УВАЖАЕМЫЕ СТУДЕНТЫ! Изучите приведенную лекцию, законспектируйте основные сведения о технологиях JAVA и .NET

Ответы на вопросы, фотоотчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 27.02.2023.

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Лекция 39 (продолжение)

Знакомство с технологией JAVA и NET

Цель: Изучить основные сведения о технологиях JAVA и .NET

Во второй части лекции подробно описываются основные особенности платформы .NET

Новая платформа программирования

• Платформа .NET позволяет реализовать проверку типовой безопасности и проверку надежности. Следствием этого является более устойчивое функционирование приложений.

• Процесс создания приложений на платформе .NET значительно облегчился по сравнению с созданием приложений на основе интерфейса 32-разрядных Windows-приложений (Win32 API) или модели компонентных объектов Microsoft (COM).

• Платформа целиком, как и некоторые ее части, может быть реализована на многих различных типах компьютеров (аналогично Java-машине).

• Имеется единая библиотека классов, используемая всеми языками, которые поддерживает платформа .NET.

• Приложения, написанные на различных языках программирования платформы .NET, могут быть легко интегрированы друг с другом.

Платформа .NET имеет также несколько важных характерных особенностей, а именно:

• каркас .NET Framework;

• общеязыковую среду выполнения CLR (Common Language Runtime);

• возможность разработки приложения на многих языках программирования, поддерживаемых платформой .NET;

• инструментальные средства разработки приложений.

Каркас NET Framework

Современный стиль программирования предполагает многократное

использование кода, содержащегося в библиотеках. Объектно-ориентированные языки программирования облегчают создание библиотек классов. Получающиеся в результате библиотеки являются гибкими, им присущ высокий уровень абстракции. Эти библиотеки могут быть расширены путем добавления новых классов, а также путем образования новых классов на основе уже существующих. При этом новые классы наследуют функциональность существующих классов.

В каркасе .NET Framework представлено более 2500 классов, содержащих повторно используемый код. Эти классы доступны в любом языке программирования, который поддерживается платформой. Библиотека классов .NET "Framework" является расширяемой. На основе ужё существующих базовых классов можно создать новые производные классы, причем производные классы могут быть реализованы на совершенно другом языке программирования.

В состав библиотеки классов .NET Framework, входят классы, которые используются при разработке Windows-приложений, а также~ *приложений с базами* данных. В библиотеке классов .NET Framework имеются также классы, обеспечивающие взаимодействие с языком XML, с моделью компонентных объектов Microsoft (COM) и с любой платформой, поддерживающей интерфейс 32-разрядных Windows-приложений (Win32 API).

Общеязыковая среда выполнения CLR (Common Language Runtime)

Среда выполнения предоставляет необходимые службы во время выполнения приложений. Традиционно каждой среде программирования соответствует своя среда выполнения. В качестве примера среды выполнения могут служить стандартная библиотека языка C++, библиотека базовых классов Microsoft (MFC), среда выполнения языка Visual Basic, а также виртуальная машина Java (Java Virtual Machine). Среда выполнения платформы .NET получила название общеязыковой среды выполнения CLR (Common Language Runtime).

Управляемый код и данные

Общеязыковая среда выполнения CLR (Common Language Runtime) предоставляет в распоряжение .NET-кода ряд служб (включая и библиотеку классов .NET Framework, которая размещается на верхнем слое CLR). Для того чтобы воспользоваться этими службами, .NET-код должен иметь предсказуемое поведение и, к тому же, быть понятным общеязыковой среде выполнения CLR. Например, для того чтобы среда выполнения могла осуществить проверку границ массивов, все массивы в .NET имеют идентичный формат. Требования типовой безопасности могут налагать на .NET-код и другие ограничения.

Ограничения, которые накладываются на .NET-код, определяются общей системой типов (Common Type System, CTS), а также ее реализацией в промежуточном языке IL, разработанном корпорацией Microsoft (Microsoft Intermediate Language— MSIL, или просто IL). Общей системой типов определены типы и операции, которые могут использоваться кодом, работающим в общеязыковой среде выполнения CLR. Так, именно общей системой типов (Common Type System, CTS) на используемые типы накладывается ограничение единичного наследования реализации. Код на промежуточном языке, разработанном корпорацией Microsoft (Microsoft Intermediate Language, MSIL), компилируется во внутренний (собственный) код платформы.

.NET – приложения содержат в себе метаданные, т.е. описание кода и данных, используемых приложением. Благодаря использованию метаданных возможно автоматическое преобразование данных в последовательную форму общеязыковой средой выполнения CLR при их сохранении.

Код, который может использовать службы, предоставляемые общеязыковой средой выполнения CLR, называется управляемым кодом.

Память для управляемых данных распределяется и освобождается автоматически. Такое автоматическое освобождение занимаемой памяти называется *сборкой мусора (garbage collection).* Сборка мусора решает все проблемы утечки памяти и им подобные.

Microsoft и Европейская Ассоциация производителей ЭВМ

Корпорация Microsoft передала с целью стандартизации спецификацию языка C# и основные части библиотеки классов .NET Framework на рассмотрение Европейской Ассоциации производителей компьютеров (European Computer Manufacturers' Association — ECMA). Техническими требованиями этой независимой международной организации по стандартам определена независимая от платформы инфраструктура универсального языка CLI (Common Language Infrastructure). Общеязыковую среду выполнения CLR можно представить себе как инфраструктуру универсального языка CLI (Common Language Infrastructure), дополненную библиотеками базовых классов BCL (Basic Class Libraries). Библиотека базовых классов BCL (Basic Class Library) поддерживает фундаментальные типы общей системы типов CTS (Common Type System), а именно: ввод/вывод файлов, строки и форматирование. Поскольку общеязыковая среда выполнения CLR зависит от используемой платформы, в ней используются модели управления процессами и памятью базовой операционной системы.

Спецификацией (техническими требованиями) Европейской Ассоциации производителей компьютеров (European Computer Manufacturers' Association — ECMA) определен универсальный промежуточный язык CIL (Common Intermediate Language). Согласно этим требованиям, разрешено интерпретировать код на промежуточном языке CIL или компилировать его в собственный (внутренний) код.

Верифицируемый код

Управляемый код может быть проверен на предмет типовой безопасности. Код, удовлетворяющий требованиям типовой безопасности, разрушить не так легко. Например, структуры данных или другие приложения, которые находятся в памяти, не могут быть повреждены в результате перезаписи буфера. Политику безопасности можно применить к коду, удовлетворяющему требованиям типовой безопасности. Например, доступ к некоторым файлам или средствам пользовательского интерфейса может быть разрешен или запрещен. Выполнение кода, происхождение которого неизвестно, можно запретить.

Однако, не все приложения, для работы которых требуется общеязыковая среда выполнения CLR, обязаны удовлетворять требованиям типовой безопасности. В частности, такая ситуация реализуется для приложений, написанных на C++. Управляемый код, написанный на C++, может использовать возможности, предоставляемые общеязыковой средой выполнения CLR, например,

сборку мусора. Но так как на C++ может быть создан и неуправляемый код, то нет никаких гарантий относительно того, что приложение, написанное на C++, будет удовлетворять требованиям типовой безопасности. В управляемом коде, написанном на C++, нельзя выполнять арифметические операции над управляемыми указателями, или приводить тип управляемого указателя к неуправляемому. Поэтому управляемый код, написанный на C++, можно проверить на безопасность. Но может случиться так, что в этом же приложении, написанном на C++, будут выполняться арифметические операции над указателями или приведение типов управляемых указателей к неуправляемым. А это, по своей сути, ненадежно.

Разработка приложений на разных языках

Как следует из ее названия, общеязыковая среда выполнения CLR поддерживает многие языки программирования. Для каждого такого языка должен быть реализован компилятор, который генерирует "управляемый код". Сама компания Microsoft реализовала компиляторы для управляемого C++, Visual Basic.NET, Jscript, а также совершенно нового языка программирования C#.

Компиляторы для более чем дюжины других языков реализуются усилиями независимых разработчиков. К числу этих языков программирования принадлежит язык COBOL (его реализацией занимается компания Fujitsu) и язык Perl (его реализацией занимается компания ActiveState). Представьте себе, что миллиарды строк кода, написанных на языке COBOL, после некоторых усилий, связанных с переносом, станут доступными в среде .NET. Чтобы воспользоваться преимуществами среды .NET, программистам, которые пишут приложения на языке COBOL, не придется переучиваться и с начала изучать совершенно новый язык программирования.

Общая картина представлена на рис. 1.1.

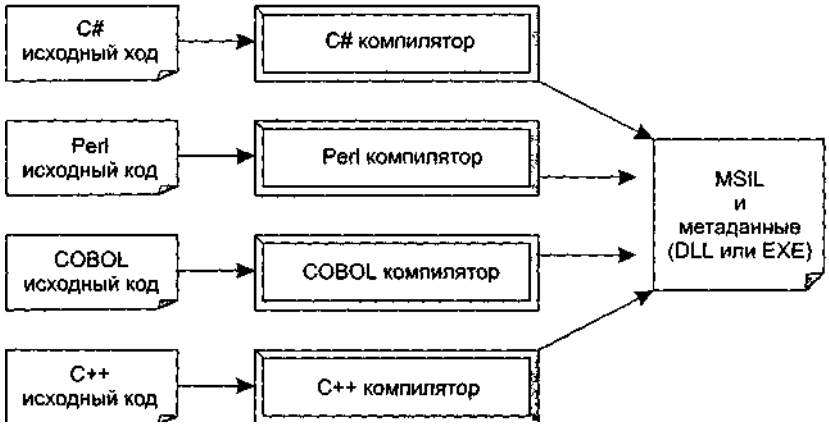

Рис. 1.1. Все компиляторы, ориентированные на .NET, генерируют IL-инструкции и метаданные

Может показаться смешным, но двоичные файлы .NET, для которых используются стандартные расширения DLL и EXE, по своему внутреннему содержанию не имеют абсолютно ничего общего с обычными исполняемыми файлами. Например, файлы DLL не предоставляют свои методы в распоряжение приложений на компьютере. В отличие от компонентов COM двоичные файлы .NET не описываются с помощью кода IDL и регистрируются в системном реестре.

Однако, пожалуй, самое важное отличие заключается в том, что двоичные файлы .NET не содержат зависящих от платформы команд. Содержимое двоичных файлов .NET — это платформенно-независимый «промежуточный язык», который официально называется Microsoft Intermediate Language (MSIL, промежуточный язык Microsoft), или простоIL.

Преимущества C#

Специально для платформы .NET Microsoft был разработан новый язык программирования C#. C# — это язык программирования, синтаксис которого очень похож на синтаксис Java (но не идентичен ему). Например, в C# (как в Java) определение класса состоит из одного файла (*.cs), в отличие от C++, где определение класса разбито на заголовок (*.h) и реализацию (*.cpp). Однако называть C# клоном Java было бы неверно. Как C#, так и Java основаны на синтаксических конструкциях C++. Если Java во многих отношениях можно назвать очищенной версией C++, то C# можно охарактеризовать как очищенную версию Java.

Синтаксические конструкции C# унаследованы не только от C++, но и от Visual Basic. Например, в C#, как и в Visual Basic, используются свойства классов. Как C++, C# позволяет производить перегрузку операторов для созданных вами типов *(Java* не поддерживает ни ту, ни другую возможность). C# — это фактически гибрид разных языков. При этом *СП* синтаксически не менее (если не более) чист, чем Java, так же прост, как Visual Basic, и обладает практически той же мощью и гибкостью, что и C++. Подводя итоги, еще раз выделим основные особенности C#.

- Указатели больше не нужны! В программах на C#, как правило, нет необходимости в работе с ними (однако если вам это потребуется, пожалуйста, — возможности для работы с указателями в вашем распоряжении).
- Управление памятью производится автоматически.
- В C# предусмотрены встроенные синтаксические конструкции для работы с перечислениями, структурами и свойствами классов.
- В C# осталась возможность перегружать операторы, унаследованные от C++. При этом значительная часть возникавших при этом сложностей ликвидирована.
- Предусмотрена полная поддержка использования программных интерфейсов. Однако в отличие от классического COM применение интерфейсов — это не единственный способ работы с типами, используя различные двоичные модули. .NET позволяет передавать объекты (как ссылки или как значения) через границы программных модулей.
- Также предусмотрена полная поддержка аспектно-ориентированных программных технологий (таких как атрибуты). Это позволяет присваивать типам характеристики (что во многом напоминает COM IDL) для описания в будущем поведения данной сущности.

Возможно, самое важное, что необходимо сказать про язык C#, — это то, что он генерирует код, предназначенный для выполнения только в среде выполнения .NET. Например, вы не сможете использовать C# для создания классического COM-сервера. Согласно терминологии Microsoft код, предназначенный для работы

в среде выполнения .NET, — это *управляемый код* (managed code). Двоичный файл, который содержит управляемый файл, называется *сборкой* (assembly). Подробнее об этом будет сказано ниже.

Инструментальные средства разработки

Настоящим ключом к успеху в разработке программного обеспечения является наличие набора эффективных инструментальных средств разработки. Компания Microsoft уже давно предлагает замечательные инструментальные средства разработки, к числу которых принадлежат Visual C++ и Visual Basic. Платформа .NET объединяет средства разработки в единую интегрированную среду, которая имеет название Visual Studio.NET.

• Среда VS.NET обладает широкими функциональными возможностями, которые могут быть использованы при создании приложения на любом языке, поддерживаемом платформой .NET.

• Платформа .NET позволяет использовать несколько языков программирования для написания приложений и имеет необходимые средства отладки.

• Среда VS.NET предоставляет множество различных конструкторов форм, баз данных и других программных элементов.

Независимые разработчики могут и в дальнейшем разрабатывать расширения среды Visual Studio.NET, а также предлагать дополнительные языки программирования и соответствующие полноценные среды разработки, поддерживаемые платформой .NET. Программы на предложенных независимыми разработчиками языках программирования смогут взаимодействовать с программами на любых языках, поддерживаемых платформой .NET. Существующий набор инструментальных средств разработки обладает широкими возможностями, которые используются при создании Web-приложений и Web-служб. Обеспечивается также всесторонняя поддержка разработки приложений с базами данных.

Важность инструментальных средств разработки

Не следует недооценивать значение инструментальных средств разработки приложений. Хорошей иллюстрацией тому может послужить случай, который произошел при работе над проектом языка Ada. Целью данного проекта было создание очень мощного языка программирования. Частью первоначального замысла было также создание стандартизованной среды программирования на языке Ada (Ada Programming Support Environment — APSE). Разработке языка программирования было уделено огромное внимание. В то же время гораздо меньше внимания было уделено надлежащей разработке среды программирования на языке Ada (APSE). Из-за этого у языка программирования Ada так и не появилась среда разработки, которая могла бы сравниться со средой разработки Visual Studio, Smalltalk, или с многочисленными интегрированными средами разработки, которые имеются для языка Java.

Преимущество среды разработки Visual Studio.NET состоит в том, что она является стандартом. Следовательно, она будет тщательно настроена для того, чтобы сделать работу в этой среде продуктивной. Компания Microsoft, по сравнению со многими более мелкими разработчиками, присутствующими на

обширном рынке инструментальных средств, располагает гораздо большими ресурсами, которые она в состоянии выделить на поддержку среды Visual Studio.NET. Платформа Java характеризуется высоко стандартизированным языком программирования и интерфейсом прикладного программирования (API). В то же время, инструментальные средства разработки, без которых написание высокопроизводительных приложений немыслимо, не являются в ней стандартизированными.

Роль языка XML

Язык XML в технологии .NET используется повсеместно. В глобальном видении развития приложений в эпоху Internet компания Microsoft также отводит ему особое место. Ниже перечислены некоторые применения языка XML в .NET.

• Язык XML используется для кодирования запросов к Web-службам и ответов, возвращаемых клиенту.

• Язык XML может использоваться для моделирования данных в наборах данных, используемых в технологии доступа к данным ADO.NET.

• Язык XML используется при создании конфигурационных файлов.

• Для некоторых языков, поддерживаемых платформой .NET, документация на языке XML может быть сгенерирована автоматически.

• Язык XML — лингва-франка (общепринятый язык) для корпоративных серверов, построенных на платформе .NET.

• Язык XML используется технологией Web-служб для описания и передачи данных.

Факторы, определяющие успех Web-служб

Перспектива Internet-приложений, как ее видит компания Microsoft, стала достоянием общественности. Окончательный успех инициативы, с которой выступила Microsoft, зависит от двух внешних факторов, которые не связаны со сферой программного обеспечения. А именно, от степени развития инфраструктуры сети Internet и успеха предложенной модели предприятия. Вопрос о том, приобретет ли технология Web-служб широкое распространение, прямо зависит от наличия сетей с высокой пропускной способностью. Такие сети уже сейчас широкодоступны. И пропускная способность их в последующие несколько лет существенно увеличится. А вот что касается перспектив предложенной модели предприятия, то они нам пока еще неизвестны!

Важно отдавать себе отчет в том, что технология .NET обладает гораздо более широкими возможностями, чем громко рекламируемые возможности Internet. Более устойчивая платформа, предназначенная для создания Windows-приложений, чрезвычайно мощная библиотека классов .NET Framework, а также инструментальные средства разработки — это именно те особенности технологии .NET, благодаря которым она выдержит испытание временем.

Резюме

Microsoft .NET — это новая платформа, построенная на верхнем слое операционной системы. Она обладает многими возможностями, которые позволяют создавать и развертывать как обычные, так и новые Web-ориентированные приложения. Web-службы позволяют использовать

функциональные возможности приложений во всей сети Internet. Как правило, для организации взаимодействия с Web-службами задействован протокол SOAP (Simple Object Access Protocol — простой протокол доступа к объектам). Поскольку в основу протокола SOAP положены широко распространенные стандарты, в частности язык разметки гипертекста HTML (Hypertext Markup Language) и язык XML (extensible Markup Language), этот протокол характеризуется высокой степенью функциональной совместимости, а значит, и высокой способностью к взаимодействию.

Платформа .NET использует управляемый код, для выполнения которого предназначена общеязыковая среда выполнения CLR. Общеязыковая среда выполнения CLR использует общую систему типов (Common Type System) Библиотека классов .NET Framework содержит огромное количество классов, которые в равной степени доступны в любом языке программирования, поддерживаемом платформой .NET. Ключевая роль в технологии .NET принадлежит языку XML Все функциональные возможности, которыми обладает платформа .NET, могут использоваться как для создания более устойчивых Windows-приложений, так и для построения Internet-приложений.