Visual Studio .Net
Язык программирования C#

Федотова Наталья Петровна

Структура

- Платформа .Net
- Среда разработки Visual Studio
- Язык программирования C#

Платформа .Net

- Функциональная совместимость с существующим программным кодом.
- Интеграция языков.
- Общий исполняющий механизм для всех поддерживающих .Net языков.
- Обширная библиотека базовых классов.
- Упрощенная модель инсталляции (без записи в реестр, версии, нет «ада dll»).

Среда разработки Visual Studio

- Среда для разработки программ
- Открытость для языков программирования
  - (C#, C++, Java, VB) + другие
- Интеллектуальность
  - Подсказки к классам, функциям, параметрам
  - Список вариантов переменных
  - Вставка блоков кода
  - Выравнивание кода
- Самодокументация
- Атрибутивное программирование
- Множество шаблонов проектов

Язык программирования C#

- C# создавался параллельно с каркасом Framework .Net и в полной мере учитывает все его возможности - как FCL, так и CLR;
- C# и ООП: даже типы, встроенные в язык, представлены классами;
- возможности наследования и универсализации;
- C# наследник C/C++, сохраняя лучшие черты.
- Общий синтаксис, знакомые операторы языка облегчают переход от C++ к C#;
- Простота и надежность (допускаются, но не поощряются такие опасные свойства C++ как указатели, адресация, разыменование, адресная арифметика);
- Сборка мусора;
- благодаря каркасу Framework .Net, ставшему надстройкой над операционной системой, программисты C# получают те же преимущества работы с виртуальной машиной, что и программисты Java.
- реализация, сочетающая построение надежного и эффективного кода, является немаловажным фактором, способствующим успеху C#.

Платформа .Net

В *каркасе Framework .Net* можно выделить
два основных компонента:

- статический ***FCL (Framework Class Library)*** –

библиотеку классов каркаса;

- динамический – ***CLR (Common Language Runtime)***

общеязыковую исполнительную среду.

FCL (Framework Class Library)

- Единство каркаса (классы библиотеки используются всеми языками).
- Встроенные примитивные типы (в Visual Basic - Integer, в языке C# - int, проецируется на один и тот же тип каркаса System.Int32).
- Структурные типы (строки, массивы, перечисления, структуры).
- Архитектура приложений
  - Windows- и консольных приложения
  - Web-приложения
  - Повторно используемые компоненты
    - библиотеки классов
    - библиотеки элементов управления
- Модульность (пространства имен и сборки)

CLR (Common Language Runtime)

- Двухэтапная компиляция
- Виртуальная машина
- Дизассемблер и ассемблер
- Метаданные
- Сборщик мусора - Garbage Collector - и управление памятью
- Исключительные ситуации
- События
- Общие спецификации и совместимые модули
- CTS (Common Type System)
- CLS (Common Language Specification)

Двухэтапная компиляция

- Компиляторы языков программирования, включенные в *Visual Studio .Net*, создают модули на промежуточном языке ***MSIL (Microsoft Intermediate Language)***, называемом далее просто - IL.
- Компиляторы создают ***управляемый модуль*** - переносимый исполняемый файл (Portable Executable или PE-файл).
- Этот файл содержит код на IL и ***метаданные*** - всю необходимую информацию как для *CLR*, так и конечных пользователей, работающих с приложением.
- В зависимости от выбранного типа проекта, PE-файл может иметь расширения exe, dll, mod или mdl.
- PE-файл, имеющий расширение exe, хотя и является exe-файлом, но это не совсем обычный исполняемый Windows файл. При его запуске он распознается как специальный PE-файл и передается *CLR* для обработки.
- Исполнительная среда начинает работать с кодом, в котором специфика исходного языка программирования исчезла. Код на IL начинает выполняться под управлением *CLR* (по этой причине **код** называется ***управляемым***).
- Исполнительную среду можно рассматривать как своеобразную виртуальную IL-машину. Эта машина транслирует "на лету" требуемые для исполнения участки кода в команды реального процессора, который в действительности и выполняет код.

Виртуальная машина

- Microsoft использовала удачный опыт виртуальной машины Java
- Платформа *.Net* перестала быть частью студии, а стала надстройкой над операционной системой.
- Теперь компиляция и создание PE-модулей на IL отделены от выполнения.
- Компиляция и трансляция – м.б. разные платформы.
- В состав *CLR* входят трансляторы JIT (Just In Time Compiler), которые и выполняют трансляцию IL в командный код той машины, где установлена и функционирует исполнительная среда *CLR*.
- Оптимизация проводится на нижнем уровне, где можно учесть даже особенности процессора.
- Благодаря этому создаются высокопроизводительные приложения.
- Следует отметить, что *CLR*, работая с IL-кодом, выполняет не только достаточно эффективную оптимизацию, но и защиту кода.

Дизассемблер и ассемблер

- Если у вас есть готовый PE-файл, то иногда полезно анализировать его IL-код и связанные с ним *метаданные*.
- Для этого есть *дизассемблер* – ildasm:
  - выполняющий дизассемблирование PE-файла
  - показывающий *метаданные*,
  - IL-код с комментариями
  - в наглядной форме.
- C:\Program Files\Microsoft Visual Studio .Net\FrameworkSDK\Bin\ildasm.exe
- Профессионалы, предпочитающие работать на низком уровне, могут программировать на языке *ассемблера* IL. C:\WINDOWS\Microsoft.Net\Framework\v1.1.4322\ilasm.exe

Сборка

- - это самоописываемый, имеющий версию двоичный файл, обслуживаемый CLR.
- Сборка имеет расширение . exe или .dll.
- Расширение то же, а исполняется по-другому.
- Однофайловые и многофайловые (модули, главный модуль, редко используемые в отдельном модуле, чтобы лишнее не загружать) сборки.
- Приватные и разделяемые сборки.

Частная сборка

- Сборки для частного использования.
- Частная сборка может находиться в корневой папке и во вложенных папках корневого каталога.
- Приложение ссылается на частную сборку по ее частному имени, которое содержится в метаданных.
- Создаваемые Вами сборки.

Разделяемая сборка

- Разделяемые сборки хранятся не в каталоге приложения, а в специальном хранилище.
- Глобальный кэш сборок GAC (Global Assembly Cache).
- C:\WINDOWS\assembly.
- Утилита gacutil.exe
- Все сборки, находящиеся в GAC, подписаны строгим именем.

Сборки со строгим именем

- Обычные сборки могут быть без труда декомпилированы, и код в них может быть повторно использован. Для коммерческих приложений это недопустимо.
- Сборки, подписанные строгим именем, позволяют обеспечить безопасность, защиту кода, облегчить применение их в несколько приложениях, а также управлять версионностью сборок.
- Строгое имя является уникальным обозначением сборки.
- Оно гарантирует невозможность замены вашей сборки другой.
- Строгое имя сборки включает в себя частное имя сборки, ее версию, открытый ключ для клиентского приложения и цифровую подпись безопасности.

Самостоятельная работа 1

- Опишите статический компонент каркаса .Net
- Опишите динамический компонент каркаса .Net

- Что происходит на первом этапе двухэтапной компиляции
- Что происходит на втором этапе двухэтапной компиляции

- Какое приложение нужно запустить, чтобы написать приложение на IL ассемблере
- Какое приложение нужно запустить, чтобы просмотреть двоичный файл в удобном виде с комментариями

- Как называется самоописываемый двоичный файл, содержащий версию
- Как называется информация из двоичного файла, необходимая среде CLR для его исполнения

- Каким ограничениям должен удовлетворять язык программирования, чтобы на нем можно было писать программы под .Net?

- Разные языки программирования содержат разные примитивные типы.

  Что в таких условиях обеспечивает межъязыковое взаимодействие?