

# Mono.Cecil

Mono's assembly swiss army knife

**Jb Evain**

Mono Team

[jbevain@novell.com](mailto:jbevain@novell.com)

**Novell®**

The background of the slide is a solid blue color with a pattern of diagonal stripes in varying shades of blue, creating a sense of motion or depth.

# Introduction

# What is Cecil?

***« Cecil is a library to generate and inspect programs and libraries in the ECMA CIL format. »***



What can you do with Cecil?

- You can inspect all the inwards of an assembly.
- You can modify them as much as you want and save them back.
- You can also create assemblies from scratch.
- As Cecil is licensed under the permissive MIT/X11 license, you can use it in all kind of applications

# Why would you want to ...

- ... inspect assemblies using something else than `System.Reflection`?
  - Because `System.Reflection` doesn't give you access to everything, and to the CIL code in particular.
  - Because `Mono.Cecil` doesn't require the assembly to be loaded in any `AppDomain`.
  - Because the types in `Mono.Cecil` are eligible for garbage collection.
  - Because you want to analyze a different version of the `mscorlib.dll` than the one which is currently used.

# Why would you want to ...

- ... generate assemblies with something different from `System.Reflection.Emit`?
  - Because you can not modify an already existing assembly with `System.Reflection.Emit`.
  - Because `System.Reflection.Emit` contains some interesting limitations and bugs. (Cecil contains some as well, but at least we can fix them).

# When you probably should not ...

- ... use Cecil?
  - When you need to emit code dynamically and use it directly. Here `System.Reflection.Emit` performs much better.

The background of the slide is a solid blue color with a pattern of diagonal stripes in varying shades of blue, creating a sense of motion and depth. The stripes are more pronounced on the right side of the slide.

Code

# Printing all the types of an assembly

```
using Mono.Cecil;

// ...

AssemblyDefinition asm =
    AssemblyFactory.GetAssembly ("Mono.Cecil.dll");

foreach (TypeDefinition type in asm.MainModule.Types) {
    Console.WriteLine (type.FullName);
}

// <Module>
// Mono.Cecil.AggressiveReflectionReader
// Mono.Cecil.ArrayDimension
// Mono.Cecil.ArrayDimensionCollection
// Mono.Cecil.ArrayType
// Mono.Cecil.AssemblyDefinition
// Mono.Cecil.AssemblyFactory
// ...
```



# The custom attributes of an assembly

```
// asm is an AssemblyDefinition ...
foreach (CustomAttribute attribute in asm.CustomAttributes) {
    Console.WriteLine (
        attribute.Constructor.DeclaringType.Name);
    foreach (object p in attribute.ConstructorParameters) {
        Console.WriteLine ("param: {0}", p);
    }
}

// AssemblyTitle
// param: Mono.Cecil
// AssemblyDescriptionAttribute
// param: Library for reading and writing CIL images
// ...
```

# Printing some CIL

```
using Mono.Cecil.Cil;

// ...

TypeDefinition asm = cecil.MainModule.Types
    ["Mono.Cecil.AssemblyDefinition"];
MethodDefinition tostr = type.Methods.GetMethod (
    "ToString") [0];

foreach (Instruction instr in tostr.Body.Instructions) {
    Console.WriteLine ("{0} {1}",
        instr.OpCode.Name,
        instr.Operand);
}

// ldarg.0
// ldfld Mono.Cecil.AssemblyNameDefinition Mono.Cecil.AssemblyDefinition::m_asmName
// callvirt System.String Mono.Cecil.AssemblyNameReference::get_FullName()
// ret
```

# Modifying the CIL

```
// tostr is a MethodDefinition

CilWorker cil = tostr.Body.CilWorker;

// import System.Console::WriteLine(string)
MethodReference writeline = asm.MainModule.Import (
    typeof (Console).GetMethod ("WriteLine", new Type [] {
        typeof (string)}));

Instruction ldstr = cil.Create (OpCodes.Ldstr,
    "Hello world of CIL injection");

cil.InsertBefore (tostr.Body.Instructions [0], ldstr);
cil.InsertAfter (ldstr, cil.Create (OpCodes.Call, writeline));

AssemblyFactory.SaveAssembly (asm, "Mono.Cecil.new.dll");
```

The background of the slide is a solid blue color with a pattern of diagonal stripes in varying shades of blue, creating a sense of motion or depth.

# Frequently Asked Questions

# Does Cecil supports generics?

- Oh my, yeah.

# Is Cecil stable?

- It is.
- The growing list of users ensures that Cecil is able to read pretty much every valid assembly.
- And even some invalid assemblies because of some \$#@\*()%#! obfuscators that just love to emit non-sense.

# Is Cecil API stable?

- The API should not change a lot from now.
- Still, because of some on-going optimization work, I don't want to promise API stability.
- I'm usually kind with my users, and when there's a big change, I provide a migration path.

# Does Cecil supports debug symbols?

- Mostly.
- On Mono, Cecil can read and emit our .mdb debug symbols store file format.
- For a variety of reasons, Cecil can only read and write .pdb files (Microsoft's store format) when running on .net.



# What's missing?

- Cecil is pretty close to be feature complete.
- One missing thing is the support for emitting multi-modules assemblies.
- Another is emitting mixed-mode assemblies.

The background of the slide is a solid blue color with a pattern of diagonal stripes in varying shades of blue, creating a sense of movement and depth.

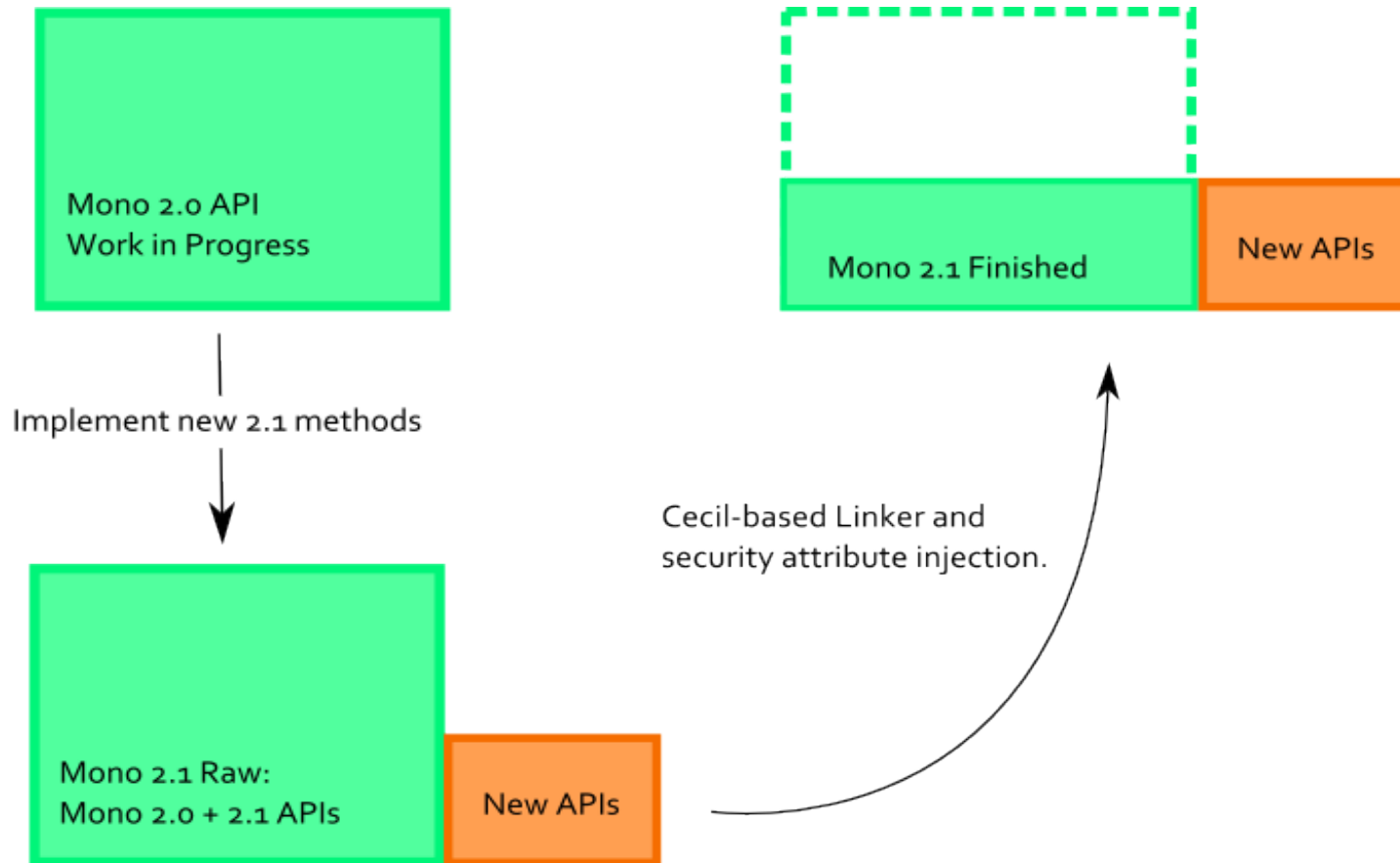
# Usage and Users

# Mono

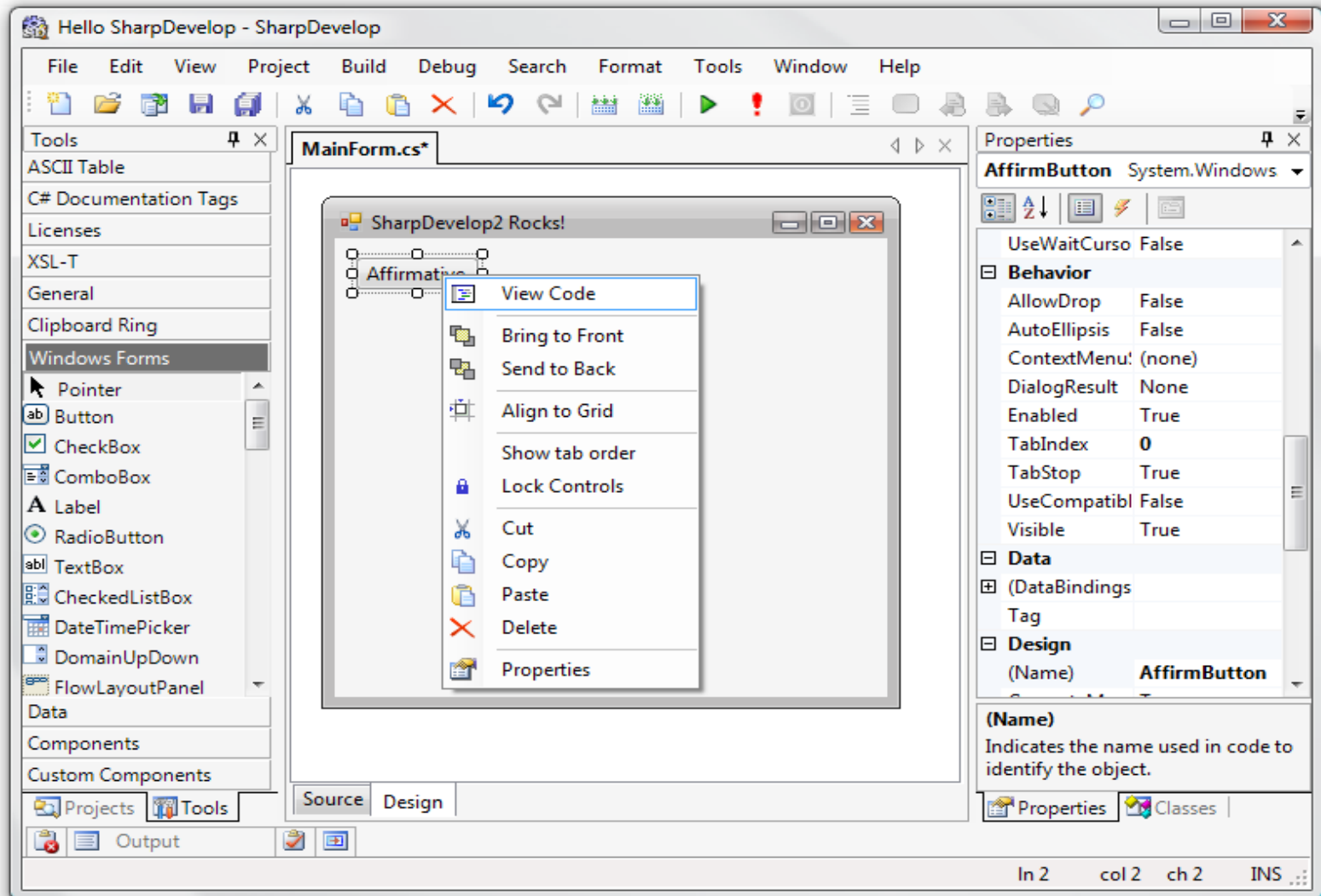
- Mono as a whole project is probably one of the biggest user of Cecil.
  - MonoDevelop,
  - Stetic,
  - mdb,
  - MoMA
  - Gendarme,
  - Monoxide,
  - the linker

# The story of moonlight libraries

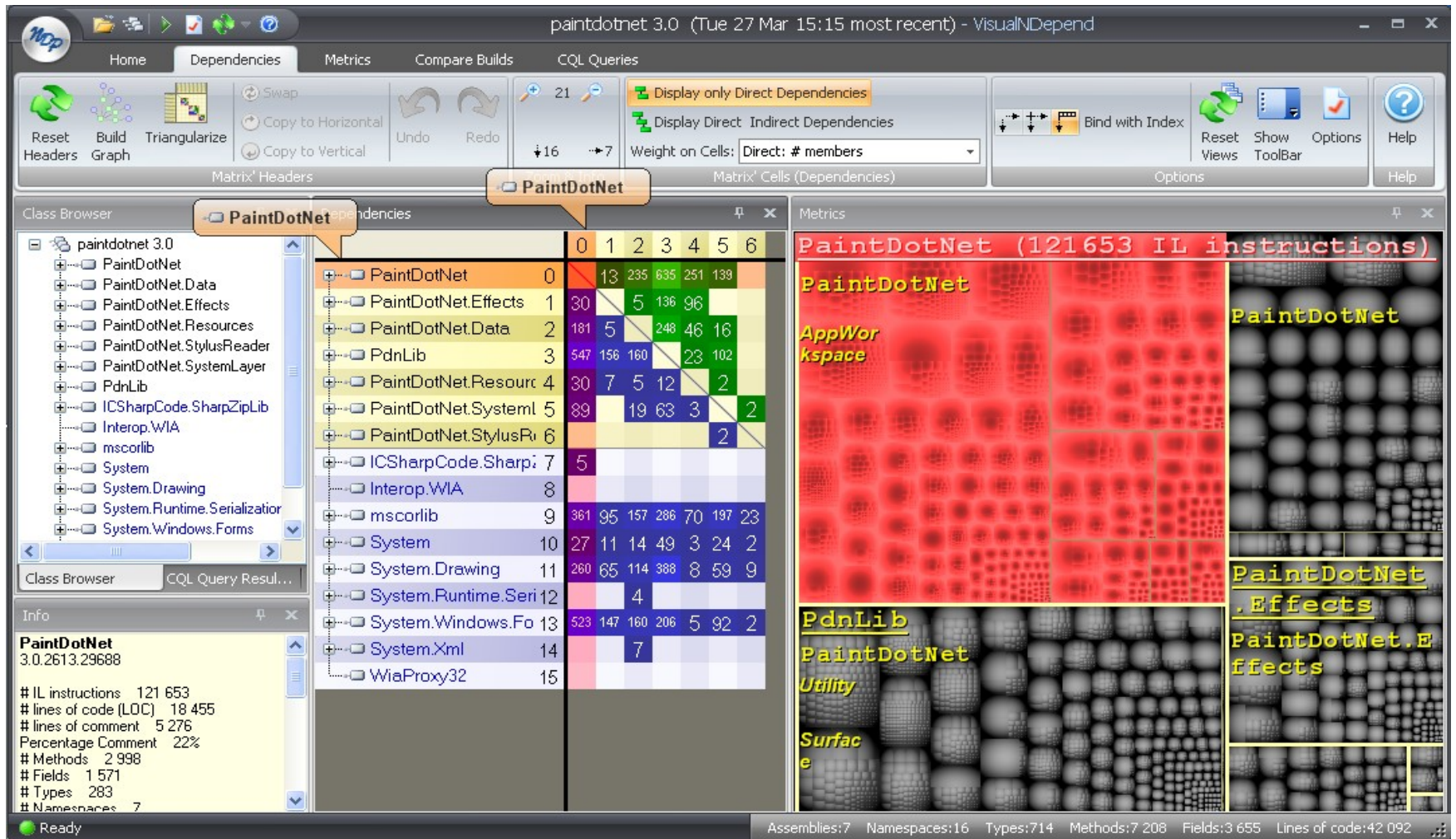
- or how Cecil became a part of the build process

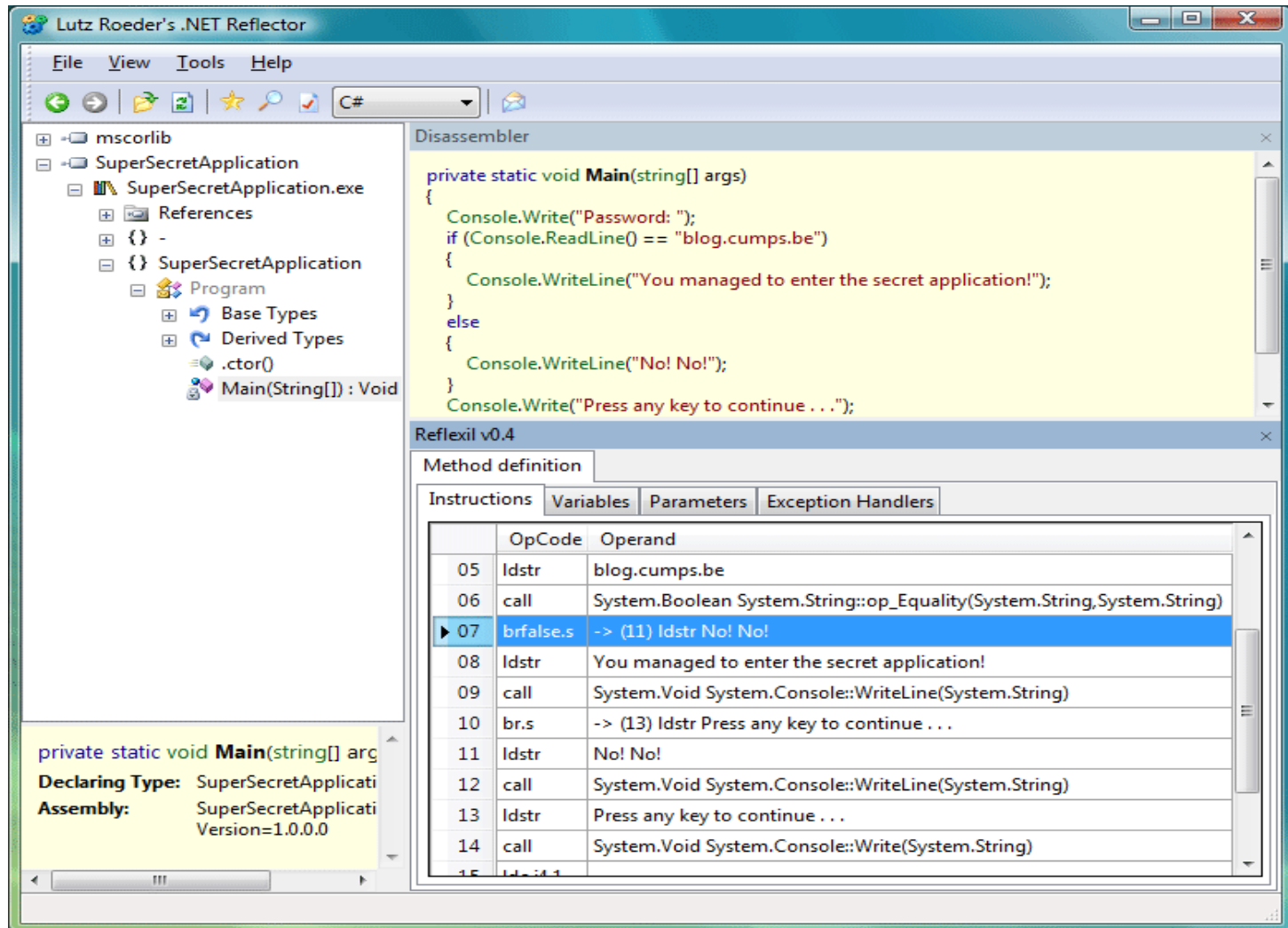


# SharpDevelop



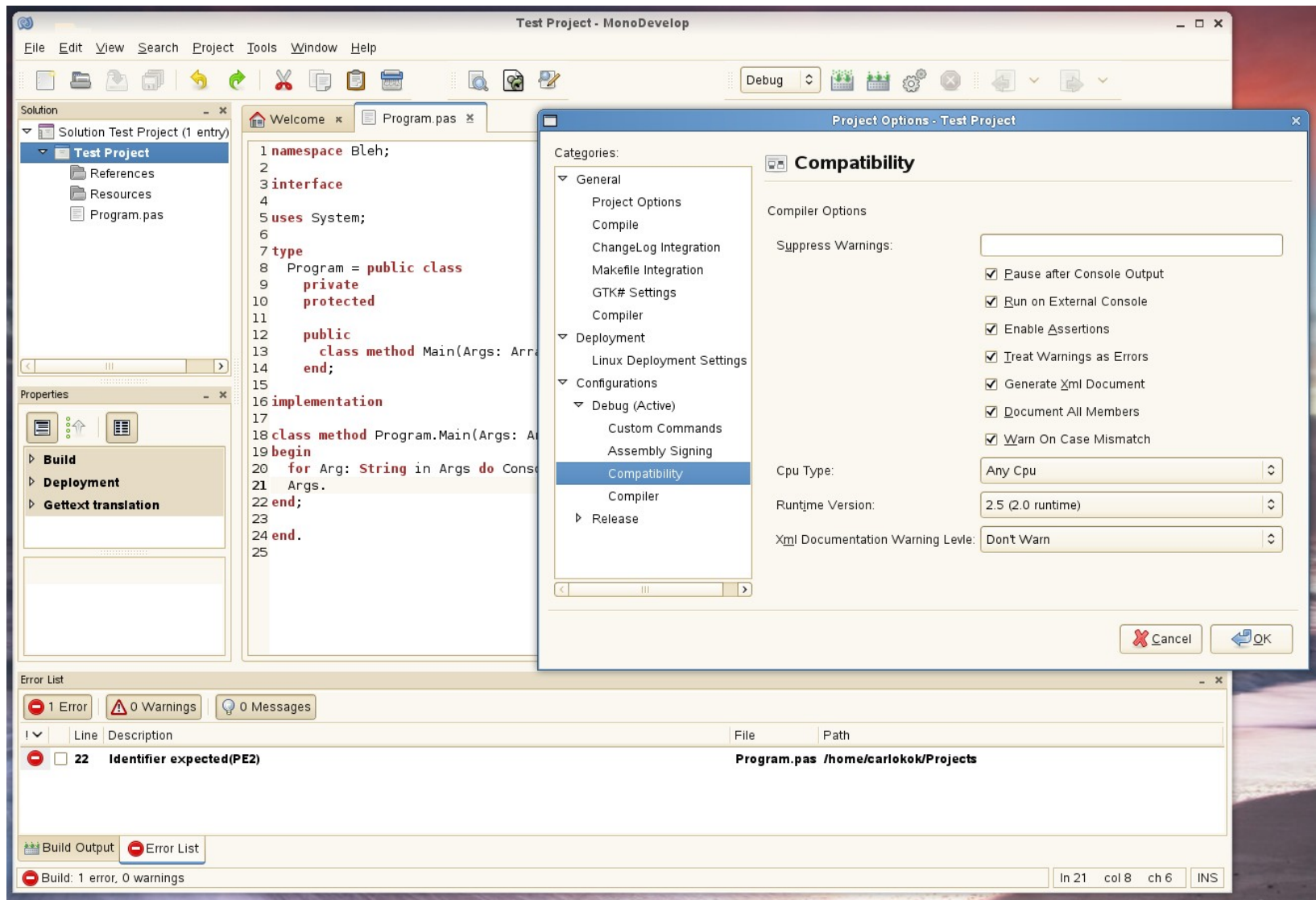
# NDepend







# RemObjects Chrome





# Other noticeable users

- db4o
- Mainsoft's Grasshopper
- some obfuscators
- a whole bunch of AOP tools
- ...

The background of the slide is a solid blue color with a pattern of diagonal stripes in varying shades of blue, creating a sense of motion or depth.

so what's next?

# What's next.

- If Cecil is almost feature complete, it could certainly use some optimization work.
- Mainsoft started a branch where Cecil delays the loading of the metadata as much as possible. Useful for people only interested in reading parts of an assembly. The code is still being reviewed, but it will certainly be merged in the trunk at some point.

The background of the slide is a solid blue color with a pattern of diagonal stripes in varying shades of blue, creating a sense of movement and depth.

Want to join the Cecil fest?

# Plenty of areas to contribute to:

- ildasm, ilasm,
- peverify,
- flowanalysis,
- gendarme,
- ...

## but we also badly need:

- documentation,
- tutorials,
- unit tests

## Reach us on:

- <http://groups.google.com/group/mono-cecil>
- <http://bugzilla.novell.com>
  - Mono-Class libraries
    - > Mono.Cecil
- #cecil on GimpNet

## Reach us on:

- <http://groups.google.com/group/mono-cecil>
- <http://bugzilla.novell.com>
  - Mono-Class libraries
    - > Mono.Cecil
- #cecil on GimpNet

The background of the slide is a solid blue color with a pattern of diagonal stripes in varying shades of blue, creating a sense of motion or depth.

Questions?



**Novell®**

## **Unpublished Work of Novell, Inc. All Rights Reserved.**

This work is an unpublished work and contains confidential, proprietary, and trade secret information of Novell, Inc. Access to this work is restricted to Novell employees who have a need to know to perform tasks within the scope of their assignments. No part of this work may be practiced, performed, copied, distributed, revised, modified, translated, abridged, condensed, expanded, collected, or adapted without the prior written consent of Novell, Inc. Any use or exploitation of this work without authorization could subject the perpetrator to criminal and civil liability.

## **General Disclaimer**

This document is not to be construed as a promise by any participating company to develop, deliver, or market a product. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. Novell, Inc. makes no representations or warranties with respect to the contents of this document, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. The development, release, and timing of features or functionality described for Novell products remains at the sole discretion of Novell. Further, Novell, Inc. reserves the right to revise this document and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes. All Novell marks referenced in this presentation are trademarks or registered trademarks of Novell, Inc. in the United States and other countries. All third-party trademarks are the property of their respective owners.

