

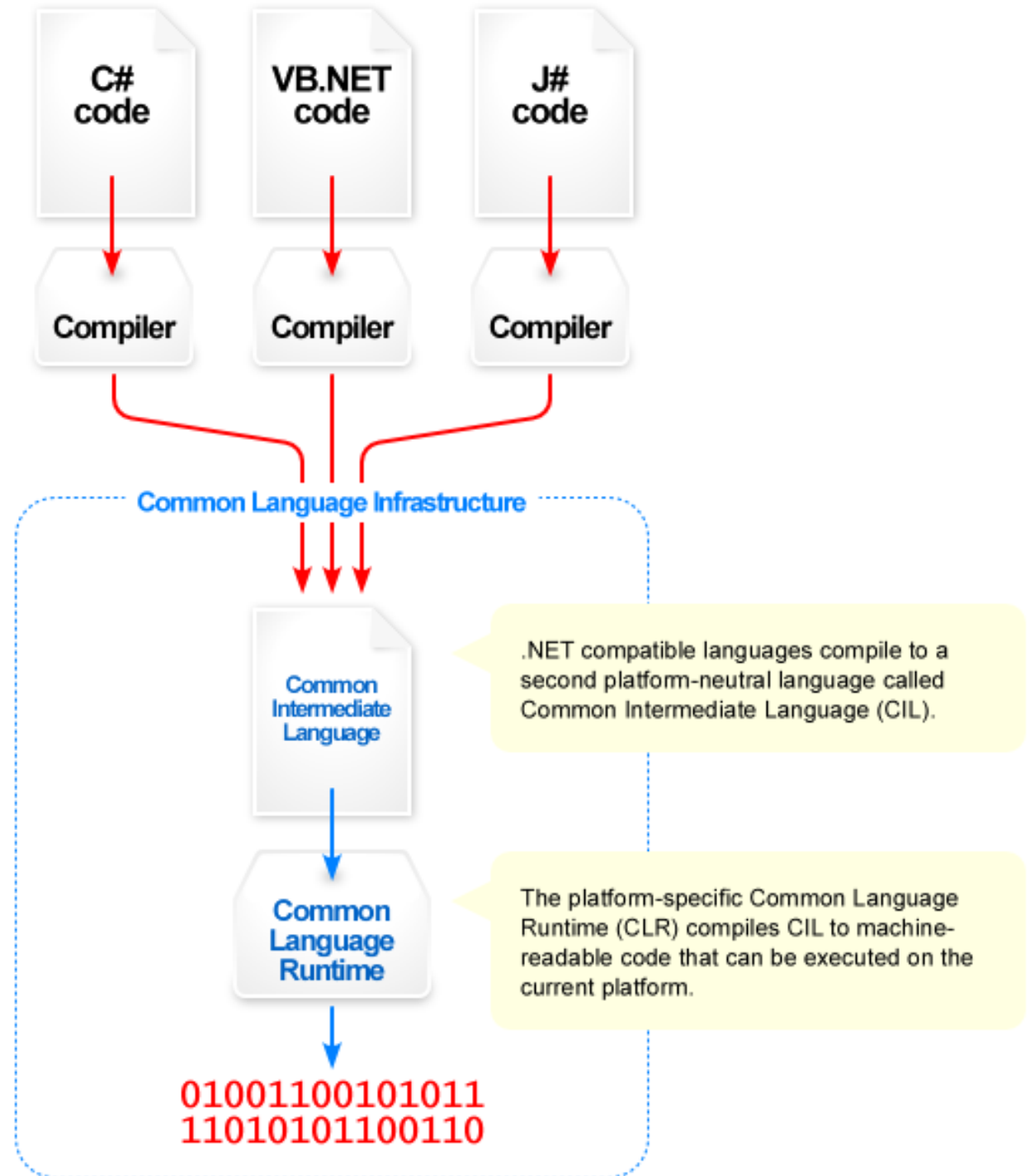
Reflection

- Технология позволяваща извличане на метайнформация за типовете по време на изпълнение на програмата

Кога се ползва Reflection?

- Ако се разработва функционалност, която ще ползва типове, които не са налични по време на проектиране на програмата;
- Ако трябва да се извлече информация за тип, за който не е известна структурата;
- Ако трябва да се прескочи ограничение на типа – пр: да се достъпи `private` поле;
- Ако трябва динамично да се конструират типове;
- ...

.NET vs Native



Дисасемблиране

The screenshot shows a .NET assembly disassembler window with the following structure:

- File View Help
- Assembly: C:\Users\Angel\Desktop\TU\OOP\WindowsFormsApplicationGraphics3\WindowsFormsApplicationGraphics\bin\Del...
- MANIFEST
- ClassLibraryFigures
 - ClassLibraryFigures.IGraphics
 - .class interface public abstract auto ansi
 - DrawRectangle : void(valuetype [System.Drawing]System.Drawing.Color,int32,int32,int32,int32)
 - ClassLibraryFigures.Rectangle
 - .class public auto ansi serializable beforefieldinit
 - <Color>k__BackingField : private valuetype [System.Drawing]System.Drawing.Color
 - <Height>k__BackingField : private int32
 - <Order>k__BackingField : private int32
 - <Position>k__BackingField : private valuetype [System.Drawing]System.Drawing.Point
 - <Width>k__BackingField : private int32
 - .ctor : void()
 - Contains : bool(valuetype [System.Drawing]System.Drawing.Point)
 - Paint : void(class ClassLibraryFigures.IGraphics)
 - get_Color : valuetype [System.Drawing]System.Drawing.Color()
 - get_Height : int32()
 - get_Order : int32()
 - get_Position : valuetype [System.Drawing]System.Drawing.Point()
 - get_Width : int32()
 - set_Color : void(valuetype [System.Drawing]System.Drawing.Color)
 - set_Height : void(int32)
 - set_Order : void(int32)
 - set_Position : void(valuetype [System.Drawing]System.Drawing.Point)
 - set_Width : void(int32)
 - Color : instance valuetype [System.Drawing]System.Drawing.Color()
 - Height : instance int32()
 - Order : instance int32()
 - Position : instance valuetype [System.Drawing]System.Drawing.Point()
 - Width : instance int32()

Assembly Manifest:

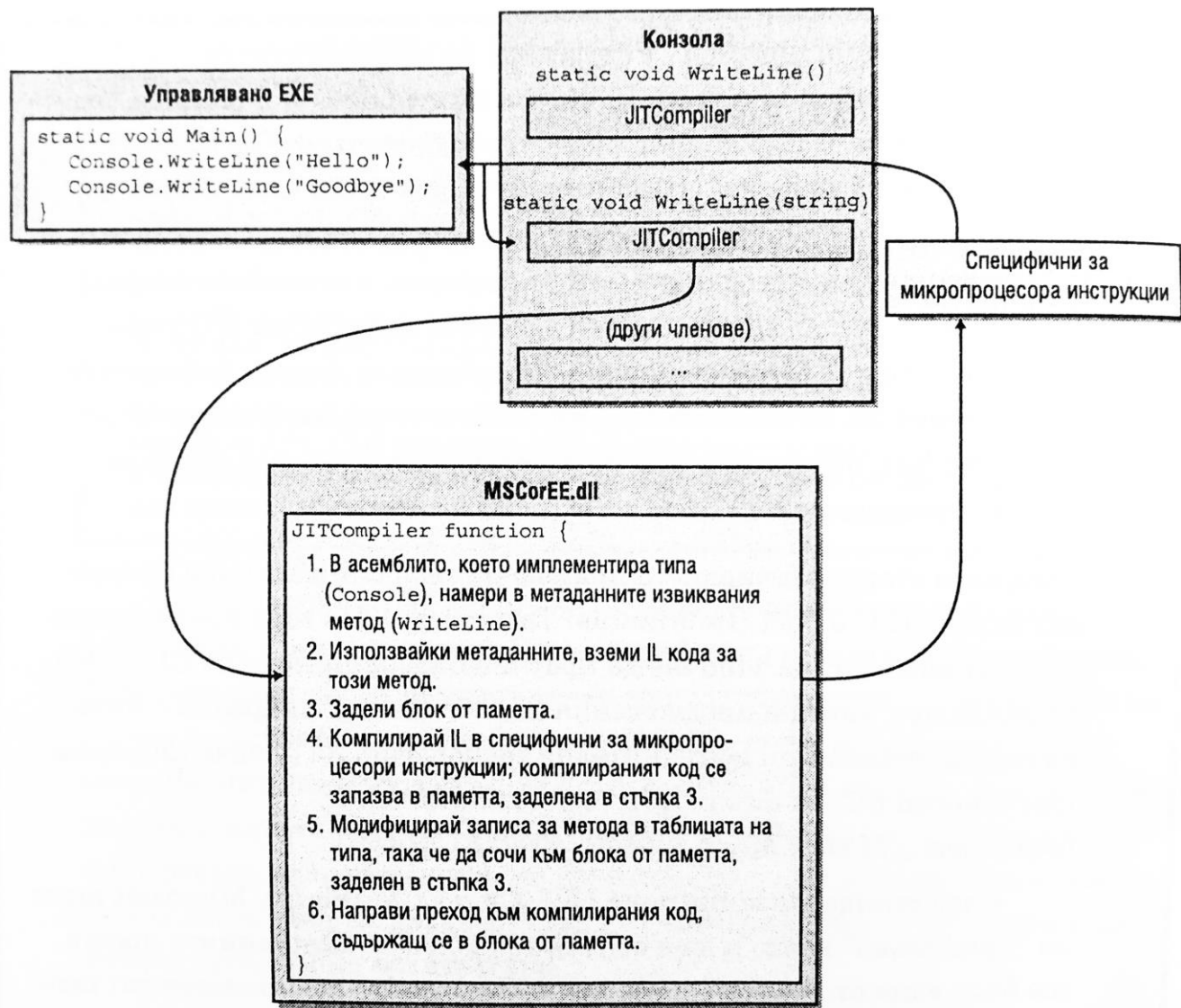
```
.assembly ClassLibraryFigures
{
  .ver 1:0:0:0
}
```

Дисасемблиране

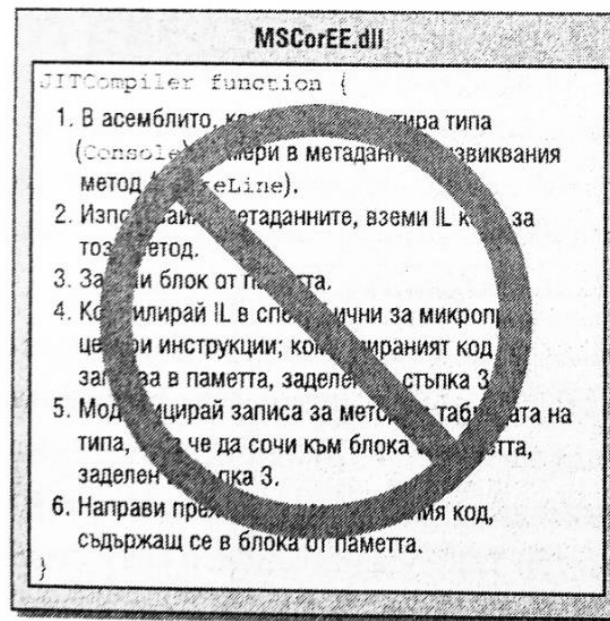
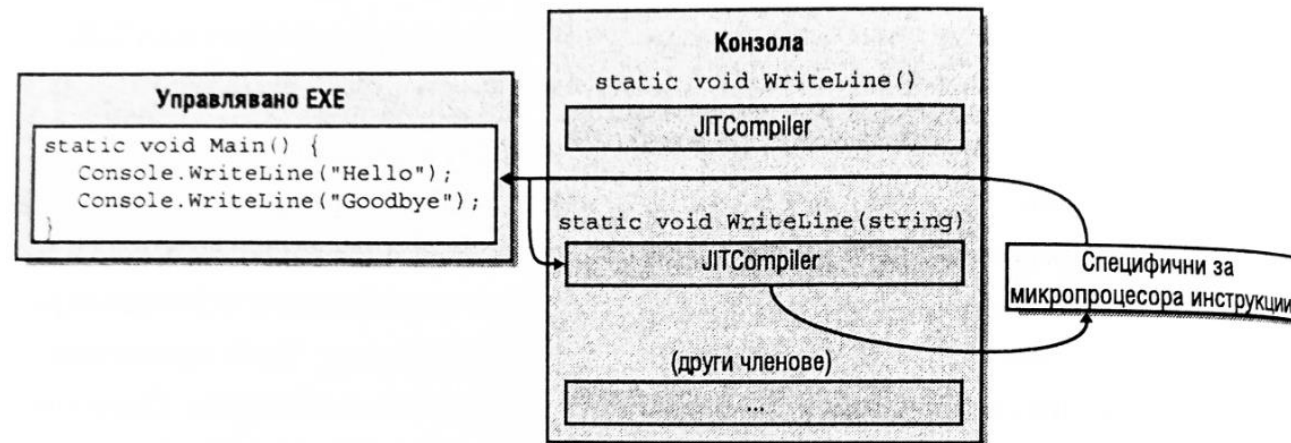
```
ClassLibraryFigures.Rectangle::Paint : void(class ClassLibraryFigures.IGraphics)
Find Find Next
.method public hidebysig instance void Paint(class ClassLibraryFigures.IGraphics graphics) cil managed
{
    // Code size          55 (0x37)
    .maxstack 6
    .locals init ([0] valuetype [System.Drawing]System.Drawing.Point CS$0$0000)
    IL_0000: nop
    IL_0001: ldarg.1
    IL_0002: ldarg.0
    IL_0003: call        instance valuetype [System.Drawing]System.Drawing.Color ClassLibraryFigures.Rectangle::get_Color()
    IL_0008: ldarg.0
    IL_0009: call        instance valuetype [System.Drawing]System.Drawing.Point ClassLibraryFigures.Rectangle::get_Position()
    IL_000e: stloc.0
    IL_000f: ldloc.s    CS$0$0000
    IL_0011: call        instance int32 [System.Drawing]System.Drawing.Point::get_X()
    IL_0016: ldarg.0
    IL_0017: call        instance valuetype [System.Drawing]System.Drawing.Point ClassLibraryFigures.Rectangle::get_Position()
    IL_001c: stloc.0
    IL_001d: ldloc.s    CS$0$0000
    IL_001f: call        instance int32 [System.Drawing]System.Drawing.Point::get_Y()
    IL_0024: ldarg.0
    IL_0025: call        instance int32 ClassLibraryFigures.Rectangle::get_Width()
    IL_002a: ldarg.0
    IL_002b: call        instance int32 ClassLibraryFigures.Rectangle::get_Height()
    IL_0030: callvirt   instance void ClassLibraryFigures.IGraphics::DrawRectangle(valuetype [System.Drawing]System.Drawing.Color,
                                                                    int32,
                                                                    int32,
                                                                    int32,
                                                                    int32)

    IL_0035: nop
    IL_0036: ret
} // end of method Rectangle::Paint
```

Първо извикване



Следващо извикване



Зареждане на асембли

```
Assembly assembly = Assembly.LoadFrom("ClassLibraryFigures.dll");

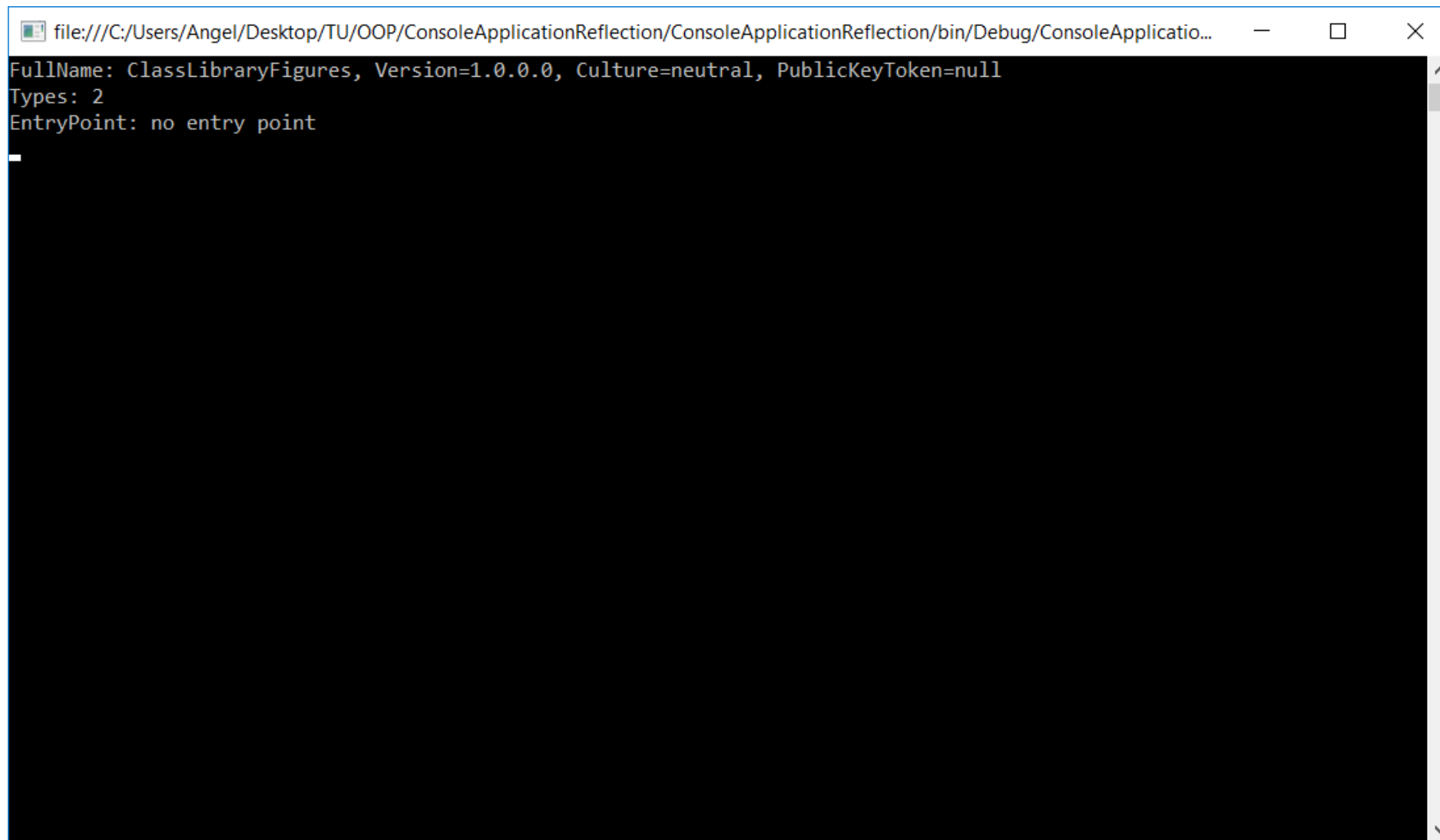
Console.WriteLine(
    "FullName: " + assembly.FullName);

Console.WriteLine(
    "Types: " + assembly.Types.Count());

Console.WriteLine(
    "EntryPoint: " + (assembly.EntryPoint != null
        ? assembly.EntryPoint.Name
        : "no entry point"));

Console.ReadLine();
```


Зареждане на асембли



A screenshot of a Windows console application window. The title bar shows the file path: `file:///C:/Users/Angel/Desktop/TU/OOP/ConsoleApplicationReflection/ConsoleApplicationReflection/bin/Debug/ConsoleApplicatio...`. The console output displays the following information:

```
FullName: ClassLibraryFigures, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null
Types: 2
EntryPoint: no entry point
```

The console window has a black background with white text. There is a small white cursor on the line following the output.

Типът Type

```
Assembly assembly = Assembly.LoadFrom("ClassLibraryFigures.dll");

IEnumerable<Type> assemblyTypes = assembly.GetTypes();

int i = 1;
foreach (Type type in assemblyTypes)
{
    Console.WriteLine("Type" + i++ + ": " + type.Name);
    Console.WriteLine("\tMethods: " + type.GetMethods().Count());
    Console.WriteLine("\tFields: " + type.GetFields().Count());
}

Console.ReadLine();
```

Типът Type



```
file:///C:/Users/Angel/Desktop/TU/OOP/ConsoleApplicationReflection/ConsoleApplicationReflection/bin/Debug/ConsoleApplicationReflection.exe
Type1: IGraphics
      Methods: 1
      Fields: 0
Type2: Rectangle
      Methods: 16
      Fields: 0
```

The screenshot shows a Windows console window with a black background and white text. The title bar at the top indicates the file path: `file:///C:/Users/Angel/Desktop/TU/OOP/ConsoleApplicationReflection/ConsoleApplicationReflection/bin/Debug/ConsoleApplicationReflection.exe`. The console output displays the results of type reflection for two types: `IGraphics` and `Rectangle`. For `IGraphics`, it shows 1 method and 0 fields. For `Rectangle`, it shows 16 methods and 0 fields. A small white cursor is visible on the line following the `Rectangle` information.

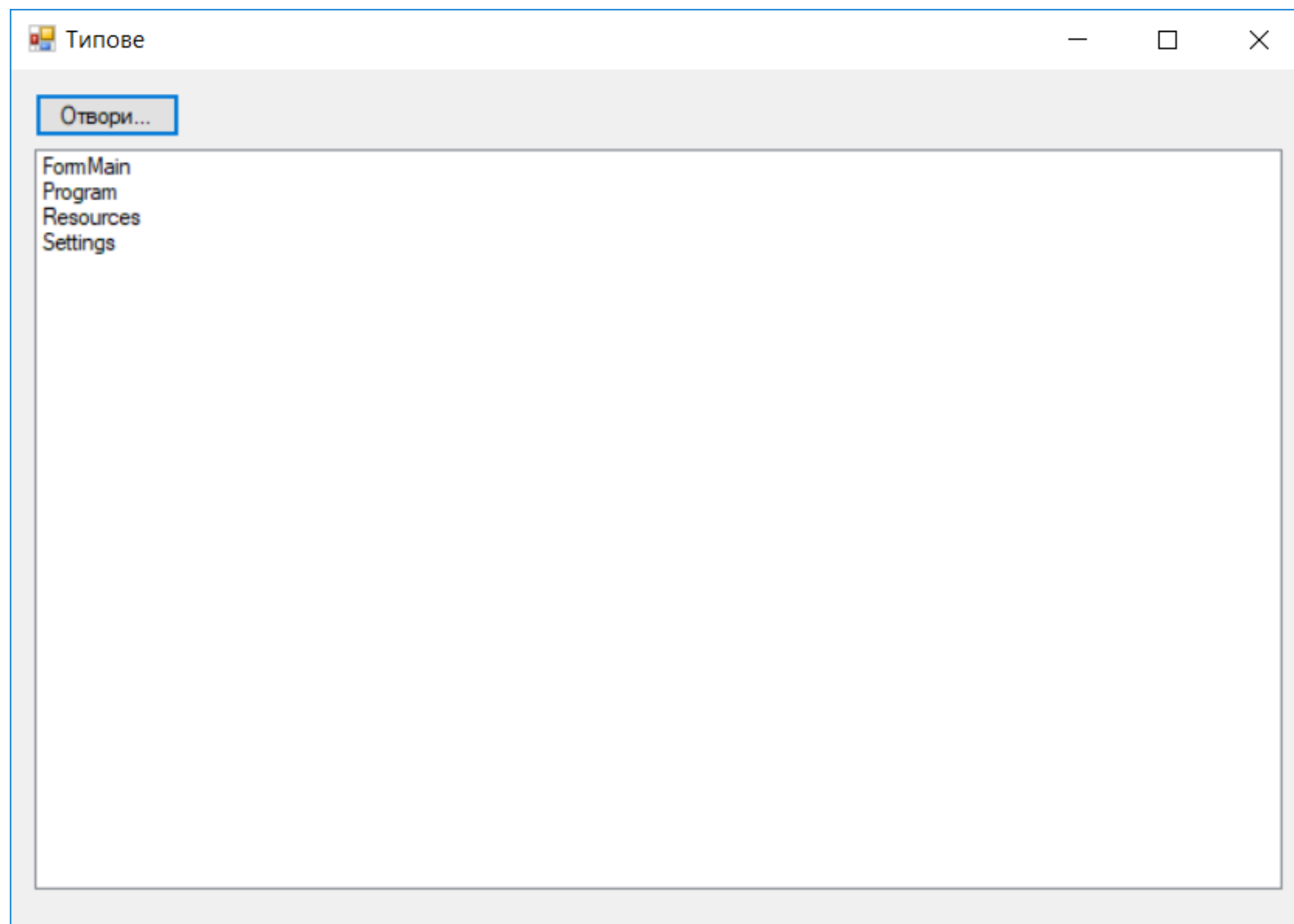
Задача

- Да се направи Windows Forms приложение, зареждащо асембли и визуализиращо типовете в него;
- Приложението да има ListBox, в който се зареждат типовете и бутон отварящ диалогов прозорец за избор на файл;
- Помощен код:

```
OpenFileDialog openFileDialog = new OpenFileDialog();
if (openFileDialog.ShowDialog() == DialogResult.OK)
{
    Assembly assembly = Assembly.LoadFile(openFileDialog.FileName);

    foreach (Type type in assembly.GetTypes())
    {
        ...
    }
}
```

Задача



Задача

```
private void buttonLoad_Click(object sender, EventArgs e)
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    if (openFileDialog.ShowDialog() == DialogResult.OK)
    {
        listBoxTypes.Items.Clear();

        Assembly assembly = Assembly.LoadFile(openFileDialog.FileName);

        foreach (Type type in assembly.GetTypes())
        {
            listBoxTypes.Items.Add(type);
        }
    }
}
```

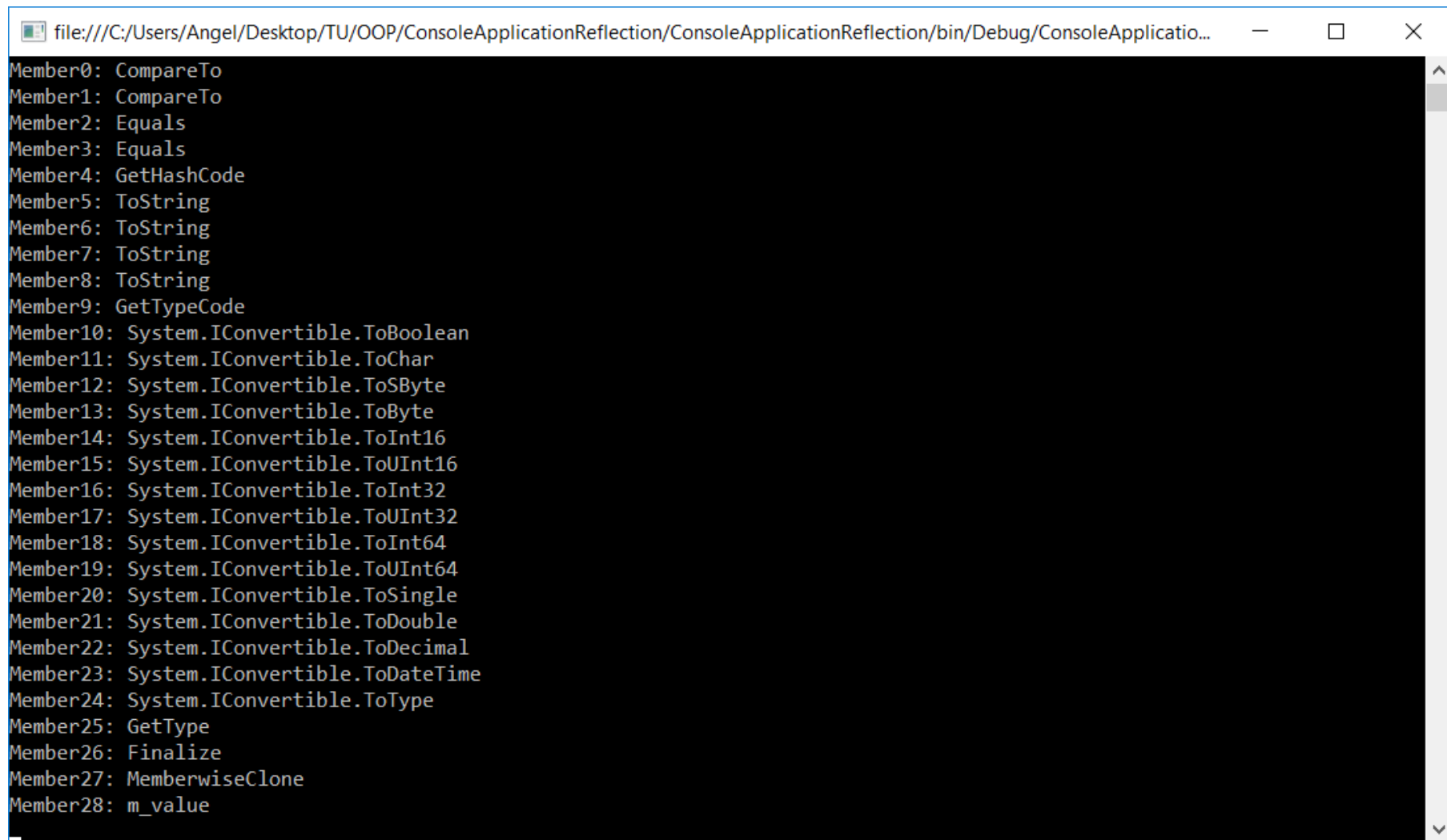
Типът Type

```
int i = 0;
Type type = typeof(int); //type = i.GetType();

foreach (MemberInfo memberInfo in type.GetMembers(
    BindingFlags.Instance |
    BindingFlags.Public |
    BindingFlags.NonPublic))
{
    Console.WriteLine("Member" + i++ + ": " + memberInfo.Name);
}

Console.ReadLine();
```

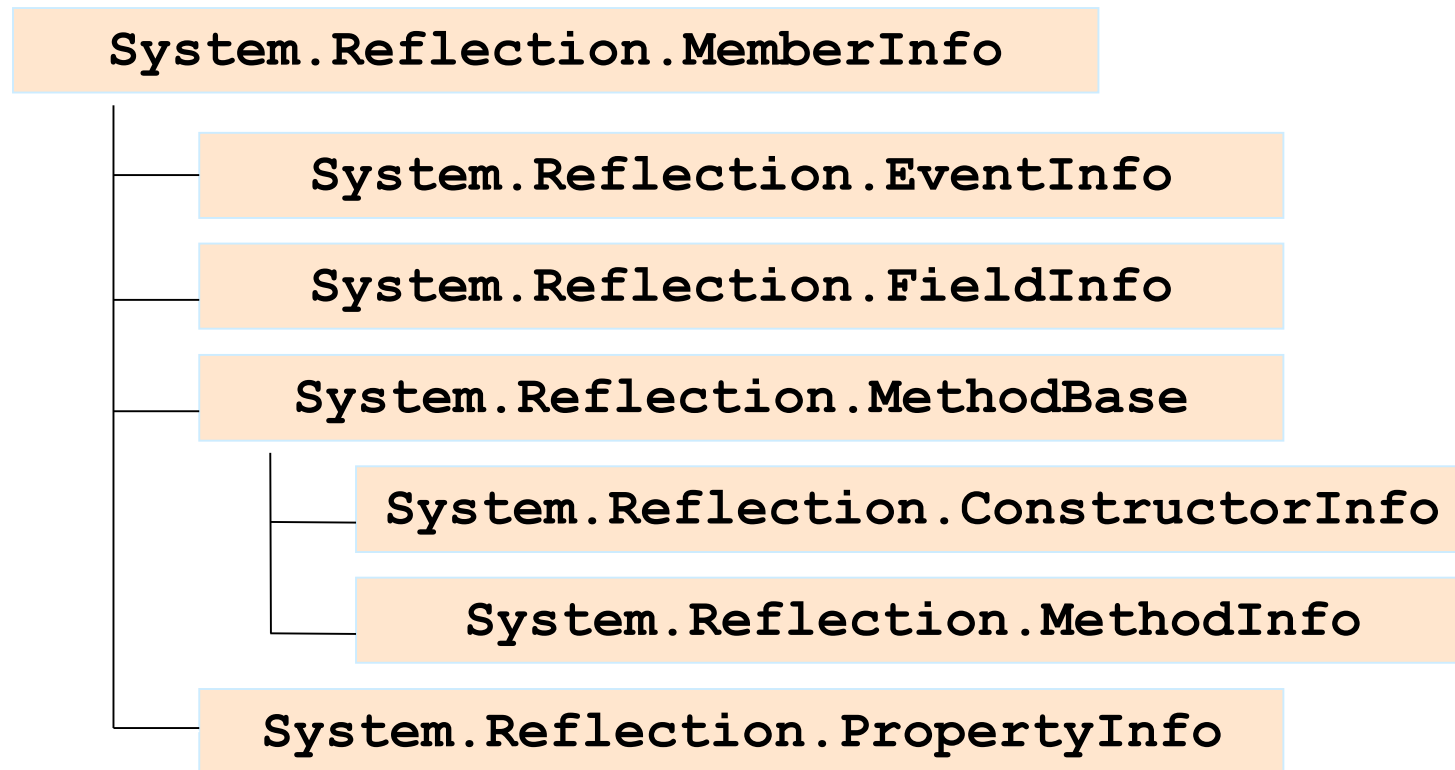
Типът Type



A screenshot of a Windows console window with a black background and white text. The window title bar shows the file path: `file:///C:/Users/Angel/Desktop/TU/OOP/ConsoleApplicationReflection/ConsoleApplicationReflection/bin/Debug/ConsoleApplicatio...`. The console displays a list of 29 members of the `Type` class, indexed from Member0 to Member28. The members include methods like `CompareTo`, `Equals`, `GetHashCode`, `ToString`, `GetTypeCode`, and various `System.IConvertible` methods (e.g., `ToBoolean`, `ToChar`, `ToSByte`, `ToByte`, `ToInt16`, `ToUInt16`, `ToInt32`, `ToUInt32`, `ToInt64`, `ToUInt64`, `ToSingle`, `ToDouble`, `ToDecimal`, `ToDateTime`, `ToType`), as well as `GetType`, `Finalize`, `MemberwiseClone`, and `m_value`.

```
Member0: CompareTo
Member1: CompareTo
Member2: Equals
Member3: Equals
Member4: GetHashCode
Member5: ToString
Member6: ToString
Member7: ToString
Member8: ToString
Member9: GetTypeCode
Member10: System.IConvertible.ToBoolean
Member11: System.IConvertible.ToChar
Member12: System.IConvertible.ToSByte
Member13: System.IConvertible.ToByte
Member14: System.IConvertible.ToInt16
Member15: System.IConvertible.ToUInt16
Member16: System.IConvertible.ToInt32
Member17: System.IConvertible.ToUInt32
Member18: System.IConvertible.ToInt64
Member19: System.IConvertible.ToUInt64
Member20: System.IConvertible.ToSingle
Member21: System.IConvertible.ToDouble
Member22: System.IConvertible.ToDecimal
Member23: System.IConvertible.ToDateTime
Member24: System.IConvertible.ToType
Member25: GetType
Member26: Finalize
Member27: MemberwiseClone
Member28: m_value
```


Основни мета типове



Задача

- Да се измени задачата, така че при двойно щракване на тип от списъка да се показва нов диалогов прозорец с всички методи на типа

```
public partial class FormType : Form
{
    private Type _type;
    1 reference
    public Type Type
    {
        get
        {
            return _type;
        }
        set
        {
            _type = value;
            listBoxMembers.Items.Clear();
            foreach (MemberInfo memberInfo in _type.GetMethods())
            {
                listBoxMembers.Items.Add(memberInfo.Name);
            }
        }
    }
}
```

```
private void listBoxTypes_DoubleClick(object sender, EventArgs e)
{
    if (listBoxTypes.SelectedItem == null)
    {
        return;
    }

    FormType formType = new FormType();
    formType.Type = listBoxTypes.SelectedItem as Type;

    formType.ShowDialog();
}
```

Задача

- Да се измени задачата, така че методите да се показват с параметрите си. За целта за всеки `MethodInfo` може да се извика метода `GetParameters()`, който връща колекция от тип `ParameterInfo`. Всеки `parameterInfo` обект има `ParameterType` – типа на параметъра и `Name` – името на параметъра.

```
set
{
    _type = value;
    listBoxMembers.Items.Clear();
    foreach (MethodInfo methodInfo in _type.GetMethods())
    {
        string parameters = "";
        foreach (ParameterInfo parameterInfo in methodInfo.GetParameters())
        {
            parameters += parameterInfo.ParameterType.Name + " " + parameterInfo.Name + ",";
        }

        if (parameters.Length > 0)
        {
            parameters = parameters.Remove(parameters.Length - 1, 1);
        }

        listBoxMembers.Items.Add(
            methodInfo.Name + "(" + parameters + ")");
    }
}
```

Създаване на инстанция чрез Type

```
class UnknownType
{
    private int _privateField = 5;

    public int GetSomeValue()
    {
        return 10;
    }
}
```

```
Type type = typeof(UnknownType);

Object o = Activator.CreateInstance(type);

Console.ReadLine();
```

Четене / задаване на поле

```
Type type = typeof(UnknownType);
```

```
Object o = Activator.CreateInstance(type);
```

```
Console.WriteLine(type  
    .GetField("_privateField", BindingFlags.Instance | BindingFlags.NonPublic)  
    .GetValue(o));
```

```
type  
    .GetField("_privateField", BindingFlags.Instance | BindingFlags.NonPublic)  
    .SetValue(o, 6);
```

```
Console.ReadLine();
```


Динамично извикване на метод

```
Type type = typeof(UnknownType);

Object o = Activator.CreateInstance(type);

Console.WriteLine(type
    .GetMethod("GetSomeValue", BindingFlags.Instance | BindingFlags.Public)
    .Invoke(o, new object[] { }));

Console.ReadLine();
```