

# C# Professional

Отражение (Reflection).

# C# Professional

После урока обязательно



Повторите этот урок в видео формате на  
[ITVDN.com](http://itvdn.com)



Проверьте как Вы усвоили данный материал на  
[TestProvider.com](http://testprovider.com)

## Отражение (Reflection).

# C# Professional

## Reflection

Рефлексия (отражение) - это процесс, во время которого программа может отслеживать и модифицировать собственную структуру и поведение во время выполнения.

Парадигма программирования, положенная в основу отражения, называется рефлексивным программированием. Это один из видов метапрограммирования.

# C# Professional

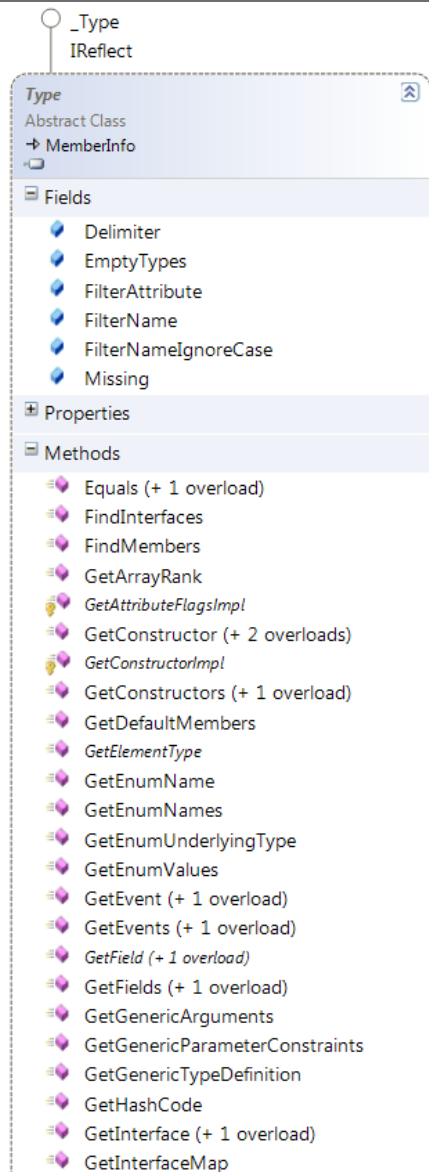
## System.Reflection

- System.Reflection
  - AmbiguousMatchException
  - Assembly
  - AssemblyAlgorithmIdAttribute
  - AssemblyCompanyAttribute
  - AssemblyConfigurationAttribute
  - AssemblyCopyrightAttribute
  - AssemblyCultureAttribute
  - AssemblyDefaultAliasAttribute
  - AssemblyDelaySignAttribute
  - AssemblyDescriptionAttribute
  - AssemblyFileVersionAttribute
  - AssemblyFlagsAttribute
  - AssemblyInformationalVersionAttribute
  - AssemblyKeyFileAttribute
  - AssemblyKeyNameAttribute
  - AssemblyName
  - AssemblyNameProxy
  - AssemblyProductAttribute
  - AssemblyTitleAttribute
  - AssemblyTrademarkAttribute
  - AssemblyVersionAttribute
  - Binder
  - ConstructorInfo
  - CustomAttributeData
  - CustomAttributeFormatException
  - DefaultMemberAttribute
  - EventInfo
  - ExceptionHandlingClause
  - FieldInfo
  - InvalidFilterCriteriaException
  - LocalVariableInfo
  - ManifestResourceInfo
  - MemberInfo
  - MethodBase
  - MethodBody
  - MethodInfo
  - Missing
  - Module
  - ObfuscateAssemblyAttribute
  - ObfuscationAttribute
  - ParameterInfo
  - Printer

**System.Reflection** - пространство имен, которое содержит классы для применения рефлексии в языке C#.

# C# Professional

## Класс



**Type** является корневым классом для функциональных возможностей рефлексии и основным способом доступа к метаданным.

С помощью членов класса **Type** можно получить сведения об объявленных в типе элементах: конструкторах, методах, полях, свойствах и событиях класса, а также о модуле и сборке, в которых развернут данный класс.

# C# Professional

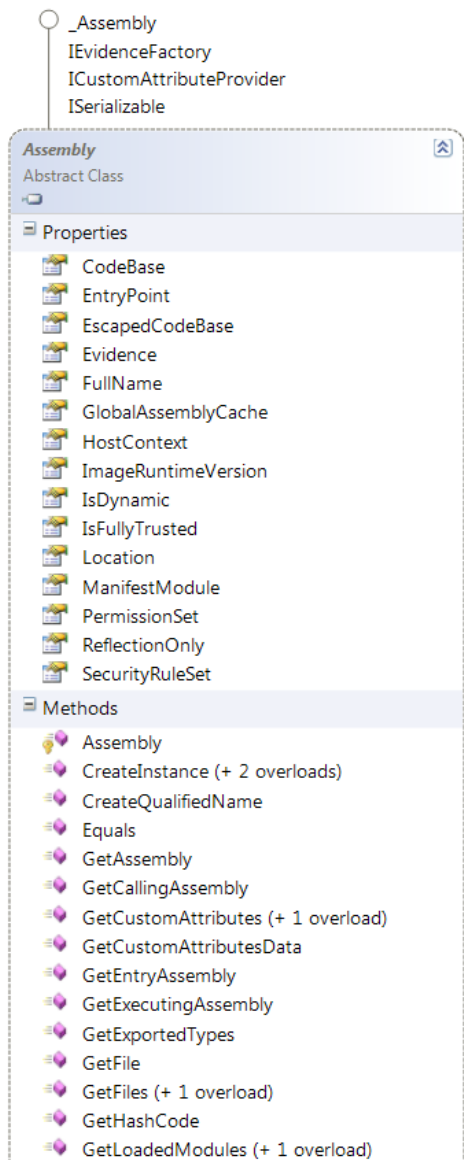
## Способы получения экземпляра

1. Вызов метода **GetType()** на экземпляре требуемого класса.
2. Вызов статического метода **GetType()** класса **Type**.
3. Использование оператора **typeof()**.

В приведенных выше примерах результатом будет ссылка на объект **Type**, содержащий информацию о целевом типе .

# C# Professional

## Класс



Класс **Assembly** представляет собой сборку, которая является модулем с возможностью многократного использования, поддержкой версий и встроенным механизмом описания общезыковой исполняющей среды.



# C# Professional

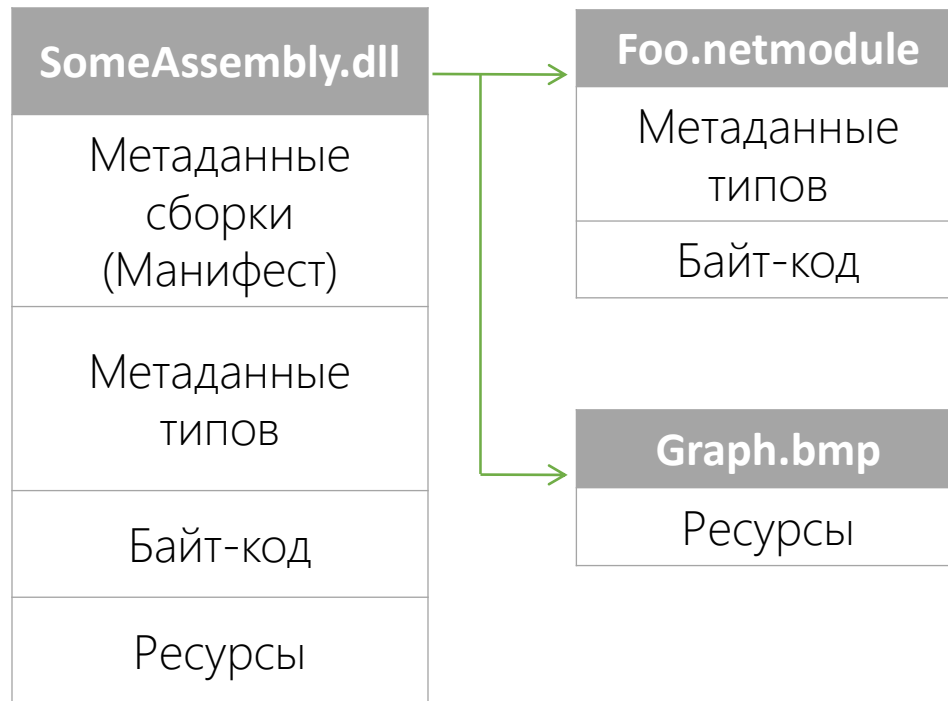
## Однофайловая

SomeAssembly.dll
Метаданные сборки (Манифест)
Метаданные типов
Байт-код
Ресурсы

Все компоненты хранятся в одном файле.

# C# Professional

## Многофайловая



Метаданные сборки обязательно должны находиться в главном файле.

Метаданные типов, код и ресурсы могут храниться как в главном файле сборки, так и во вспомогательных файлах.

# C# Professional

## Манифест

Метаданные сборки (манифест) состоят из описания сборки:  
имя, версия, строгое имя, информация о культуре.

# C# Professional

## Информация описывающая тип

Метаданные типов включают пространство имен и имя типа, члены типа и параметры, если имеются.

# C# Professional

## Byte-code

Байт-код (псевдокод) — машинно-независимый код низкого уровня, генерируемый транслятором и исполняемый интерпретатором. Большинство инструкций байт-кода эквивалентны одной или нескольким командам ассемблера. Трансляция в байт-код занимает промежуточное положение между компиляцией в машинный код и интерпретацией.

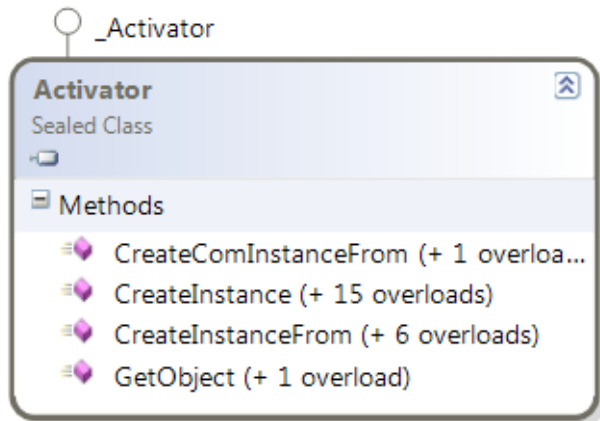
Байт-код называется так, потому что длина каждого кода операции — один байт, но длина кода команды различна. Каждая инструкция представляет собой однобайтовый код операции от 0 до 255, за которым следуют такие параметры, как регистры или адреса памяти.

# C# Professional Resources

Ресурсы – это объекты, которые используются кодом:  
строки, изображения, различные файлы.

# C# Professional

## Класс



Класс **Activator** содержит методы для локального создания типов объектов.

Метод **CreateInstance()** создает экземпляр типа, определенного в сборке путем вызова конструктора, который наилучшим образом соответствует заданным аргументам.

## Q&A



# Информационный видеосервис для разработчиков программного обеспечения

