การใช้งานตัวอย่างโปรแกรม PDKSample

บริษัท อาร์ แอนด์ ดี คอมพิวเตอร์ ซิสเท็ม จำกัด

PDKSample_SourceCode_Usage_R221026.odt

1. ความสำคัญของไฟล์ DLL และ DLD

1.1 ไฟล์ RDNIDLibD.dll คือไฟล์ Library สำหรับใช้ในการติดต่อกับเครื่องอ่านบัตรสมาร์ตคาร์ด และบัตรประชาชน โปรแกรมตัวอย่างทั้งหมดจะมีการเรียกใช้ฟังก์ชันภายในของไฟล์ DLL นี้ เสมอ

1.2 ไฟล์ RDNIDLib.dld คือไฟล์ที่ใช้ในการตรวจสอบสิทธิ์ในการใช้งาน (License File) ฟังก์ชันใน DLL จะมีการตรวจสอบเครื่องอ่านบัตรสมาร์ตคาร์ดที่ต่ออยู่ว่ามีสิทธิ์ในการใช้งานหรือไม่โดยการตรวจสอบกับไฟล์ DLD นี้ เราสามารถวางไฟล์ DLD อยู่ที่ใดก็ได้ที่โปรแกรมสามารถเข้าถึง แต่โดยปกติแนะนำให้อยู่ที่เดียวกับไฟล์ DLL เพื่อให้ง่ายต่อการตรวจสอบ

1.3 ดังนั้นผู้พัฒนาโปรแกรมจำเป็นจะต้องนำไฟล์ DLL และ DLD ไปวางไว้ในจุดที่โปรแกรม EXE ทำงาน ทั้งที่ขณะอยู่ระหว่างการพัฒนา (ในโฟลเดอร์ Debug และ Release) และการติดตั้งลงเครื่องของผู้ใช้งาน (ผ่านการสร้างเป็น Package สำหรับการเผยแพร่)

2. ไลบรารี DLL

2.1 การใช้ไลบรารี DLL แบบ 32 บิต (RDNIDLibD.dll)

- เพื่อให้ไลบรารี RDNIDLibD.dll ที่จะใช้ในการพัฒนาโปรแกรมนี้สามารถใช้งานกับระบบปฏิบัติการ Windows รุ่นต่าง ๆ และเครื่องคอมพิวเตอร์ที่ใช้หน่วยประมวลผลกลางได้ทั้งแบบ 32 และ 64 บิต ไลบรารี DLL ตัวนี้จึงพัฒนาขึ้นมาบนฐานคำสั่งแบบ 32 บิตทั้งหมด ซึ่งข้อดีคือสามารถใช้กับ Windows ทุกรุ่นไม่ว่าจะเป็น 32 และ 64 บิตได้
- อย่างไรก็ตามในการนำไปใช้กับ C# ในชุดพัฒนา Visual Studio ผู้พัฒนาจะต้องแจ้งระบุ Platform หรือ Target CPU ให้กับคอมไพเลอร์เพื่อให้คอมไพเลอร์จะได้เลือกการแปลที่รองรับคำสั่งแบบ 32 บิต โดยกำหนด Platform หรือ Target CPU เป็น x86 หากกำหนดเป็นแบบอื่น เช่น x64 หรือ AnyCPU จะใช้งานโปรแกรมบนระบบ Windows 64 บิตไม่ได้ (สำหรับ Visual Studio จะแจ้งรหัสข้อผิดพลาดเป็น Exception from HRESULT:0x8007000B)
- วิธีการตั้ง Platform หรือ Target CPU ขอให้ดูจากรายละเอียดในหัวข้อ **การตั้ง Platform เป็น x86 สำหรับ Visual Studio** ที่อยู่ท้ายเอกสารนี้

3. การทดลองตัวอย่างโปรแกรมสำหรับ C#

3.1 ตัวอย่างโปรแกรมนี้ สร้างมาจาก Visual Studio 2019 (VS2019) จึงสามารถโหลดเข้า VS 2019 ได้โดยตรง และสามารถนำไปใช้กับ Visual Studio รุ่นอื่น ๆ ที่สูงกว่า ได้ด้วย

3.2 จะต้องมีการนำไฟล์ DLL และ DLD ไปไว้ในโฟลเดอร์เดียวกับไฟล์ EXE ที่คอมไพล์ได้ หรือที่จะนำไปใช้งานเสมอ ทั้งในระหว่างการพัฒนาและในการติดตั้งให้กับผู้ใช้งาน ขอให้ดูวิธีการในหัวข้อถัดไป

3.3 โปรแกรมนี้เป็นเพียงตัวอย่าง ผู้พัฒนายังต้องมีการเพิ่มคำสั่งในเรื่องของการป้องกันการคลิกปุ่มอ่านซ้ำซ้อนทั้งที่ยังอ่านบัตรไม่เสร็จ และเรื่องของการดักจับข้อผิดพลาดต่าง ๆ ของโปรแกรม

3.4 หากจะสร้างโซลูชันขึ้นมาใหม่หมด ต้องไม่ลืมที่จะ Add ไฟล์ DLL, DLD และการกำหนด Platform ให้เป็น x86 ด้วย

3.5 ไฟล์ EXE ที่อยู่ในชุดโปรแกรม เป็นไฟล์ที่คอมไพล์มาจากตัวอย่างโปรแกรมที่ให้ไว้ สามารถลองเรียกใช้งานได้ทันที

4. การ Add ไฟล์ .CS, DLL และ DLD เข้าไปไว้ใน Solution ของ Visual Studio

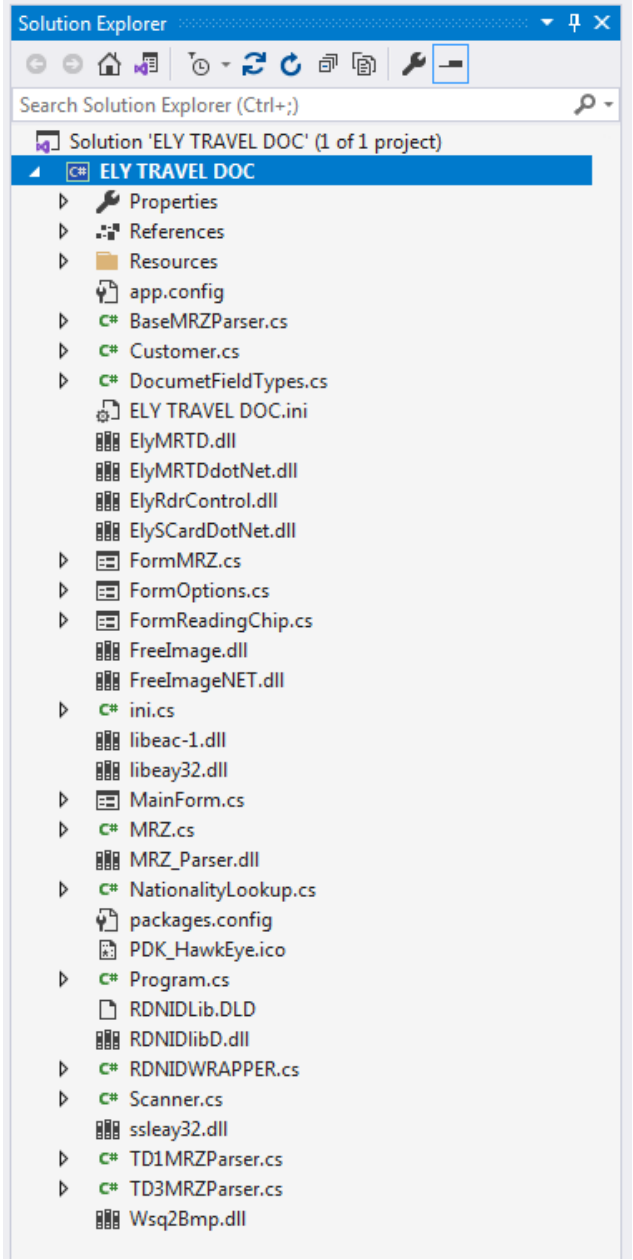
กรณีจะไม่ใช้ Project ตัวอย่างเป็นตัวตั้งต้น แต่จะสร้าง Project ขึ้นมาใหม่หมด จะต้องนำเอาไฟล์ .CS, DLL และ DLD ทั้งหมดในชุดพัฒนานี้ไปรวมไว้ใน Solution ของ Project ใหม่ตั้งแต่ขั้นตอนการพัฒนา วิธีการมีดังนี้

4.1 การ Add ไฟล์ .CS, DLL แบบ Win32 และ DLD เข้าไปไว้ใน Solution

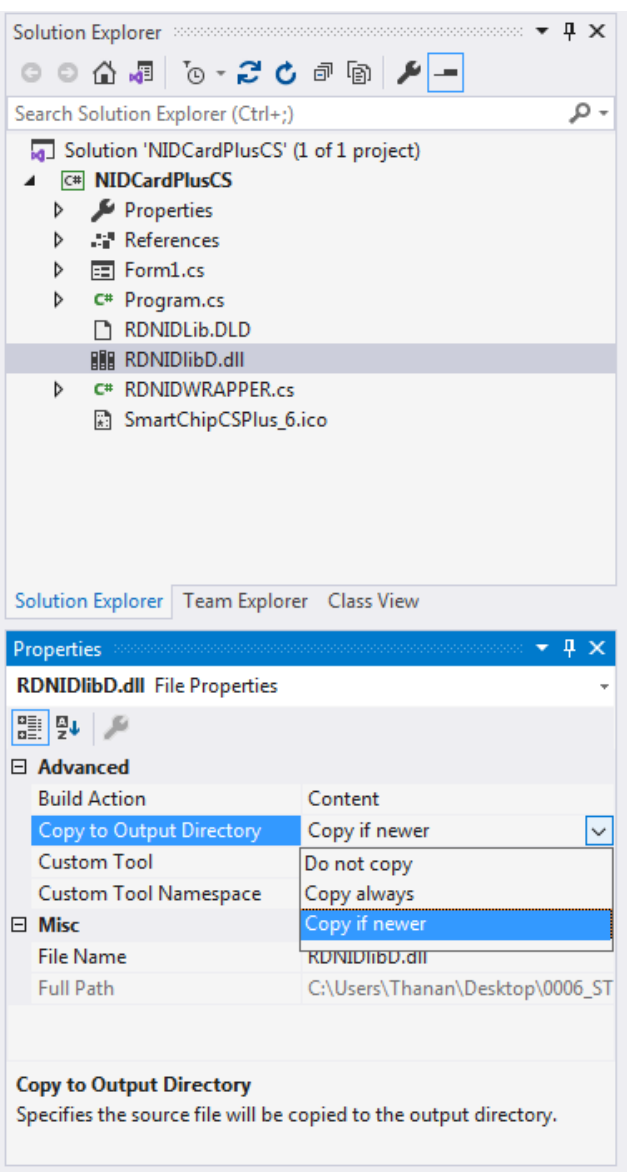
- เปิด Solution ของโปรแกรมขึ้นมา
- ที่เมนู คลิก Project > Add Existing Item... > Object of type = All Files (*.*) เลือกไฟล์ DLD ที่อยู่ใน SDK เสร็จแล้วจะเห็นไฟล์ DLD ปรากฏใน Solution Explorer (ช่องด้านบนขวามือ)
- สั่ง Add แบบเดียวกันนี้กับไฟล์ .CS และ DLL ให้ครบทุกไฟล์ (ดูให้ครบตามที่แสดงในภาพ หรือให้ดูตาม Project PDKSample ที่ให้ไว้ในตัวอย่าง Project)
- ไฟล์ DLL และ DLD ทุกไฟล์ที่ Add เข้ามาใน Solution Explorer ให้คลิกที่ชื่อไฟล์ จะมีข้อมูล Properties ของไฟล์แสดงในช่อง Properties ด้านล่าง ให้คลิกที่ Build Action แล้วเลือกเป็น Content คลิกที่ Copy to Output Directory แล้วเลือกเป็น Copy if newer ทำแบบเดียวกันนี้ทุกไฟล์

หมายเหตุ

- ให้สังเกตว่าการทำข้างต้นเป็นการ Add Existing Item ไม่ใช่การ Add Reference



ไฟล์ .CS, DLL และ DLD ที่ต้อง Add Existing Item



ตัวอย่างการตั้ง Copy to Output Directory ใน Build Action ให้เป็น Copy if newer สำหรับไฟล์ DLL ทุกไฟล์ และไฟล์ DLD

4.2 การ Add ไฟล์ DLL แบบ DotNet เข้าไปไว้ใน Reference

- ให้ Add ไฟล์ DLL ต่อไปนี้ไปไว้ใน Reference

- ElyMRTDdotNet.dll
- ElySCardDotNet.dll
- FreeImageNET.dll
- MRZ_ParserDotNet.dll
- Wsq2Bmp.dll
- Newtonsoft.Json.dll

■ ให้สั่ง Project > Add Reference... >Browse เลือกไฟล์ DLL ที่ต้องการ Add Reference

หมายเหตุ

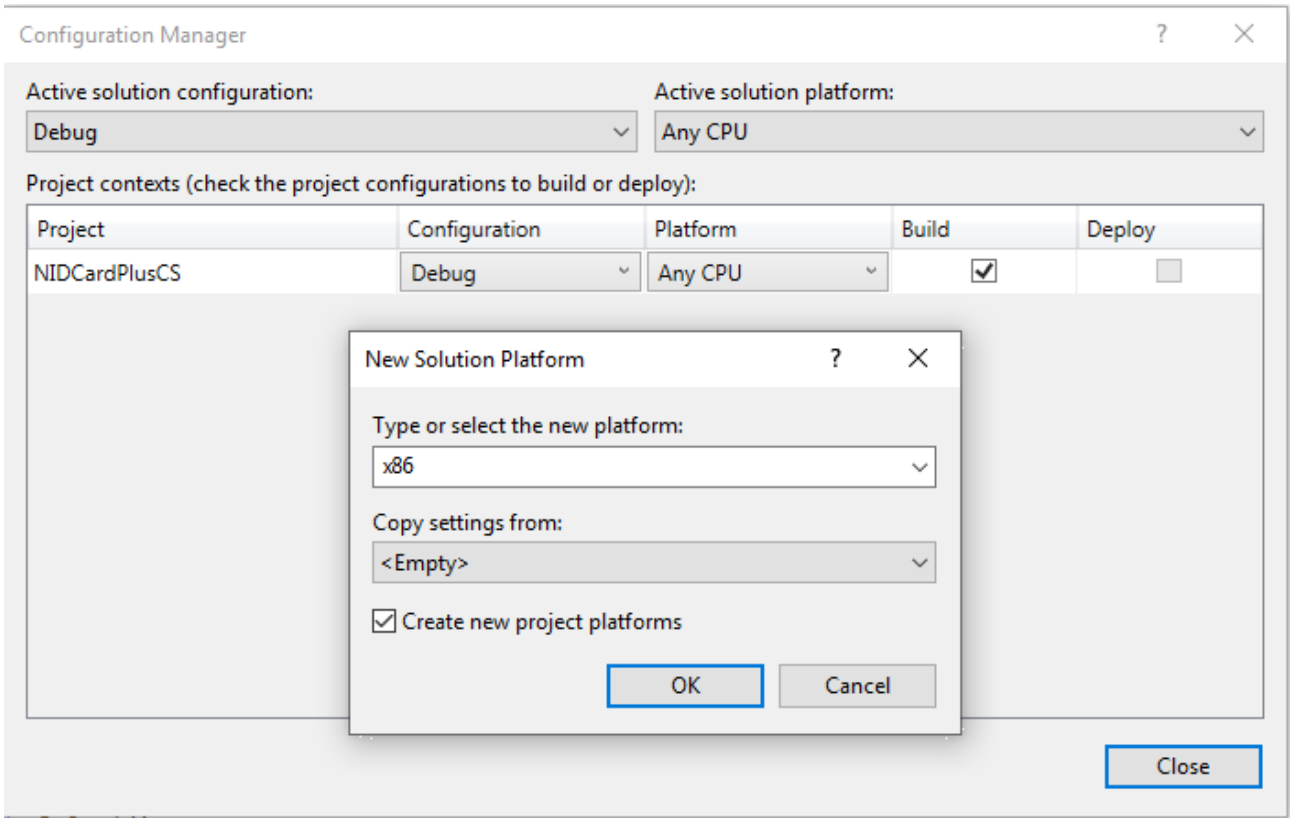
■ โปรแกรมตัวอย่าง PDKSample ที่อยู่ในชุดพัฒนา ได้มีการรวมไฟล์ .CS, DLD และ DLL ทั้งหมดไว้ใน Solution เรียบร้อยแล้ว

5. การตั้ง Platform เป็น x86 สำหรับการพัฒนาโปรแกรมแบบ 32 บิต ใน Visual Studio

5.1 เพื่อให้โปรแกรมที่พัฒนาขึ้นมาจาก Microsoft Visual Studio สามารถใช้งานได้กับทั้ง Windows 32 บิต และ 64 บิต ผู้พัฒนาจำเป็นต้องกำหนด Platform ในการคอมไพล์ให้ถูกต้อง วิธีการคือ ใช้ Configuration Manager ในการเลือก Platform โดยเลือกเป็น x86

5.2 การตั้ง Platform สำหรับ Visual Studio

- ที่แถบเมนูด้านบน คลิกเปิดเมนูของ Build จะเห็น Configuration Manager อยู่ ให้เรียกใช้งาน Configuration Manager
  - หากไม่พบเมนู Build ให้เลือก Tools>Settings>Expert Settings
  - หากไม่พบบรรทัดเมนู Configuration Manager ให้ทำดังนี้
    - คลิกที่เมนู Tools > Options
    - เฉพาะ Visual Studio บางรุ่น ให้เลือกใส่เครื่องหมายถูกในช่อง Show all settings (อยู่มุมล่างซ้ายของหน้าต่าง Options)
    - คลิกที่ Projects and Solutions และ General
    - เลือก Show advanced build configurations แล้วคลิก OK
    - กลับไปลองคลิกเปิดเมนูของ Build อีกที จะเห็น Configuration Manager อยู่บรรทัดล่าง ให้เรียกใช้งาน Configuration Manager
- เมื่อเรียกแล้ว จะมีหน้าต่าง Configuration Manager ขึ้นมา
- ที่ช่อง Active solution platform คลิกที่ปุ่มลูกศรลง แล้วเลือก x86 แล้วคลิก Close
  - หากไม่พบ x86 ให้สร้างขึ้นมาใหม่เอง โดยเลือก <New...>
  - จะมีหน้าต่าง New Solution Platform แสดงขึ้นมา ที่ช่อง Type or select the new platform ให้เลือกเป็น x86 แล้วคลิก OK



5.3 หลังจากตั้ง Platform เป็น x86 แล้ว ไฟล์ .EXE ที่คอมไพล์ได้ จะอยู่ในโฟลเดอร์ bin\x86