## 2. Общая характеристика платформы MS.Net

## 2.1 Основные понятия

**Система программирования –** это язык программирования и совокупность программных средств, поддерживающих разработку и исполнение программ, написанных на этом языке.

Для выполнения программа должна быть загружена в **среду исполнения**.

Часть программы, которая хранится в одном файле, называется **модулем.** В простейшем случае вся программа хранится в одном файле. Имена файлов, как правило, назначает разработчик, а расширения файлов назначаются автоматически по правилам, принятым в среде исполнения.

Модуль, содержащий программу на языке высокого уровня, называется **исходным модулем.**

Модуль, содержащий программу в виде, готовом для загрузки в среду исполнения, называется **исполняемым модулем.**

Загрузке программы может предшествовать ряд преобразований, целью которых является приведение программы к виду, необходимому для загрузки в среду исполнения.

Различают две основные схемы преобразования исходного модуля в исполняемый модуль: **компиляциия и интерпретация**.

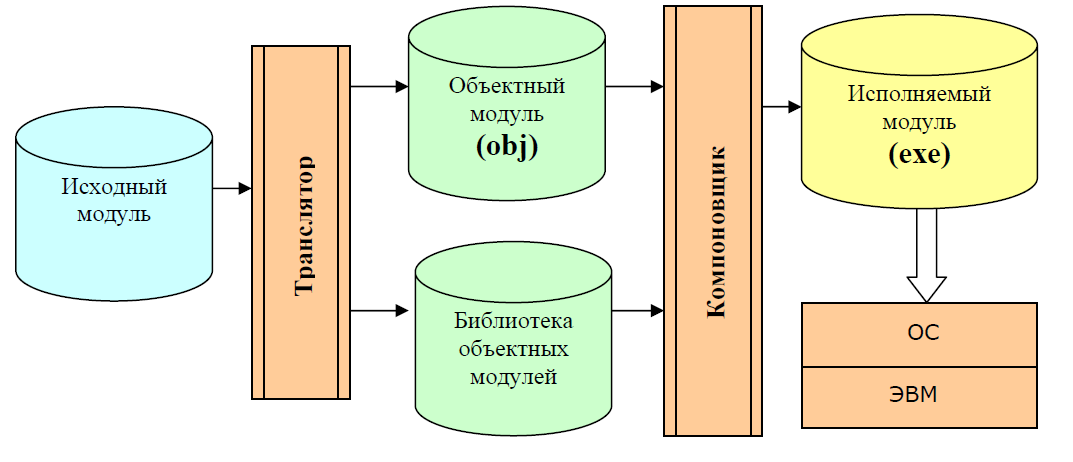


Рисунок 1 Схема компиляции

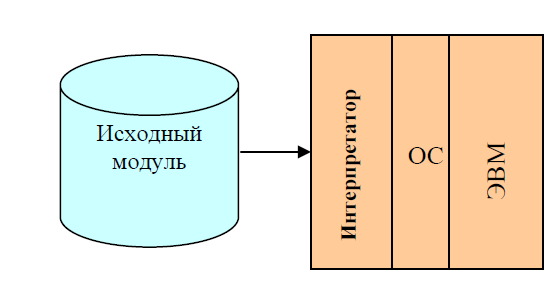


Рисунок 2 Схема интерпретации

При разработке программ обычно используются ранее созданные подпрограммы, которые хранятся **в библиотеке стандартных подпрограмм** в виде, пригодном для загрузки в среду исполнения. Подключение стандартных подпрограмм может выполняться в ходе выполнения программы (динамически компонуемые библиотеки) или предварительно до загрузки исполняемого кода в среду исполнения (статически компонуемые библиотеки). В последнем случае модуль, полученный транслятором, называют объектным модулем.

Подключение стандартных подпрограмм возлагается на специальную программу - **компоновщик (редактор связей).** Транслятор и компоновщик являются составными частями системы программирования.



Рисунок 3 Система программирования

Средства разработки, объединенные в систему на основе общего интерфейса и общей базы данных, образуют **среду программирования.**

## 2.2. Общая характеристика платформы MS.Net

Платформа MS.Net предназначена для разработки и исполнения приложений различных типов:

* автономное консольное приложение с использованием текстового интерфейса пользователя;
* автономное Windows-приложение с использованием графического интерфейса пользователя;
* библиотека классов, которые предназначены для использования в других приложениях;
* Web-приложение, доступ к которому выполняется через браузер и которое по запросу формирует Web-страницу и отправляет ее клиенту по сети;
* Web-сервис – компонент, методы которого могут вызываться через Интернет.

## 2.3. Выполнение программы в .NET

MSIL или IL (Microsoft Intermediate Language ) – промежуточный язык, который не содержит команд, зависящих от языка, ОС и типа компьютера.

CLR (Common Language Runtime) – общеязыковая среда выполнения, выполняет программу на языке IL. Может быть реализована для любой ОС.

JIT (just in time) – компилируются только те части программы, которые нужно выполнить в данный момент.

Сборка – файл с расширением exe или dll, который содержит код на языке IL и метаданные.

Метаданные – сведения об объектах, используемых в программе и самой сборке.

Управляемый код - исходный код должен быть переведен на специально разработанный для платформы промежуточный язык MSIL (MS Common Intermediate Language, CIL). Для исполнения кода на промежуточном языке приложения используется специальная программная компонента платформы – общеязыковая среда исполнения CLR(Common Language Runtime).

Небезопасный код - исходный код должен быть переведен на язык машинных команд. Машинный код исполняется непосредственно под управлением операционной системы.

CLR

Рисунок 4. Выполнение программы в .NET

Исходные модули транслируются на промежуточный язык MSIL. Код на промежуточном языке рассматривается средой исполнения CLR как исполняемый модуль. Среда исполнения представляет собой программу, надстроенную над операционной системой Windows и выполняемую под ее управлением. C другой стороны, среда исполнения представляет собой функциональный аналог ЭВМ, - виртуальную машину, в которой выполняются программы на промежуточном языке.

При вызове метода среда исполнения активизирует транслятор JIT, который переводит код метода с промежуточного языка в машинный код и сохраняет его в памяти. При повторном вызове метода повторная трансляция не выполняется, используется машинный код, сохраненный в памяти. За счет такого подхода достигается эффективность, соизмеримая с эффективностью неуправляемого кода и экономия расхода основной памяти за счет перевода в машинный код не всей программы, а только тех методов, которые были реально вызваны в процессе выполнения программы.

Таким образом, каркас платформы образуют две компоненты:

* Статическая компонента – **базовая библиотека классов**, содержащая обширный набор готовых к использованию программных компонент на промежуточном языке. Базовая библиотека классов является общей для всех языков программирования, поддерживаемых в платформе.
* Динамическая компонента – общеязыковая среда исполнения (CLR).

Интегрированная среда разработки MS Visual Studio.NET представляет собой программную компоненту, поддерживающую процесс разработки программ. С помощью средств MS Visual Studio.NET выполняется

* редактирование исходного кода,
* выполнение приложения в отладочном режиме,
* визуальное отображение логической структуры приложения,
* выдача справочной информации по самой среде, платформе и языкам программирования, что является традиционным для большинства интегрированных сред,
* возможность автоматической генерации исходного кода по визуальному представлению диаграммы классов в нотации UML и наоборот, построение визуального представления диаграммы классов по исходному коду программы.



Рисунок 5 Структура консольного приложения в среде MS VS.Net

Приложение в процессе разработки называют **проектом.** Проект логически объединяет все необходимые для создания приложения файлы. Описание структуры проекта хранится в специальном файле с расширением csproj.

Несколько проектов логически могут быть объединены в одно **решение**. Структура решения хранится в файле с расширением sln. С помощью инспектора решения Solution Explorer, входящего в состав интегрированной среды можно просматривать логическую структуру всех проектов, включенных в решение и выполнять операции по изменению логической структуры решения и проектов.

## 2.4. Общая система типов CTS

**CTS (общая система типов)** представляет собой формальную спецификацию, в которой описано то, как должны быть определены типы для того, чтобы они могли обслуживаться в [CLR-среде](http://professorweb.ru/my/csharp/charp_theory/level1/1_4.php). Внутренние детали CTS обычно интересуют только тех, кто занимается разработкой инструментов и/или компиляторов для платформы .NET. Т.е. CTS описывает целую развитую иерархию типов, включающую хорошо определенные точки, в которых код может определять свои собственные типы. Иерархическая структура общей системы типов (CTS) отражает объектно-ориентированную методологию одиночного наследования IL и показана на рис. 6.

Рисунок 6. Иерархическая структура общей системы типов (CTS)

Абсолютно всем .NET-программистам важно уметь работать на предпочитаемом ими языке с пятью типами из CTS.

1. **Типы классов.** В каждом совместимом с .NET языке поддерживается, как минимум, понятие типа класса (class type), которое играет центральную роль в [объектно-ориентированном программировании (ООП)](http://professorweb.ru/my/glossariy/glossy/files/100.php#o1). Каждый класс может включать в себя любое количество методов/функций (таких как конструкторы, свойства, методы и события) и полей/данных.
2. **Интерфейсы** представляют собой именованный набор определений абстрактных методов, которые могут реализоваться в данном классе или структуре.
3. Понятие **структуры** аналогично понятию структуры в С и других ЯВУ, и представляет собой "облегченный" тип класса с основанной на использовании значений семантикой.
4. **Перечисления** представляют собой программную конструкцию, которая позволяет группировать данные в пары "имя-значение". По умолчанию для хранения каждого элемента выделяется блок памяти, соответствующий 32-битному целому, однако при необходимости (например, при программировании с расчетом на устройства, обладающие малыми объемами памяти, вроде мобильных устройств Windows) это значение можно изменить.
5. **Делегаты**являются .NET-эквивалентом безопасных в отношении типов указателей функций в стиле С. Делегаты очень удобны, когда требуется обеспечить один объект возможностью перенаправлять вызов другому объекту и образовывать основу для архитектуры обработки событий .NET.

В CTS содержится четко определенный набор фундаментальных типов данных. Хотя в каждом отдельно взятом языке для объявления того или иного встроенного типа данных из CTS обычно предусмотрено свое уникальное ключевое слово, все эти ключевые слова в конечном итоге соответствуют одному и тому же типу в сборке mscorlib.dll.

|  |  |
| --- | --- |
| **Классы типов данных C#** | |
| Типы данных в CTS | Ключевое слово в C# |
| System.Byte | byte |
| System.SByte | sbyte |
| System.Int16 | short |
| System.Int32 | int |
| System.Int64 | long |
| System.UInt16 | ushort |
| System.UInt32 | uint |
| System.UInt64 | ulong |
| System.Single | float |
| System.Double | double |
| System.Object | object |
| System.Char | char |
| System.String | String |
| System.Boolean | bool |