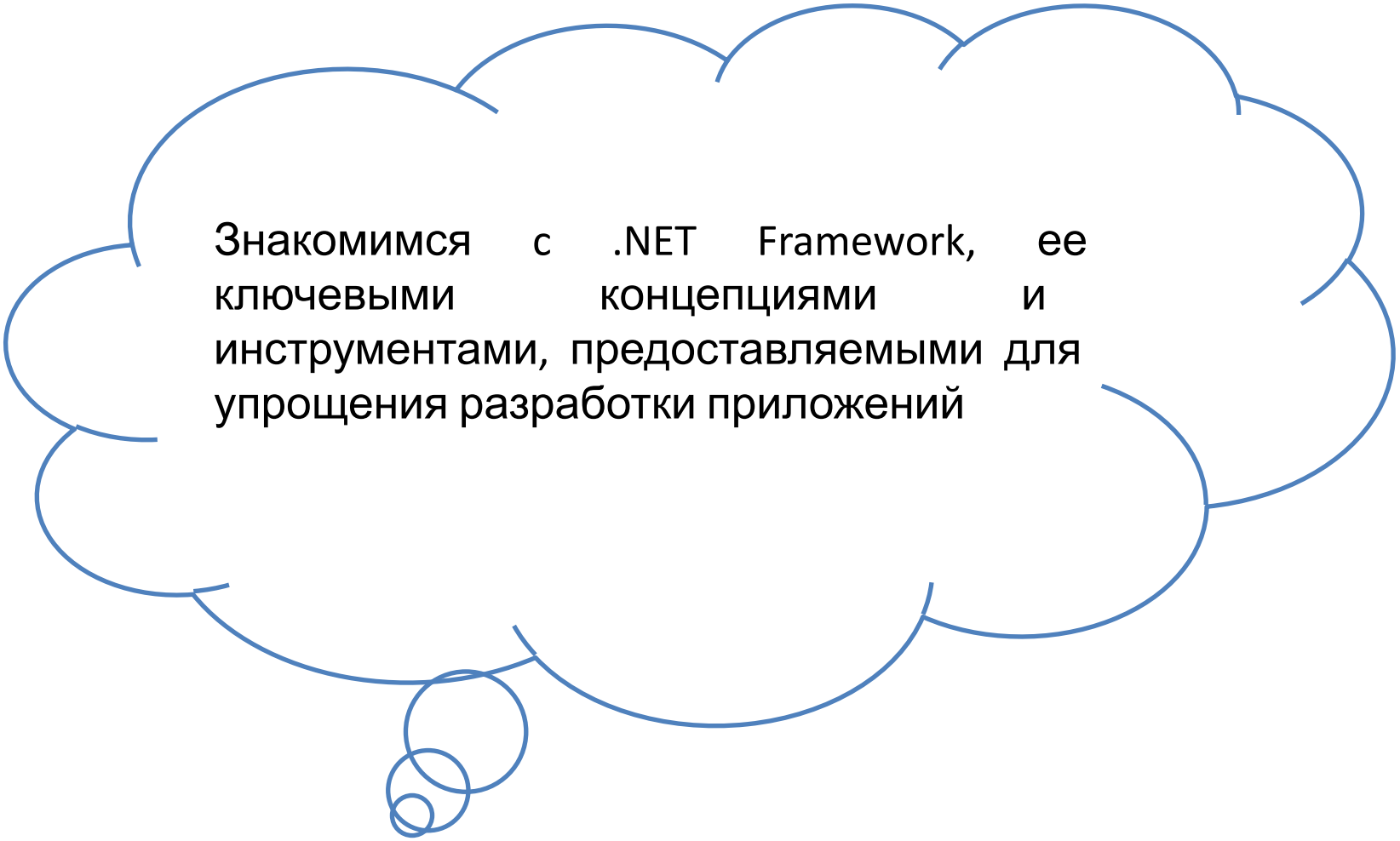
Курс “Языки программирования”

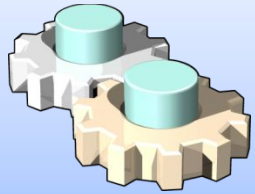
Лекция 1.
Архитектура платформы .Net

Знакомство с платформой .NET Framework

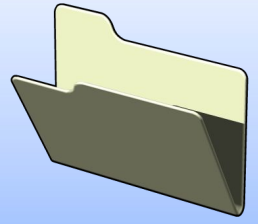


Платформа .NET Framework

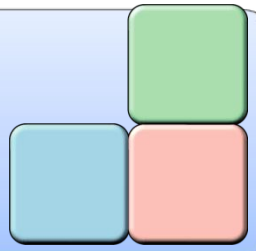
Среда CLR
(Common Language Runtime, CLR)



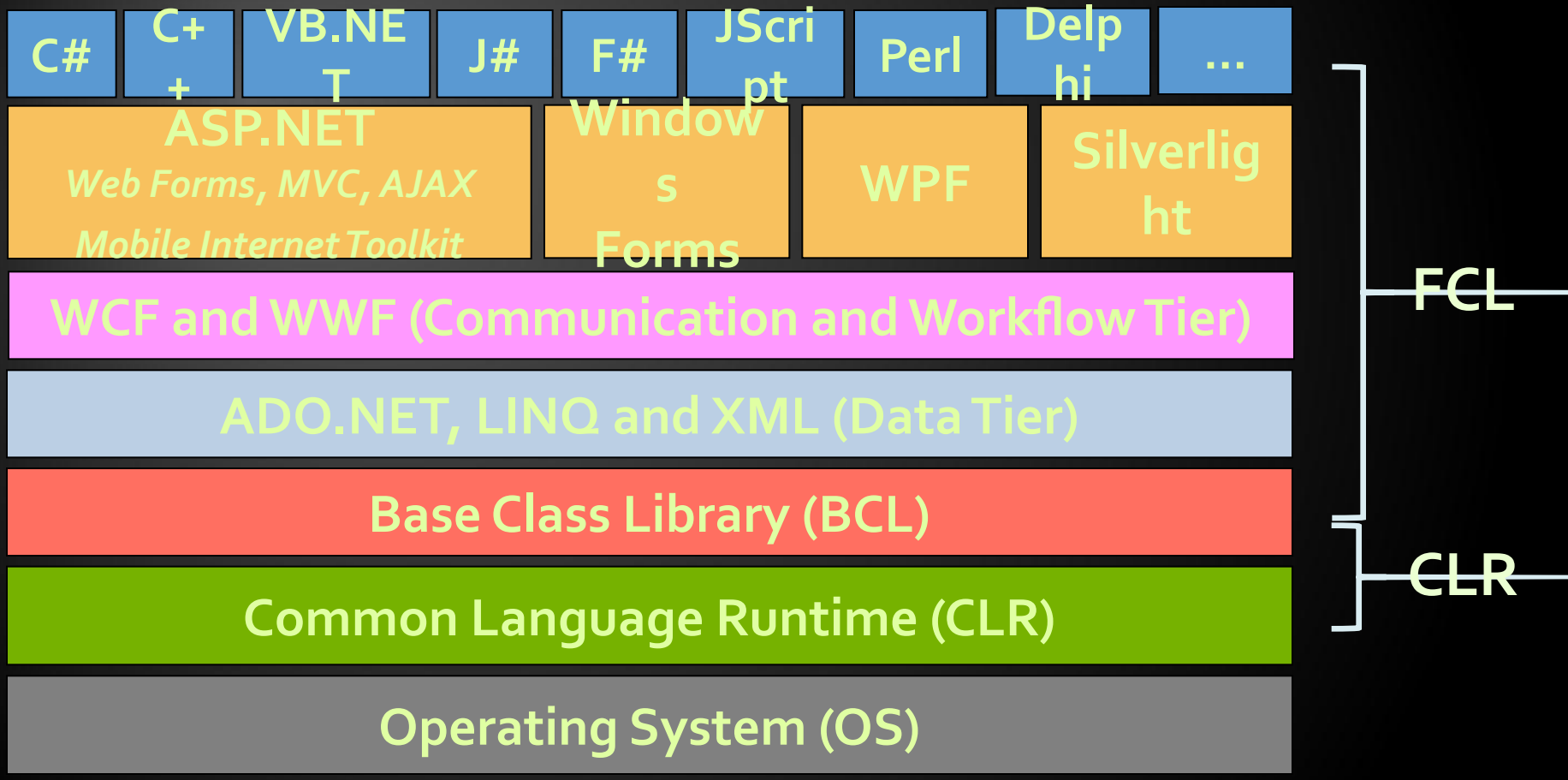
Библиотека классов .NET Framework
(.NET Framework Class Library)



Фреймворки для разработки приложений
(Development Frameworks)



Внутри .NET Framework



CLR

CLR (Common Language Runtime) - общеязыковая исполняющая среда или .NET runtime)

Managed code (управляемый код) - кода программы, исполняемой под «управлением» CLR

Компиляция кода в .NET выполняется в два этапа:

1. **Compilation of source code to Microsoft Intermediate Language (MSIL/IL).**
2. **Compilation of IL to platform-specific code by the CLR.**

Microsoft Intermediate Language is the key to providing many of the benefits of .NET.

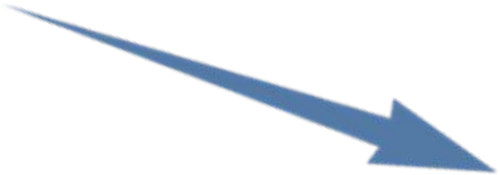
IL

Microsoft Intermediate Language - независимый от процессора набор инструкций, который можно эффективно преобразовать в машинный код.

IL

C# code

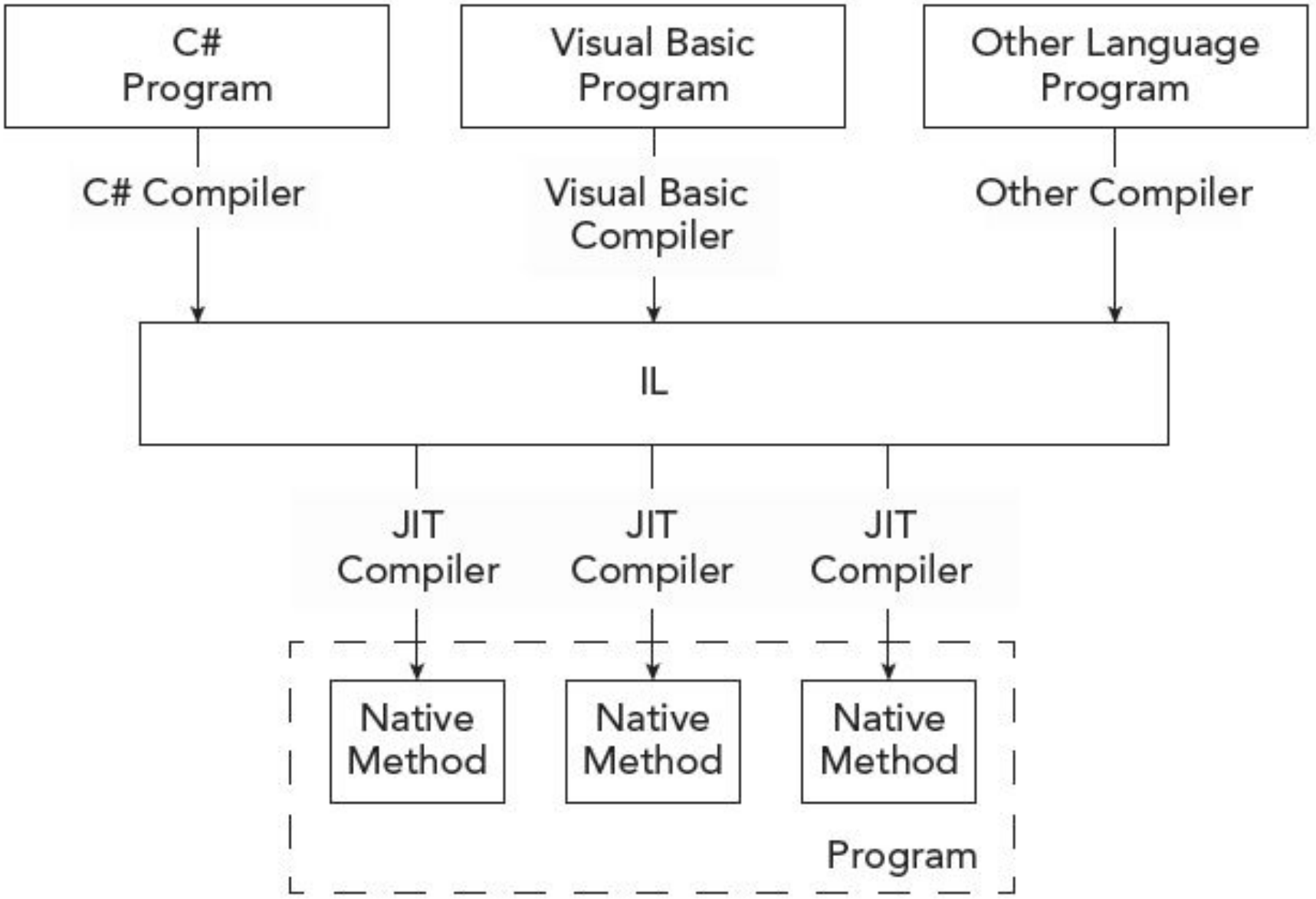
```
static void Main(string[] args)
{
  foreach (string arg in args)
Console.WriteLine(arg);
  Console.WriteLine("Press Enter to continue");
  Console.ReadLine();
}
```



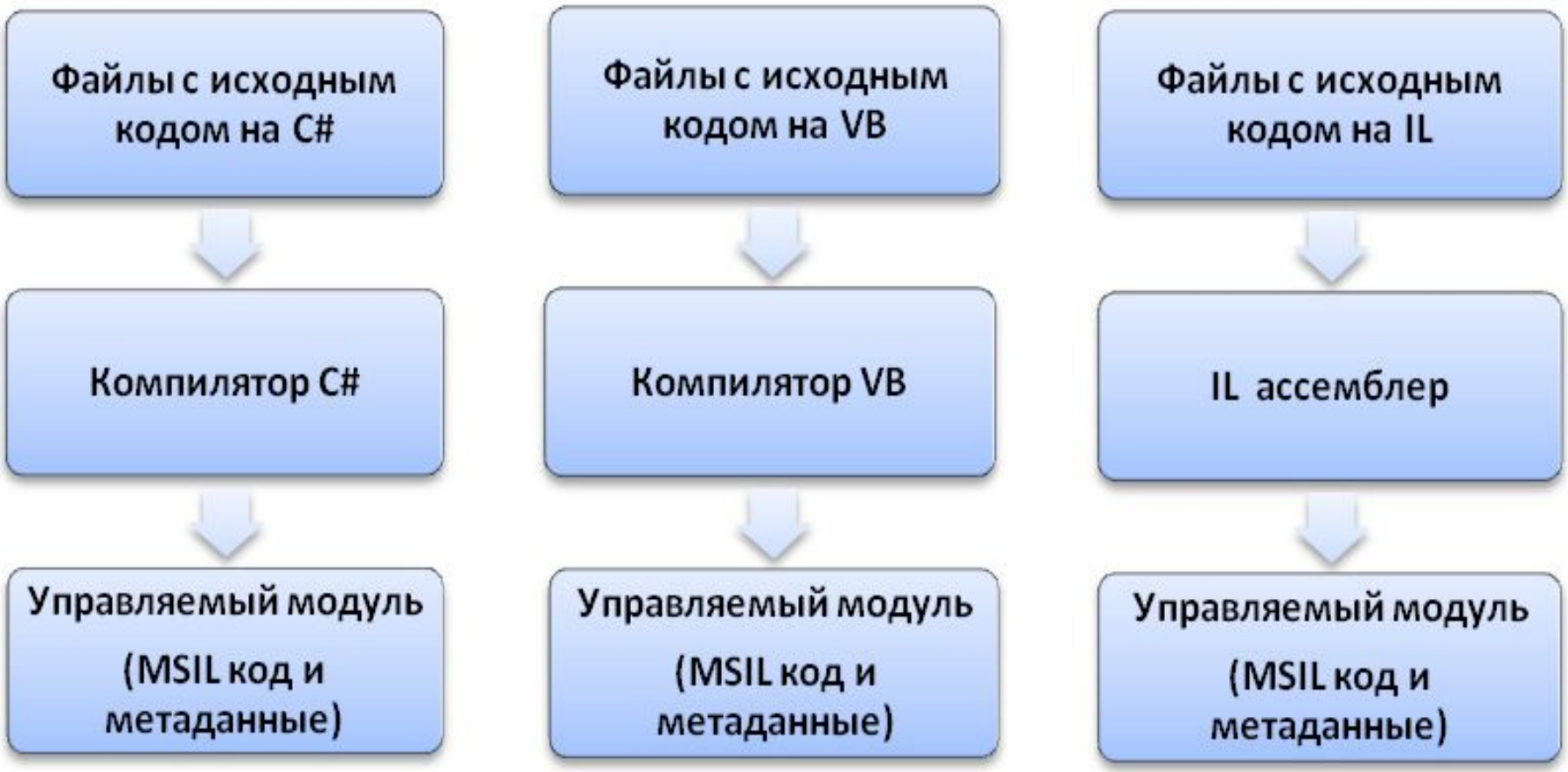
IL code

```
.method private hidebysig static void
Main(string[] args) cil managed
{
.entrypoint
// Code size 51 (0x33)
.maxstack 2
.locals init ([0] string arg,
[1] string[] CS$6$0000,
[2] int32 CS$7$0001,
[3] bool CS$4$0002)
IL_0000: nop
IL_0001: nop
IL_0002: ldarg.0
IL_0003: stloc.1
IL_0004: ldc.i4.0
IL_0005: stloc.2
IL_0006: br.s IL_0017
IL_0008: ldloc.1
IL_0009: ldloc.2
IL_000a: ldelem.ref
IL_000b: stloc.0
IL_000c: ldloc.0
IL_000d: call void
[mscorlib]System.Console::WriteLine(string)
IL_0012: nop
IL_0013: ldloc.2
…
```

CLR


C# Program
Visual Basic Program
Other Language Program
C# Compiler
Visual Basic Compiler
Other Compiler
IL
JIT Compiler
JIT Compiler
JIT Compiler
Native Method
Native Method
Native Method
Program

Управляемые модули, MSIL код и метаданные

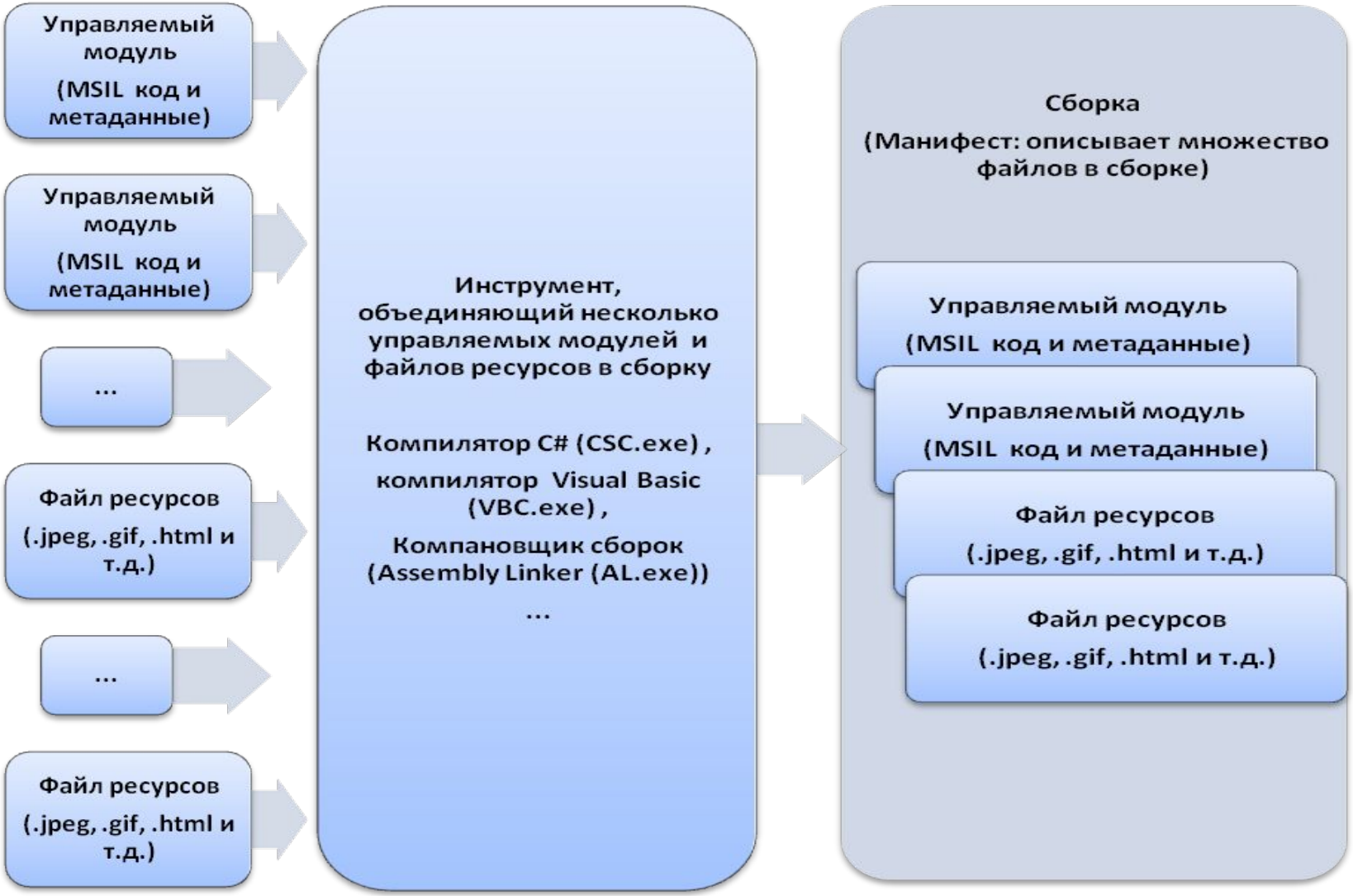


Сборки в .NET

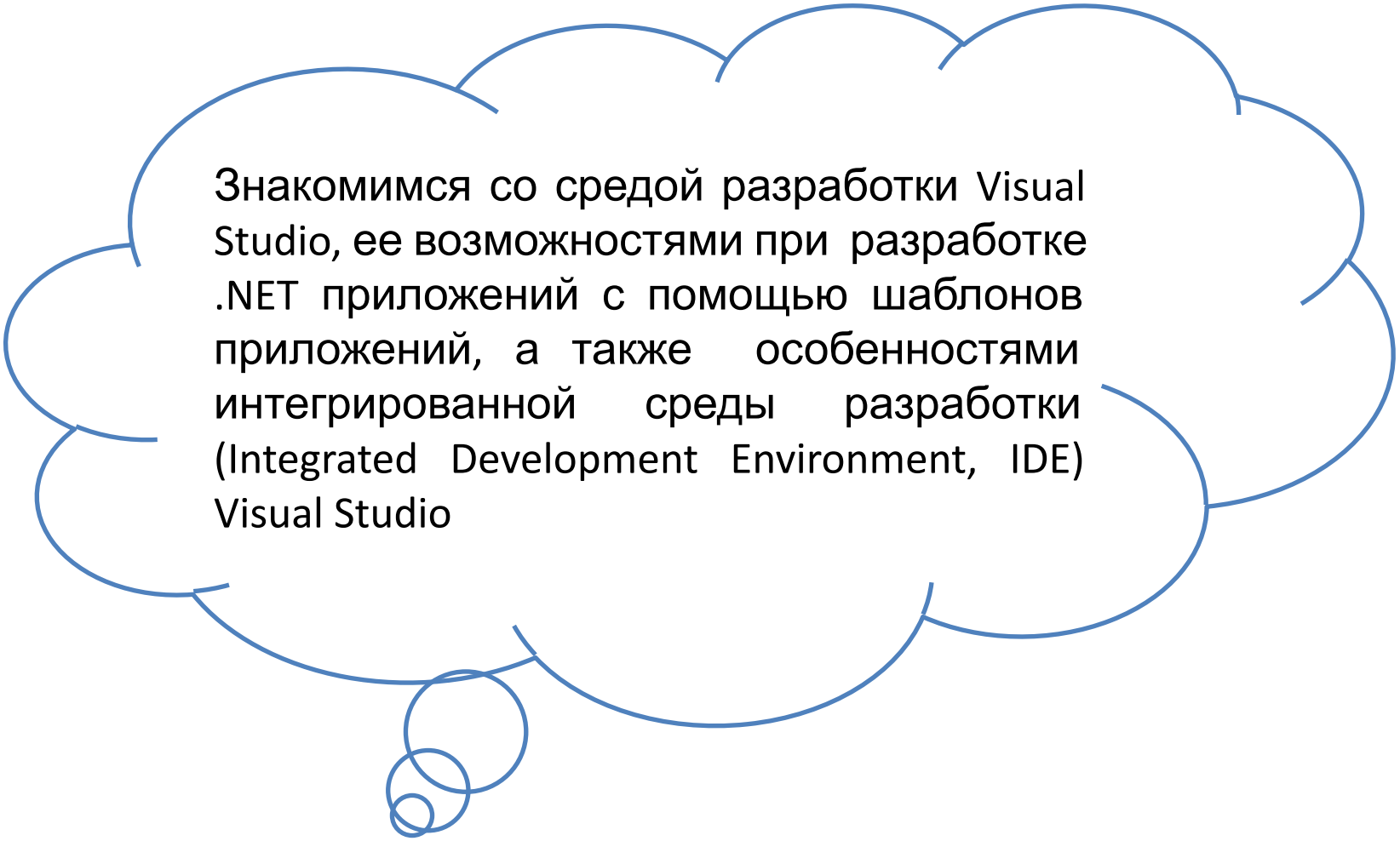
Логическая группировка одного или нескольких управляемых модулей и файлов ресурсов

Самая маленькая единица с точки зрения повторного использования, безопасности и управления версиями

Сборки в .NET


Управляемый модуль
(MSIL код и метаданные)
Управляемый модуль
(MSIL код и метаданные)
...
Файл ресурсов
(.jpeg, .gif, .html и т.д.)
...
Файл ресурсов
(.jpeg, .gif, .html и т.д.)
Инструмент, объединяющий несколько управляемых модулей и файлов ресурсов в сборку
Компилятор C# (CSC.exe),
компилятор Visual Basic (VBC.exe),
Компановщик сборок (Assembly Linker (AL.exe))
...
Сборка
(Манифест: описывает множество файлов в сборке)
Управляемый модуль
(MSIL код и метаданные)
Управляемый модуль
(MSIL код и метаданные)
Файл ресурсов
(.jpeg, .gif, .html и т.д.)
Файл ресурсов
(.jpeg, .gif, .html и т.д.)

Создание проектов в Visual Studio 2010



Основные возможности Visual Studio

- Интегрированная среда разработки (Integrated development environment, IDE)
- Быстрая разработка приложений (Rapid application development)
- Сервер и доспуп к данным (Server and data access)
- Возможности отладки (Debugging features)
- Обработка ошибок (Error handling)
- Справка и документация (Help and documentation)

Шаблоны в Visual Studio

Шаблоны:

обеспечивают стартовый код, который можно использовать для быстрого создания функционирующего приложения

включают поддержку компонентов и элементов управления, относящихся к типу проекта

обеспечивают настройку Visual Studio IDE согласно типу разрабатываемого приложения

обеспечивают добавление ссылки на любую начальную сборку, обычно требующуюся соответствующему типу приложения

Шаблоны в Visual Studio

Шаблон	Описание
Console Application	Предоставляет параметры среды, инструменты, ссылки на проекты и стартовый код для разработки приложения, выполняемое в интерфейсе командной строки. Этот тип приложения считается простым по сравнению с шаблоном приложения Windows Forms, потому что отсутствует графический интерфейс пользователя.
WPF Application	Предоставляет параметры среды, инструменты, ссылки на проекты и стартовый код для создания богатых графических приложений Windows. Приложения WPF позволяет создавать новое поколение приложений Windows, с гораздо большим контролем над дизайном пользовательского интерфейса.
Class Library	Предоставляет параметры среды, инструменты и стартовый код для построения .dll сборок. Этот тип файла можно использовать для хранения функциональностей, на которые можно ссылаться из других приложений.

Шаблоны в Visual Studio

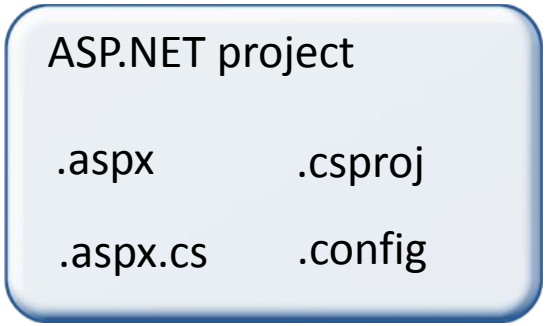
Шаблон	Описание
Windows Forms Application	Предоставляет параметры среды, инструменты, ссылки на проекты и стартовый код для построения графических приложений Windows Forms.
ASP.NET Web Application	Предоставляет параметры среды, инструменты, ссылки на проекты, и стартовый код для создания серверных, компируемых веб-приложений ASP.NET.
ASP.NET MVC Application	Предоставляет параметры среды, инструменты, ссылки на проекты и стартовый код, чтобы создать Model-View-Controller (MVC) веб-приложение. ASP.NET MVC веб-приложение отличается от стандартного веб-приложений ASP.NET тем, что архитектура приложения позволяет отделить уровень представления (presentation layer), слой бизнес-логики (business logic layer) и уровень доступа к данным (data access layer).

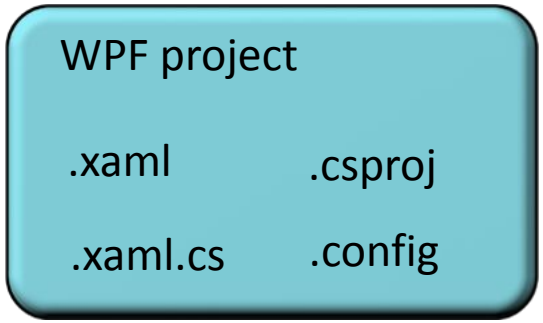
Шаблоны в Visual Studio

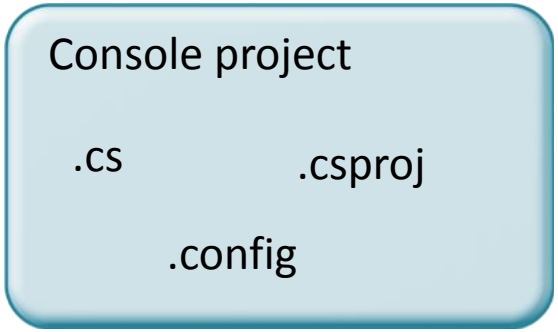
Шаблон	Описание
Silverlight Application	Предоставляет параметры среды, инструменты, ссылки на проекты и стартовый код для создания богатого графического веб-приложения.
WCF Service Application	Предоставляет параметры среды, инструменты, ссылки на проекты и стартовый код для построения Service Orientated Architecture (SOA) сервисов.

Структура проектов и решений Visual Studio

Visual Studio использует решения и проекты как концептуальные контейнеры для организации исходных файлов в процессе разработки. Классификация исходных файлов таким образом, упрощает компоновку и развертывание процесса для приложений .NET Framework







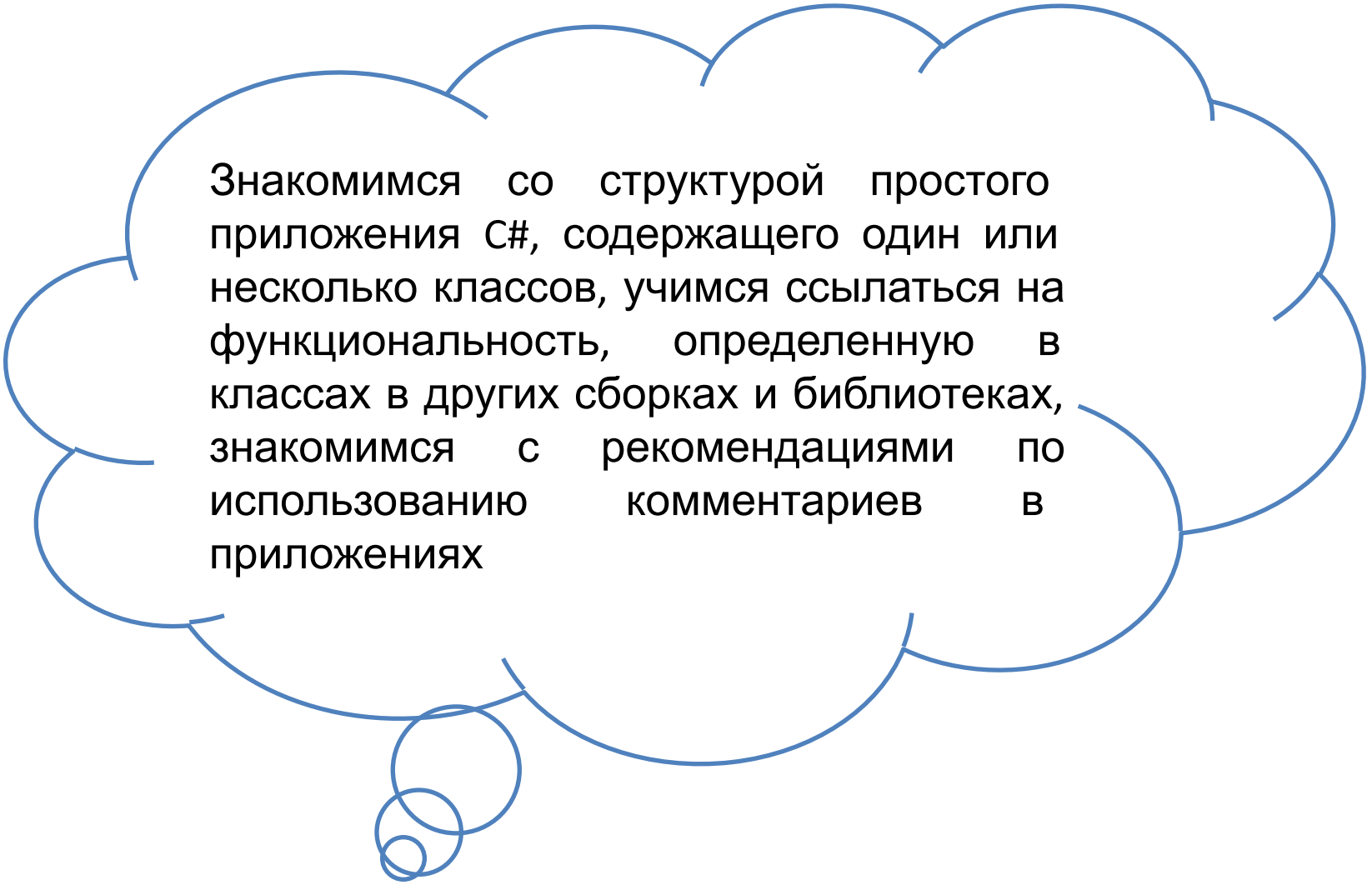
Структура проектов и решений Visual Studio

Файл	Описание
.cs	Файлы кода, которые могут принадлежать к одному проектному решению. Этот тип файла может быть одним из следующих: •модуль •файл Windows Forms •файл классов
.csproj	Файлы проекта, которые могут принадлежать к нескольким проектным решениям. Файл .csproj также хранит параметры проекта.
.aspx	Файлы, представляющие веб-страницы ASP.NET. Файл ASP.NET может содержать код Visual C# или использоваться сопровождающим .aspx.cs файлом для хранения кода в дополнение к разметке страницы.
.config	Файлы конфигурации - это XML-файлы, использующиеся для хранения настроек на уровне приложения, например, таких как строки подключения к базе данных, которые затем можно изменять без повторной компиляции приложения.
.xaml	XAML файлы используются в WPF и Silverlight Microsoft® приложениях для определения элементов пользовательского интерфейса.

Структура проектов и решений Visual Studio

Файл	Описание
.sln	Файл решения Visual Studio 2010, обеспечивающий единую точку доступа к нескольким проектам, элементам проекта и элементам решения. Файл .sln стандартный текстовый файл, который не рекомендуется изменять извне Visual Studio 2010.
.suo	Файл параметров пользователя решения, который хранит все настройки, которые изменяются при настройке Visual Studio 2010 IDE.

Написание приложений на C#



Классы и пространства имен

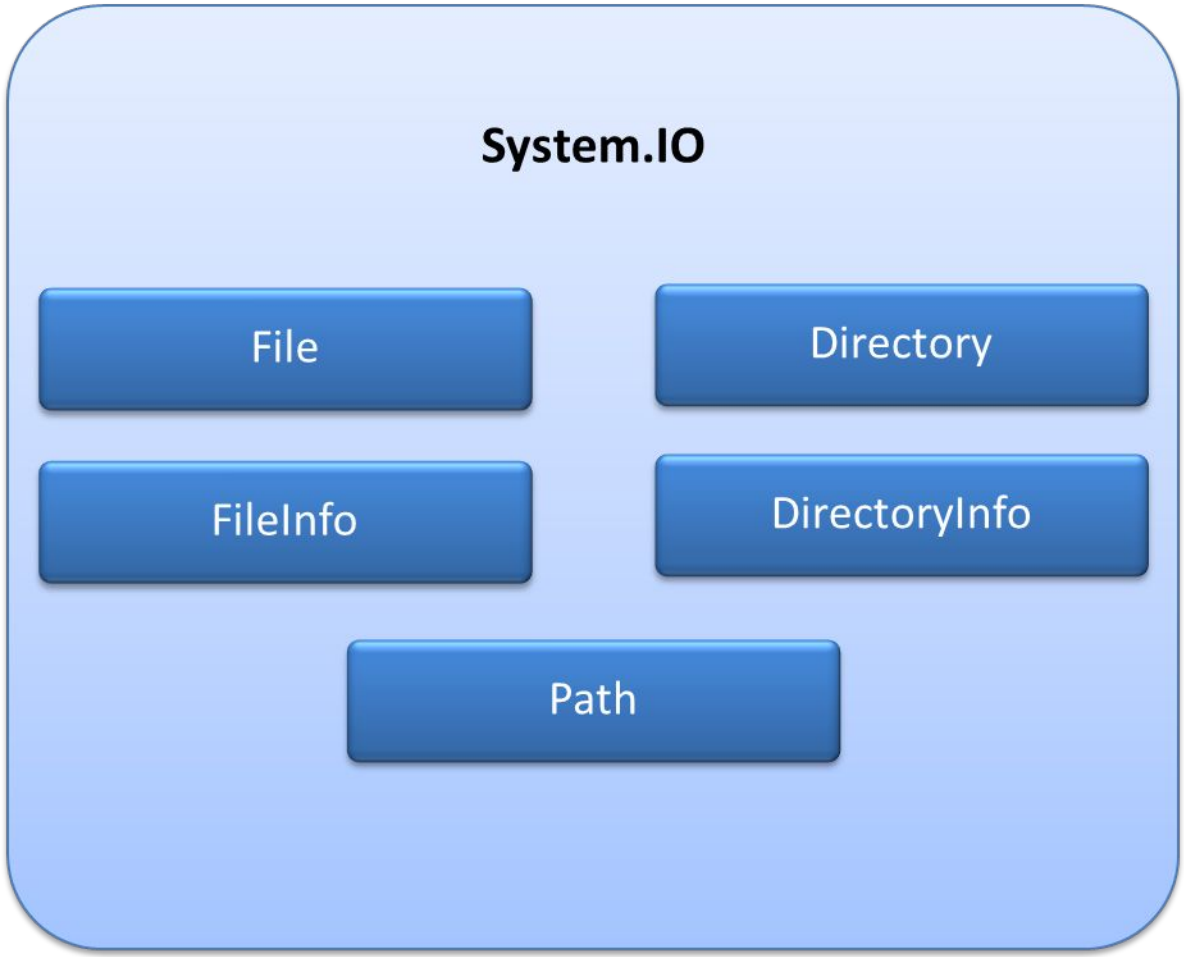
Класс

- Класс по существу чертеж, определяющий характеристики сущности, включает в себя свойства, определяющие типы данных, которые может содержать объект, и методы, описывающие поведение объекта
- Классы хранятся в сборках

Пространство имен

- Пространство имен представляет собой логический набор классов
- Пространство имен является средством для устранения неоднозначности классов, которые могут иметь одинаковые имена в различных сборках

Классы и пространства имен



Классы и пространства имен

Для использования класса, определенного в .NET Framework, следует выполнить следующие задачи:

Добавить ссылку на сборку, которая содержит скомпилированный код для класса

Импортировать пространство имен, которое содержит класс

```
using System;
using System.IO;
using System.Collections;
```

Структура консольного приложения

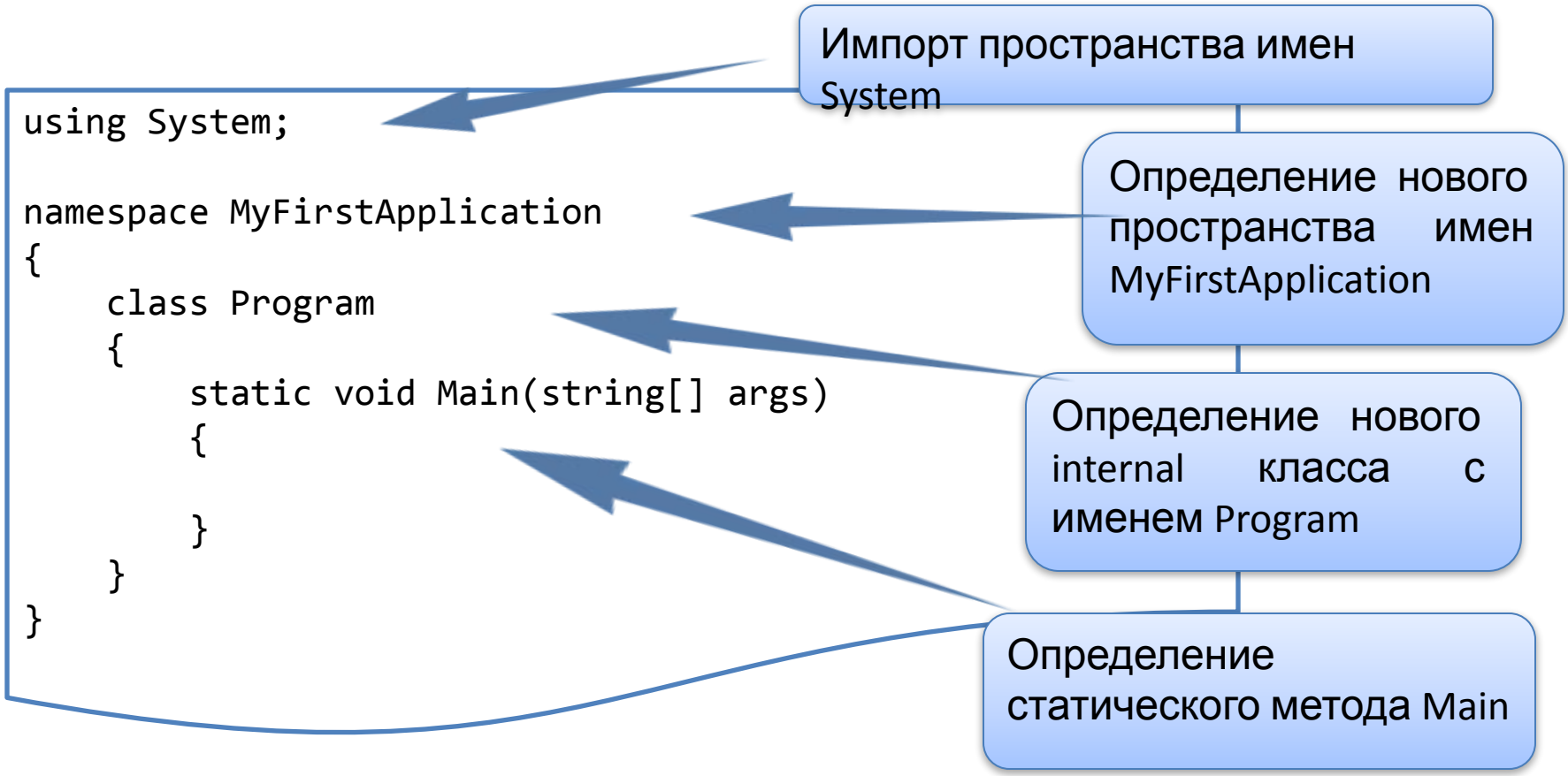
При создании нового консольного приложения с помощью шаблона Console Application, Visual Studio выполняет следующие задачи:

Создает новый файл с расширением .csproj для представления консольного проекта и структуры всех компонентов по умолчанию в консольном проекте

Добавляет ссылки на сборки библиотеки классов .NET Framework, которые обычно требуются консольным приложениям. Этот набор включает в себя сборку System

Создает файл Program.cs с методом Main, предоставляющий точку входа в консольное приложение

Структура консольного приложения



Выполнение ввода и вывода с использованием консольного приложения

```
using System;
...
Console.Clear();
```

```
using System;
...
int nextCharacter = Console.Read();
```

```
using System;
...
ConsoleKeyInfo key = Console.ReadKey();
```

```
using System;
...
string line = Console.ReadLine();
```

```
using System;
...
Console.Write("Hello there!");
```

```
using System;
...
Console.WriteLine("Hello there!");
```

Выполнение ввода и вывода с использованием консольного приложения

Метод	Описание
Clear()	Очищает окно и буфер консоли от данных. кода. using System; ... Console.Clear(); // clears the console display
Read()	Читает следующий символ из консоли. using System; ... int nextCharacter = Console.Read();
ReadKey()	Читает следующий символ или клавишу из окна консоли. using System; ... ConsoleKeyInfo key = Console.ReadKey();

Выполнение ввода и вывода с использованием консольного приложения

Метод	Описание
ReadLine()	Считывает следующую строку символов из окна консоли. using System; ... string line = Console.ReadLine();
Write()	Пишет текст в окне консоли. using System; ... Console.Write("Hello there!");
WriteLine()	Пишет текст в следующую строку в окне консоли. using System; ... Console.WriteLine("Hello there!");

Рекомендации по комментированию приложений C#

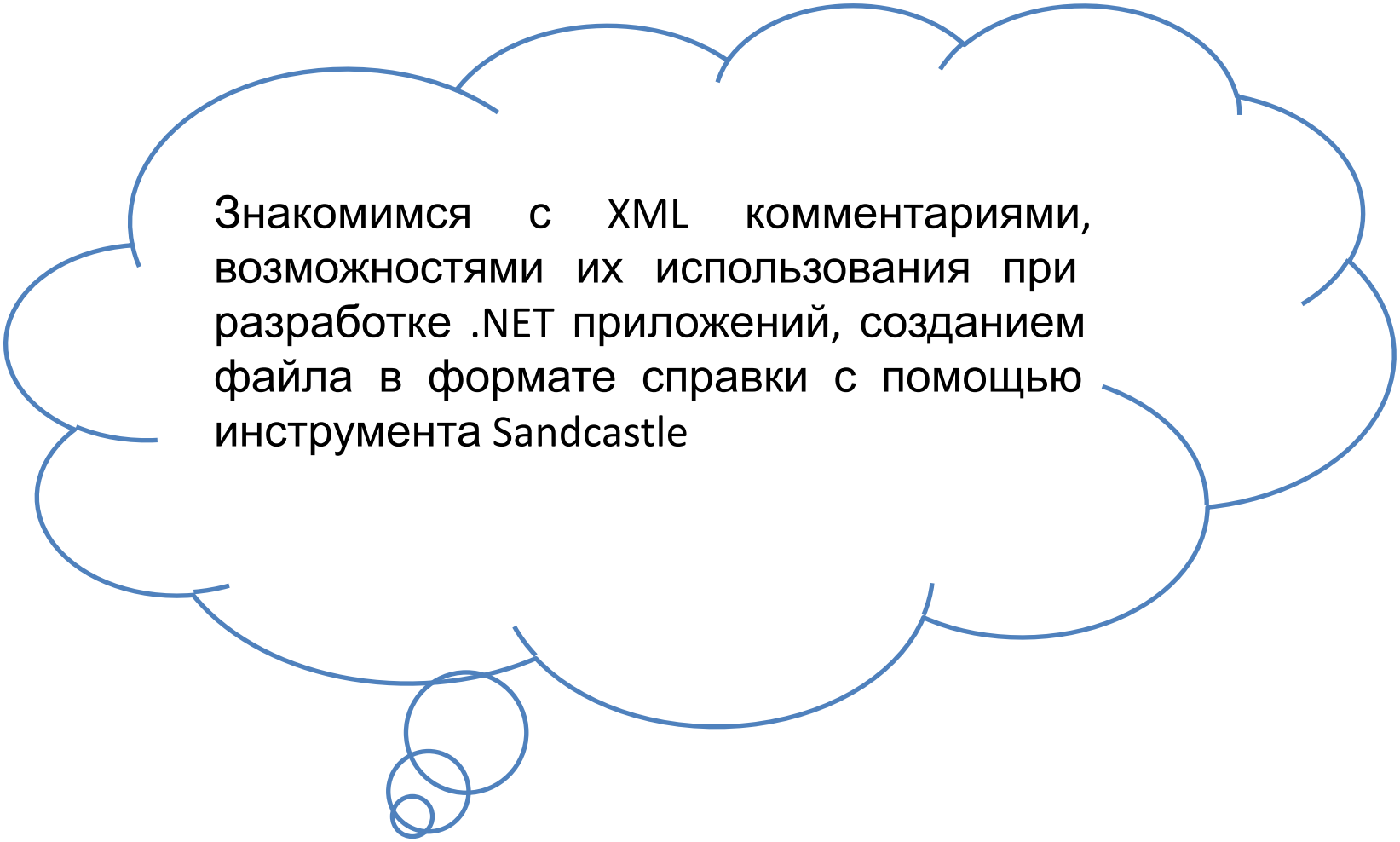
1. В начале процедуры следует поместить блок комментария, который должен включать информацию о цели процедуры, возвращаемом значении, аргументах и так далее

2. В длинных процедурах комментарии используются для того, чтобы выделить единицы работы в рамках процедуры

3. При объявлении переменных комментарии используются для указания того, как переменная будет использоваться

4. При написании структурного решения комментарии используются для указания, как решение будет выполнено и что оно означает

Документирование приложений



XML комментарии

В Visual Studio можно добавить комментарии к исходному коду, который будет обработан в XML файл

XML файл может быть включен в процесс создания справочной документации по классу или использован для поддержки IntelliSense

```
/// <summary> The Hello class prints a greeting on the screen
/// </summary>
public class Hello
{
    /// <summary> We use console-based I/O. For more information about
    /// WriteLine, see <seealso cref="System.Console.WriteLine()"/>
    /// </summary>
    public static void Main()
    {
        Console.WriteLine("Hello World");
    }
}
```

Общие теги XML комментариев

<summary> … </summary>

<remarks> … </remarks>

<example> … </example>

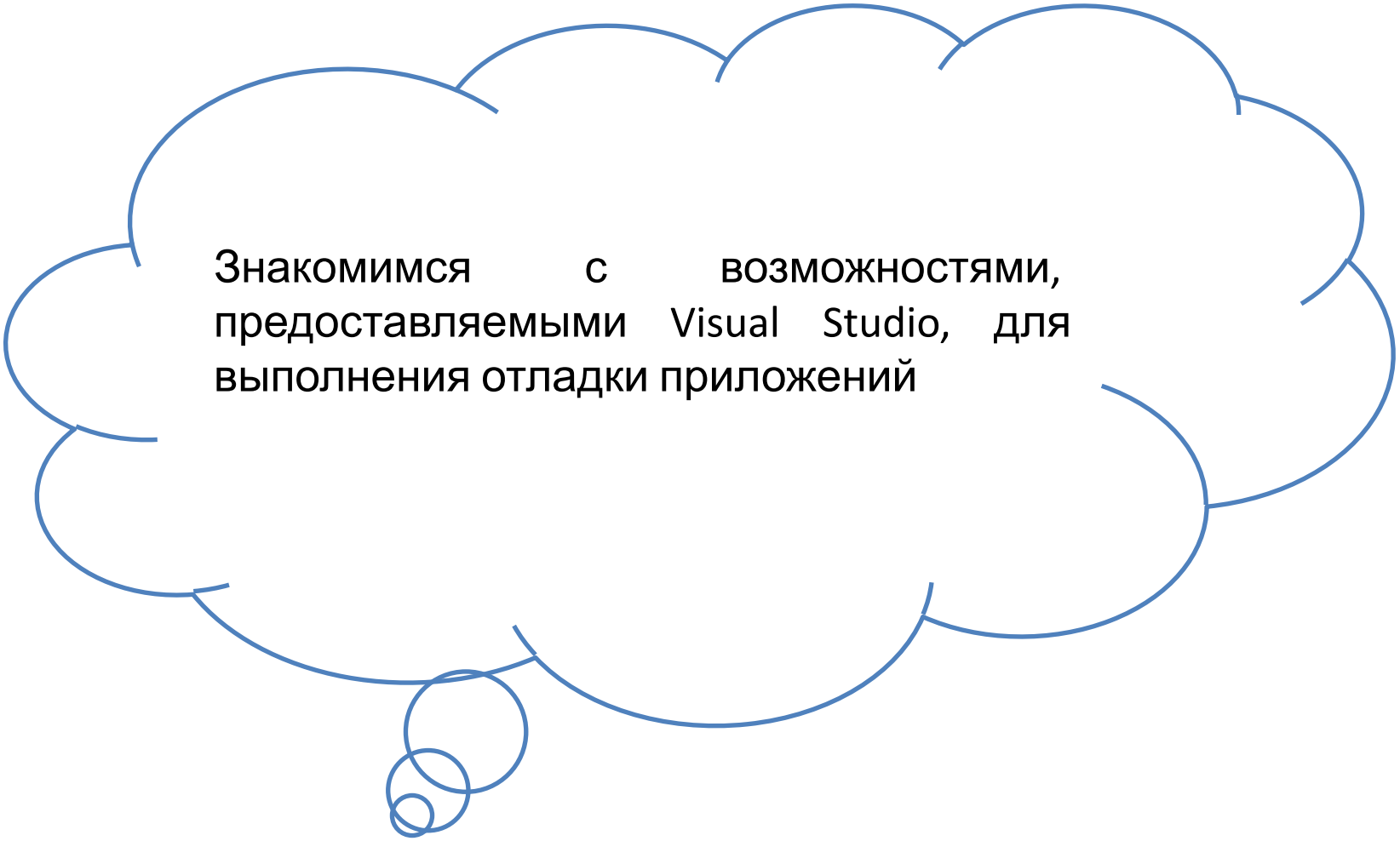
<code> … </code>

<returns> … </returns>

Общие теги XML комментариев

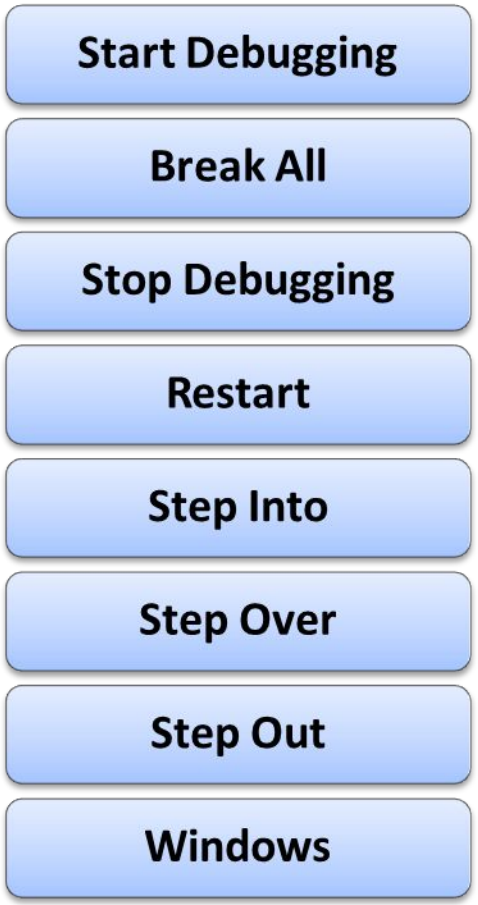
Тег	Назначение
<summary> … </summary>	Предоставляет краткое описание. Для более подробного описания используются теги <remarks>.
<remarks> … </remarks>	Содержит подробное описание. Этот тег может содержать вложенные разделы (пункты), списки и другие типы тегов.
<example> … </example>	Предоставляет пример того, как метод, свойство или другой член библиотеки должен быть использован. Этот тег часто связано с использованием вложенных тегов <code>.
<code> … </code>	Указывает, что прилагаемый текст является кодом приложения.
<returns> … </returns>	Документирует возвращаемое значение и тип метода.

Отладка приложений с помощью Visual Studio



Отладка в Visual Studio

Visual Studio предоставляет несколько инструментов, которые помогают выполнять отладку кода. Эти инструменты можно использовать во время разработки кода, во время тестовой фазы или после того, как приложение было выпущено



Отладка в Visual Studio

Опция меню	Кнопка панели инструментов	Клавиши быстрого доступа	Описание
Start Debugging	Start/continue	F5	Эта кнопка доступна, когда приложение не работает и находится в режиме приостановки. Кнопка запускает приложение в режиме отладки или возобновляет в случае режима приостановки.
Break All	Break all	CTRL+ALT+BREAK	Эта кнопка вызывает обработку приложения в режим pause и break. Кнопка доступна, когда приложение выполняется.
Stop Debugging	Stop	SHIFT+F5	Эта кнопка останавливает отладку. Она доступна, когда приложение работает или в режиме приостановки.

Отладка в Visual Studio

Опция меню	Кнопка панели инструментов	Клавиши быстрого доступа	Описание
Restart	Restart	CTRL+SHIFT+F5	Эта кнопка равносильна остановке, следующей за стартом, что приводит к перезапуску приложения с самого начала. Она доступна, когда приложение работает или в режиме приостановки.
Step Into	Step into	F11	Эта кнопка используется для пошагового выполнения кода.
Step Over	Step over	F10	Эта кнопка используется для пошагового выполнения кода.
Step Out	Step out	SHIFT+F11	Эта кнопка используется для пошагового выполнения кода.
Windows	Windows	Various	Эта кнопка обеспечивает доступ к различным окнам отладки, каждое из которых имеет свое собственное сочетание клавиш.

Использование точек останова

При запуске приложения в режиме отладки, можно приостановить выполнение и войти в режим прерывания (break mode)

В режиме прерывания можно:

- просматривать и изменять значения переменных
- выполнять дополнительный код
- вычислять выражения
- многое другое

Функции пошагового выполнения кода Through and Over Code

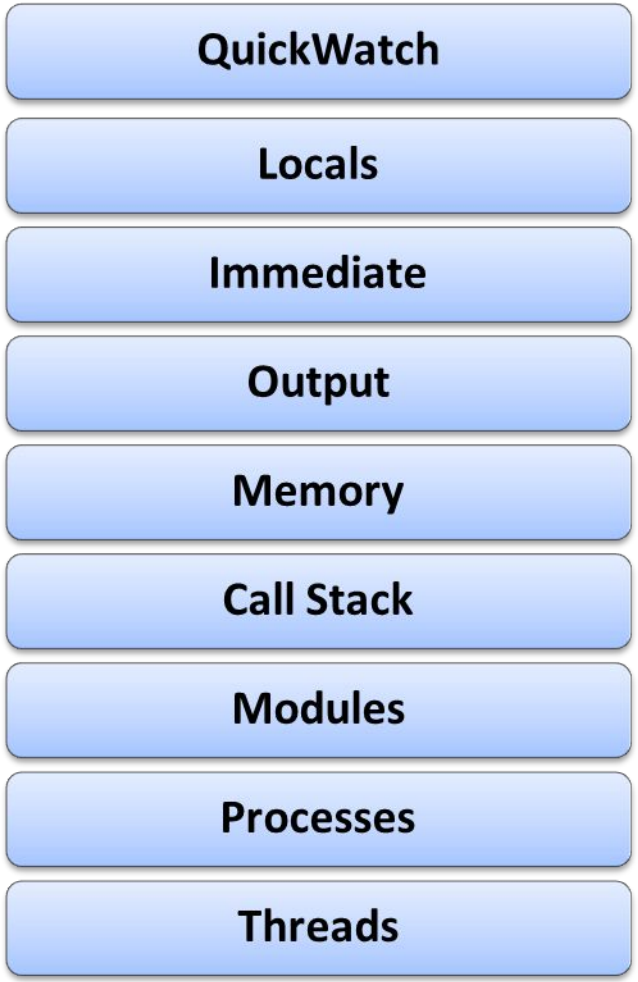
Код можно пошагово выполнять, чтобы увидеть, как именно происходит выполнение приложения

Существуют три функции отладки, которые необходимы для пошагового выполнения кода

- Step into
- Step over
- Step out

Использование Debug Windows

Visual Studio включает в себя несколько окон, которые можно использовать для отладки приложений



Использование Debug Windows

Окно	Описание
QuickWatch	Это модальное окно, которое позволяет вычислять переменные и выражения. Для его ипользования нужно набрать имена переменных или выражений в поле Expression, а затем нажать кнопку Reevaluate, чтобы посмотреть значение и тип переменной или результата выражения. Чтобы закрыть окно QuickWatch нужно нажать Close.
Locals	Это окно позволяет просматривать и редактировать локальные (в пределах области видимости) переменные. Можно расширить переменные, просмотреть члены и редактировать содержание некоторых переменных в столбцах Value..
Immediate	Это окно позволяет вычислять выражения, выполнять операторы и распечатывать значения переменных. Можно использовать это окно в контексте команды Visual Studio 2010 Debug.Print, чтобы печатать значения переменной или выражения.
Output	В этом окне можно просматривать ошибки и информационные сообщения. Одним из основных видов использования этого окна, является посмотр следов приложения с помощью метода System.Diagnostics.Debug.WriteLine().

Использование Debug Windows

Окно	Описание
Memory	Это окно позволяет просмотреть и отредактировать содержимое памяти, который использует приложение. Это функция может заставить приложение вести себя непредсказуемо, если только не использовать ее с осторожностью.
Call Stack	Это окно позволяет просматривать стек вызовов методов, которые используются для достижения текущего местоположения кода. Текущая позиция отображается в верхней части окна, а ниже показан ряд вызовов, которые обрабатываются приложением для достижения этого расположения.
Modules	Это окно позволяет просматривать информацию о модулях (сборках и исполняемых файлах), которые использует приложение. Каждый модуль перечисляется вместе с его местоположением, версией и другой информацией.
Processes	В этом окне можно просматривать информацию о процессах, к которым присоединен отладчик.
Threads	В этом окне можно проверять и контролировать потоки приложения.

Спасибо за внимание