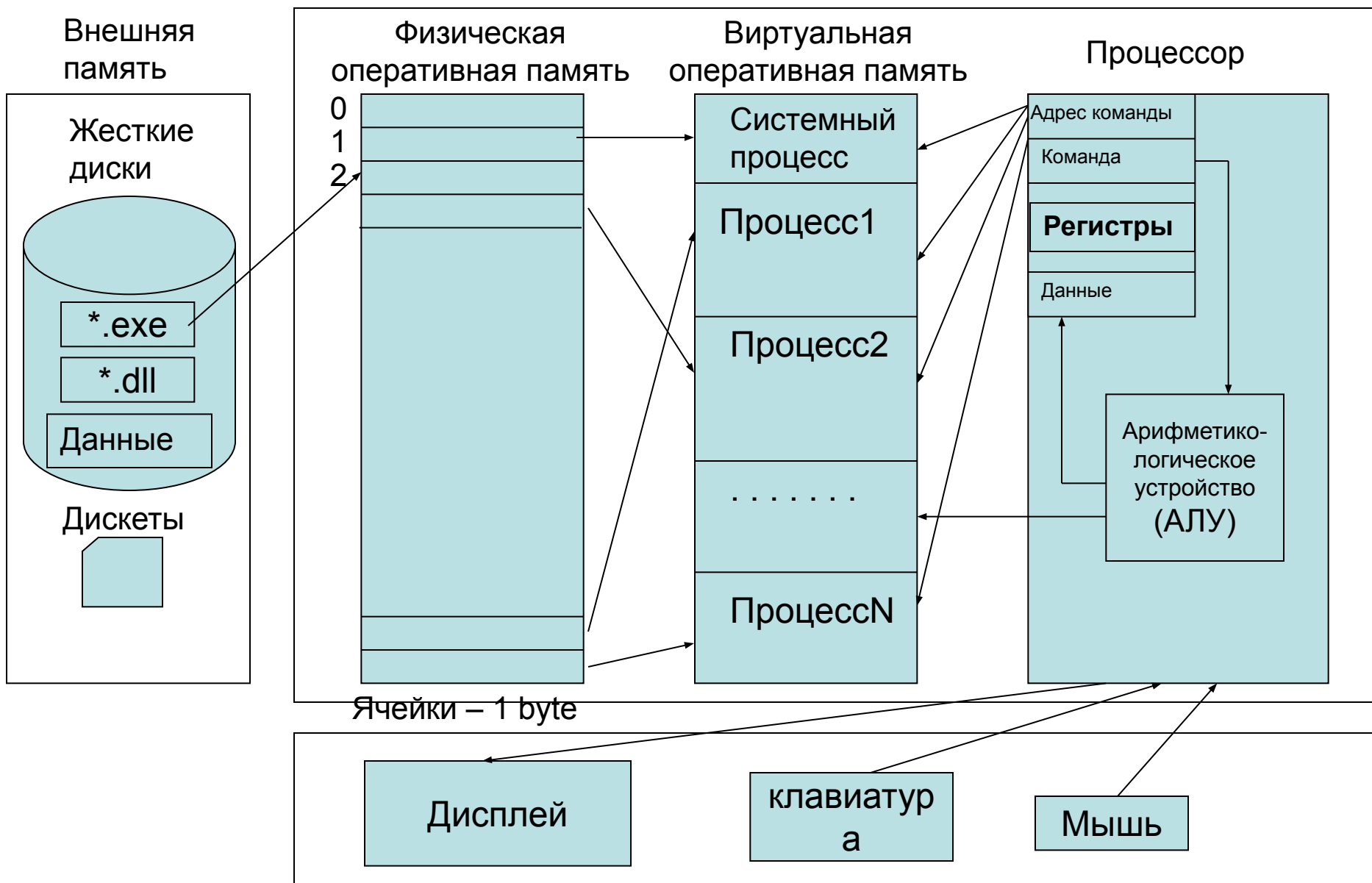


Новая платформа программирования .Net

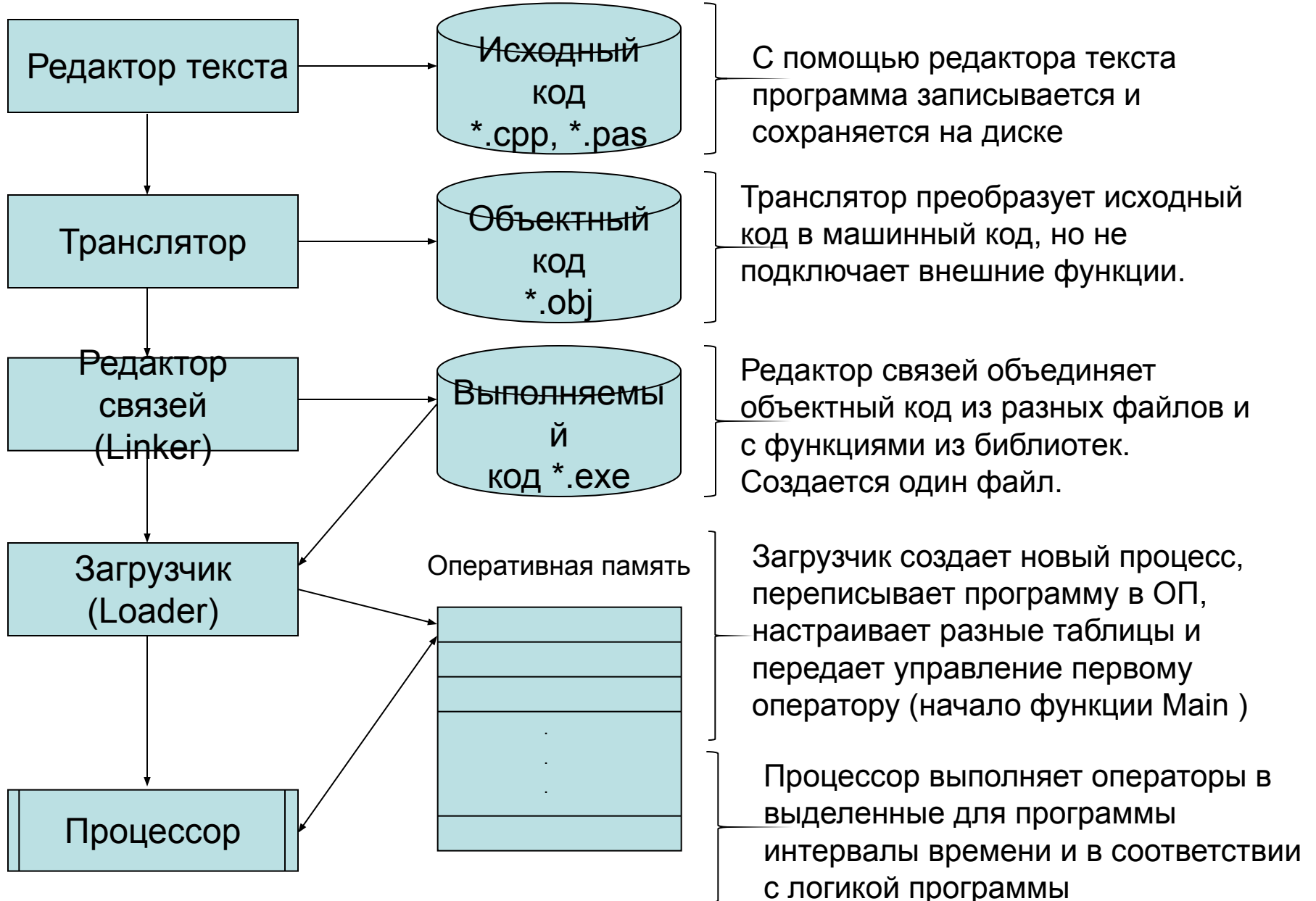
Логика работы компьютера

- Внешние устройства
 - Внешняя память
 - Жесткие диски (до 1 Тб)
 - Дискеты
 - ...
 - Экран
 - Клавиатура
- Основной блок
 - Физическая оперативная память (до 1 Гб)
 - Процессор
 - Регистры
 - Арифметико-логическое устройство
 - Виртуальная оперативная память
 - Процессы
 - Системный
 - Пользовательские

Пояснение работы компьютера и программ



Классическая последовательность создания программы



Проблемы программного обеспечения

- Взаимодействие программных модулей (компонент)
 - локальный компьютере
 - в сети (локальной и глобальной)
- Переносимость между разными платформами (портативность)
 - 32 и 64 битные
 - настольные и портативные
- Безопасность
- Эффективность

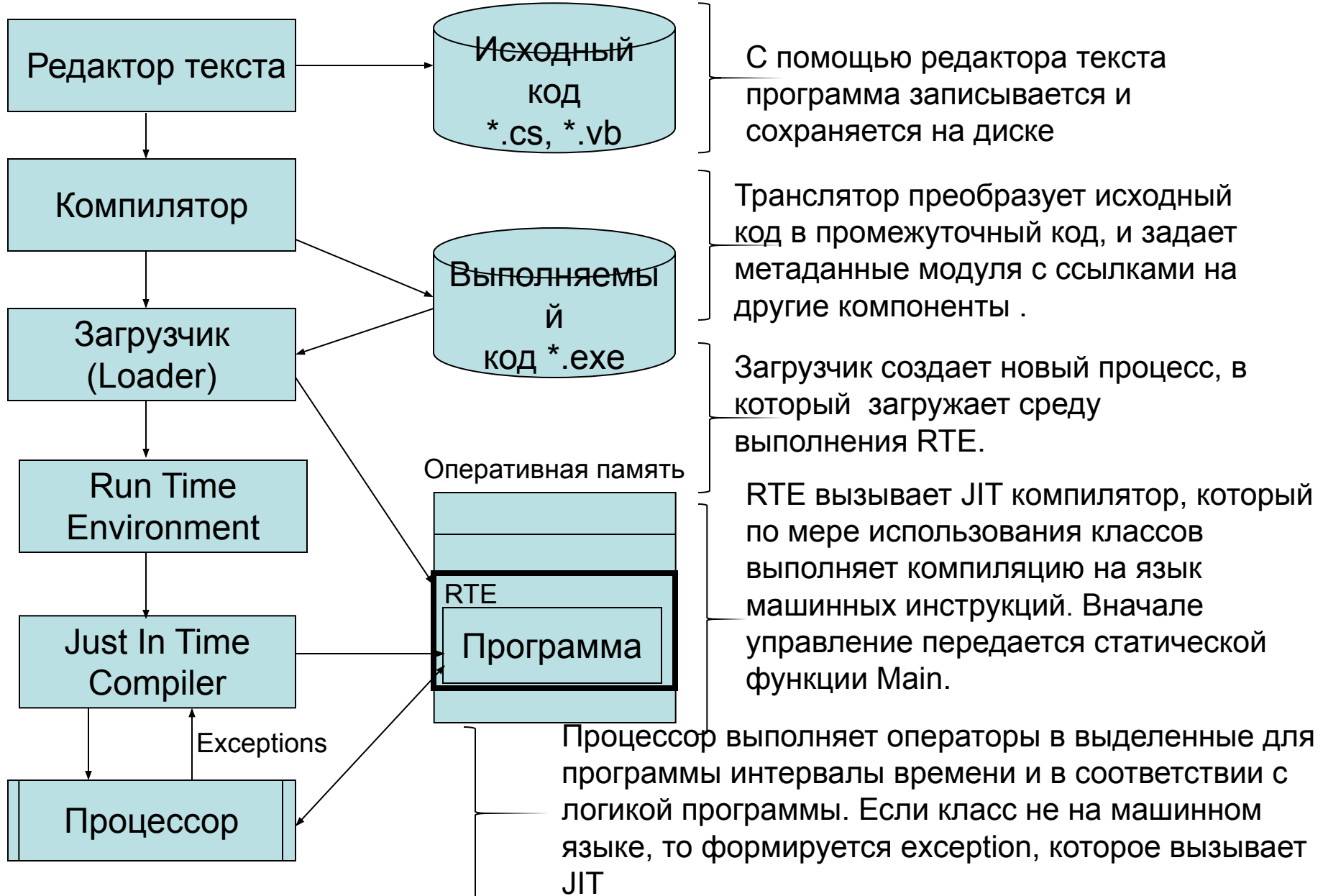
Технология .Net

- Новый комплексный подход компании Microsoft к решению проблем создания и работы программного обеспечения
- Разработка программного обеспечения для распределенных систем
- Разработка программного обеспечения для мобильных систем

Основные идеи .Net технологии

1. Общий промежуточный язык
(Common Intermediate Language - CIL)
Все компиляторы .Net создают программу на специальном языке CIL
2. Общая среда выполнения
(Common Language Runtime - CLR)
Все программы выполняются под управлением специальной программы (CLR)
3. Framework Class Library (FCL)
При выполнении программы, написанные на любом языке, используют общую библиотеку

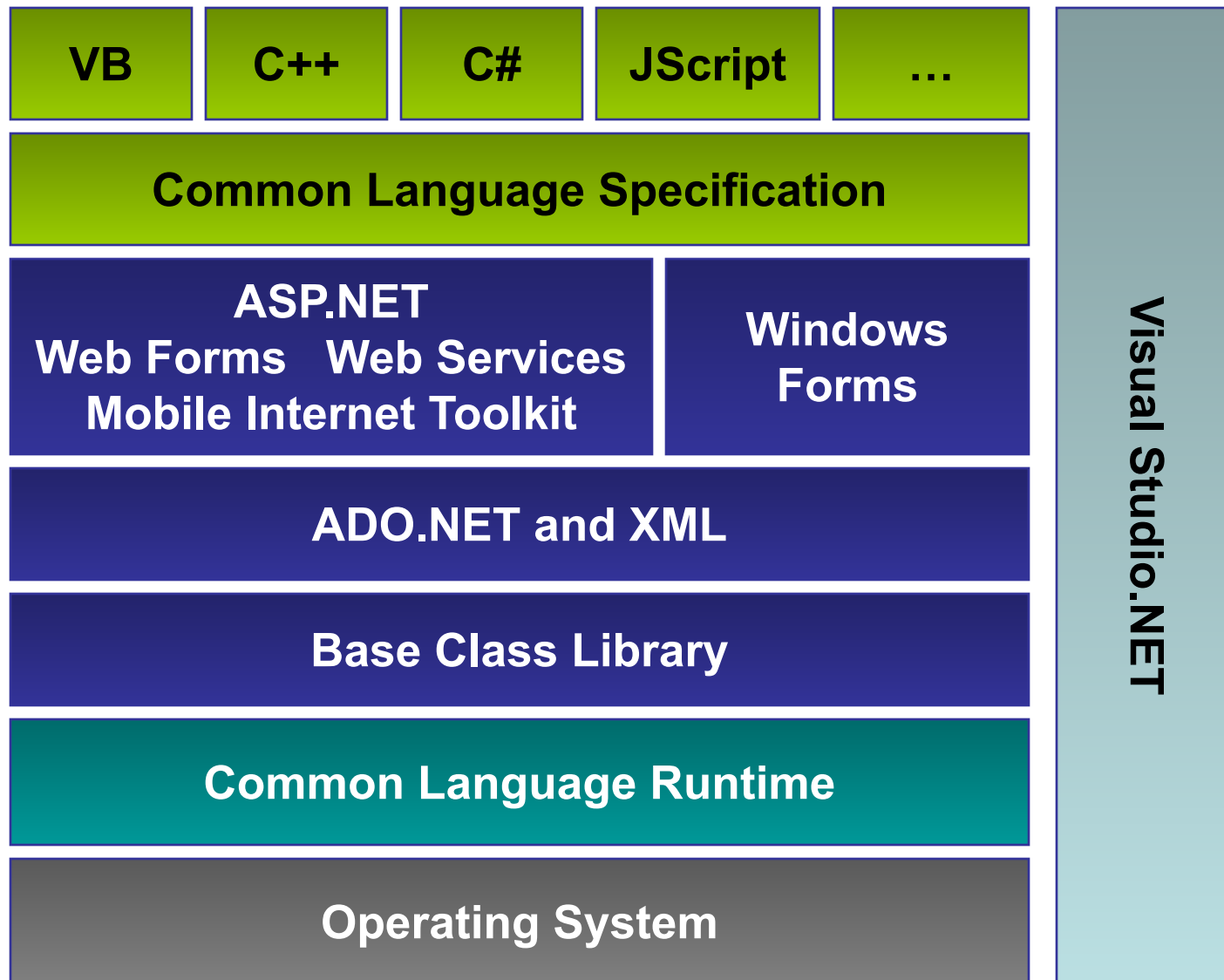
Последовательность создания и выполнения программы на платформе .Net



Состав платформы .NET (the .NET Framework)

- Общая среда выполнения (Common Language Runtime)
 - Runtime 'engine' для управляемого кода
 - Управление потоками и Памятью
 - Хорошо гранулированная, ясная защищенность (security)
 - Межъязыковое управление исключениями, диагностика, отладка
- Библиотека классов (.NET Framework Class Libraries)
 - Набор иерархически организованных библиотек классов
 - Используется всеми языками .NET
 - Встроенная общая система типов данных (common type system)
 - Объектно-ориентированная, расширяемая
- Компиляторы/инструменты
 - VB .NET, C# and C++/debugger

Платформа .NET Framework



Факты об .NET Framework

- .NET Framework SDK свободно распространяется
- .NET Framework SDK включает компиляторы для языков: C#, VB.NET и C++.
- Программирование на .NET Framework SDK **НЕ** требует наличия среды разработки Visual Studio .NET
- Имеются бесплатные версии среды разработки Visual.Studio (Express Edition)
- .NET Framework SDK включает набор инструментов, запускаемых из командной строки, такие как компиляторы, отладчики, и разные утилиты
- Rotor это открытый код реализации .NET Common Language Runtime (CLR) и C# языка

Два типа программ в ОС Windows

- Программы (exe модули) в виде набора инструкций процессора (native code)
 - выполняются процессором непосредственно
 - все ранее созданное программное обеспечение
- Программы имеющие специальную структуру на промежуточном языке - управляемый код (managed code)
 - создаются на платформе .Net
 - выполняются в среде CLR

Типы программных модулей на .Net платформе

- сборки (assembly)
 - exe (может быть запущен на выполнение)
 - dll (библиотека классов, может использоваться в других программах, которые на нее ссылаются - reference)
- Специальные модули (не включает метаданные о сборке, а только метаданные с описанием типов)
 - netmodule (может быть включен в сборку).

Создание сборки в результате компиляции в .NET



Новый тип программы – Сборка (assembly)

- Сборка (assembly) – включает 1 или более управляемых модулей (УМ)
- Управляемый модуль (managed module) – содержит 1 или более классов
- Один класс должен включать 1 статический метод (static method) Main()
- В методе Main должно быть решение задачи, или создание экземпляров класса, которые решают задачу

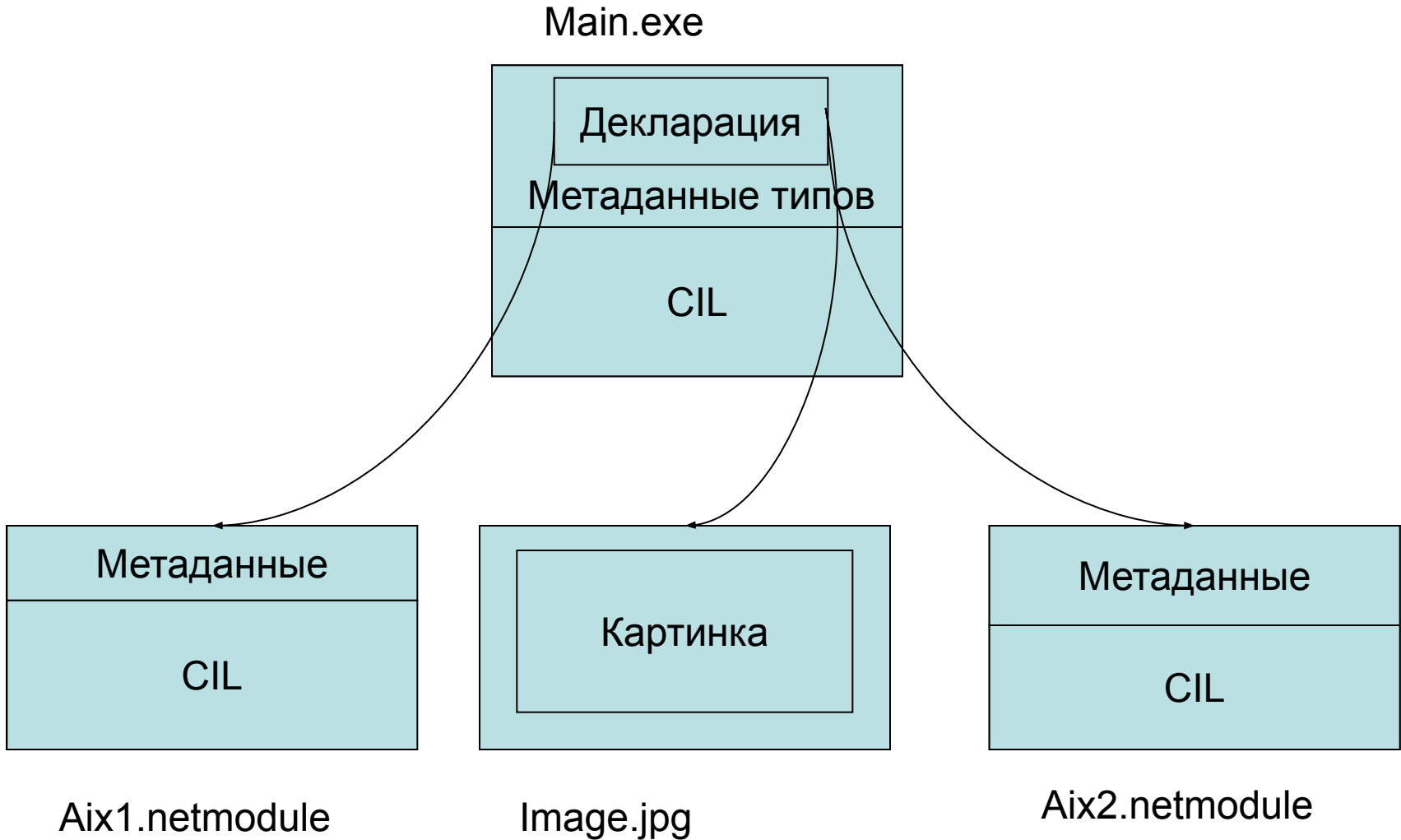
Сборка (продолжение)

- Компилятор сразу создает управляемый модуль и сборку
- Все модули сборки хранятся (один или несколько файлов) хранятся в одном каталоге
- Утилита AL.exe – для создания многофайловыхборок (может быть на разных языках)
- В сборке есть *декларация* – дополнительные метаданные, которые описывают состав сборки

Структура программных единиц

Метаданные	
Описание сборки (только у сборок)	Метаданные модуля – декларация (Manifest)
Описание доступных классов	Метаданные типов (Type Metadata)
Код программы на IL языке	IL code
Встроенные ресурсы (меню, рисунки, ...)	Resources

Многофайловая сборка



Вызов компилятора

- `csc.exe progr.cs`
`/reference:System.Drawing.dll,System.Windows.Forms.dll`
`/target:exe /out:myprg.exe`
- По умолчанию подключается модуль
– *mscorlib.dll*
- `/reference:<подключаемые библиотеки>`
- `/target:<тип результата>`
 - `exe` – консольное приложение
 - `winexe` – GUI приложение
 - `library` – библиотека классов (dll)
 - `module` – управляемый модуль
- `/out: <имя полученного файла>`

Промежуточный язык

- Microsoft Intermediate Language (MSIL) является языком ассемблера виртуальной машины. Однако реально система команд этой машины переводится в исполняемый код конкретного процессора перед исполнением (так называемая компиляция времени исполнения)
- При этом выполняется довольно сложный анализ типов программы и проверки условий корректности кода

Общий промежуточный язык (CIL)

- Псевдоассемблер – определяет набор команд виртуального процессора (примерно 100 команд)
- Использует стековую модель выполнения (сперва значения загружаются в стек, вызывается команда операции, а затем результаты сохраняются в памяти)
- При запуске программы CLR компилирует с CIL в машинные коды
- Утилита `ildasm.exe` - дизассемблер

Трансляция в MSIL

Исходный текст на C#

```
using System;

class Fib // числа Фибоначчи
{
    public static void Main (String [] args)
    {
        int a = 1, b = 1;
        for (int i = 1; i != 10; ++ i)
        {
            Console.WriteLine (a);
            int c = a + b;
            a = b; b = c;
        }
    }
}
```

Трансляция в MSIL

Сгенерированный код (начало)

```
// объявление имени assembly
.assembly fib as "fib" {
// здесь могут быть параметры assembly
}
.class public Fib
{
    .method public static void Main ()
    {
        .entrypoint // означает начало assembly

        // декларация локальных переменных:
        .locals (int32 a, int32 b)
        ldc.i4.1 // загрузка константы 1
        stloc    a // сохранение 1 в a (a = 1)
        ldc.i4.1
        stloc    b // аналогично: b = 1
        ldc.i4.1 // загрузка 1 на стек
                // (счетчик цикла)
```

Трансляция в MSIL

Сгенерированный код (окончание)

Loop:

```
ldloc    a
call     void System.Console::WriteLine(int32)
                                               // печать a
ldloc    a // stack: 1 a
ldloc    b // stack: 1 a b
add      // stack: 1 (a+b)
ldloc    b
stloc    a // a = b
stloc    b // b = (a+b)
ldc.i4.1
add      // инкремент счетчика
dup
ldc.i4.s 10
bne.un.s Loop // сравнение и переход
              // на следующую итерацию
pop      // удаление счетчика цикла со стека
ret
```

```
}
}
```


Достоинство MSIL

- Многоплатформенность
- Интеграция языков программирования
- Возможность отладки многоязыковых приложений
- Единая модель обработки ошибок

Ассемблер и дизассемблер MSIL

- Ассемблер ILAsm.exe (входит в .NET Framework)
- Дизассемблер ILDasm.exe (не входит в .NET Framework, но входит в VS.NET)

MSIL и интеллектуальная собственность

Способы защиты вашей интеллектуальной
собственности:

- Использование утилит, «искажающих» имена
- Размещение части модулей на сервере
- Реализация части алгоритмов в неуправляемых модулях

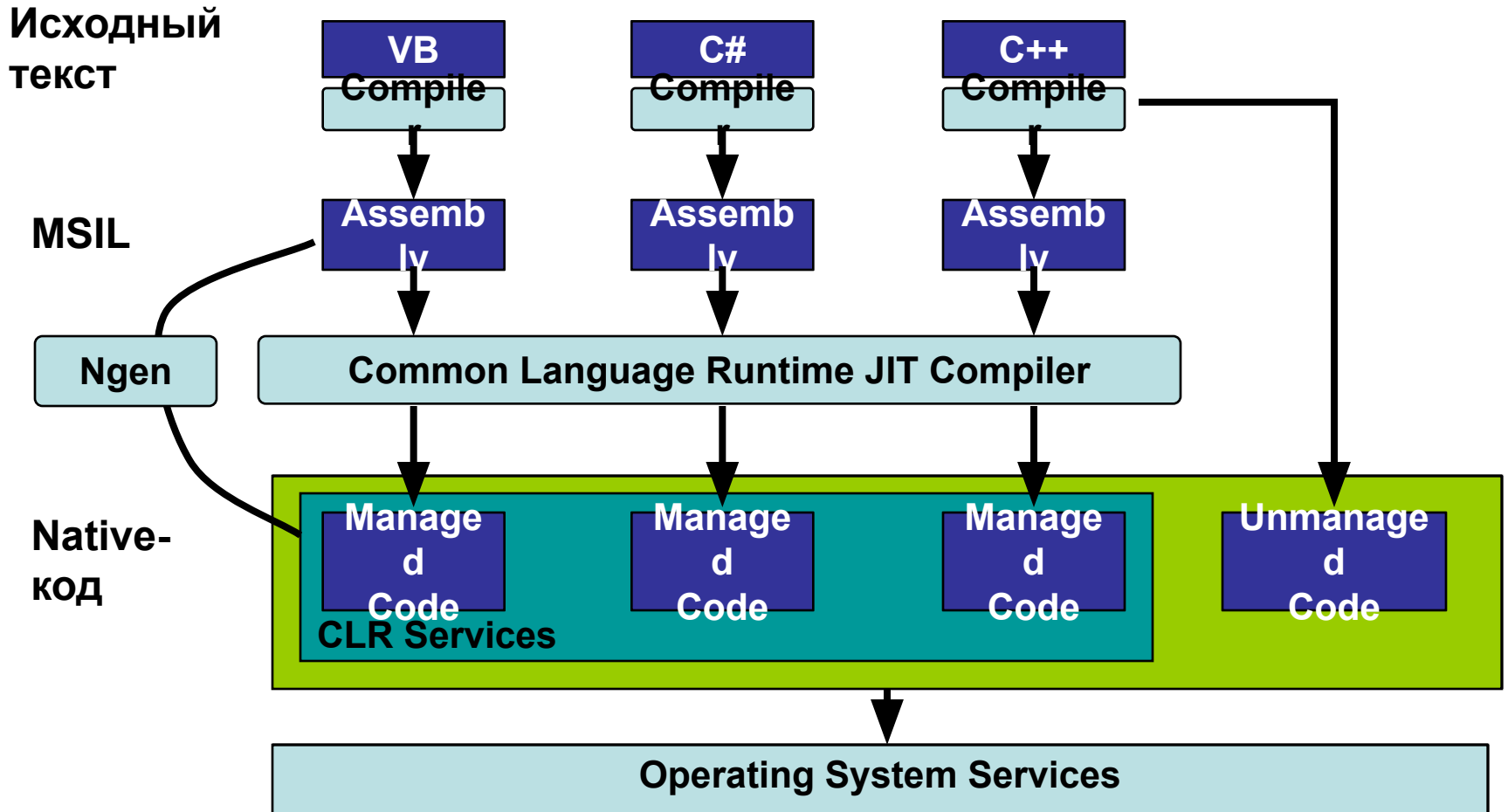
MSIL и безопасность

- При компиляции IL в команды процессора выполняется верификация (проверка кода на безопасность)
- Верификация основывается на метаданных
- При обнаружении небезопасного кода возбуждается исключение (`System.Security.VerificationException`)
- Не исполняется для небезопасного кода (например, помеченного с помощью ключевого слова `unsafe` в C#)

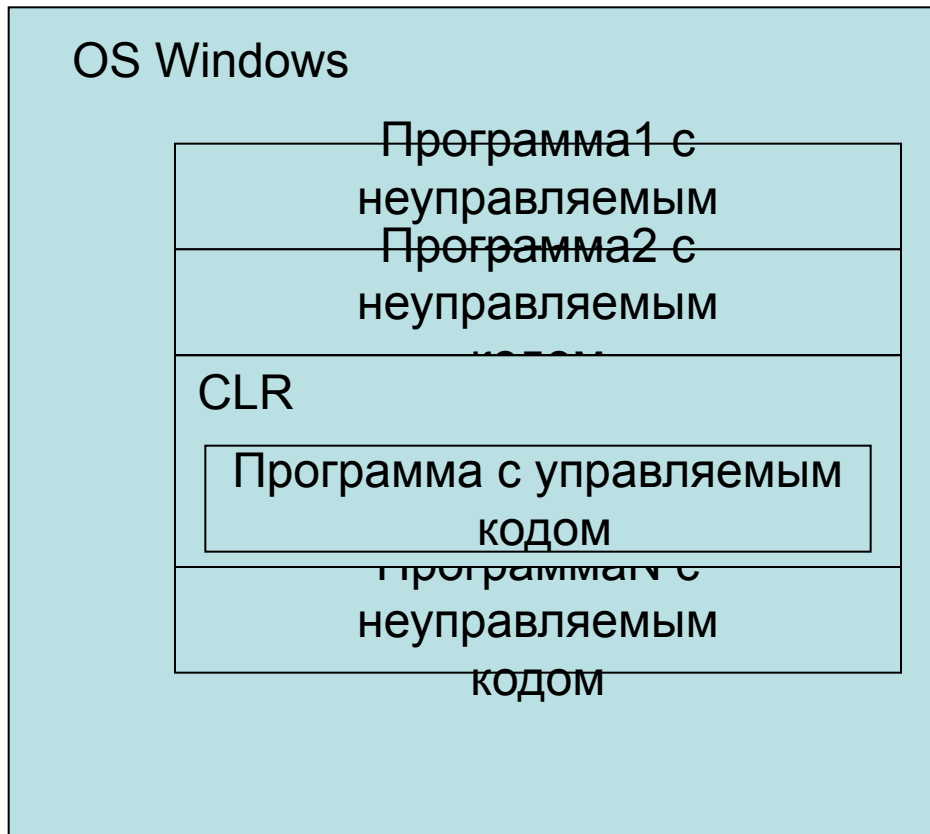
Common Language Runtime (CLR)

- При запуске программы (управляемых приложений, managed application)
- Запускается CLR
- CLR загружает программу и запускает компилятор с CIL в машинный код (x86)
- Запускает полученный машинный код
- CLR контролирует работу программы

Исполнение в .NET



Загрузка и управление программами



Служебные программы

- Компилятор с языка C# - csc.exe
- Дизассемблер с CIL (VStudio) - ildasm.exe
- Редактор связей между сборками (assembler Linker) - al.exe
- Установка (/i) и удаление (/u) общедоступных сборок - gacutil.exe

Базовая библиотека классов

- Base Class Library (BCL)
- Framework Class Library (FCL)

Библиотека классов .NET Framework (FCL)

- Более 7000 типов (классы, интерфейсы, перечисления и делегаты)
- Некоторые классы до 100 методов.
- Все языки используют одни и те же типы
- Библиотека разделена на иерархические пространства имен (около 100)
- Физически классы размещаются в DLL.
- Классы одного и того же пространства имен могут находиться в разных DLL
- 2 800 method calls for Microsoft Win32 API
- 184 000 method calls for .Net Framework Class Library

Организация FCL

- Размещены в наборе библиотек - dll
- В разных библиотеках включены разные пространства имен – namespaces
- Объекты одного пространства имен могут включаться в разные библиотеки
- В пространствах имен включены классы, структуры, ...

Пространство имен – name space

- Разделение объектов по иерархически упорядоченным группам
- Иерархическое пространство имен
<имяПространства>.<имяТипа>.<имяПодтипа>.<имяСобственное>
 - Вложенность нескольких имен (как почтовый адрес)
 - Значительно понижается вероятность совпадения имен классов разработанных разными компаниями
- Для описания связей между классами (классы близкие по функциональности включены в одно пространство)
- Пространство имен включает
 - Классы (Class)
 - Интерфейсы (Interface)
 - Перечисление (Enum)
 - Делегаты (Сигнатуры классов, Delegate)
 - Другие пространства имен
- В одном модуле могут включаться разные пространства имен

Пространства имен FCL (FCL Namespaces)

System.Web

Services

Description

Discovery

Protocols

Caching

Configuration

UI

HtmlControls

WebControls

Security

SessionState

System.Windows.Forms

Design

ComponentModel

System.Drawing

Drawing2D

Printing

Imaging

Text

System.Data

ADO

SQL

Design

SQLTypes

System.Xml

XSLT

Serialization

XPath

System

Collections

Configuration

Diagnostics

Globalization

IO

Net

Reflection

Resources

Security

ServiceProcess

Text

Threading

Runtime

InteropServices

Remoting

Serialization

Основные пространства имен FCL

- `System` – общие базовые типы
- `System.VisualBasic` – базовые типы для VBasic
- `System.Drawing` – классы для рисования
- `System.Windows.Forms` – классы для приложений с графическим интерфейсом
- `System.Data` – классы для работы с данными в БД
- `System.Web` – классы для ASP.NET и Web-форм
- `System.Net` – классы для работы с сетевыми протоколами
- `System.Web.Services` - классы для разработки Web сервисов
- `System.Web.UI` – основные классы используемые ASP.Net

Ссылки (Reference) и операторы using

- **Reference** – ссылка на библиотеки, которые должны загружаться во время работы программы
- Библиотеки – наборы классов
- Хранятся в файлах Dynamic Link Libraries (DLL)
- Оператор **using** – используется для сокращения записи наименований классов

Просмотр библиотеки классов (Object browser)

- View/Object Browser
- Просмотр пространств имен – классов
- Просмотр всех элементов классов с описанием назначения и состава параметров