Язык программирования VB.NET

лекция 1
Платформа .NET Framework

Вид обучения-Бакалавр

2 курс, 4 семестр

Интегрированная среда разработки приложений - IDE

- Интегрированная среда (integrated development environment - IDE) - набор инструментов для разработки и отладки программ, имеющий общую интерактивную графическую оболочку, поддерживающую выполнение всех основных функций **жизненного цикла** разработки программы - набор и редактирование исходного текста (кода), компиляцию (сборку), исполнение, отладку, профилирование и др.

История интегрированных сред

- Турбо-среды (Turbo Pascal, Turbo C, Turbo C++, Delphi и др.) фирмы Borland
- GNU Emacs [2] - многоязыковая и многоплатформная интегрированная среда разработки, реализованная для MS DOS, затем для Windows, OpenVMS и для Linux.
- интегрированная среда для разработки программ на объектно-ориентированном языке Smalltalk фирмы Xeror PARC, в которой впервые появилось понятие байт-кода и понятие just-in-time компилятора.

Основные возможности IDE

- Единая интерактивная оболочка, обеспечивающая вызов всех других компонент, не выходя из среды, с широким использованием функциональных клавиш;
- Текстовый редактор для набора и редактирования исходных текстов программ;
- Система поддержки сборки (build);
- Отладчик (debugger) для отладки программ в среде с помощью типичного набора команд;
- Современные текстовые редакторы в интегрированных средах обеспечивают также режим автоматического завершения кода (code completion).

Новые компоненты IDE

- Профилировщик (profiler)
- Рефакторинг (refactoring)
- Генератор тестов (unit test generator)
- Система управления версиями исходных кодов (source code control system)
- Инструменты поддержки командной разработки программ (teamwork) - этапов жизненного цикла программы
- Инструменты анализа кода (code analysis)
- Инструменты визуализации сгенерированного бинарного кода - методов, переменных, их имен и т. д.

Продолжение

- Инструменты "запутывания" кода (obfuscation),
- Поддержка создания различных видов программных проектов (projects) и решений (solutions) на основе типовых шаблонов кода (code patterns); механизм разработки расширений (plug-ins, add-ins, add-ons).
- Поддержка моделирования структуры программ на языке моделирования UML (Unified Modeling Language).

Microsoft .NET Framework

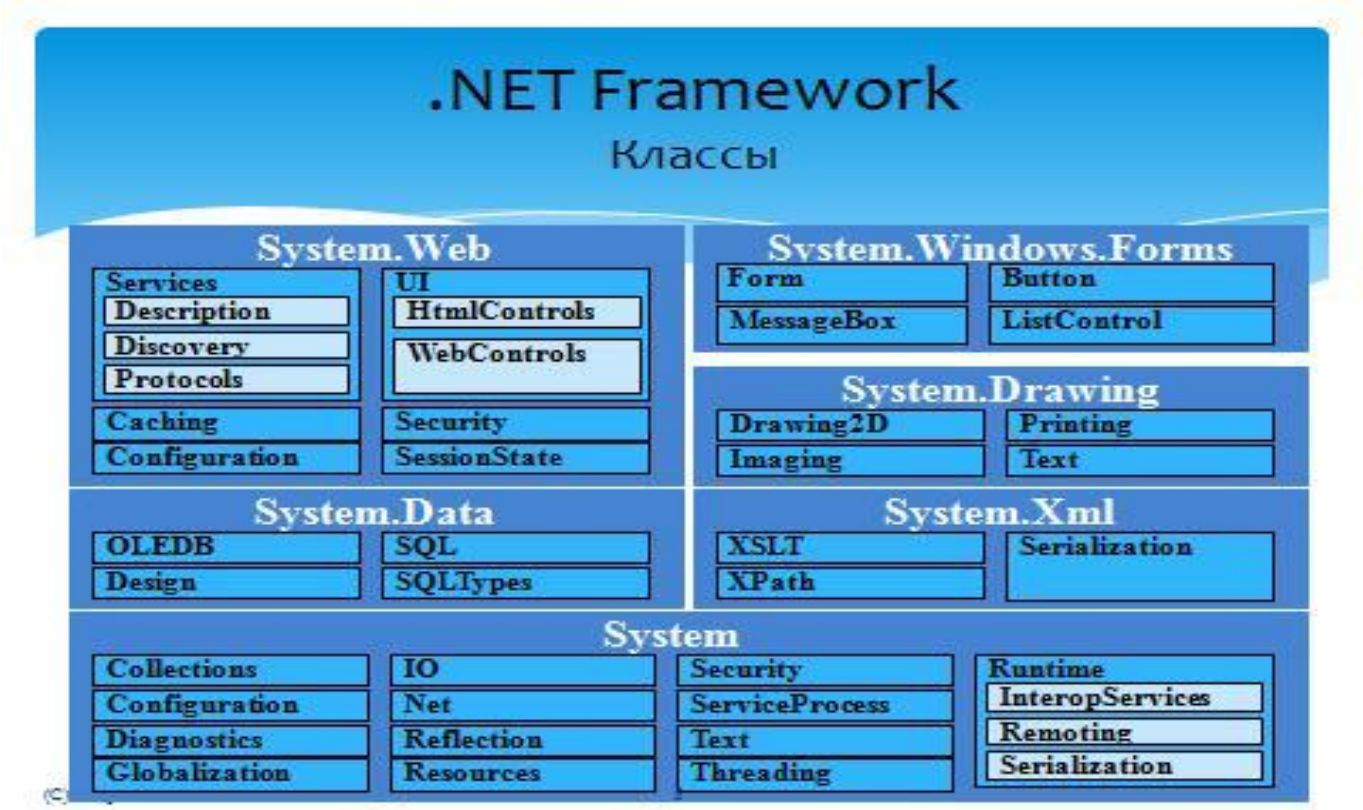
- Разработка платформы .NET началась в 1998 году. Изначально ей дали рабочее название Project 42, которое затем было изменено на COM Object Runtime (сокращенно, COR).
- Видимо, аббревиатура COR использовалась достаточно длительное время, так как ее до сих пор можно найти в названиях dll-файлов и именах библиотечных функций.
- Потом платформа сменила еще несколько названий: Lightning, COM+ 2.0, Next Generation Web Services и, в конце концов, стала называться .NET Framework.
- Спецификация основной части платформы .NET стандартизована ассоциацией ECMA (European Computer Manufacturers Association). Это означает, что корпорация Microsoft приветствует независимые реализации платформы.

Платформа .NET Framework

- .NET - платформа надежного и безопасного многоязыкового программирования.
- Она основана на единой для всех языков инфраструктуре (CLI), общей системе типов (CTS), общей системе поддержки выполнения (CLR).
- **метаинструменты** - программы, для которых другие программы выступают в роли данных.
- *Совокупность средств, с помощью которых программы пишутся, корректируются, преобразуются в машинные коды, отлаживаются и запускаются, называют средой разработки или оболочкой.*

Архитектура .NET Framework

- В основе .NET лежит единая объектно-ориентированная модель классов



Пространство имен

- В основе .NET лежит единая объектно-ориентированная модель классов, в которой все классы унаследованы от базового класса Object. Классы разбиты на пространства имен для избежания накладок при совпадении имен. Основные сервисы .NET сосредоточены в пространстве имен System (например, там находится упоминавшийся выше класс Object).
- Пространства имен имеют много уровней вложенности (System.WinForms или System.Web.UI.WebControls).

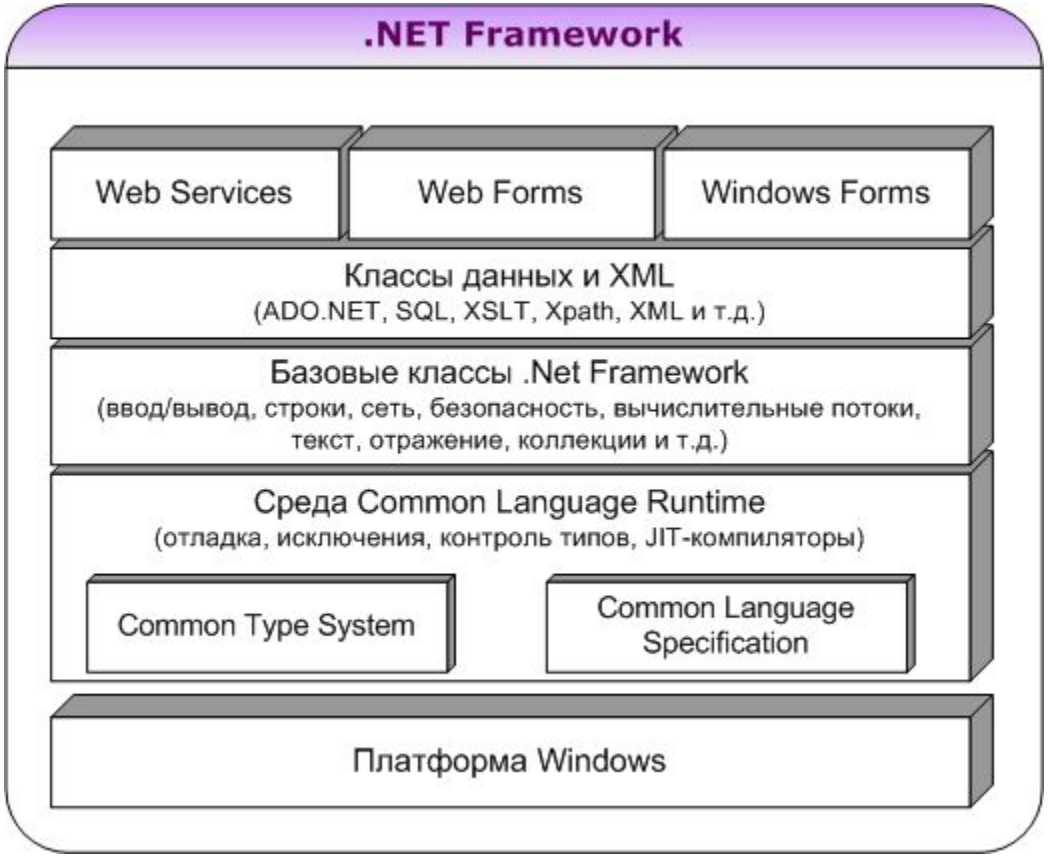
Разработка метаинструментов

- Мы будем называть метаинструментами программы, для которых другие программы выступают в роли данных.
- Метаинструменты используются для разработки, тестирования, анализа и преобразования программ.
- Это могут быть компиляторы, средства быстрой разработки приложений (RAD), оптимизаторы, отладчики, верификаторы, профайлеры и т.п.

.NET Framework включает в себя:

- Общеязыковую объектно-ориентированную среду выполнения CLR (Common Language Runtime), совместно используемую этими языками для создания приложений.
- Библиотеку классов под общим именем FCL (Framework Class Library), включающую ADO.NET, ASP.NET, Windows Forms, Windows Presentation Foundation (WPF) и Windows Workflow Foundation (WF).
- Языки программирования C# и Visual Basic, F#, Managed C++ , JScript .NET

Общая схема архитектуры .NET



Common Language Runtime

- Common Language Runtime (сокращенно CLR) можно назвать "двигателем" платформы .NET.
- Его задача - обеспечить выполнение приложений .NET, которые, как правило, закодированы на языке CIL, рассчитаны на автоматическое управление памятью и вообще требуют гораздо больше заботы, чем обычные приложения Windows.
- Поэтому CLR занимается управлением памятью, компиляцией и выполнением кода, работой с потоками управления, обеспечением безопасности и т.п.

.NET Framework Class Library

- FCL включает в себя Common Language Specification (CLS – общая языковая спецификация), которая устанавливает: основные правила языковой интеграции. Спецификация CLS определяет минимальные требования, предъявляемые к языку платформы .NET. Компиляторы, удовлетворяющие этой спецификации, создают объекты, способные взаимодействовать друг с другом. Поэтому любой язык, соответствующий требованиям CLS, может использовать все возможности библиотеки FCL.
- Библиотека разбита на несколько модулей таким образом, что имеется возможность использовать ту или иную ее часть в зависимости от требуемых результатов.
- Часть FCL посвящена описанию базисных типов. Тип — это способ представления данных; определение наиболее фундаментальных из них облегчает совместное использование языков программирования с помощью .NET Framework. Все вместе это называетсяCommon Type System (CTS — единая система типов).

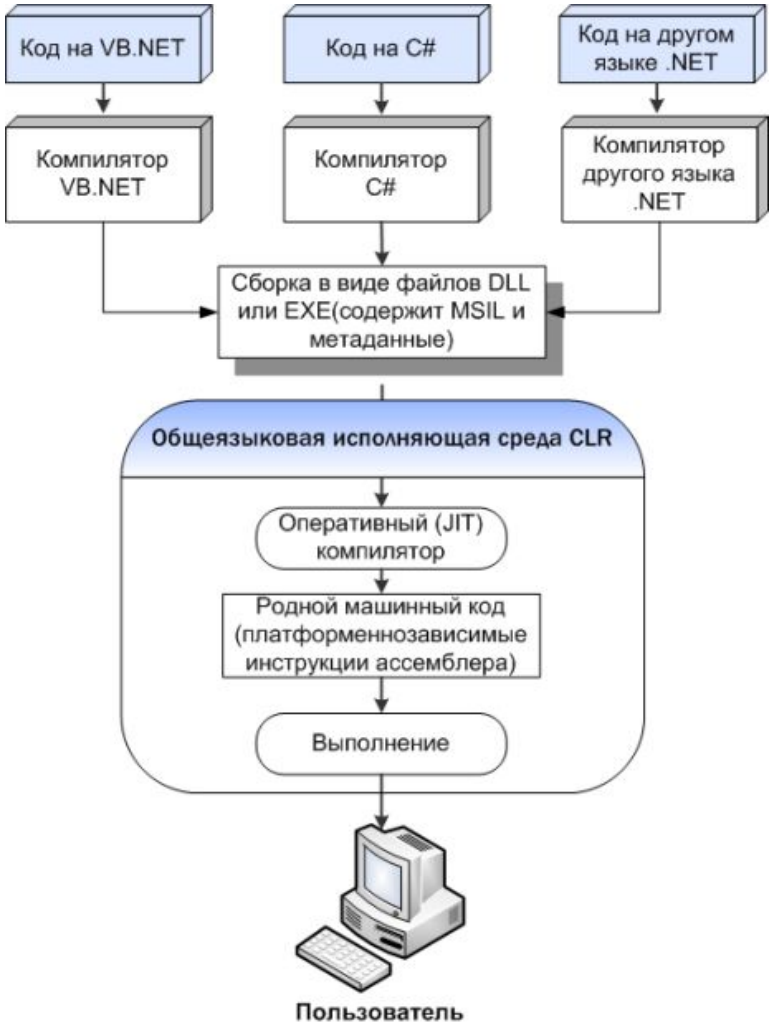
Разновидности типов в CTS

- Типы-значения - размещаются в стеке виртуальной машины .NET; (int, double, unsigned int, native int, структуры),
- Типы-ссылки – размещаются в куче. Управляемый указатель в .NET содержит ссылку на тип объекта (на метаданные), благодаря чему тип и атрибуты любого объекта можно проверить во время выполнения. Неуправляемый указатель (unmanaged pointers) - обычные адреса без явного хранения типа содержащейся в памяти по этим адресам информации.

Понятие сборки

- Сборка представляет собой набор файлов, модулей и дополнительной информации, которые должны обеспечить простую установку приложения и последующую работу. Таким образом, можно говорить и о том, что повторное использование приложений может быть реализовано с помощью интеграции различных сборок.

Схема компиляции .NET-приложения



Спецификация CLI

- Общая система типов (Common Type System, сокращенно CTS)
- Виртуальная система исполнения (Virtual Execution System, сокращенно VES)
- Система метаданных (Metadata System) - предназначена для описания типов.
- Общий промежуточный язык (Common Intermediate Language, сокращенно CIL) - независимый от платформы объектно-ориентированный байт-код
- Общая спецификация языков (Common Language Specification, сокращенно CLS) - соглашение между разработчиками языков программирования и разработчиками библиотек классов, в котором определено подмножество CTS и набор правил.

JIT-компиляция

- Ключевой особенностью выполнения программ в среде .NET является JIT-компиляция.
- Аббревиатура JIT расшифровывается как Just-In-Time, и термин JIT-компиляция можно перевести как компиляция программ "на лету". JIT-компиляция заключается в том, что CIL-код, находящийся в запускаемой сборке, тут же компилируется в машинный код, на который затем передается управление.
- Такая схема выполнения программ в среднем является более эффективной, чем интерпретация инструкций CIL, так как потеря времени на предварительную компиляцию CIL-кода с лихвой компенсируется высокой скоростью работы откомпилированного кода.
- В .NET реализованы два JIT-компилятора: один компилирует сборку непосредственно перед ее выполнением, а другой позволяет откомпилировать ее заранее и поместить в так называемый кэш откомпилированных сборок. JIT-компилятор первого типа вызывается автоматически при запуске программы, а JIT-компилятор второго типа реализован в виде служебной программы ngen, которая входит в состав .NET Framework SDK.

Сборка мусора

- Преждевременное освобождение памяти (premature free). -Это случается, если мы пытаемся использовать объект, память для которого была уже освобождена. Указатели на такие объекты называются висящими (dangling pointers), а обращение по этим указателям дает непредсказуемый результат.
- Двойное освобождение (double free). -Иногда бывает важно не перестараться и не освободить ненужный объект дважды.
- Утечки памяти (memory leaks)- Когда мы постоянно выделяем новые блоки памяти, но забываем освобождать блоки, ставшие ненужными, память в конце концов заканчивается.
- Фрагментация адресного пространства (external fragmentation).

Достоинства платформы .NET

- Безопасные типы и общее повышение безопасности приложений
- Единая модель обработки ошибок
- Межъязыковое взаимодействие (language *interoperability*)
- Единая среда разработки, позволяющая проводить межязыковую отладку
- Расширенные возможности повторного использования кода

Недостатки платформы .NET

- Существенное замедление выполнения программ;
- при создании платформы, основной упор был сделан на C++/Java-подобные языки;
- наблюдается и движение с противоположной стороны: уже сегодня стандарты некоторых языков программирования претерпевают значительные изменения для того, чтобы эти языки могли быть поддержаны в .NET.

ASP.NET-технология разработки WEB-приложений.

- AJAX – технология программирования клиентской стороны (Asynchronus JavaScript and XML), позволяет странице связываться с сервером и обновлять содержимое без полной обратной отправки.
- Компонент ASP.NET MVC (Model-View-Controller), отделяет бизнес-логику от пользовательского интерфейса.
- ASP.NET Dynamic Data- вспомогательная платформа, позволяющая быстро создавать управляемые данными приложения.
- Технология Silverlight- позволяет разным браузерам в разных ОС выполнять код .NET