

Платформа Microsoft .NET

*Кафедра системного программирования
С.-Петербургского государственного
университета
Сорокин Антон,
e-mail: anton.v.sorokin@gmail.com*

Критерий оценки

- Оценка за полугодие – часть годовой
- Посещаемость лекций ($\geq 50\%$, min 7)
- Зачёт по спецсеминару (у кого есть семинар)
- Устный экзамен



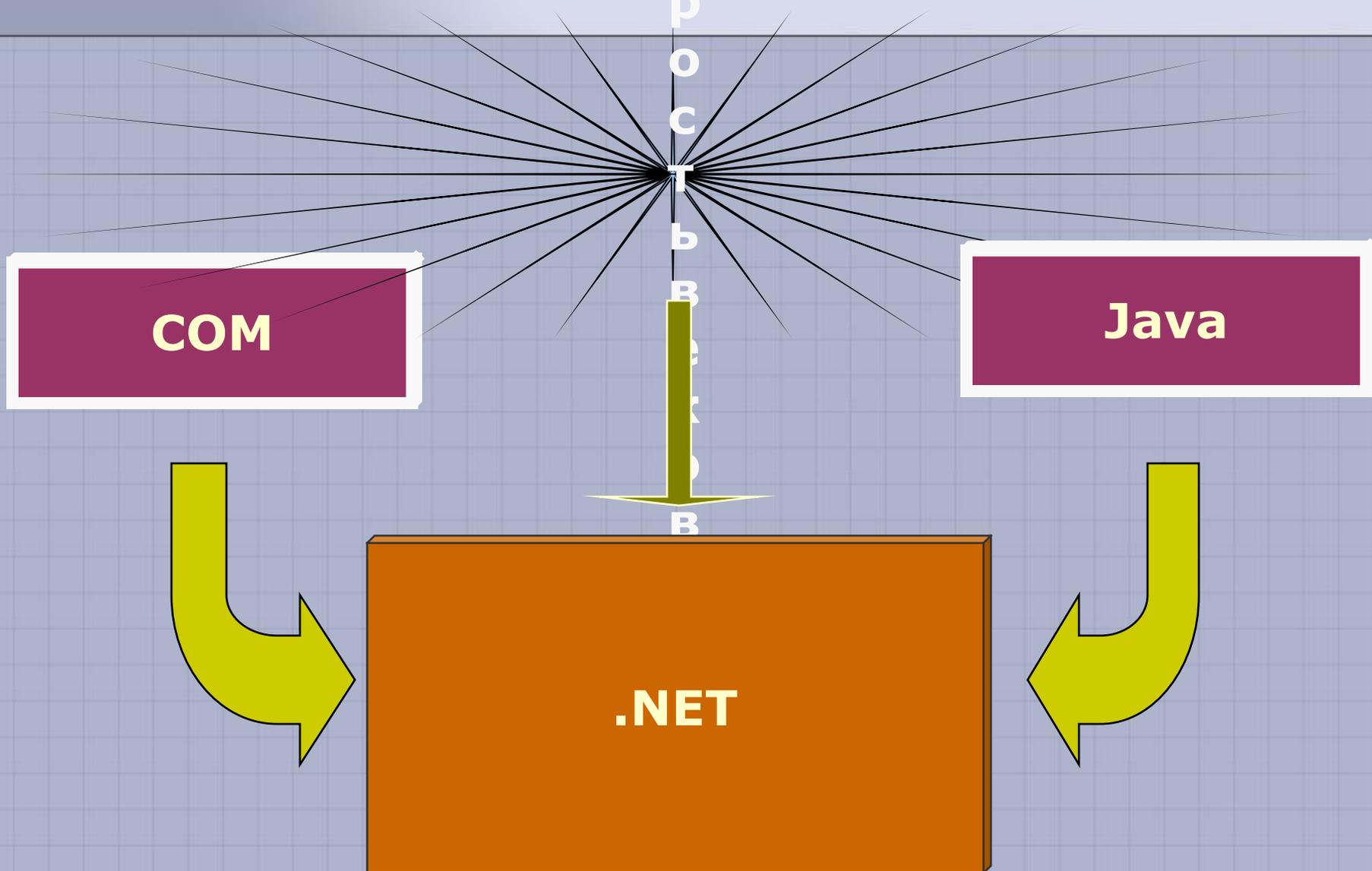
Темы курса

- Обзор платформы Microsoft.NET
- **Обзор .NET Framework**
- Обзор C#
- Windows Forms
- ADO.NET
- ASP.NET
- Веб-сервисы и SOA
- Перспективные технологии Microsoft.NET
- Бонусные лекции (если успеем)

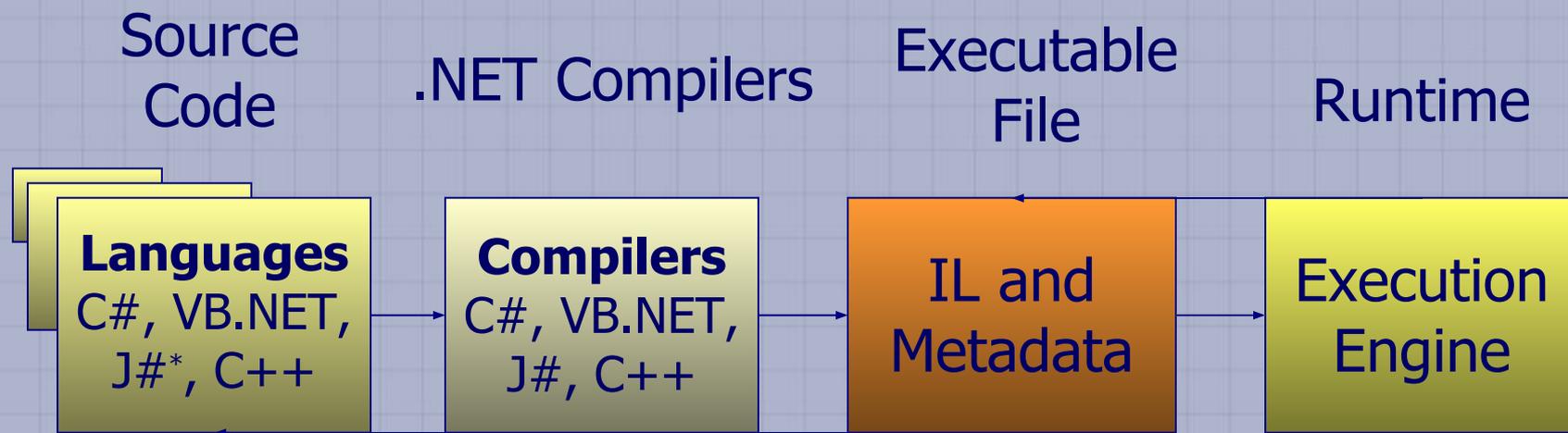
Лекция 1: Обзор .NET Framework

- **Архитектура .NET**
 - Common Language Runtime
 - MSIL
 - Framework Class Library

Откуда есть пошел .NET



Общая идея архитектуры .NET



Концептуально .NET представляет собой единение основных идей, лежащих в основе Java и COM.

**J# больше нет*

JIT-компиляция

Если у вас на всё хватает времени, то либо у вас много времени, либо маленькое всё

- Just in time compilation – компиляция во время исполнения
- Переводит команды IL в инструкции конкретной машины
- Компиляция происходит на уровне отдельных методов
- Компиляция происходит только при первом обращении к коду

JIT-компиляция (продолжение)

- JIT-компиляторы:
 - С оптимизациями
 - Без оптимизаций
- Платформы с ограниченными ресурсами: только упрощенный JIT-компилятор, так как он требует меньше памяти/времени

Лекция 2: Обзор .NET Framework

- Архитектура .NET
- **Common Language Runtime**
- MSIL
- Framework Class Library

Многоязычные приложения

Совместная работа программ на различных языках обеспечивается следующим:

- Общая управляемая среда (CLI)
- Общая система типов (CTS)
- Общезыковая спецификация (CLS)
- Метаданные (об этом на следующей лекции)

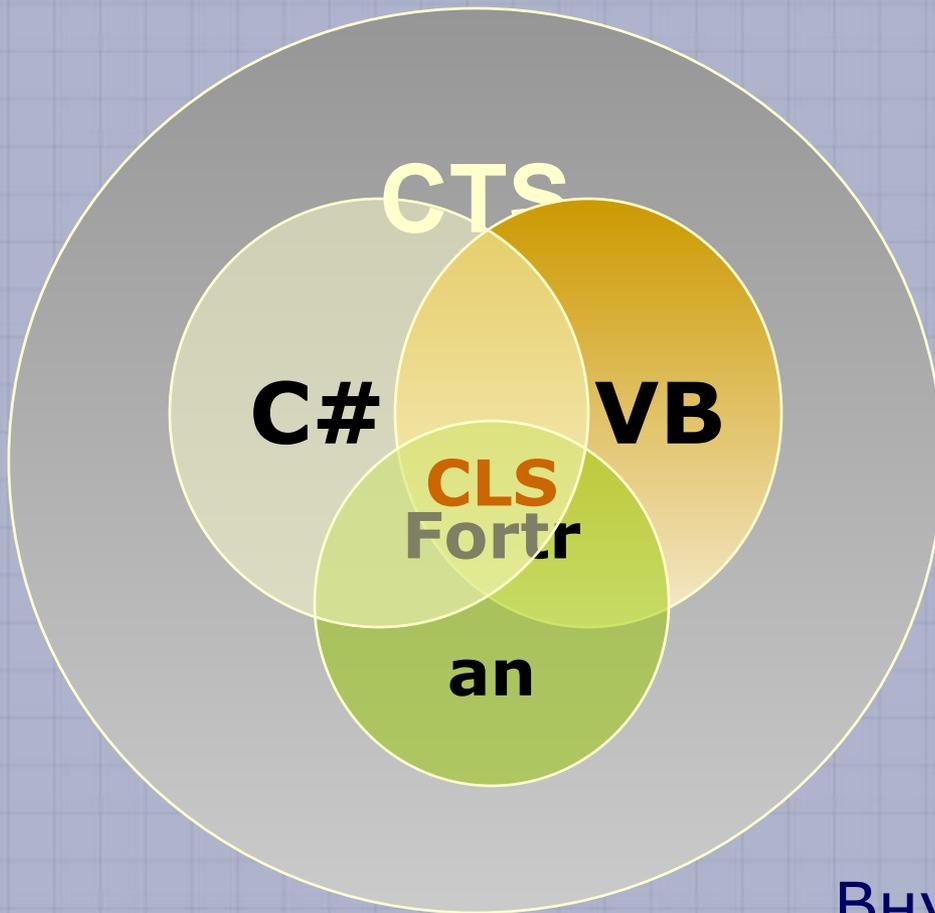
CLI

- Common Language Infrastructure – стандарт ECMA/ISO для единой инфраструктуры языков, по которому реализована .NET
- Минимальный набор возможностей компилятора, требуемый для работы в .NET
- В частности, определение правил для типов, чтобы они были доступны из других CLI-совместимых языков

CTS

- Common Type System
- Общее определение типов и их поведения:
 - Примитивные типы, типы-значения и ссылочные
 - Объектные и интерфейсные типы
 - Boxing и unboxing
- Например, определяется набор видов членов классов, видимостей, правила наследования типов, виртуальных функций и т.п.

CTS и CLS



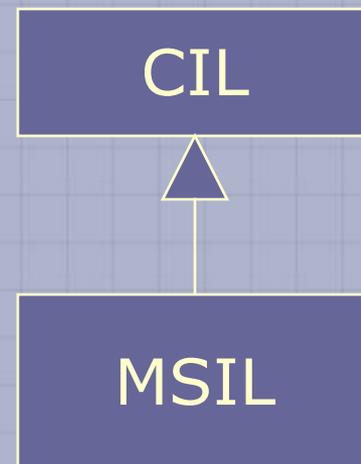
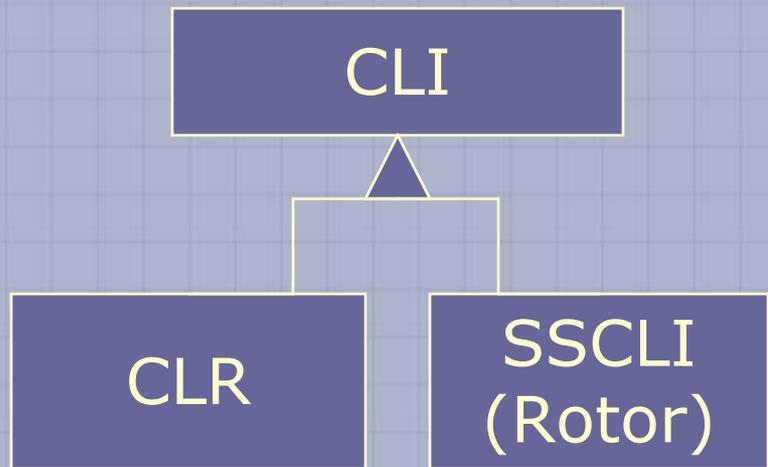
Языки .NET (кроме IL) поддерживают только подмножество CTS. Минимальный набор функций, который должны поддерживать все языки, определяется CLS

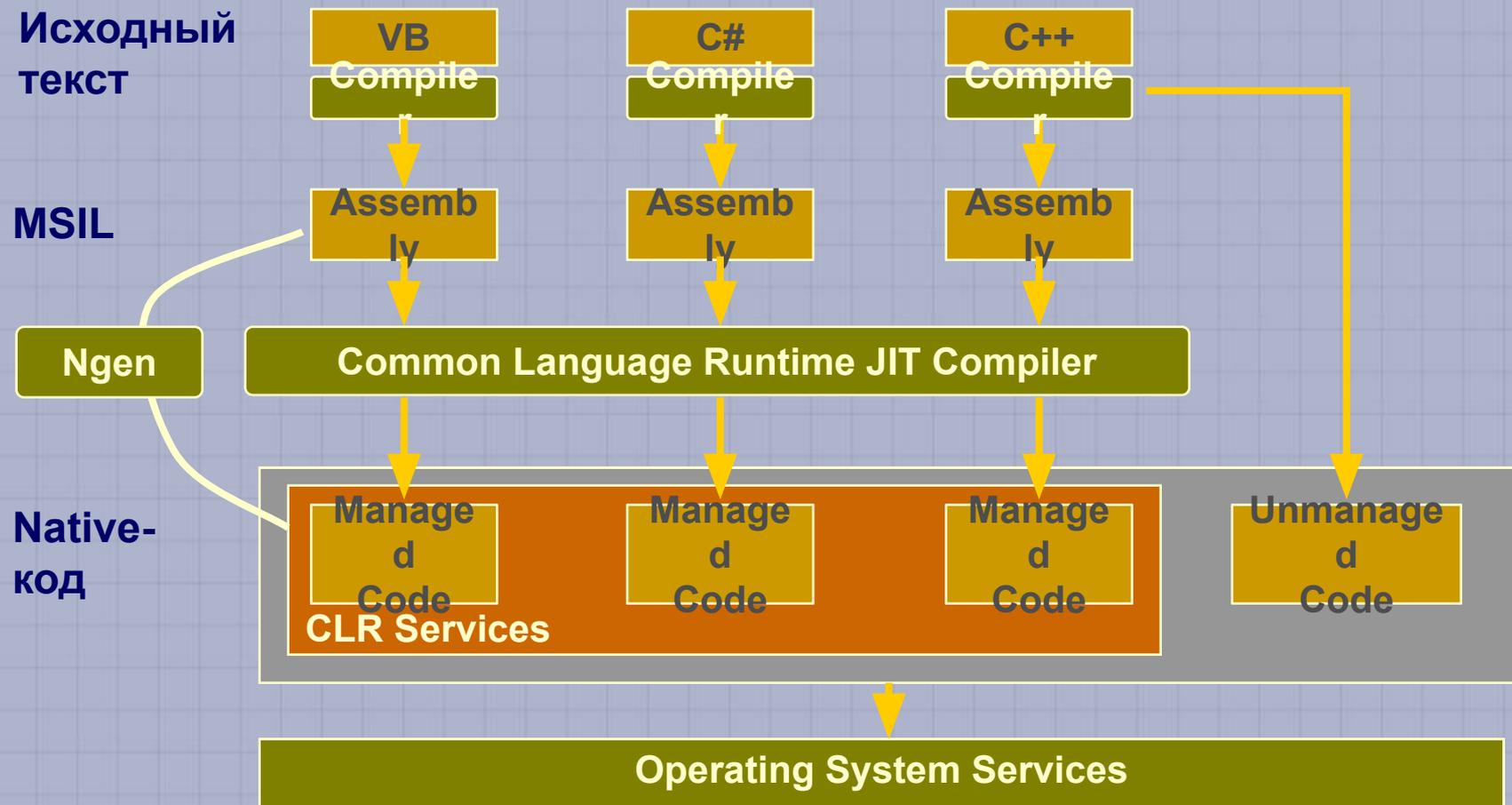
Внутренние типы могут не удовлетворять CTS

Наведём порядок

- CLI – стандарт ECMA 335 (2001 год), стандарт ISO 23271 (2003 год), Microsoft + HP + Intel
- CLR – конкретная реализация CLI от MS
- CIL и MSIL – стандарт и конкретная реализация промежуточного языка
- SSCLI – академическая open-source реализация CLI
 - Rotor – codename для SSCLI. Поддержаны – Win, FreeBSD, MacOS X

Картинка





- CLR обеспечивает исполнение скомпилированного в MSIL приложения
- CLR отвечает за сборку мусора
- CLR обеспечивает управление версиями

Лекция 2: Обзор .NET Framework

- Архитектура .NET
- Common Language Runtime
- **MSIL**
- Framework Class Library

MSIL

- Что такое IL
- Пример трансляции в IL
- Основные черты IL
- Что даёт IL
- Ассемблер и дизассемблер IL
- IL и интеллектуальная собственность
- IL и верификация

Что такое MSIL

- Microsoft Intermediate Language (MSIL) является языком ассемблера некоторой 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 типовой анализ программы и проверки условий корректности кода

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

}
}
```

Основные черты MSIL

- Машина является стековой, причем стек является статически типизированным
- Стек используется, как правило, только для хранения промежуточных результатов
- Большинство команд MSIL получают свои аргументы на стеке, удаляют их со стека и помещают вместо них результат(ы) вычисления
- Машина является объектно-ориентированной: структура MSIL отражает разбиение кода на классы, методы и т.п.

Что даёт MSIL

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



Ассемблер и дизассемблер IL

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



MSIL и IP

- IL – весьма высокоуровневый ассемблер
- ILDasm дает возможность восстановить исходный код приложения (включая имена методов)

Способы защиты вашей интеллектуальной собственности:

- Использование утилит, «мешающих» имена (Dotfuscator, .NET Reactor, CodeVeil)
- Размещение части модулей на сервере
- Реализация части алгоритмов в неуправляемых модулях (например, C++)

MSIL и верификация

- 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#)
- В .NET SDK входит программа PEVerify, которая осуществляет проверку типовой корректности исполняемого модуля
- Верификацию можно отключить

MSIL и верификация

- **Неуправляемый код:** может повредить данные другого процесса
- **Управляемый код:** проверено, мин нет
- Верифицированные управляемые приложения можно исполнять совместно в рамках одного процесса

Лекция 2: Обзор .NET Framework

- Архитектура .NET
- Common Language Runtime
- MSIL
- **Framework Class Library**

БИБЛИОТЕКА, и, ж.

1. Учреждение, собирающее и хранящее произведения печати и письменности для общественного пользования, а также осуществляющее справочно-библиографическую работу. *Публичная б. Научная, детская б. Передвижная б.*
2. Собрание книг, произведений печати, а также помещение, где они хранятся. *Б. учёного. Домашняя б.*
3. Название серии книг, объединённых тематически или по назначению, жанру. *Б. путешествий. Б. поэта (поэтическая).*

FCL

**Слон – это мыш, выполненная по
государственным спецификациям.
Роберт Хайнлайн**

- Содержит несколько тысяч классов, покрывающих всю функциональность WinAPI и не только
- Разбита на пространства имен (namespaces)

Классы FCL (выдержка)

System.Web

Services**Description****Discovery****Protocols****UI****HtmlControls****WebControls****Caching****Configuration****Security****SessionState**

System.Windows.Forms

Form**MessageBox****Button****ListControl**

System.Drawing

Drawing2D**Imaging****Printing****Text**

System.Data

OleDb**Design****SQL****SQLTypes**

System.Xml

XSLT**XPath****Serialization**

System

Collections**Configuration****Diagnostics****Globalization****IO****Net****Reflection****Resources****Security****ServiceProcess****Text****Threading****Runtime****InteropServices****Remoting****Serialization**

Текущая поддержка .NET

**Осуществляются наши самые смелые мечты!
Может, пора исполняться и несмелым?**

- .NET Framework можно бесплатно скачать с сайта Microsoft
- Windows 2003 и более поздние: .NET поставляется с ОС. Самый последний .NET (3.5) входит в Windows Server 2008 и Windows 7
- Языки: MS – C#, VB.NET, C++ (with Managed Extensions). Десятки реализаций других языков сторонними фирмами
- Mono, dotGNU – open source под другие платформы
- SSCLI (ROTOR) – доступно с исходными кодами, бесплатно для академических (некоммерческих) целей
- .NET 3.5 и Visual Studio 2008 – вышли осенью 2007 и весной 2008 соответственно
- <http://www.tipsntracks.com/7/the-donet-framework-versions-road-map.html> – вся история

Следующая лекция

Сборки (assemblies)