Основы C#
Лекция 1.

.NET Основные понятия
Пространство имен.

.NET ОСНОВНЫЕ ПОНЯТИЯ

Платформа Microsoft.NET (.NET Framework)

Microsoft.NET (.NET Framework) – программная платформа.

Платформа .NET Framework предоставляет комплексную модель программирования для создания всех типов приложений, от мобильных и веб-приложений до классических приложений.

Последняя версия платформы .NET Framework 4.6 (Включено в версию Visual Studio 2015)

Содержит следующие основные компоненты:

- the common language runtime (CLR)
- the .NET Framework class library (.NET FCL)

CLS (Common Language Specification)

Или общая спецификация языков программирования.

Это набор конструкций и ограничений, которые являются руководством для создателей библиотек и компиляторов в среде .NET Framework. Библиотеки, построенные в соответствии с CLS, могут быть использованы из любого языка программирования, поддерживающего CLS. Языки, соответствующие CLS (к их числу относятся языки Visual C#, Visual Basic, Visual C++), могут интегрироваться друг с другом.

CLS – это основа межъязыкового взаимодействия в рамках платформы Microsoft.NET.

CLR (Common Language Runtime)

– Среда Времени Выполнения или Виртуальная Машина. Обеспечивает выполнение сборки. Основной компонент .NET Framework.

Под Виртуальной Машиной понимают абстракцию инкапсулированной (обособленной) управляемой операционной системы высокого уровня, которая обеспечивает выполнение программного кода.

Задачи CLR (Common Language Runtime)

- управление кодом (загрузку и выполнение),
- управление памятью при размещении объектов, изоляцию памяти приложений,
- проверку безопасности кода, преобразование промежуточного языка в машинный код,
- доступ к метаданным (расширенная информация о типах),
- обработка исключений, включая межъязыковые исключения,
- взаимодействие между управляемым и неуправляемым кодом (в том числе и COM- объектами),
- поддержка сервисов для разработки (профилирование, отладка и т.д.).

Компоненты CLR

- Ядро (mscoree.dll)
- Библиотеки базовых классов (mscorlib.dll)

FCL (.NET Framework Class Library)

соответствующая CLS спецификации объектно-ориентированная библиотека классов, интерфейсов и системы типов (типов-значений), которые включаются в состав платформы Microsoft .NET.

Данная библиотека обеспечивает доступ к функциональным возможностям системы и предназначена в качестве основы при разработке .NET приложений, компонент, элементов управления.

FCL (.NET Framework Class Library)

.NET FCL могут использовать ВСЕ .NET-приложения, независимо от назначения, архитектуры, используемого при разработке языка программирования.

В частности, содержит:

- встроенные (элементарные) типы, представленные в виде классов
- классы для разработки графического пользовательского интерфейса (Windows Form)
- классы для разработки Web-приложений и Web-служб на основе технологии ASP.NET (Web Forms)
- классы для разработки XML и Internet-протоколами (FTP, HTTP, SMTP, SOAP)
- классы для разработки приложений, работающих с базами данных (ADO.NET),
- и т д

MSIL (Microsoft Intermediate Language/ IL – Intermedia Language)

– промежуточный язык платформы Microsoft.NET.

Исходные тексты программ для .NET приложений пишутся на языках программирования, соответствующих спецификации CLS.

Для языков программирования, соответствующих спецификации CLS может быть построен преобразователь в MSIL.

Таким образом, программы на этих языках могут транслироваться в промежуточный код на MSIL.

Благодаря соответствию CLS, в результате трансляции программного кода, написанного на разных языках, получается совместимый IL код.

МЕТАДАННЫЕ

Создаются при преобразовании программного кода в MSIL.

Блок МЕТАДАННЫХ содержит информацию о данных, используемых в программе, это наборы таблиц, содержащих информацию о типах данных, определяемых в модуле, о типах данных, на которые ссылается данный модуль.

МЕТАДАННЫЕ: ФУНКЦИИ

- сохранения информации о типах. При компиляции не требуются заголовочные и библиотечные файлы. Всю необходимую информацию компилятор читает непосредственно из управляемых модулей,
- верификации кода в процессе выполнения модуля,
- управления динамической памятью (освобождение памяти) в процессе выполнения модуля,
- при разработке программы стандартными инструментальными средствами (Microsoft Visual Studio.NET) на основе метаданных обеспечивается динамическая подсказка (IntelliSense).

Исполняемый модуль

Управляемый исполняемый модуль (управляемый модуль) это результат трансляции .NET приложения.

Это стандартный переносимый исполняемый (PE – Portable Executable) файл Windows.

Элементы управляемого модуля

Заголовок PE	Показывает тип файла (например, DLL), содержит временную метку (время сборки файла), содержит сведения о процессорном коде.
Заголовок CLR	Содержит информацию для среды выполнения модуля (версию требуемой среды исполнения, характеристики метаданных, ресурсов и т.д.).
Метаданные	Таблицы метаданных: 1. типы, определённые в исходном коде, 2. типы, на которые имеются в коде ссылки.
IL	Код, который создаётся компилятором при компиляции исходного кода. На основе IL в среде выполнения впоследствии формируется множество команд процессора.

Управляемый модуль

Управляемый модуль содержит управляемый код.

Управляемый код - это код, который выполняется в среде CLR.

Код строится на основе объявляемых в исходном модуле структур и классов, содержащих объявления методов.

Управляемому коду должен соответствовать определенный уровень информации (метаданных) для среды выполнения. Код C#, Visual Basic, и JScript является управляемым по умолчанию.

Одной из особенностей управляемого кода является наличие механизмов, которые позволяют работать с УПРАВЛЯЕМЫМИ ДАННЫМИ.

Управляемые данные

Управляемые данные - объекты, которые в ходе выполнения кода модуля размещаются в управляемой памяти (в управляемой куче) и уничтожаются сборщиком мусора CLR.

Данные C#, Visual Basic и JScript .NET являются управляемыми по умолчанию.

Сборка (Assembly)

Управляемые модули объединяются в сборки.

Сборка является логической группировкой одного или нескольких управляемых модулей или файлов ресурсов.

Управляемые модули в составе сборок исполняются в Среде Времени Выполнения (CLR).

Сборка может быть:

- исполняемым приложением (файл с расширением .EXE)
- библиотечным модулем (файл с расширением .DLL).

JIT- компилятором (just in time – в нужный момент)

Перевод IL кода осуществляется JIT- компилятором (just in time – в нужный момент), который активизируется CLR по мере необходимости и выполняется процессором.

При этом результаты деятельности JIT-компилятора сохраняются в оперативной памяти.

Между фрагментом оттранслированного IL кода и соответствующим блоком памяти устанавливается соответствие, которое в дальнейшем позволяет CLR передавать управление командам процессора, записанным в этом блоке памяти, минуя повторное обращение к JIT-компилятору.

Выводы

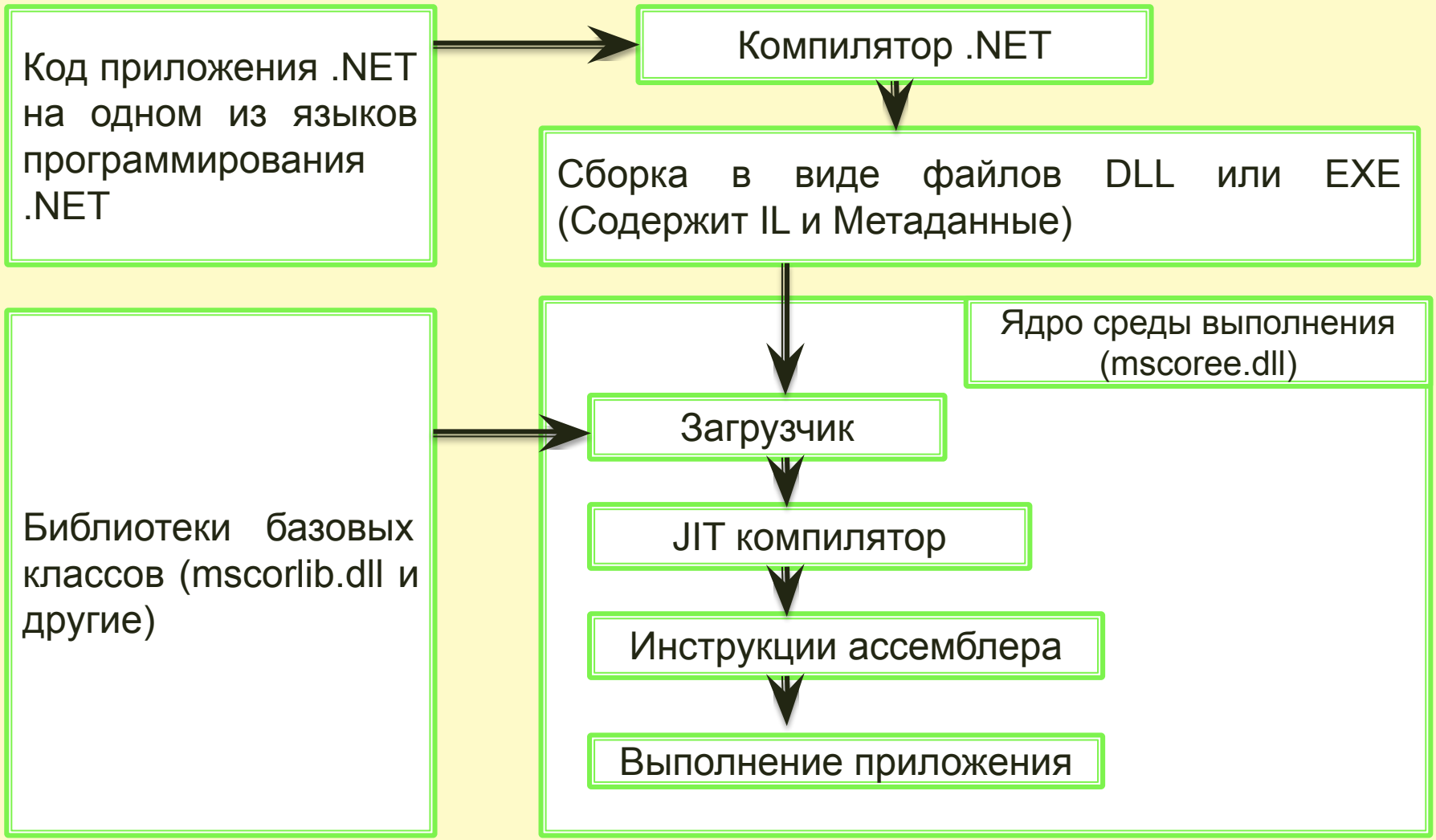
Блок метаданных CLR обеспечивает ЭФФЕКТИВНОЕ взаимодействие выполняемых .NET приложений.

Для CLR все сборки одинаковы, независимо от того на каких языках программирования они были написаны.

Фактически CLR разрушает границы языков программирования (cross-language interoperability).

Благодаря CLS и CTS .NET-приложения фактически оказываются приложениями на MSIL (IL).

Структура среды выполнения CLR



ПРОСТРАНСТВО ИМЕН.

Пространство имен

Пространство имен определяет декларативную область, которая позволяет отдельно хранить множества имен.

По существу, имена, объявленные в одном пространстве имен, не будут конфликтовать с такими же именами, объявленными в другом.

Классы группируются по пространствам имён, которые имеют (как правило) вложенную структуру.

Наиболее часто используемое пространство имён – System. Оно объединяет те классы из .NET Framework, которые наиболее часто используются в программах на C#

Пространство имен

При многократном обращаться к классам из одного и того же пространства имен, можно упростить составные имена, используя в начале программы (до определения класса) специальный оператор:

using <имя_пространства_имен>;

После такого оператора для обращения к статическому члену класса из данного пространства имен можно использовать сокращенное квалифицированное имя

имя_класса.имя_члена

Пространство имен

Нахождения модуля числа `a`

```
System.Math.Abs(a)
```

Поместив в программу оператор

```
using System;
```

Можно использовать сокращенное выражение:

```
Math.Abs(a)
```