Язык программирования C#

Лекция 1.

Литература

1. Павловская Т.А. C#. Программирование на языке высокого уровня. Учебник для вузов. – СПб.: Питер, 2007. -432с.
2. Бишоп Дж., Хорспул Н. C# в кратком изложении. М.: БИНОМ. Лаборатория знаний, 2005. - 472 с.
3. Шилдт Г. C# 4.0. Полное руководство. М.: ИД Вильямс, 2013. – 1056 с.
4. Эндрю Троелсен. Язык программирования C# 2010 и платформа .NET 4. М.: ИД Вильямс, 2011. – 1392 с.
5. Кариев Ч. Создание Windows-приложений на основе Visual C#. Web: http://www.intuit.ru/studies/courses/106/106/info
6. Павловская Т. Программирование на языке высокого уровня C#. Web: http://www.intuit.ru/studies/courses/629/485/info

Уровни языков программирования:

- машинные;
- машинно-ориентированные (ассемблеры);
- машинно-независимые (языки высокого уровня).

В качестве примера рассмотрим программу на языке ассемблера для IBM PC. Программа вычисляет значение a = b + c для целых a, b и c:

```
        .MODEL
        .SMALL
        .DATA
b       DW   5
c       DW   3
a       DW   ?
        .CODE
begin   MOV AX,@DATA
        MOV  DS,AX
        MOV  AX,B
        ADD  AX,C
        MOV  A,AX
        MOV  AH,4CH
        INT  21H
        END  begin
```

Директива .MODEL задает механизм распределения памяти под данные и команды.

Директива .DATA определяет начало участка программы с данными.

Директивы DW задают типы переменных и их значения.

Директива .CODE определяет начало участка программы с командами.

Команды MOV AX,@DATA и MOV DS,AX записывают адрес сегмента данных в регистр DS (Data Segment).

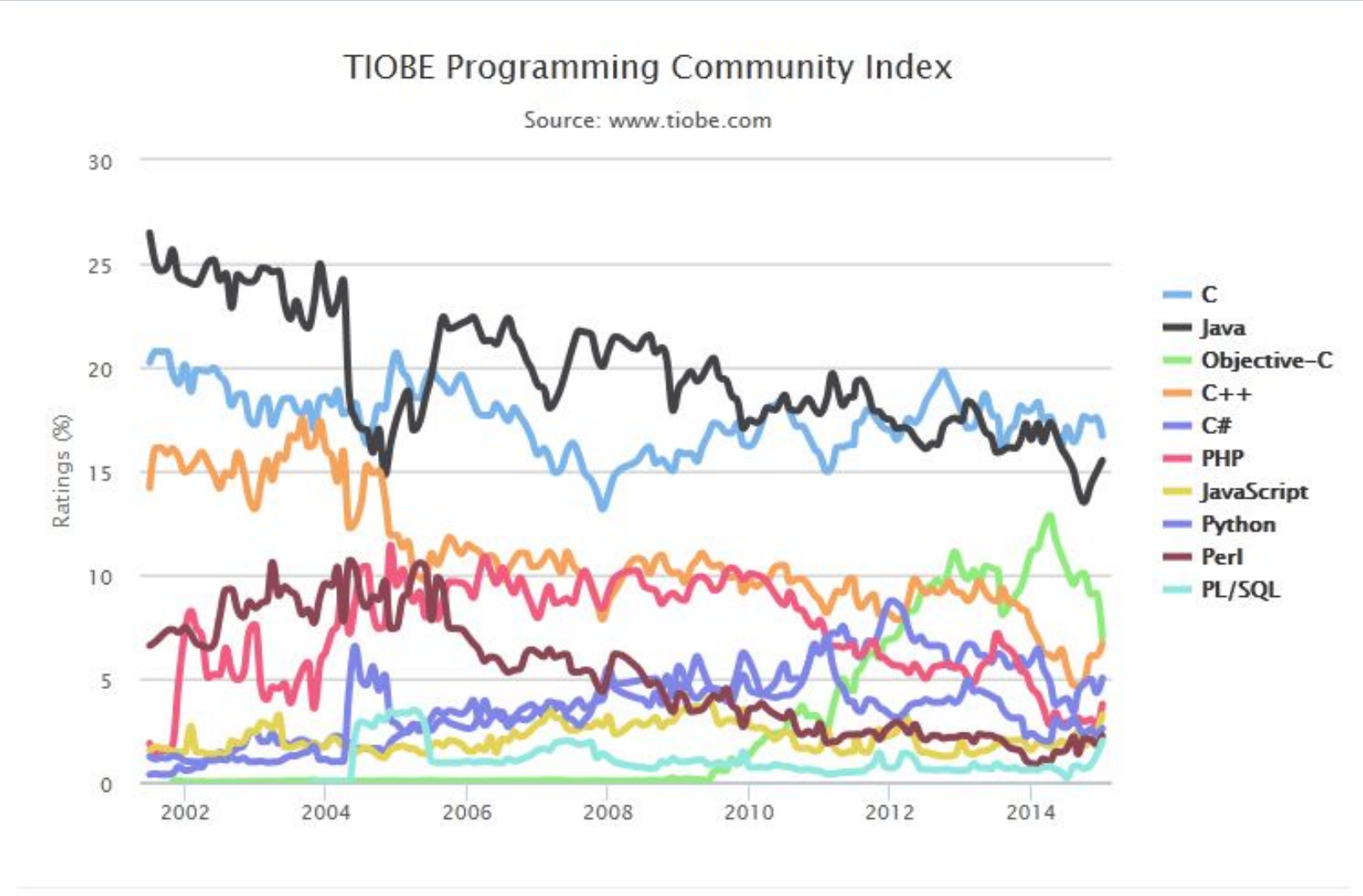
Для вычисления a используются команды MOV AX, B, ADD AX,C и MOV A,AX.

В директиве END задана метка первой выполняемой программы программы begin.

Программа на Бейсике	Программа на Паскале	Программа на СИ

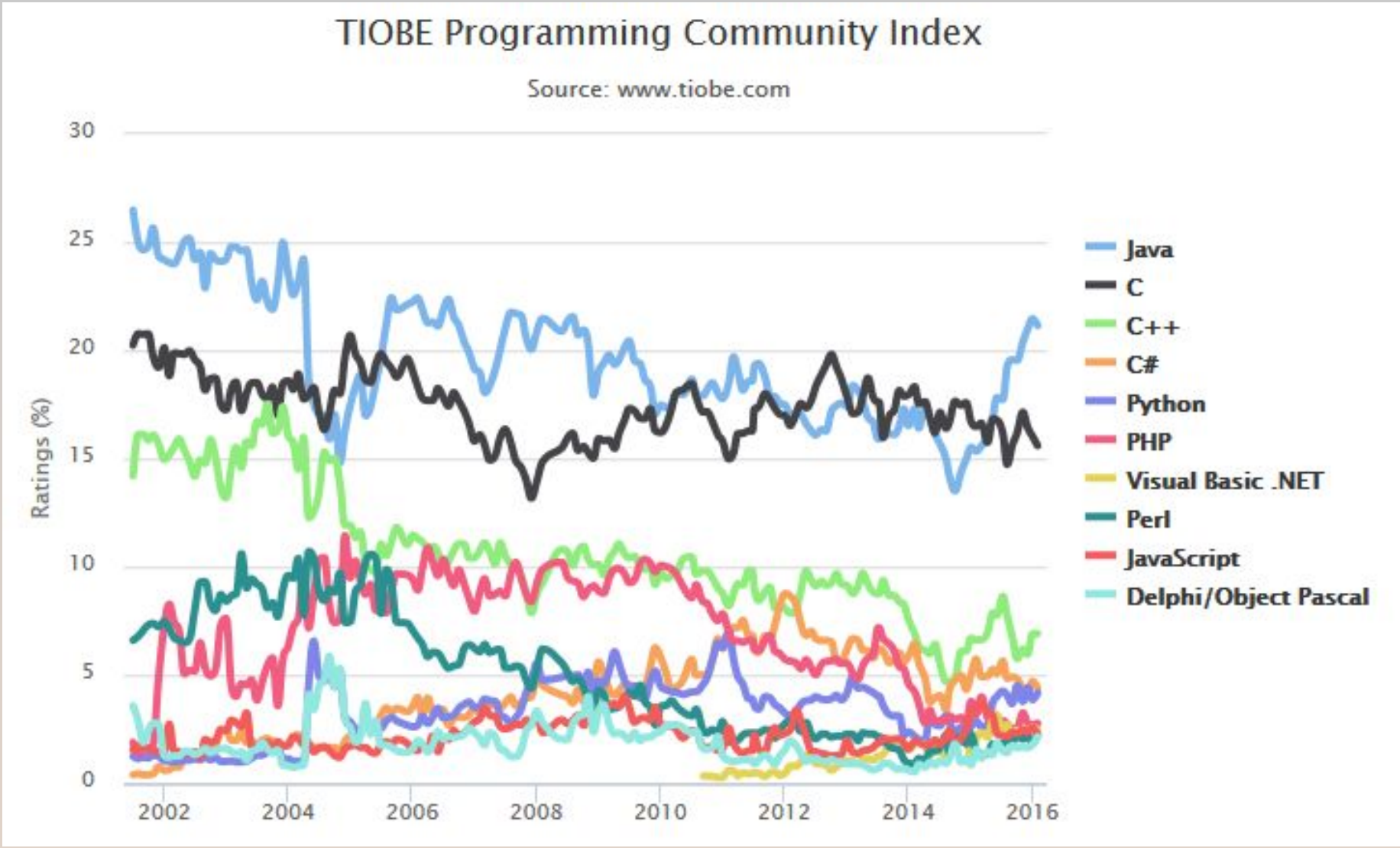
INPUT "N = "; N : DIM A(N)
FOR I = 1 TO N
PRINT "A("; I; ") =";
INPUT A(I)
NEXT I
S = 0
FOR I = 1 TO N
S = S + A(I)
NEXT I
PRINT "Сумма ="; S
END
	Program Summa;
	Type Mas = Array [1 .. 100] of Real;
	Var A : Mas;
	i, n: Integer;
	S : Real;
	BEGIN
	Write('n = '); ReadLn(n);
	For i : = 1 to n do
	begin
	Write('A[', i, '] = ');
	ReadLn(A[i]);
	end;
	S : = 0;
	
	For i : = 1 to n do
	S : = S + A[i];
	WriteLn('S = ', S:8:2);
		# include <stdio.h>
		# include <conio.h>
		main()
		{
		float a[100], s; int i, n;
		clrscr(); printf("n=");
		scanf("%i", &n);
		for (i = 1; i <= n; i++) {
		printf("a[%i]=", i);
		scanf("%f", &a[i]); }
		s = 0;
		for (i = 1; i <= n; i++)
		s = s + a[i];
		printf("s = % f \ n", s);
		return 0;
		}

TIOBE Index for January 2015

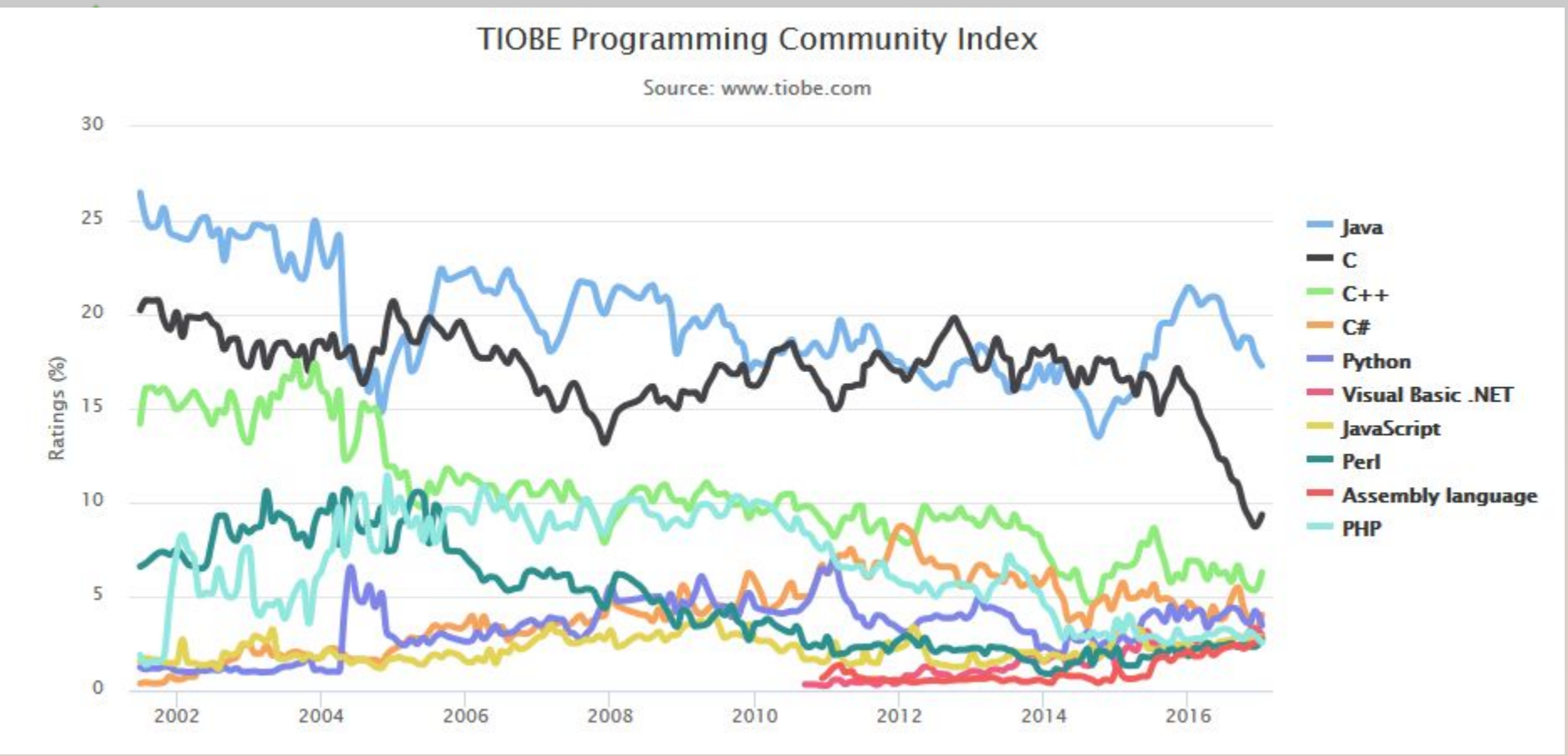


http://www.tiobe.com/tiobe-index//

TIOBE Index for February 2016



TIOBE Index for January 2017



История

- 1956 г. - язык FORTRAN
- 1972 г. - язык C (Bell Laboratories)
- 1974 г. - язык PASCAL (Никлаус Вирт)
- 1983 г. – язык C++ (Бьерн Страуструп)
- 1995 г. – языки Java (Sun Microsystems) и JavaScript (Netscape Communication)
- 2000 г. - платформа .Net и язык C# (Microsoft)

Платформа Microsoft .NET

.NET – это платформа, которая содержит средства для разработки любых современных новых приложений и одновременно – средства для обеспечения их работы. Она состоит из нескольких основных составляющих:

- **.Net Framework** – фундамент платформы, обеспечивающий создание, запуск и управление выполнением программы и библиотечных компонент;
- **.Net Enterprise Servers** – семейство корпоративных серверов;
- **Visual Studio .Net** – средство для разработки приложений;
- **Клиенты** – операционные системы типа Windows;
- **XML Web-службы** – блоки, расположенные по всей сети Internet, с которыми могут взаимодействовать приложения, основываясь на спецификациях языка XML, протоколах HTTP или SOAP.

Framework .Net - единый каркас среды разработки

В каркасе Framework .Net можно выделить два основных компонента:

- статический - FCL (Framework Class Library) - библиотеку классов каркаса - набор сборок в формате DLL, содержащих несколько тысяч определений типов, каждый из которых предоставляет некоторую функциональность
- динамический - CLR (Common Language Runtime) - общеязыковую исполнительную среду - исполняющая среда для программ, написанных на .NET-совместимых языках программирования. CLR — это по сути виртуальная машина, в которой функционируют приложения .NET.

Microsoft начала разрабатывать .NET Framework в конце 1990-х под именем «Next Generation Windows Services» (NGWS). В 2000 году была выпущена первая бета-версия .NET 1.0.

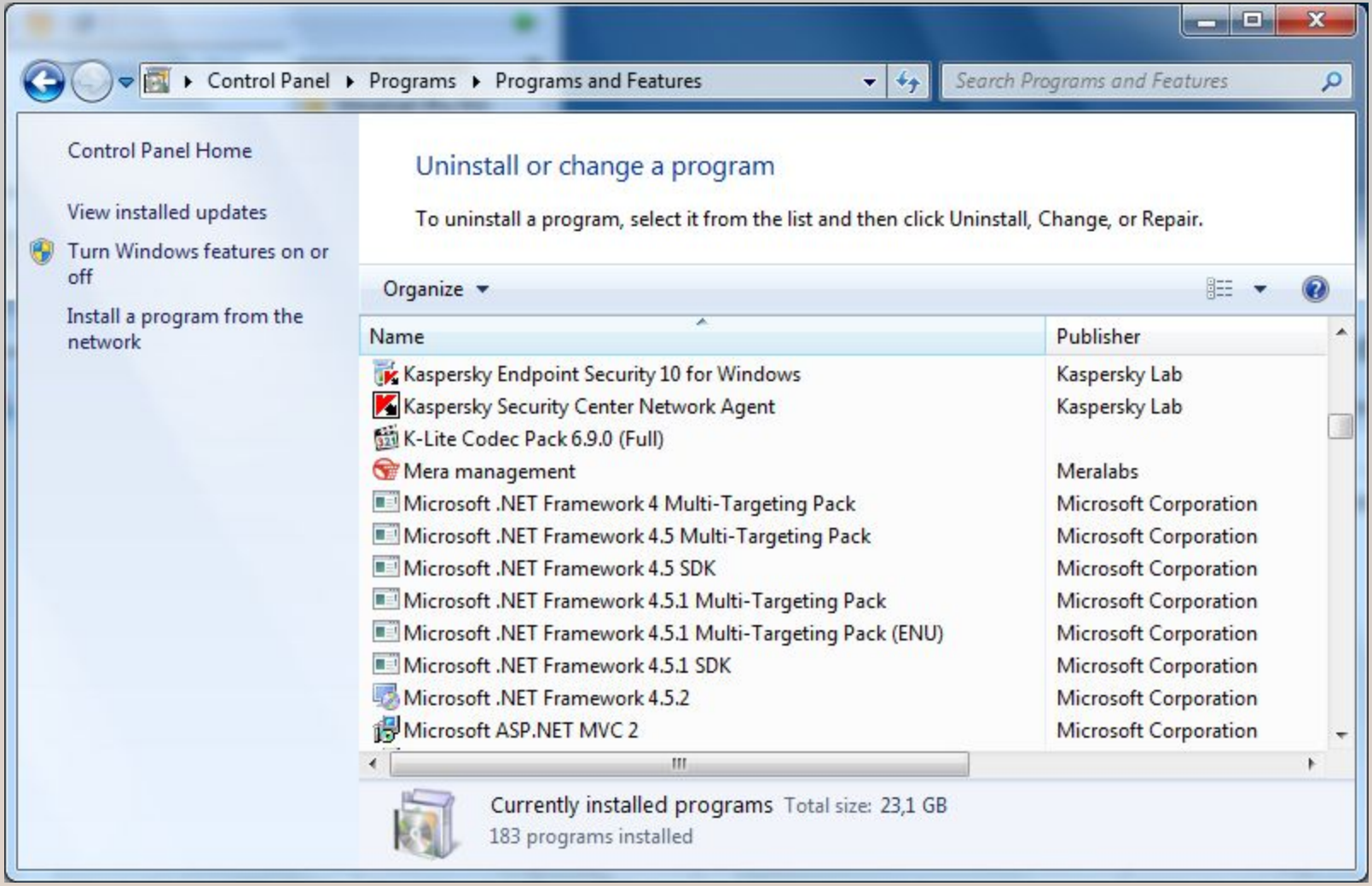
Версии .NET Framework:

Версия	Дата выхода	Visual Studio	По умолчанию в Windows
1.0	1 мая 2002 года	Visual Studio .NET	-
1.1	1 апреля 2003 года	Visual Studio .NET 2003	Windows Server 2003
2.0	11 июля 2005 года	Visual Studio 2005	Windows Server 2008 R2, Windows Server 2008 SP2, Windows Server 2003
3.0	6 ноября 2006 года	Visual Studio 2005 + расширения	Windows Vista, Windows Server 2008, Windows Server 2008 R2
3.5	9 ноября 2007 года	Visual Studio 2008	Windows 7, 8, 8.1, 10, Windows Server 2008 R2
4.0	12 апреля 2010 года	Visual Studio 2010	-
4.5	15 августа 2012 года	Visual Studio 2012	Windows 8, Windows Server 2012
4.5.1	2013-10-17	Visual Studio 2013	Windows 8.1, Windows Server 2012 R2

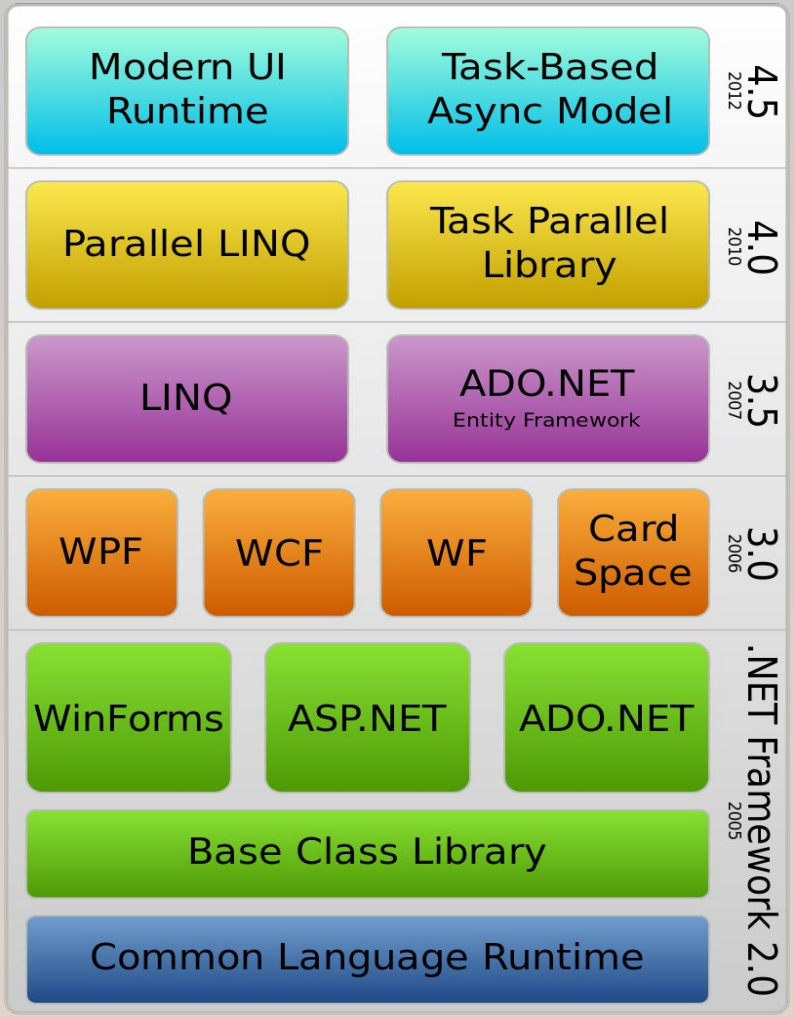
Продолжение.
Версии .NET Framework:

Версия	Дата выхода	Visual Studio	По умолчанию в Windows
4.5.2		-	-
.NET 4.6		Visual Studio 2015	Windows 10
Net 4.6.1		-	Windows 10 обновл.
.NET 4.6.2	20 июля 2016 года	-	Windows 10 Anniversary Update

https://msdn.microsoft.com/ru-ru/library/bb822049(v=vs.110).aspx


Control Panel ▸ Programs ▸ Programs and Features
Search Programs and Features
Control Panel Home
View installed updates
Turn Windows features on or off
Install a program from the network
Uninstall or change a program
To uninstall a program, select it from the list and then click Uninstall, Change, or Repair.
Organize
Name
Publisher
Kaspersky Endpoint Security 10 for Windows
Kaspersky Lab
Kaspersky Security Center Network Agent
Kaspersky Lab
K-Lite Codec Pack 6.9.0 (Full)
Mera management
Meralabs
Microsoft .NET Framework 4 Multi-Targeting Pack
Microsoft Corporation
Microsoft .NET Framework 4.5 Multi-Targeting Pack
Microsoft Corporation
Microsoft .NET Framework 4.5 SDK
Microsoft Corporation
Microsoft .NET Framework 4.5.1 Multi-Targeting Pack
Microsoft Corporation
Microsoft .NET Framework 4.5.1 Multi-Targeting Pack (ENU)
Microsoft Corporation
Microsoft .NET Framework 4.5.1 SDK
Microsoft Corporation
Microsoft .NET Framework 4.5.2
Microsoft Corporation
Microsoft ASP.NET MVC 2
Microsoft Corporation
Currently installed programs Total size: 23,1 GB
183 programs installed

Платформа .NET



The .NET Framework Stack

FCL

CLR

CLR – это совокупность утилит и служб, которые включают в себя все необходимые для создания, запуска и выполнения приложения компоненты. CLR включает:

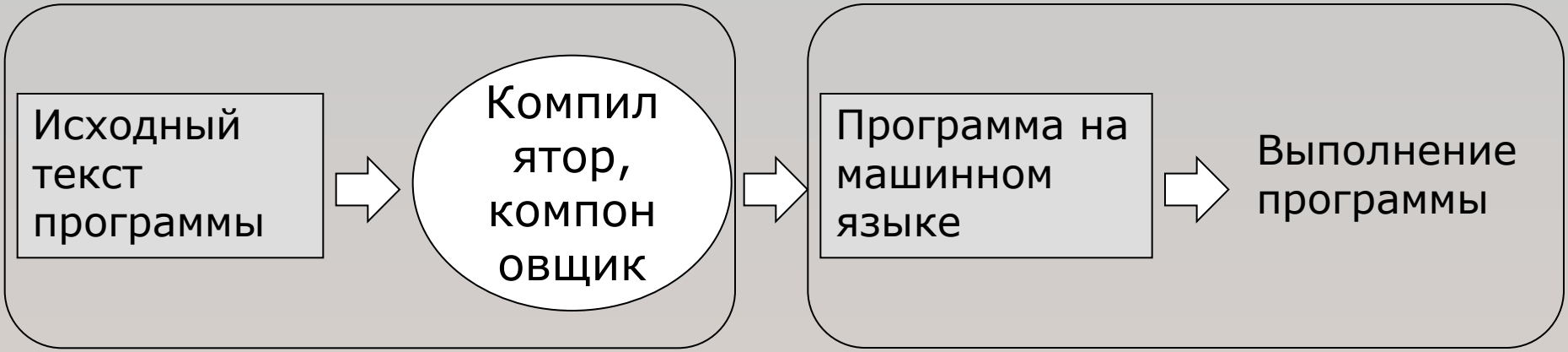
- **Class Loader** – управляет загрузкой классов;
- **MSIL** (Microsoft Intermediate Language).
- **Code Manager** – управляет запуском кода;
- **Carbage Collector** – сборщик мусора, предназначенный для освобождения неиспользуемых приложением ресурсов памяти;
- **Security Engine** - обеспечивает механизм защиты приложений;
- **Debugger** – позволяет пользователю отлаживать приложение и отслеживать выполнение запущенного приложения;
- **Type checker** – гарантирует правильное приведение типов данных;
- **Exception manager** – обеспечивает механизм передачи исключения;
- **Thread support** – компонент, отвечающий за поддержку классов для создания многопоточных приложений;
- **COM marshaler** – обеспечивает взаимодействие с COM-компонентами.

CLR (Common Language Runtime)

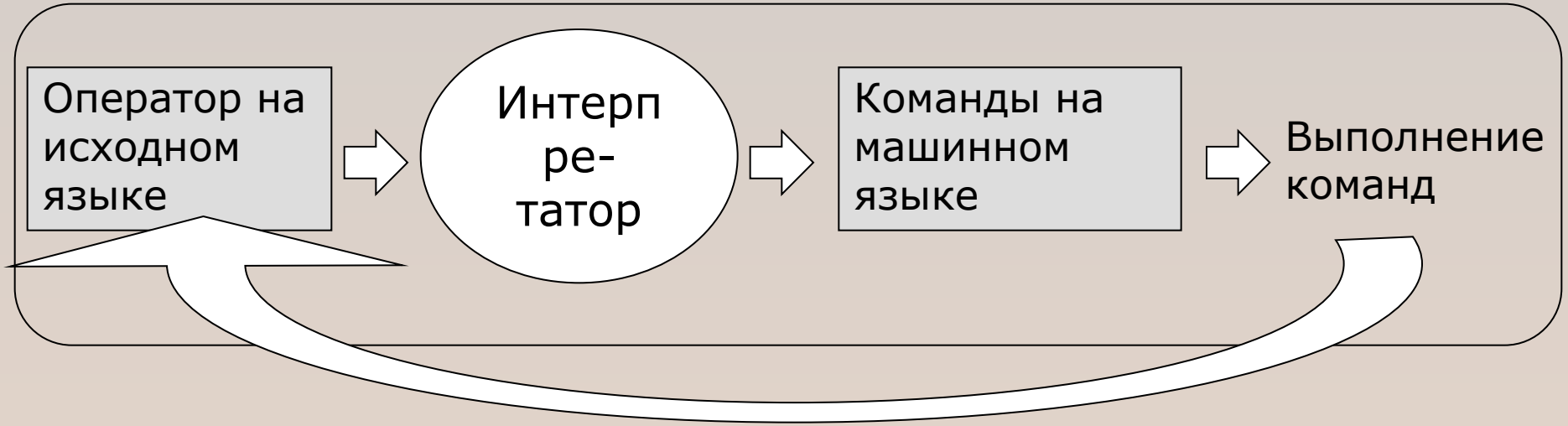
- ✔ C++/CLI
- ✔ C#
- ✔ F#
- ✔ Visual Basic
- ✔ Iron Python
- ✔ Iron Ruby
- ✔ Ассемблер Intermediate Language (IL)

Трансляция

Компиляция



Интерпретация



Гибридная схема трансляции

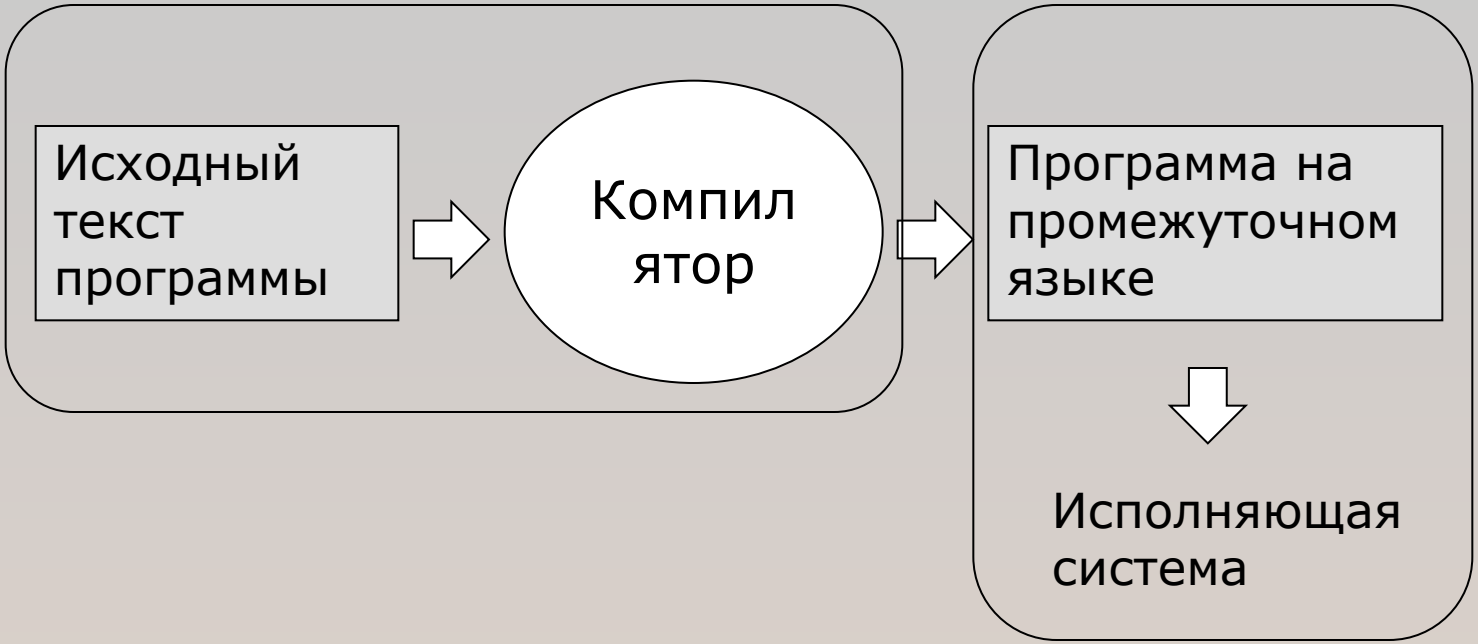
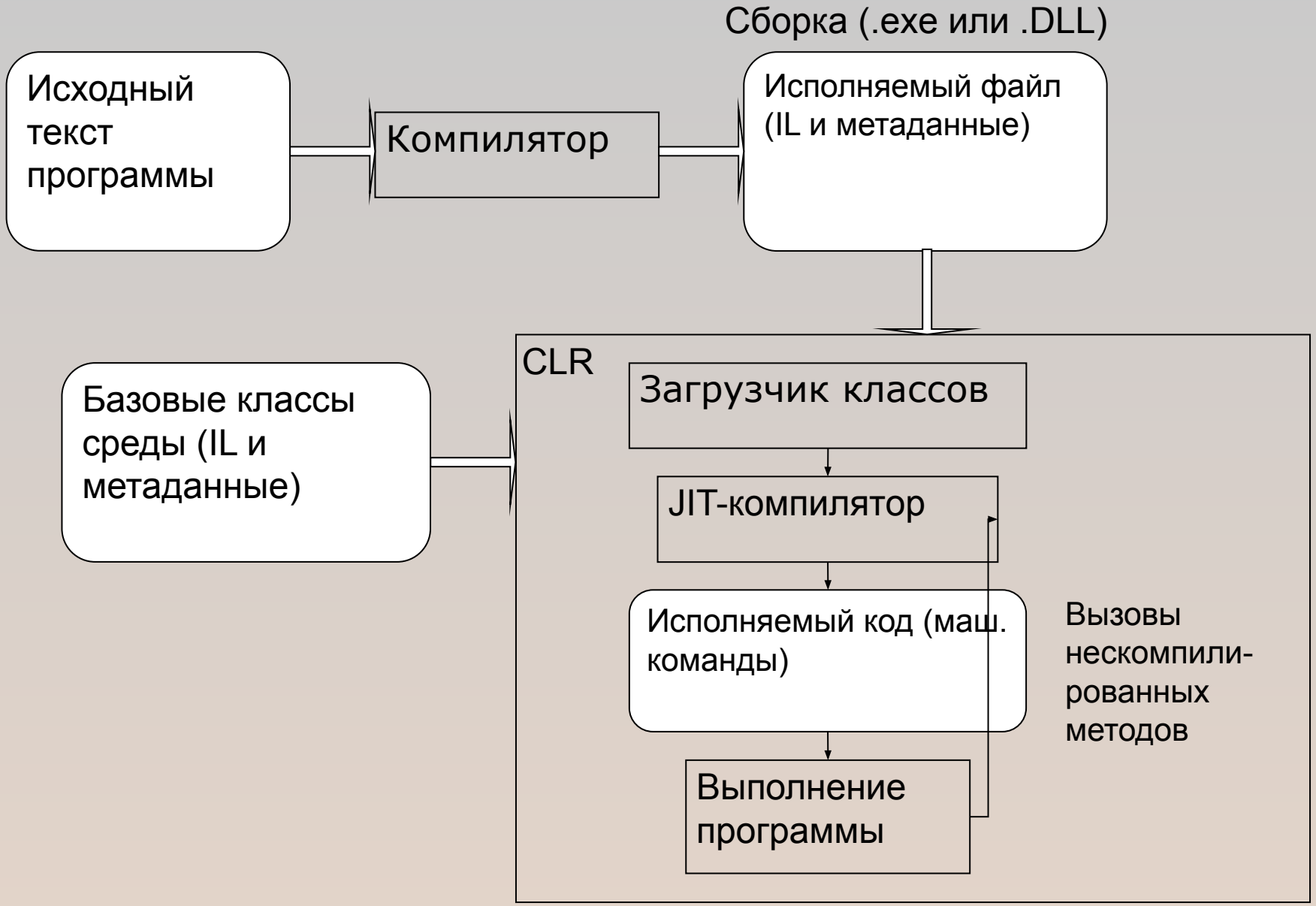
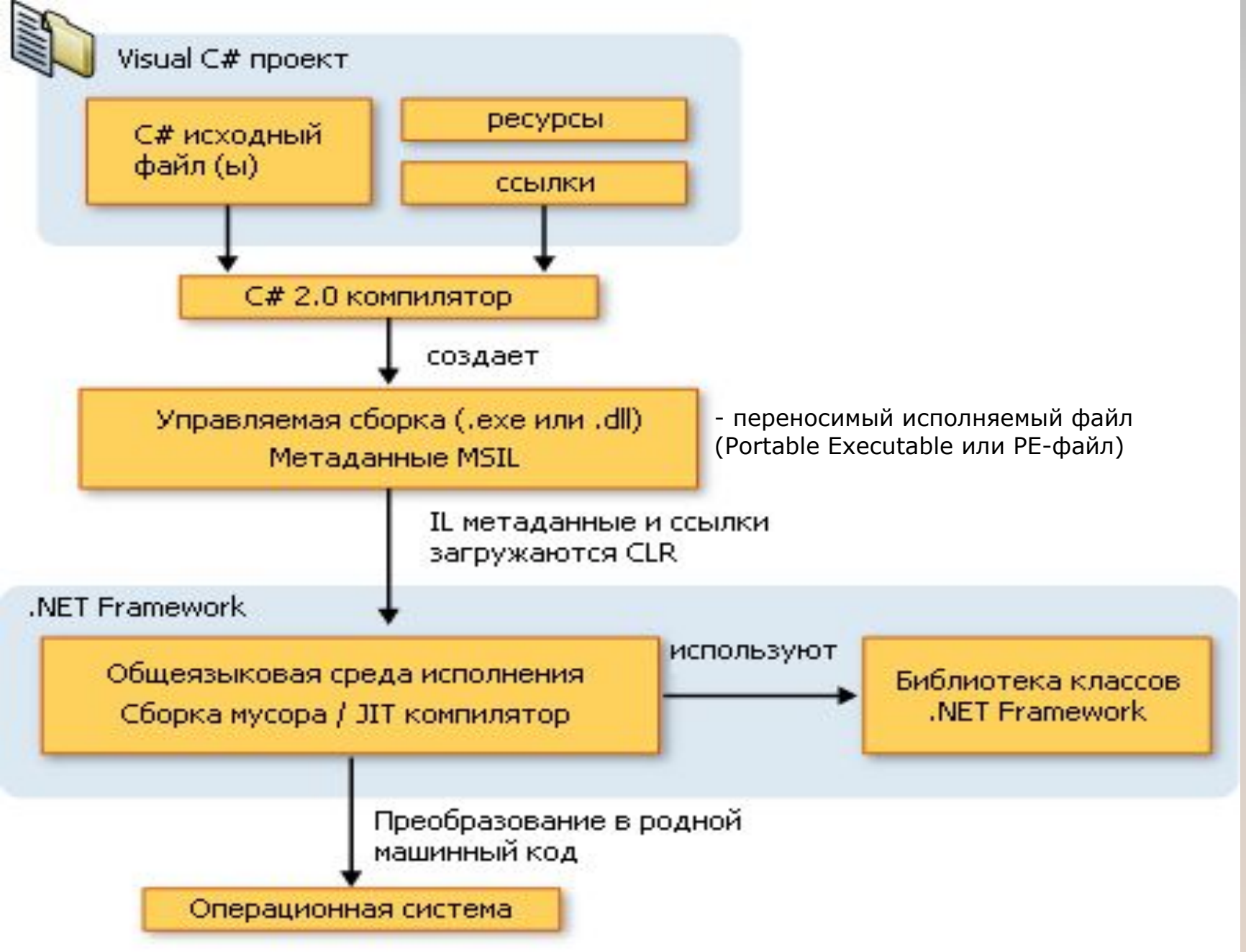


Схема выполнения программы в .NET




Visual C# проект
C# исходный файл (ы)
ресурсы
ссылки
C# 2.0 компилятор
создает
Управляемая сборка (.exe или .dll)
Метаданные MSIL
- переносимый исполняемый файл (Portable Executable или PE-файл)
IL метаданные и ссылки загружаются CLR
.NET Framework
Общеязыковая среда исполнения
Сборка мусора / JIT компилятор
используют
Библиотека классов .NET Framework
Преобразование в родной машинный код
Операционная система

Компиляция исходного кода в управляемые модули

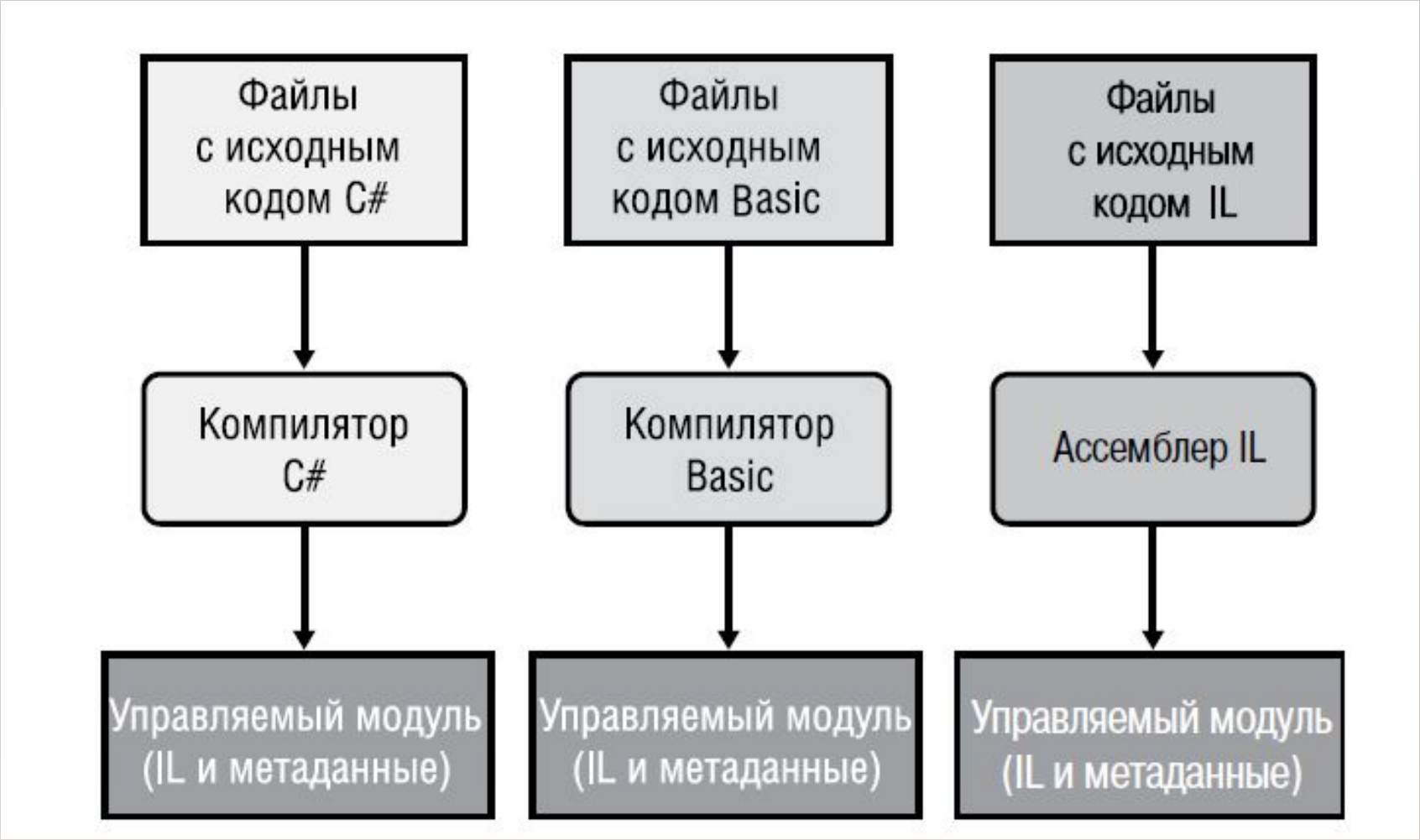
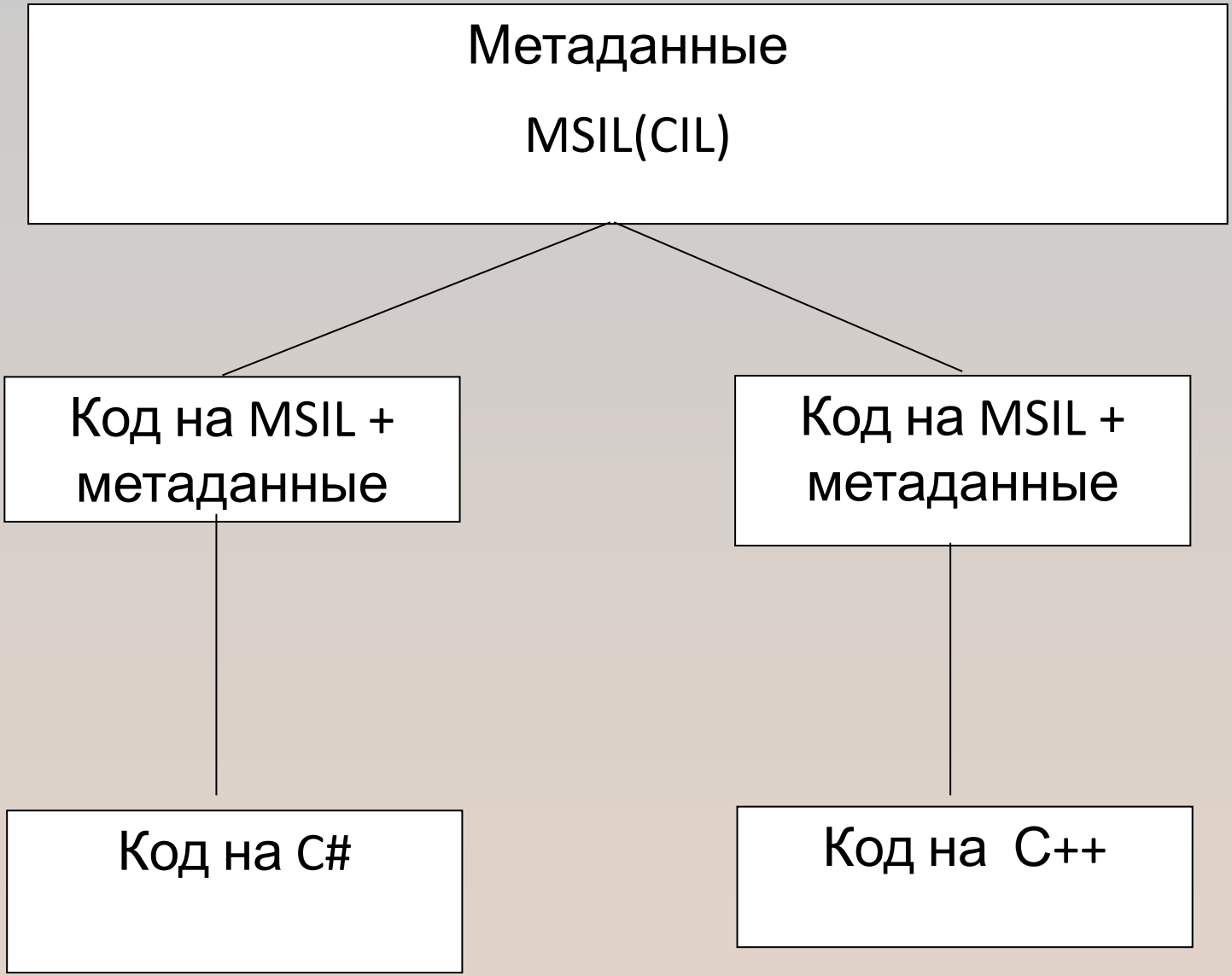
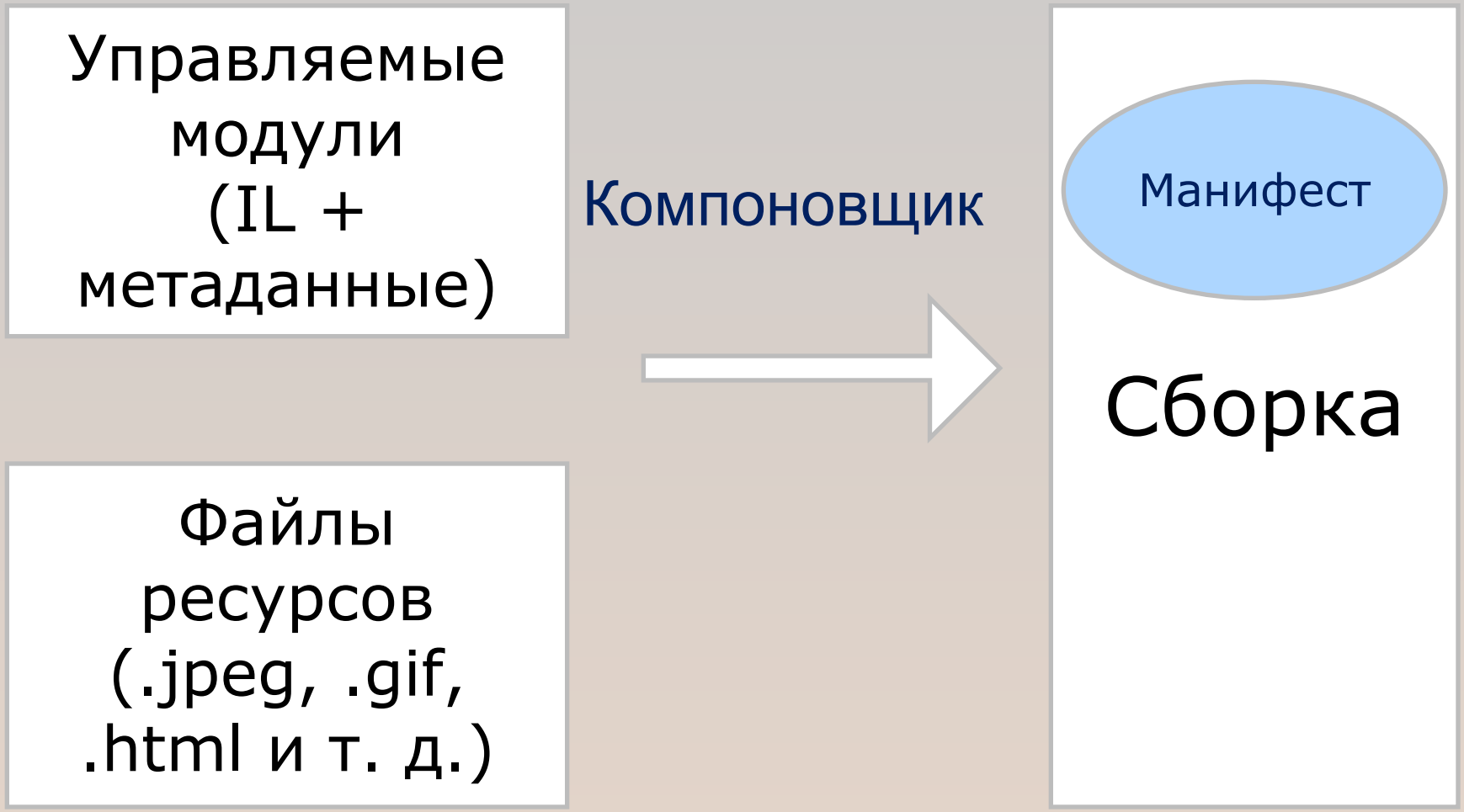


Схема формирования сборного кода в MSIL из модулей на разных языках программирования



Объединение управляемых модулей в сборку



Понятие сборки (assembly)

единица повторного использования кода, в которой поддерживается система управления версиями и заложена система управления безопасностью программного обеспечения

Манифест - служебная информация, которая включается в сборку дополнительно к программному коду

Процесс создания приложения:

- Каждый файл компилируется в исполняемый модуль, который является совокупностью кода на промежуточном языке программирования и набором метаданных, полностью описывающий все типы данных и структур, классов и объектов, входящих в модуль;
- С помощью программы - сборщика al.exe (Assembler Linker) модули и другие файлы объединяются в сборку. При этом они могут войти в сборку непосредственно, а могут располагаться в каталогах сборки, сама сборка при этом содержит ссылки на эти файлы;
- Сборка транслируется средой CLR в автокод, которая после этого начинает управлять выполнением программы, вызывая в процессе выполнения необходимые классы и методы классов из нужных стандартных библиотек.

Посмотрим по порядку, что же происходит с кодом

1. Вы пишете исходный код на C#.
2. Затем вы компилируете его с помощью компилятора языка C# в EXE-файл.
3. Компилятор создает MSIL-код и помещает в раздел "только-на-чтение" выходного файла стандартный PE-заголовок (признак машино-независимой выполняемой программы для Win32). При создании выходного файла компилятор импортирует из CLR функцию _CorExeMain.
4. Когда приложение начинает выполняться, ОС загружает этот PE, а также все нужные DLL, в частности, библиотеку, которая экспортирует функцию _CorExeMain (mscoree.dll).
5. Загрузчик ОС выполняет переход в точку входа PE, устанавливаемую компилятором. Это ничем не отличается от процедуры загрузки в Windows любого другого PE. Однако так как ОС не в состоянии выполнить MSIL-код, то фактически в точке входа содержится заглушка, в которой установлена команда перехода к функции jCorExeMain из mscoree.dll.
6. Функция JCorExeMain переходит к выполнению MSIL-кода, помещенного в PE.
7. Так как MSIL-код не может быть выполнен непосредственно (ведь это не машинный код), CLR компилирует его с помощью оперативного (just-in-time, или JIT) компилятора в команды процессора. Эта компиляция выполняется только для непосредственно вызываемых методов программы. Откомпилированный выполняемый код сохраняется на машине и перекомпилируется только в случае изменения исходного кода.

Манифест

- ✔ Версия сборки
- ✔ Список файлов, входящих в сборку
- ✔ Имена и версии сборок, которые используются данной сборкой
- ✔ Права на запуск и использование

Метаданные

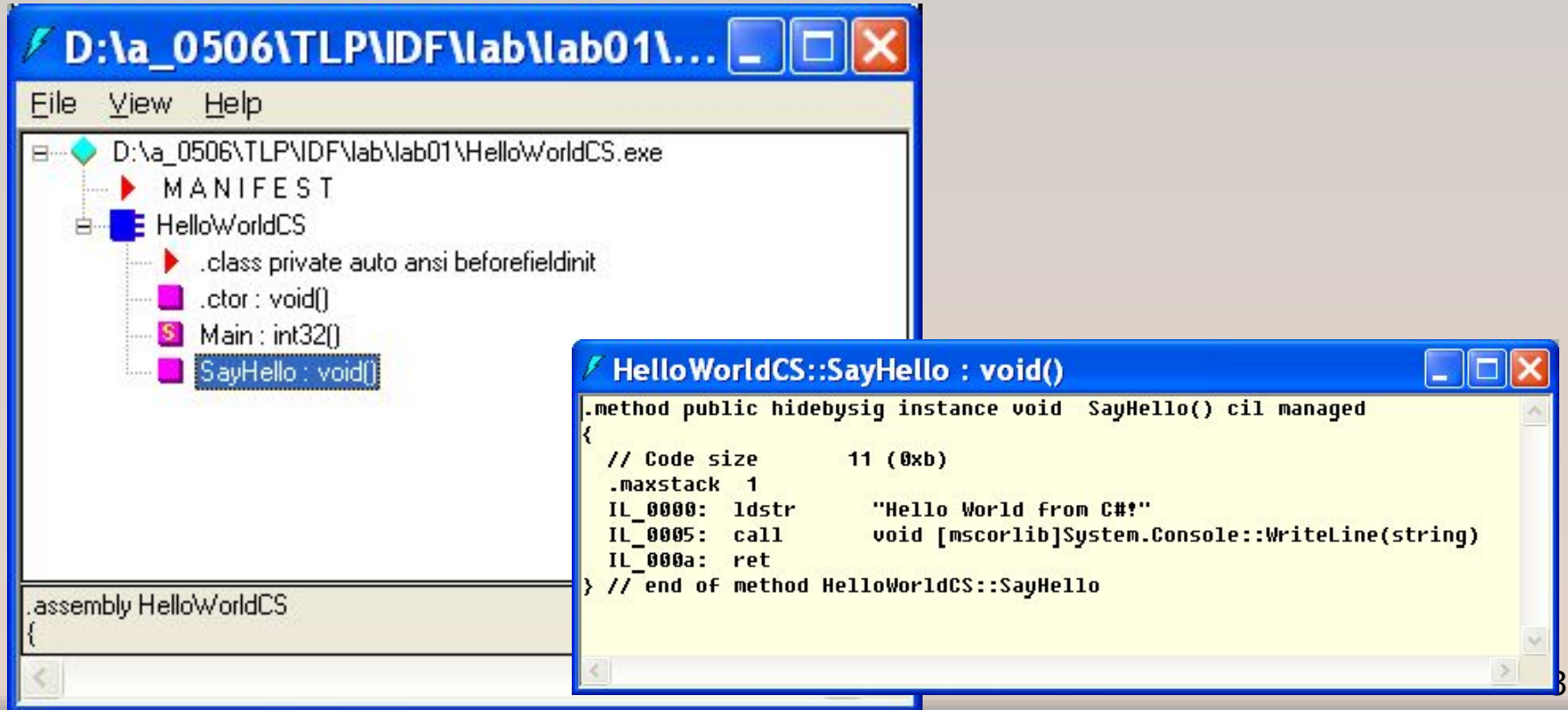
это таблицы, описывающие типы данных и их члены, определенные в исходном коде, а также таблицы, описывающие типы данных, и их члены, на которые имеются ссылки в исходном коде

IL код

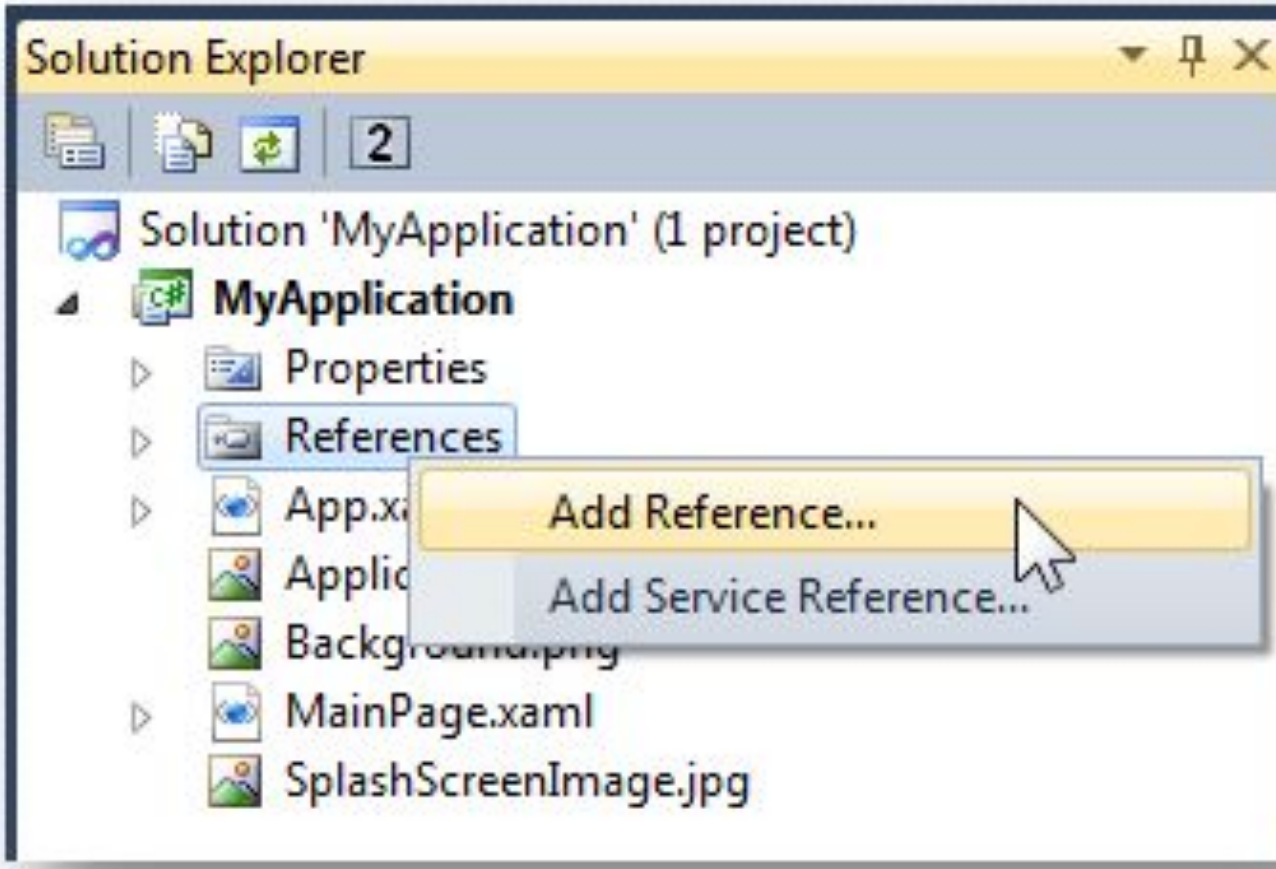
не зависящий от процессора объектно-ориентированный машинный язык

ILDASM Tool - **Ildasm.exe (IL Disassembler)**

Дизассемблер IL — сопутствующее средство Ассемблера IL (Ilasm.exe).Ildasm.exe принимает входной исполняемый файл (PE), содержащий код на промежуточном языке (IL), и создает соответствующий текстовый файл в качестве входных данных для Ilasm.exe.
Это средство автоматически устанавливается с Visual Studio.



Добавление сборок в проект



Конфигурационный файл

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <startup>
        <supportedRuntime version="v4.0
sku=".NETFramework,Version=v4.5" />
    </startup>
</configuration>
```

- C# создавался параллельно с каркасом Framework .Net и в полной мере учитывает все его возможности - как FCL, так и CLR;
- C# является полностью объектно-ориентированным языком, где даже типы, встроенные в язык, представлены классами;
- C# является мощным объектным языком с возможностями наследования и универсализации;
- C# является наследником языков C/C++, сохраняя лучшие черты этих популярных языков программирования. Общий с этими языками синтаксис, знакомые операторы языка облегчают переход программистов от C++ к C#;
- сохранив основные черты своего великого родителя, язык стал проще и надежнее. Простота и надежность, главным образом, связаны с тем, что на C# хотя и допускаются, но не поощряются такие опасные свойства C++ как указатели, адресация, разыменование, адресная арифметика;
- благодаря каркасу Framework .Net, ставшему надстройкой над операционной системой, программисты C# получают те же преимущества работы с виртуальной машиной, что и программисты Java. Эффективность кода даже повышается, поскольку исполнительная среда CLR предоставляет собой компилятор промежуточного языка, в то время как виртуальная Java-машина является интерпретатором байт-кода;
- мощная библиотека каркаса поддерживает удобство построения различных типов приложений на C#, позволяя легко строить Web-службы, другие виды компонентов, достаточно просто сохранять и получать информацию из базы данных и других хранилищ данных;
- реализация, сочетающая построение надежного и эффективного кода, является немаловажным фактором, способствующим успеху C#.

Особенности C#, заимствованные у Java

- **Классы**

Классов в C#, как и в Java очень много.

- **Интерфейсы**

Как и в Java, интерфейс - абстрактное определение коллекции методов.

- **Булевы операции**

Прямого преобразования между булевым типом любым другим типом данных нет.

- **Ошибки**

Как и в Java, управлять обработкой ошибок можно захватывая объекты исключения.

- **Управление памятью**

Существует автоматическая сборка "мусора", которая обеспечивается .NET.

Особенности, заимствованные у C и C ++

- **Компиляция**

Программы выполняют компиляцию непосредственно в стандартную двоичную выполнимую форму.

- **Структуры**

Структуры C# - подобны структурам в C++ и должны содержать определения данных и методы. Однако, в отличие от C++, структуры в C# не поддерживают наследование. Однако, подобно Java, структуры могут реализовывать интерфейсы.

- **Препроцессор**

Существуют директивы препроцессора для условной компиляции, предупреждений, ошибок и контроля.

- **Перегрузка операторов**

Некоторые операторы могут быть перегружены, а некоторые нет.

Особенности, уникальные для C#

- **Определения в namespace**

Когда вы создаете программу, вы создаете один или более классов в namespace. В нем же (вне класса) возможно объявление интерфейсов, enums и structs. Используя ключевые слова вы можете адресовать содержимое другого namespace.

- **Фундаментальные типы данных**

В C# существует более широкое разнообразие типов данных чем в C, C ++ или Java. Типы - bool, byte, ubyte, short, ushort, int, uint, long, ulong, float, double, and decimal. Подобно Java, все типы имеют установленный размер. Подобно C и C ++ все типы могут быть знаковыми и без знаковыми. Подобно Java, char содержит 16-ти битный unicode символ. В C# новым типом данных является тип decimal, который может содержать до 28 десятичных цифр.

Особенности, уникальные для C#

- **Два фундаментальных класса**

Класс object - базовый класс всех классов. Класс string - также базовый класс. Являясь частью языка он используется компилятором, когда вы создаете строку в вашей программе, заключая ее в кавычки.

- **Ассемблирование**

Ассемблирование - коллекция компилируемых классов и способность к выполнению других элементов языка, которые объединены в отдельном файле. Если это программа, файл имеет расширение EXE. Если это библиотека - DLL.

Дизассемблер:

1)C:\Program Files\Microsoft Visual Studio .Net\ FrameworkSDK\Bin\ildasm.exe

2) JetBrains dotPeek

Особенности, уникальные для C#

- **Признаки**

Каждый член класса имеет признаки: public, protected, internal, protected internal, or private.

- **Прохождение аргумента**

Методы могут объявляться для принятия некоторого числа аргументов. По умолчанию происходит передача значений фундаментальным типам данных. Ключевое слово ref может использоваться для передачи значения по определенной ссылке, которая позволяет возвращать значение. Ключевое слово out также вызывает переход по ссылке без передачи значения.

Особенности, уникальные для C#

- **Виртуальные методы**

Прежде, чем метод в базовом классе будет переписан, он должен быть объявлен как virtual. Метод в подклассе, который будет переписан, должен быть объявлен с помощью ключевого слова override. Это предотвратит случайную перезапись метода.

- **Свойства**

C# позволяет определять свойства внутри любого класса. Внутри C# класса, каждому свойству дается имя и тип данных. Ключевые слова set accessor и get accessor используется для объявления выполняемого кода при чтении или обновлении свойства.

Особенности, уникальные для C#

- **Delegate и callback**

объект delegate содержит информацию, необходимую для вызова определенного метода. К объекту delegate можно обратиться для безопасного запроса к представленному методу. Метод callback - пример delegate. Ключевое слово event используется в определении методов, которые вызываются при возникновении события.

- **Определение версий**

C# позволяет разработчикам поддерживать множество версий классов в двоичной форме, помещая их в различных namespace.

Особенности, уникальные для C#

- **Явные и неявные преобразования**

Подобно Java, C# учитывает неявное преобразование фундаментальных типов данных, пока нет вероятности потери данных (преобразование типа byte в int), но если есть вероятность потери данных (преобразование int в byte) выполняется явное преобразование. C# расширяет эту способность для других элементов программы

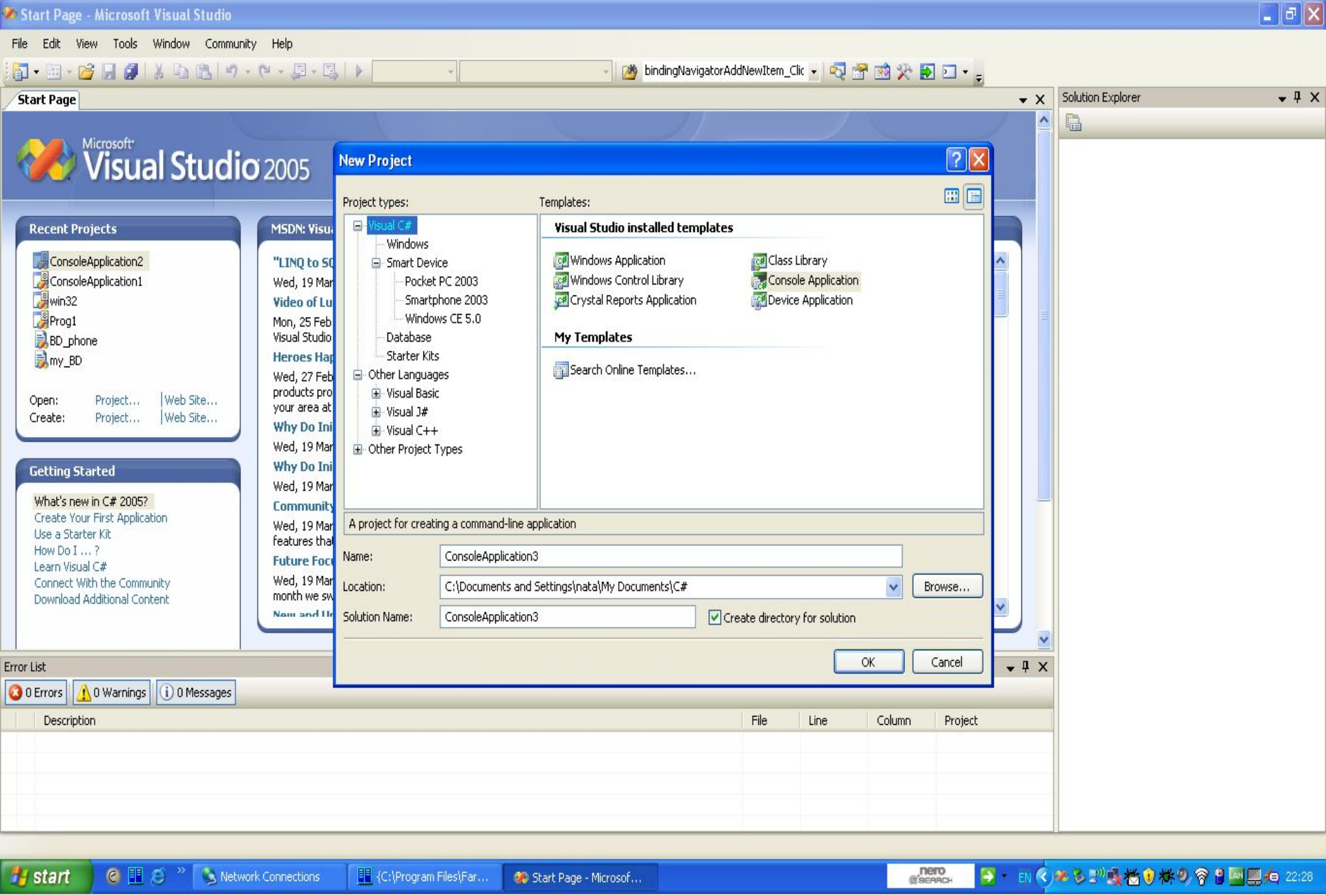
- **Внешне выполняемые методы**

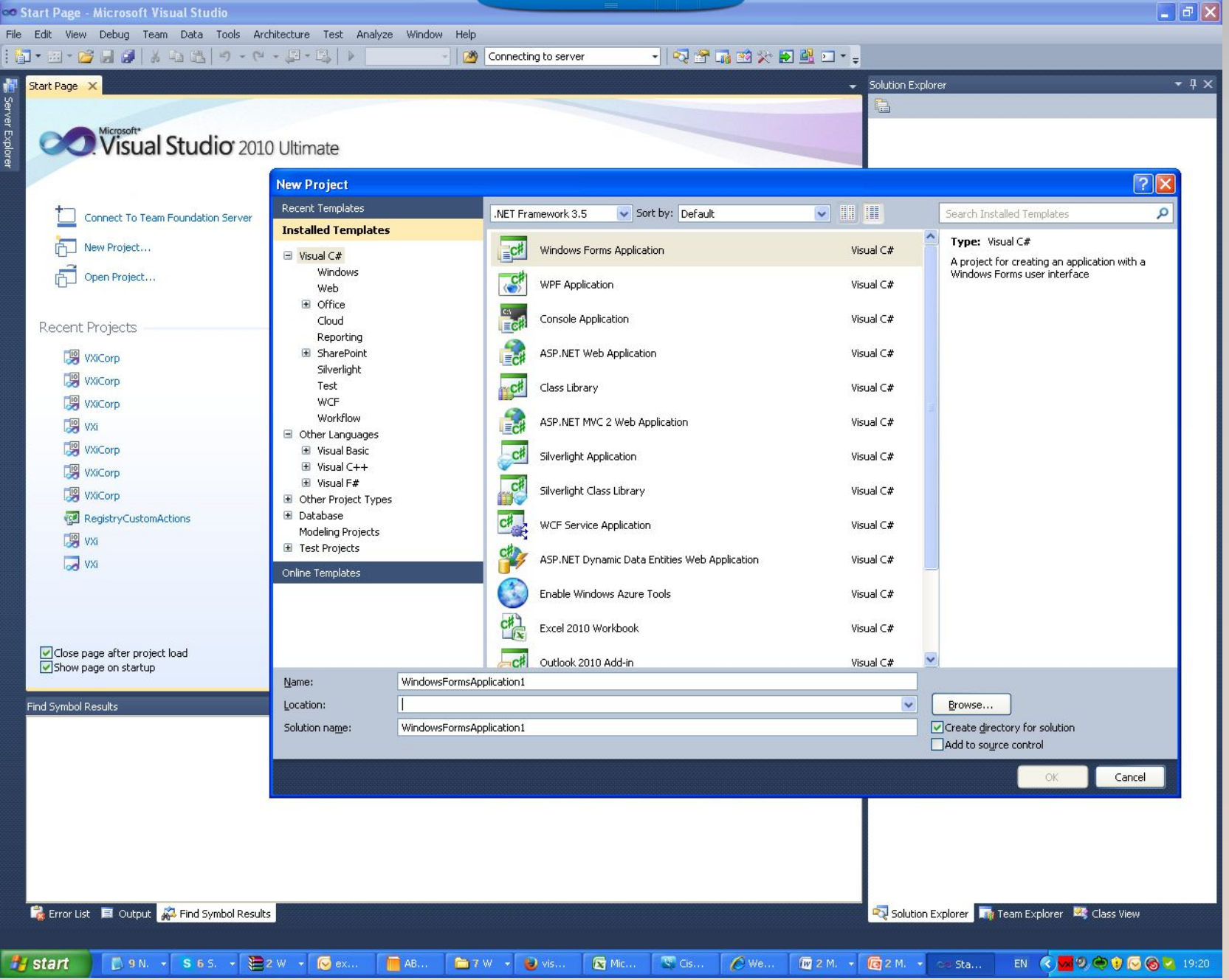
Методы в классе могут выполняться внешне. В следующем примере, статический метод RemoveDirectory выполняется в библиотеке под именем kernel32.dll:

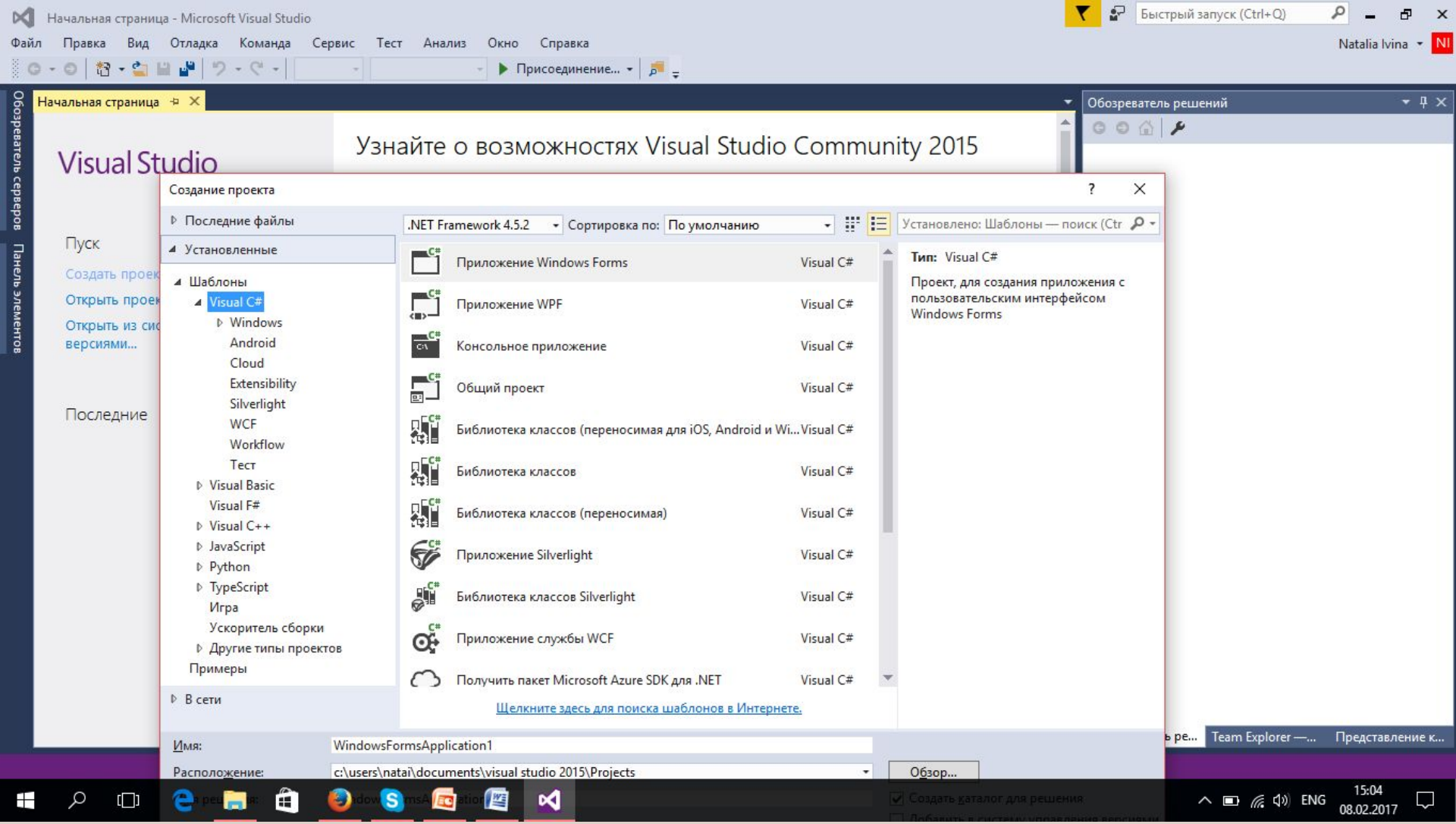
```
class Path {[DllImport("kernel32",
setLastError=true)]static extern bool
RemoveDirectory(string name);}
```

Среды разработки, поддерживающие .NET:

Среда разработки	Разработчик	ОС
Microsoft Visual Studio	**Microsoft**	**Windows**
SharpDevelop	**ICSharpCode Team**	**Windows**
MonoDevelop	**MonoDevelop Team** (корпорация Xamarin, ранее Novell). Проект возглавляет Мигель де Икаса, известный разработчик, основатель проекта GNOME.	**Linux, BSD, Mac OS X, Windows**
Embarcadero RAD Studio (Delphi for .NET)	**Embarcadero Technologies**	**Windows**
Zonnon	**Автор Юрг Гуткнехт (Швейцарский федеральный технологический институт)**	
PascalABC.NET	**Руководитель проекта Михалкович С.С. (Южный федеральный университет)**	


Start Page - Microsoft Visual Studio
File Edit View Tools Window Community Help
Start Page
Microsoft Visual Studio 2005
Recent Projects
ConsoleApplication2
ConsoleApplication1
win32
Prog1
BD_phone
my_BD
Open: Project... Web Site...
Create: Project... Web Site...
Getting Started
What's new in C# 2005?
Create Your First Application
Use a Starter Kit
How Do I ... ?
Learn Visual C#
Connect With the Community
Download Additional Content
New Project
Project types:
Templates:
Visual C#
Windows
Smart Device
Pocket PC 2003
Smartphone 2003
Windows CE 5.0
Database
Starter Kits
Other Languages
Visual Basic
Visual J#
Visual C++
Other Project Types
Visual Studio installed templates
Windows Application
Class Library
Windows Control Library
Console Application
Crystal Reports Application
Device Application
My Templates
Search Online Templates...
A project for creating a command-line application
Name: ConsoleApplication3
Location: C:\Documents and Settings\nata\My Documents\C#
Browse...
Solution Name: ConsoleApplication3
Create directory for solution
OK
Cancel
Solution Explorer
Error List
0 Errors
0 Warnings
0 Messages
Description
File
Line
Column
Project
start
Network Connections
{C:\Program Files\Far...
Start Page - Microsof...
EN
22:28


Start Page - Microsoft Visual Studio
File Edit View Debug Team Data Tools Architecture Test Analyze Window Help
Connecting to server
Start Page
Solution Explorer
Microsoft Visual Studio 2010 Ultimate
Connect To Team Foundation Server
New Project...
Open Project...
Recent Projects
VXiCorp
VXiCorp
VXiCorp
VXi
VXiCorp
VXiCorp
VXiCorp
RegistryCustomActions
VXi
VXi
Close page after project load
Show page on startup
New Project
Recent Templates
Installed Templates
Visual C#
Windows
Web
Office
Cloud
Reporting
SharePoint
Silverlight
Test
WCF
Workflow
Other Languages
Visual Basic
Visual C++
Visual F#
Other Project Types
Database
Modeling Projects
Test Projects
Online Templates
.NET Framework 3.5
Sort by: Default
Search Installed Templates
Windows Forms Application Visual C#
WPF Application Visual C#
Console Application Visual C#
ASP.NET Web Application Visual C#
Class Library Visual C#
ASP.NET MVC 2 Web Application Visual C#
Silverlight Application Visual C#
Silverlight Class Library Visual C#
WCF Service Application Visual C#
ASP.NET Dynamic Data Entities Web Application Visual C#
Enable Windows Azure Tools Visual C#
Excel 2010 Workbook Visual C#
Outlook 2010 Add-in Visual C#
Type: Visual C#
A project for creating an application with a Windows Forms user interface
Name: WindowsFormsApplication1
Location:
Solution name: WindowsFormsApplication1
Browse...
Create directory for solution
Add to source control
OK
Cancel
Find Symbol Results
Error List
Output
Find Symbol Results
Solution Explorer
Team Explorer
Class View
start
EN
19:20


Начальная страница - Microsoft Visual Studio
Быстрый запуск (Ctrl+Q)
Файл Правка Вид Отладка Команда Сервис Тест Анализ Окно Справка
Natalia Ivina
Присоединение...
Начальная страница
Обозреватель серверов
Панель элементов
Visual Studio
Узнайте о возможностях Visual Studio Community 2015
Пуск
Создать проек
Открыть проек
Открыть из сис
версиями...
Последние
Обозреватель решений
Создание проекта
Последние файлы
Установленные
Шаблоны
Visual C#
Windows
Android
Cloud
Extensibility
Silverlight
WCF
Workflow
Тест
Visual Basic
Visual F#
Visual C++
JavaScript
Python
TypeScript
Игра
Ускоритель сборки
Другие типы проектов
Примеры
В сети
.NET Framework 4.5.2
Сортировка по: По умолчанию
Установлено: Шаблоны — поиск (Ctr
Приложение Windows Forms Visual C#
Приложение WPF Visual C#
Консольное приложение Visual C#
Общий проект Visual C#
Библиотека классов (переносимая для iOS, Android и Wi... Visual C#
Библиотека классов Visual C#
Библиотека классов (переносимая) Visual C#
Приложение Silverlight Visual C#
Библиотека классов Silverlight Visual C#
Приложение службы WCF Visual C#
Получить пакет Microsoft Azure SDK для .NET Visual C#
Щелкните здесь для поиска шаблонов в Интернете.
Тип: Visual C#
Проект, для создания приложения с пользовательским интерфейсом Windows Forms
Имя: WindowsFormsApplication1
Расположение: c:\users\natai\documents\visual studio 2015\Projects
Обзор...
Team Explorer —...
Представление к...
ENG
15:04
08.02.2017

visual studi... | Compar... | Mono — Ви... | MonoDevel... | extern (спра... | c# - Where t... | [Solved] Ho... | C# произно... | Список вер... | Версии и за...

https://www.visualstudio.com/vs/compare/

Поиск

	Visual Studio Community	Visual Studio Professional	Visual Studio Enterprise	Visual Studio Test Professional	MSDN Platforms
Supported Usage Scenarios	●●●○	●●●●	●●●●	●●●●	●●●●
Debugging and Diagnostics	●●●○	●●●○	●●●●	○○○○	○○○○
Testing Tools	●○○○	●○○○	●●●●	●●○○	●●○○
Integrated Development Environment	●●●○	●●●○	●●●●	○○○○	○○○○
Development Platform Support	●●●●	●●●●	●●●●	○○○○	○○○○
Xamarin Mobile Development	●●○○	●●○○	●●●●	○○○○	○○○○
Architecture and Modeling	●○○○	●○○○	●●●●	○○○○	○○○○
Lab Management	○○○○	○○○○	●●●●	●●●●	●●●●
Team Foundation Server features	○○○○	●●●○	●●●●	●●●●	●●●●
Collaboration Tools	●●●●	●●●●	●●●●	●●●○	●●●○
Team Collaboration Benefits	Included with all subscriptions				
Subscriber Benefits	Included with annual cloud subscriptions and standard subscriptions				
Visual Studio Dev Essentials Benefits	Free for all developers [14]				

Feedback

ENG 14:57 08.02.2017