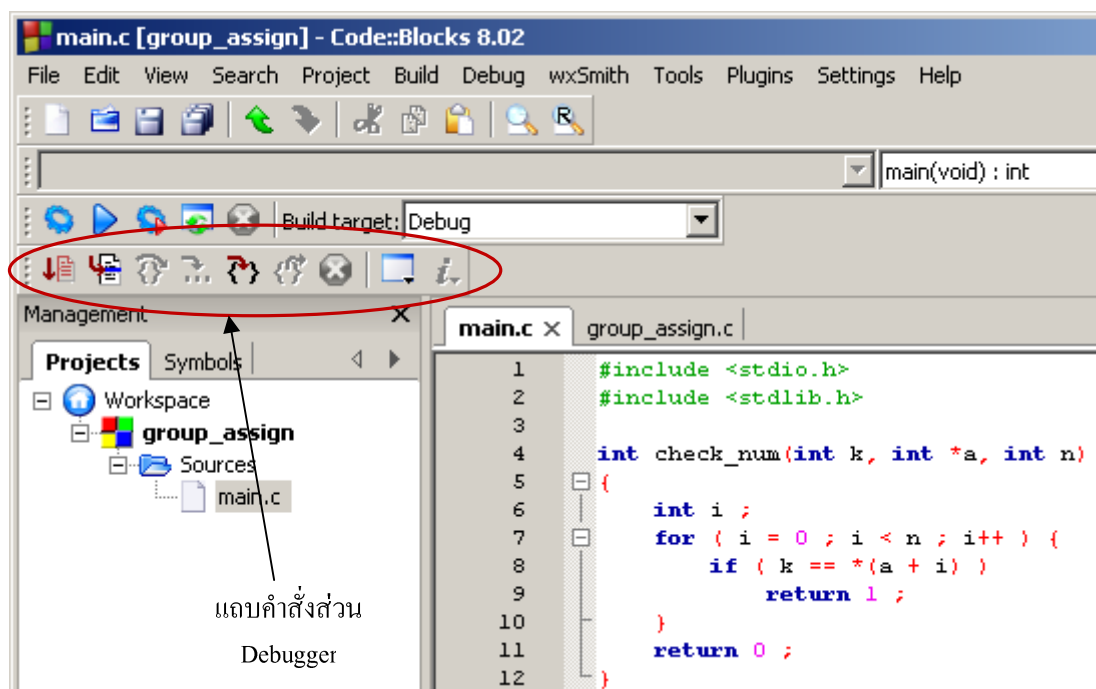
สิ่งแรกสุดที่ควรทำในการใช้งาน Debugger ในโปรแกรม code::blocks คือ ทำการกำหนด Breakpoint ซึ่งเป็นจุดที่จะให้ Debugger หยุด เราอาจจะกำหนดจุดเดียวหรือหลายจุดก็ได้ หรืออาจจะกำหนดเพิ่มเติมหลังจากเริ่มงานของ Debugger ก็ได้เช่นกัน การกำหนด Breakpoint สามารถทำได้โดยคลิกขวาที่บริเวณบรรทัดที่ต้องการ แล้วเลือก Toggle breakpoint หรือเลือกคำสั่ง Debug > Toggle Breakpoint หรือกดปุ่ม F5 บรรทัดที่เป็น Breakpoint จะถูกแสดงด้วยสัญลักษณ์วงกลมสีแดงอยู่บนบรรทัด (รูปที่ 23)

การเริ่มต้นการทำงานของ Debugger ทำได้โดยคลิกที่ปุ่ม Debug/Continue หรือเลือกเมนู Debug > Start หรือกดปุ่ม F8 Debugger จะรันถึง Breakpoint จุดแรกที่กำหนด พร้อมด้วยการเปลี่ยน tab ด้านล่างเป็น Debug tab เพื่อแสดงข้อความเกี่ยวกับกระบวนการ debug อนึ่ง ขั้นตอนปัจจุบันของโปรแกรม Debugger จะถูกแสดงด้วยรูปสามเหลี่ยมสีเหลือง (รูปที่ 24) จากนั้น เราสามารถเลือกทำงานต่อไปนี้

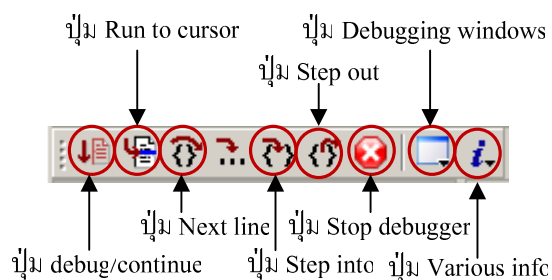
วิธีการใช้โปรแกรม Code::Blocks

1. Continue เพื่อให้โปรแกรม Debugger ทำงานจนถึง Breakpoint ต่อไป โดยการคลิกที่ปุ่ม Debug/Continue หรือเลือกเมนู Debug > Continue หรือกดปุ่ม Ctrl-F7
2. Next line เพื่อให้โปรแกรม Debugger ทำงานในคำสั่งถัดไป โดยการคลิกที่ปุ่ม Next line หรือเลือกเมนู Debug > Next line หรือกดปุ่ม F7

ในระหว่างที่ทำการใช้โปรแกรม Debugger เราสามารถเพิ่มตัวแปรที่ต้องการดูค่า โดยการเลือกเมนู Debug > Edit Watch แล้วใส่ชื่อตัวแปรลงไปในวินโดว์ที่ปรากฏขึ้น (รูปที่ 25) ซึ่งค่าตัวแปรที่กำหนดนี้จะได้รับการแสดงในวินโดว์ watches ดังตัวอย่างที่แสดงในรูปที่ 26 การทำเช่นนี้ จะเป็นสิ่งที่มีประโยชน์อย่างมากในการ Debug เนื่องจากจะสามารถตรวจสอบตำแหน่งของข้อผิดพลาดในโปรแกรมได้ ท้ายที่สุด เราสามารถหยุดการทำงานของโปรแกรมได้โดยการคลิกที่ปุ่ม Stop Debugger หรือเลือกเมนู Debug > Stop debugger

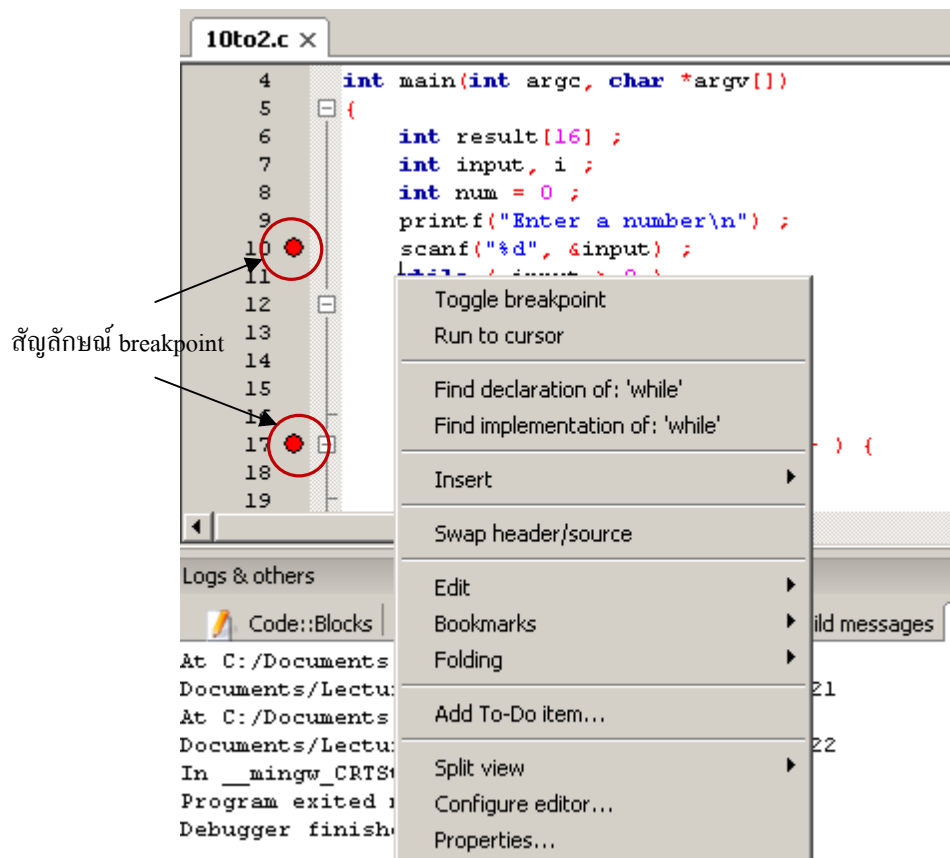


รูปที่ 21 แถบคำสั่งส่วน Debugger

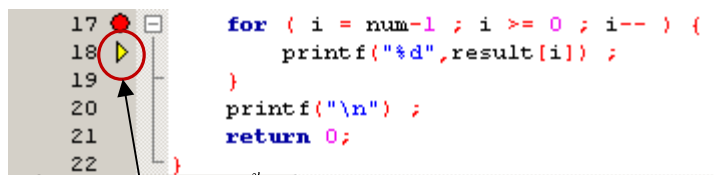


รูปที่ 22 ปุ่มต่างๆบนแถบคำสั่งส่วน Debugger

วิธีการใช้โปรแกรม Code::Blocks

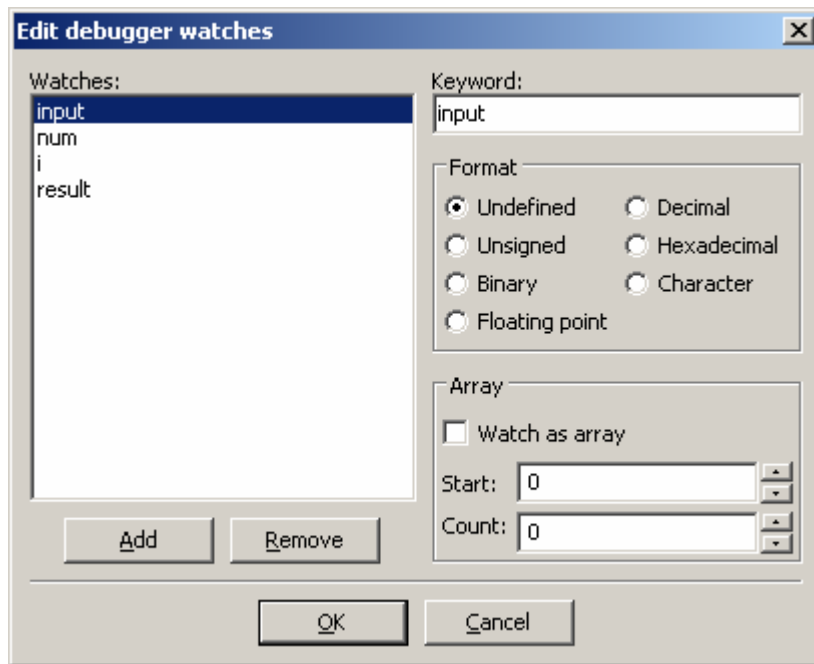


รูปที่ 23 คำสั่งกำหนด breakpoint และสัญลักษณ์ breakpoint

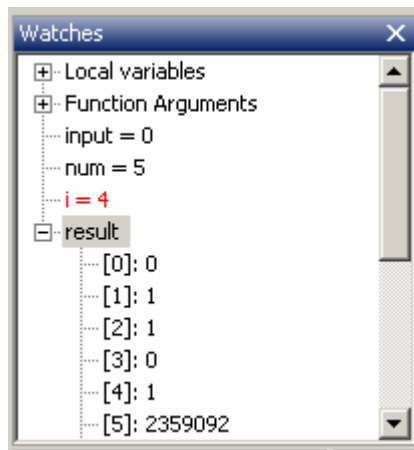


สัญลักษณ์แสดงขั้นตอนปัจจุบันของโปรแกรม Debugger

รูปที่ 24 สัญลักษณ์แสดงขั้นตอนปัจจุบันของโปรแกรม Debugger



รูปที่ 25 วินโดว์สำหรับใส่ค่าตัวแปรที่ต้องการตรวจสอบค่าในระหว่างการ debug



รูปที่ 26 วินโดว์แสดงค่าตัวแปรที่ต้องการตรวจสอบค่าในระหว่างการ debug