Лабораторна робота 1

Створення та дослідження збірок .Net. Ознайомлення з пакетом утиліт SDK .Net Framework

Мета лабораторної роботи - дослідити структуру керованого коду .Net Framework та ознайомитись з утилітами створення та підписання збірок .Net.

Теоретичні відомості

Збірки .Net Framework

Збірка (assembly) – файл або сукупність файлів, що складають логічну одиницю розгортання, виконання, захисту та керування версійністю в .Net Framework. До збірки звичайно входять керований код (managed code) та файли ресурсів. Для створення керованого коду використовуються спеціально розроблені для .Net Framework компілятори, здатні генерувати з вихідного тексту на мові високого рівня (C++ з керованим розширенням, C#, Visual Basic, J# та ін.) код на загальній проміжній мові (common intermediate language, CIL), який виконується загальномовним виконуючим середовищем (common language runtime, CLR). Використання єдиної проміжної мови дозволило досить легко поєднувати у збірку складові, написані на різних мовах високого рівня, а роботу CLR зробити не залежною від обраної мови програмування. Файли ресурсів створюються спеціальними засобами, що входять до складу інтегрованого середовища розробки програмного забезпечення Microsoft Visual Studio. Net.

Кожний модуль керованого коду складається з коду, представленого на мові CIL, та метаданих, що описують вміст модуля і зовнішні зв'язки з іншими модулями. Метадані є обов'язковим компонентом, який генерується автоматично кожним CIL–сумісним компілятором. Тобто керований код має важливу властивість самоопису, що застосовується компіляторами, збірками – користувачами та спеціальними утилітами (наприклад, утилітою IntelliSenge – засобом відображення контекстно-залежного списку доступних методів та властивостей у редакторі середовища Visual Studio).

Головний файл збірки утримує в собі додаткові метадані, що називаються декларацією (manifest). В декларації вказується ім'я збірки, список файлів, що складають збірку, опис типів даних, що імпортуються, інформація про версію збірки та виробника, права доступу та регіональні стандарти (рис. 1, 2).

Також збірки можуть посилатись на інші збірки без обмежень за кількістю посилань (рис. 3).

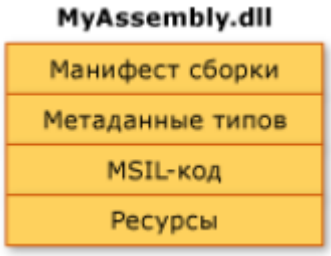

Рис.1 Структура однофайлової збірки

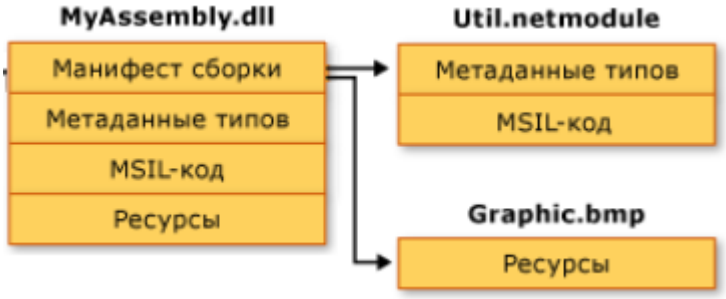

Рис.2 Структура багатофайлової збірки

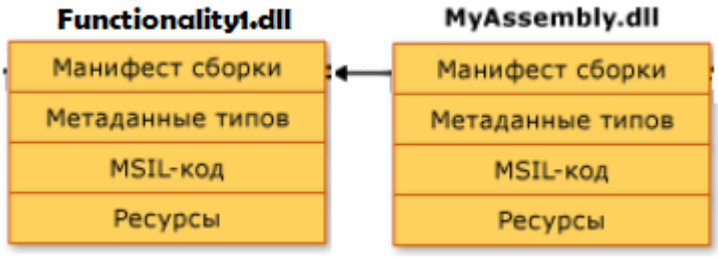

Рис.3 Збірка з посиланням на іншу збірку

У разі відсутності спеціальних вказівок компілятори генерують несуворо іменовані (weakly named) збірки. Така збірка не має криптографічного підпису, і для її ідентифікації CLR використовує тільки ім’я, що вказується в декларації. В іншому випадку створюються суворо іменовані (strongly named) збірки, які зберігають відкритий ключ розробника та цифровий підпис, що є унікальним згенерованим хэш-значенням. Цифровий підпис, що генерується за допомогою закритого ключа розробника збірки, надає збірці стійкості до підробок. Для ідентифікації суворо іменованих збірок використовується комбінація імені збірки, відкритого ключа, номеру версії та рядка регіональних стандартів.

Суворо іменовані збірки можуть бути розташованими у глобальному кеші збірок (global assembly cache, GAC), який відіграє роль глобального репозитарію збірок для загального їх використання різними додатками.

Для суворо іменованих збірок в .Net Framework передбачено механізм контролю версій зі спеціальною політикою зв’язування та перенаправлення CLR на різні версії суворо іменованих збірок.

Утиліти пакету SDK .Net Framework

Для створення та підписання збірок разом з Visual Studio .Net поставляється пакет утиліт SDK, в склад якого входять набір компіляторів, утиліти AL (для поєднання модулів у збірки), SN (для генерування файла з ключами), GACUTIL (для роботи з глобальним кешем збірок), дизамблер ILDASM (для відображення метаданих з метою отримання інформації про збірку) та ін.

Формат команди компіляції вихідних текстів.

<компілятор> [/target:<тип скомпільованого модуля>] [/out:<ім'я скомпільованого модуля>] [/addmodule:<список керованих модулів, що використовуються>] [/reference:<список динамічних бібліотек>] <список файлів вихідних текстів, що компілюються>:

- csc /target:exe class1.cs

компіляція вихідного тексту class1.cs на мові C# у консольний додаток з таким самим ім'ям;

- vbc /target:module class2.vb class3.vb

компіляція вихідних текстів на мові Visual Basic у керований модуль .netmodule з ім'ям файлу, в якому знаходиться точка входу Main;

- csc /target:library /out: newlib /reference: existlib.dll class4.cs

компіляція вихідного тексту на мові C# class4.cs у динамічну бібліотеку з іменем newlib, що посилається на існуючу бібліотеку existlib;

- csc /target:library /addmodule:module1.netmodule class5.cs

компіляція вихідного тексту на мові C# class5.cs у динамічну бібліотеку з тим же ім'ям із включенням до неї керованого модуля module1, що отримано з вихідного тексту на будь-якій мові, сумісній з платформою .Net (наприклад, C#, VB, J# та ін.).

Формат команди створення збірок утилітою AL

al [target:<тип збірки>] [/out:<ім'я збірки>] [/version:<версія збірки>] [/keyfile:<файл з ключем>] <список керованих модулів>

- al /target:library /out:lib1.dll module1.netmodule module2.netmodule

зв'язування керованих модулів module1 и module2 у динамічну бібліотеку lib1.dll;

- al /keyfile:key1.snk /target:library /out: lib2.dll /version:1.0.0.0 module3.netmodule

створення збірки lib2 версії 1.0.0.0 з суворим ім'ям, заданим у ключовому файлі key1.snk;

- al /keyfile:public.snk /delaysign /out:lib3.dll /version:1.0.0.0 module4.netmodule module5.netmodule

створення збірки lib3 з відкладеним підписуванням. При цьому використовується лише відкритий ключ без підпису, але збірка має властивості суворо підписаної.

Формат команди створення ключів для формування суворого ім'я збірки

sn <опція> [<параметри>]

- sn /D assembly1.dll assembly2. dll

перевірка збірок на ідентичність за суворим ім'ям;

- sn /k keyfile1.snk

створення пари відкритого та закритого ключів виробника та збереження їх у файлі keyfile1.snk;

- sn /p keyfile2.snk publickey.snk

отримання відкритого ключа виробника із файла з ключами keyfile2.snk у файл publickey.snk;

- sn /R lib1.dll keyfile3.snk

підписування збірки lib1.dll, створеної з відкладеним підписанням;

- sn/Vr lib2.dll

відміна верифікації збірки lib2.dll, створеної з відкладеним підписанням;

- sn/Vu lib3.dll

дозвіл верифікації збірки lib3.dll

Формат команди роботи з GAC

gacutil <опція> [параметри]

- gacutil /i lib1.dll

розташування збірки lib1.dll у GAC;

- gacutil /dll

перегляд змісту GAC;

- gacutil /l lib2.dll

перегляд всіх збірок, розташованих у GAC, з ім’ям lib2.dll;

- gacutil /u lib3.dll

видання збірки lib3.dll з GAC.

Завдання

1. Ознайомитись з утилітою командного рядка .Net.
2. Створити декілька збірок, розмістивши них простий код на мові C# (наприклад, класи Студент з можливістю обчислення розміру стипендії, переведення з курсу на курс та Викладач з можливістю визначити стаж роботи, дисципліни, що викладає, або класи Коло, Квадрат, Трикутник з можливістю обчислення периметру та об’єму фігур). Скомпілювати збірки як динамічні бібліотеки з командного рядка.
3. Створити консольний додаток, що використовує створену збірку, компілюючи його з командного рядка. Дослідити структуру збірки та додатку, використовуючи дизасемблер ILDASM.
4. Дослідити динамічне зв’язування додатку зі збіркою. Для цього потрібно змінювати місце розташування складових збірки у підкаталогах додатку та налагоджувати його конфігурацію у файлі .exe.config.
5. Створити відкритий та закритий ключі виробника та підписати збірку, вказавши її версію.
6. Дослідити механізм верифікації збірок, перекомпільовуючи збірки та змінюючи їх версії.
7. Розташувати збірку у глобальному кеші GAC.

8. Дослідити механізм перенаправлення CLR на різні версії суворо іменованих збірок, використовуючи файл конфігурації.
9. Дослідити механізм відкладеного підписання та продемонструвати його використання на прикладі створених збірок.
10. Результати роботи оформити як .doc-файл, в якому представити скріншоти утиліти командного рядка з поясненнями до них.

Контрольні запитання

1. Опишіть архітектуру .Net Framework.
2. Розкрийте поняття керованого коду.
3. Розкрийте призначення та склад збірки .Net.
4. Пояснить призначення середовища CLR.
5. Поясніть сутність та мету використання мови CIL.
6. Вкажіть основні утиліти пакету SDK та їх призначення.
7. Поясніть устрій суворого імені збірки.
8. Поясніть різницю між суворо та несуворо іменованими збірками.
9. Поясніть сутність механізму контролю версій, реалізованому в .Net Framework.
10. Поясніть призначення глобального кешу збірок.
11. Розкрийте сутність та призначення механізму відкладеного підписання.

Список літератури

1. Шилд Г. C# : учебный курс.- СПв.:Питер;К.: Издательская группа BHV, 200X.-512с.

2. Просиз Дж. Программирование для Microsoft .Net/ Пер. с англ.- М.: Издательско-торговый дом «Русская редакция», 2003.-704с.

3. Рихтер Дж. Программирование на платформе Microsoft .Net Framework/Пер. с англ.-М.: Издательско-торговый дом «Русская редакция», 2003.-512с.

4. https://msdn.microsoft.com/ru-ru/library/k3677y81(v=vs.110).aspx