Объектно-ориентированное программирование

...

Мущук Артур Николаевич

кафедра Программной инженерии ауд. 206-1

Необходимый материал

1) Лекции
2) Книги
3) Материалы

   https://diskstation.belstu.by:5001/

1. Знакомство с .NET Framework
2. Типы данных. Базовые синтаксические конструкции
3. Операторы
4. Принципы ООП
5. Понятия класса и объекта, поля и метода. Модификаторы доступа.
6. Наследование классов. Абстрактные классы.
7. интерфейсы
8. Структуры
9. Исключения
10. Обобщения
11. Делегаты, лямбды и события
12. Коллекции
13. LINQ
14. Рефлексия
15. Сериализация

.NET, CLR, C#

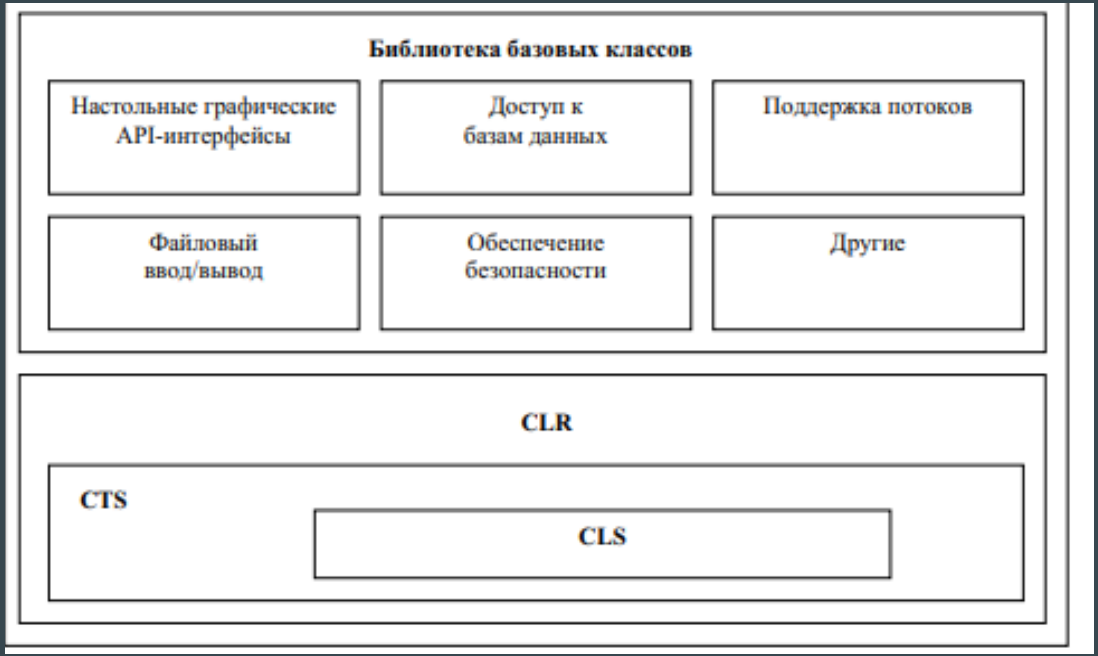
.NET FRAMEWORK

.Net Framework – платформа, которая создана ***Microsoft*** для разработки приложений

Платформа — в общем смысле, это любая существующая среда разработки и выполнения

.NET FRAMEWORK. Структура:

- общеязыковая исполняющая среда (CLR) Common Language Runtime) Виртуальная Машина. Обеспечивает выполнение сборки (управление памятью, загрузка сборок, безопасность, обработка исключений, синхронизация)
- библиотека классов (FCL). (.NET Framework Class Library) спецификации объектно-ориентированной библиотеки классов, интерфейсов и системы типов (типов-значений)соответствующая CLS



.NET Framework Class Library

Веб-службы.

Приложения Web Forms/MVC

Приложения WPF/WinForms

Консольные приложения Windows

Приложения ASP.NET

Службы Windows;

.NET Framework Class Library

System - все базовые типы, исп. в приложении
System.Data - типы для взаимодействия с БД
System.IO - типы ввода/вывода, обхода потока файлов
System.Text - типы для работы с разными кодировками

.NET FRAMEWORK – решение следующих проблем

- **1. Интеграция языков программирования.**
- **CLS (Common Language Specification)** – общеязыковая спецификация, представляет собой набор правил, которые во всех подробностях описывают минимальный и полный комплект функциональных возможностей, которые должен обязательно поддерживать каждый отдельно взятый .NET-компилятор
- **CTS (Common Type Systems)**- спецификацию типов, которые должны поддерживаться всеми языками ориентированными на CLR. Описывает все, что касается определения и поведения полей, методов, свойств, событий и т.д.

2. Работа на многих платформах.

- При компиляции кода компиляторы NET Framework генерируют код на промежуточном языке (**CIL**, **Common Intermediate Language**). При исполнении CLR транслирует CIL-код в команды соответствующего процессора.

- **3. Упрощенное повторное использование кода.**

- CLR позволяет сборки разработанные на одном языке использовать в других языках.

- **4. Автоматическое управление памятью.**

- CLR автоматически отслеживает использование ресурсов. Сборщик мусора.

5. Проверка безопасности типов.

- При работе в CLR практически исключена возможность записать (стереть) данные в область памяти, которая для этого не предназначена. Нет возможности передать управление в произвольную точку.

6. Единый принцип обработки сбоев.

- Один из самых неприятных моментов в Window-программирование – это отсутствие единой системы обработки ошибок и сбоев: возврат функций, коды состояний, исключения и т.п. Для обработки ошибок и сбоев в CLR используется только механизм исключений.

- 7. **Взаимодействие с существующим кодом.**

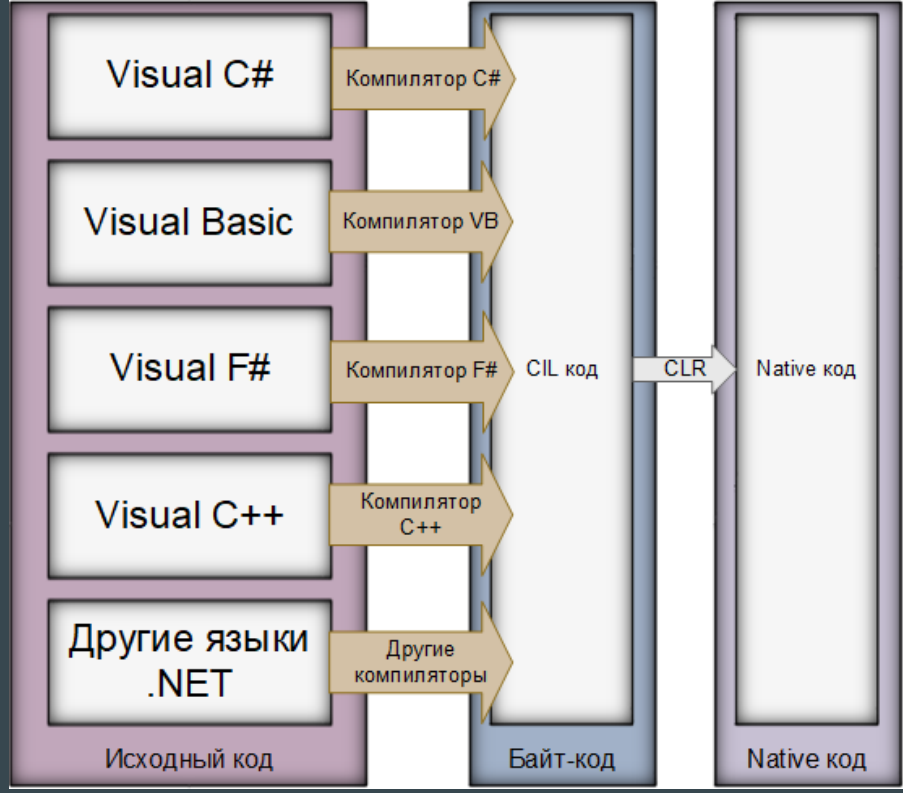
- Поддерживаются функции Win32 DLL – библиотек.

8. Проблемы с версиями.

- В Windows возникают проблемы связанные с совместимостью DLL-библиотек. .NET Framework приложение всегда работает с компонентами с которыми компилировалось и тестировалось приложение.

CLR
Common Language Runtime

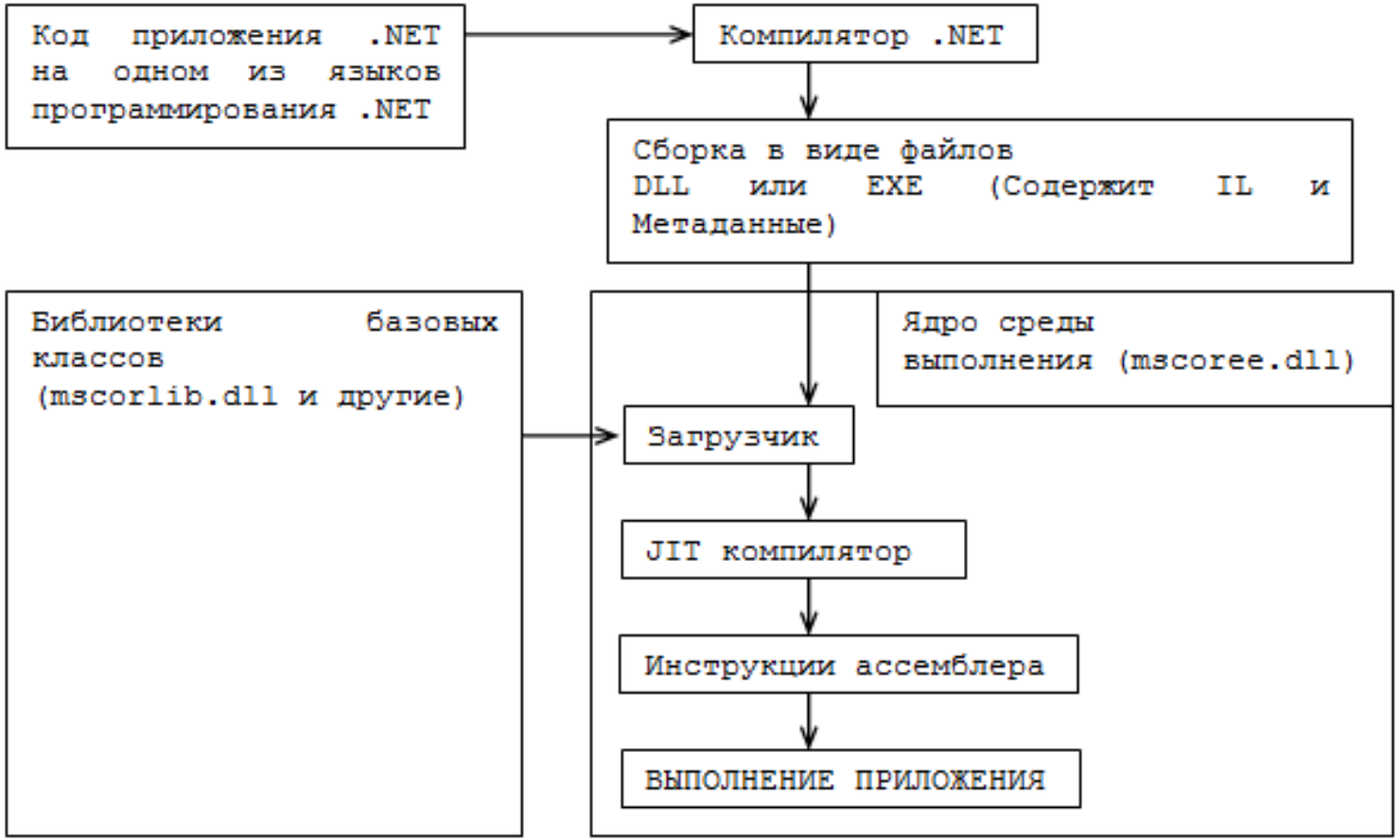
это исполняющая среда для выполнения IL (промежуточного кода).



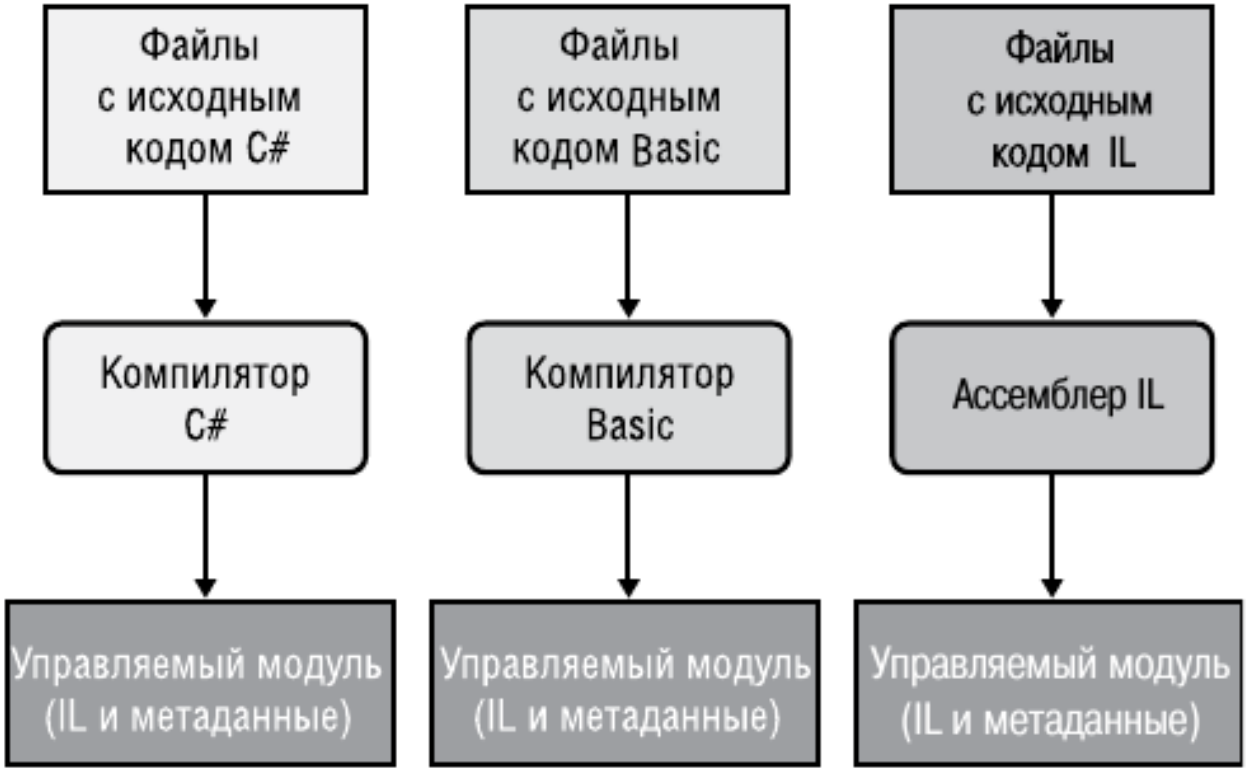
Функции:

- управление памятью;
- выполнение потоков;
- выполнение кода;
- проверка безопасности кода;
- управление системными службами и т.д.

Структура среды выполнения CLR



Компиляция исходного кода в управляемые модули



assembly

Управляемый модуль - portable executable (PE)

.Части управляемого модуля

Заголовок PE32 или PE32+ тип файла: GUI, CUI, DLL;
Заголовок CLR
Метаданные
Код Intermediate Language (IL)

Метаданные

двоичный набор таблиц данных:

типы и их члены

портируемые типы и их члены

Назначение:

1)устраняют необходимость в заголовочных файлах (прототипы);

2) используются в VS для подсказок;

3) используется при верификации кода на предмет безопасных операций;

4) можно сериализовать объект одной машине и восстановить состояние объекта на другой машине;

5) используются при сборке мусора.

Манифест

Манифест - ключевой компонент сборки.

Он хранит в себе:

- Имя сборки
- Номер версии
- Язык и региональные стандарты
- Список всех файлов сборки
- и т.д

```
// Metadata version: v2.0.50727
.assembly extern mscorlib
{
  .publickeytoken = (B7 7A 5C 56 19 34 E0 89 )
  .ver 2:0:0:0
}
.assembly extern System
{
  .publickeytoken = (B7 7A 5C 56 19 34 E0 89 )
  .ver 2:0:0:0
}
.assembly extern System.Windows.Forms
{
  .publickeytoken = (B7 7A 5C 56 19 34 E0 89 )
  .ver 2:0:0:0
}
.assembly extern System.Xml
{
  .publickeytoken = (B7 7A 5C 56 19 34 E0 89 )
  .ver 2:0:0:0
}
.assembly extern System.Drawing
{
  .publickeytoken = (B0 3F 5F 7F 11 D5 0A 3A )
  .ver 2:0:0:0
}
.assembly extern System.Core
{
  .publickeytoken = (B7 7A 5C 56 19 34 E0 89 )
  .ver 3:5:0:0
}
.assembly 'БУ_БКУ'
{
  .custom instance void [mscorlib]System.Runtime.CompilerServ

  .custom instance void [mscorlib]System.Reflection.AssemblyP
```

Сборки

- *Сборка* (assembly) — 1) это абстрактное понятие, для логической группировки одного или нескольких управляемых модулей или файлов ресурсов.
- 2) дискретная единица многократно используемого кода внутри CLR

Exe, dll

Сборка определяет следующие сведения:

1. **Код, выполняемый средой CLR.**

   Имейте в виду, что каждая сборка может иметь только одну точку входа: DllMain, WinMain или Main.

Сборка определяет следующие сведения:

2.Граница безопасности.

Сборка представляет собой единицу, для которой запрашиваются и предоставляются разрешения.

Сборка определяет следующие сведения:

3. **Граница типа**. Каждое удостоверение типа включает имя сборки, в которой располагается данный тип.

Тип с именем MyType, загруженный в области действия одной сборки, не совпадает с типом MyType, загруженным в области действия другой сборки.

Сборка определяет следующие сведения:

4.**Граница области действия ссылок.**

Манифест сборки содержит метаданные, используемые для разрешения типов и для выполнения связанных с ресурсами запросов. Манифест указывает типы и ресурсы, предоставляемые за пределами сборки, а также перечисляет другие сборки, от которых она зависит.

Сборка определяет следующие сведения:

5. **Граница версий.**

Сборка является наименьшей единицей с поддержкой версий в среде CLR. Версия для всех типов и ресурсов в одной сборке назначается как единому целому.

Сборка определяет следующие сведения:

6. **Единица развертывания.**

При запуске приложения могут присутствовать лишь сборки, первоначально вызванные приложением.

Типы сборок:

- с нестрогими именами (weakly named assemblies)
- со строгими именами (strongly named assemblies).
  - подписаны при помощи пары ключей, уникально идентифицирующей издателя сборки (безопасность, управление ее версиями, развертывание в любом месте пользовательского жесткого диска или в Интернете)
  - атрибуты: имя файла (без расширения), номер версии, идентификатор регионального стандарта и открытый ключ.

Литература

- Рихтер CLR via C#
- https://github.com/sidristij/dotnetbook/blob/master/book/ru/readme.md