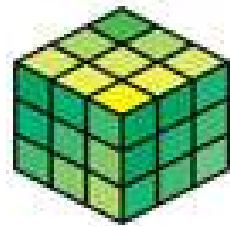




How CROSS PLATFORM is your .NET?

Jean-Baptiste
Evain



essaim









Summer of Code

db4objects

6

222

Mono

pour la recherche
et l'enseignement

agenda

1.

qu'est ce que
Mono ?

2.

ce qu'il y a
d'intéressant dans
Mono ?

3.

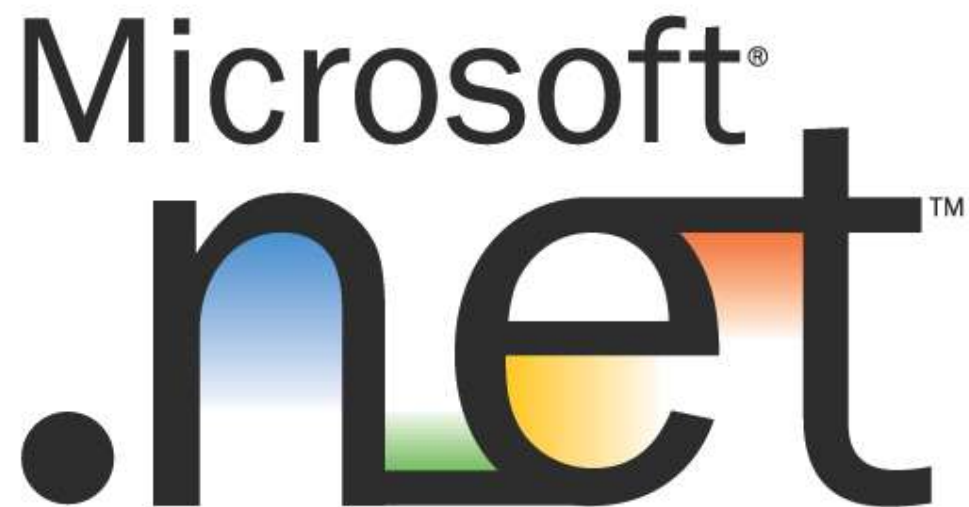
les alternatives à
Mono ?

4.

ce qu'on peut
faire avec Mono ?

1. qu'est ce que
Mono ?

Microsoft®
.**net**™

The Microsoft .NET logo is centered on a black background. It features the word "Microsoft" in a black sans-serif font with a registered trademark symbol (®) at the end. Below it, the ".net" is written in a larger, bold, black sans-serif font. The ".net" has a colorful gradient fill: the dot is black, the 'n' has a blue-to-white gradient, the 'e' has a green-to-yellow gradient, and the 't' has an orange-to-white gradient. A small trademark symbol (™) is located at the top right of the 't'.

« An architecture
for creating and
running component
software »

1.

environnement
d'exécution

2.

framework

3.

ensemble d'outils
et de compilateurs

bâti sur des
standards

CLI: Common Language Infrastructure ECMA-335

C#

ECMA - 334



European Cartons Makers Association

European Computer Manufacturers Association

ECMA-Script
(JavaScript)

Eiffel

ISO



open - source

Novell.

libre

ASP.NET

ADO.NET

Windows.Forms

Microsoft Compatibility
Libraries

Java Compatibility

iFolder

Evolution#

GTK#

Gnome#

Novell.LDAP

Rendezvous: mDNS

MySQL/Postgress/ZipLib

Apache Mono

Mozilla

Mono Libraries

Compilers and Tools

Mono Runtime

CLI: Common Language Infrastructure

runtime:

- Gestion des types (CLS)
- Gestion de la mémoire
- Gestion des exceptions
- Gestion des méta données

runtime:

- Support de plusieurs langages
- Support de l'interopérabilité avec du code natif
- Sécurité au déploiement
- Sécurité à l'exécution

cil:

- Common Intermediate Language
- Généré par tout compilateur .net
- Indépendant de l'architecture
- Conçu pour être compilé
 - « Just In Time » Compilation
 - « Ahead Of Time » Compilation

assembly:

- L'unité de déploiement
- Auto descriptive
 - Méta données
- Gestion des versions
- Gestion de la sécurité

simplification des
modèles précédents



portable

\neq OS

Linux

Windows

NT, 2k, XP, 2k3

Mac OS X

*BSD

Solaris

≠ architectures

x86

PowerPC

x86 - 64

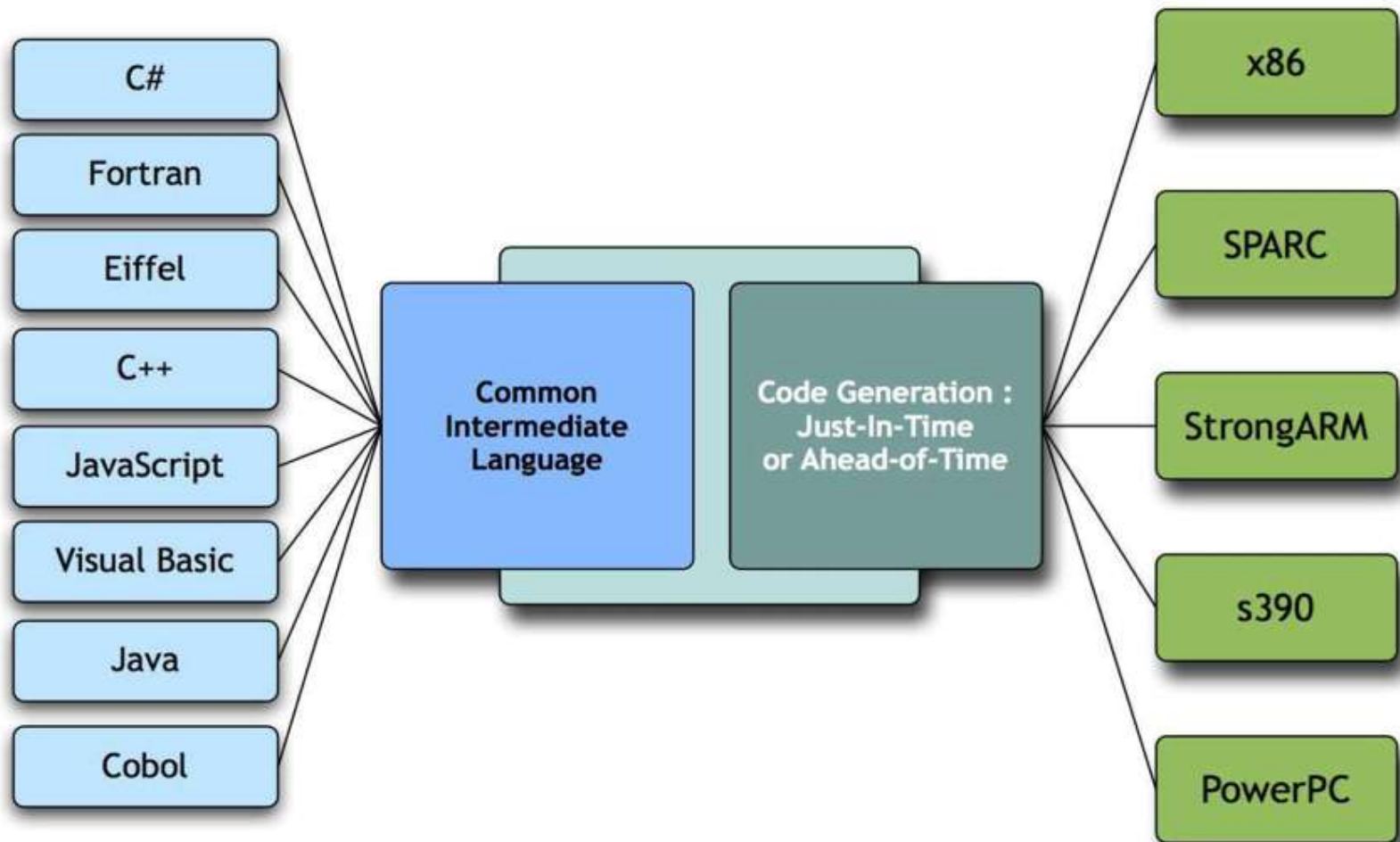
Sparc

s390

IA64

Arm





sinon

actif

20 développeurs
plein temps

~100 contributeurs
réguliers

défi technologique

~ excitant ~

si tant est qu'on
peut trouver un
ramasse miettes
excitant

2. ce qu'il y a
d'intéressant dans
Mono ?

2.1. runtime

Just In Time
compiler

JIT:

- 2 représentations
 - CIL
 - Arbres
 - Proche du code natif
- Optimisations
 - SSA, SSAPRE, HSSA, DEADCE, ABC removal, inlining, ...

Garbage Collector



GC :

- Boehm GC
 - Très utilisé
- Conservateur
- Peu précis
- En changement

et puis

runtime:

- Vérification du code CIL
- Code Access Security
- Chargement dynamique de code
- En constante amélioration

2.2 framework

compatibilité

Librairies :

- Core librairies
- Web: ASP.NET 1.1
- Services Web
- Données: ADO.NET
- Clients riches: Windows.Forms
- Portage d'application facilité si certaines règles ont été respectées

oui

bon

d' accord

les Windows . Forms

c'est pas fini

enfin pas tout à
fait

mais ça avance
bien

et puis c'est pas
si grave

les plus

en plus :

- Gtk#
- Accès aux données générique
 - Avec encore plus de providers
- Mono.C5
- Mono.Posix
- Mono.Cecil
- ...

juste un mot sur

Mono . Cecil

parce que

c'est moi qui l'ai
fait

cecil:

- Librairie de manipulation d'assemblies
- Multiples usages
- Utilisé dans de nombreux outils:
 - MonoDevelop, mdb, AspectDNG, db4o, NDepend, EUSS, ...

ce qu'il manque

COM+ / Enterprise Services

2.3. outils

1.

compilateur c#

mcs :

- Écrit en C#
 - Self hosting
- 54K LOC
- Rapide (~2.5s pour se recompiler)
- Base pour plusieurs compilateurs

dérivés :

- gmcs
 - mcs + support des génériques
- mbas
 - Compilateur Visual-Basic.NET
 - alpha



2.

debugger

mdb :

- Écrit principalement en C#
 - Avec un peu de glue C pour le runtime
- Au jour d'aujourd'hui
- Toujours en développement
- Limité à Linux x86 ou x86-64

3.

assembleur /
désassembleur

```

.namespace Mono.Cecil
{
    .class public auto ansi sealed beforefieldinit PointerType
        extends Mono.Cecil.TypeSpecification
        implements Mono.Cecil.IPointerType, Mono.Cecil.ITypeSpecification, Mono.Cecil.ITypeReference,
Mono.Cecil.IType, Mono.Cecil.IReflectionVisitable, Mono.Cecil.IMemberReference,
Mono.Cecil.IMetadataTokenProvider, Mono.Cecil.IGenericParameterProvider,
Mono.Cecil.ICustomAttributeProvider {

        // method line 1981
        .method public hidebysig specialname rtspecialname
            instance default void .ctor (class Mono.Cecil.TypeReference pType) cil managed
        {
            // Method begins at RVA 0xcc6c
            // Code size 8 (0x8)
            .maxstack 8
            IL_0000: ldarg.0
            IL_0001: ldarg.1
            IL_0002: call instance void class Mono.Cecil.TypeSpecification::.ctor(class
Mono.Cecil.TypeReference)
            IL_0007: ret
        } // end of method PointerType::instance default void .ctor (class Mono.Cecil.TypeReference pType)

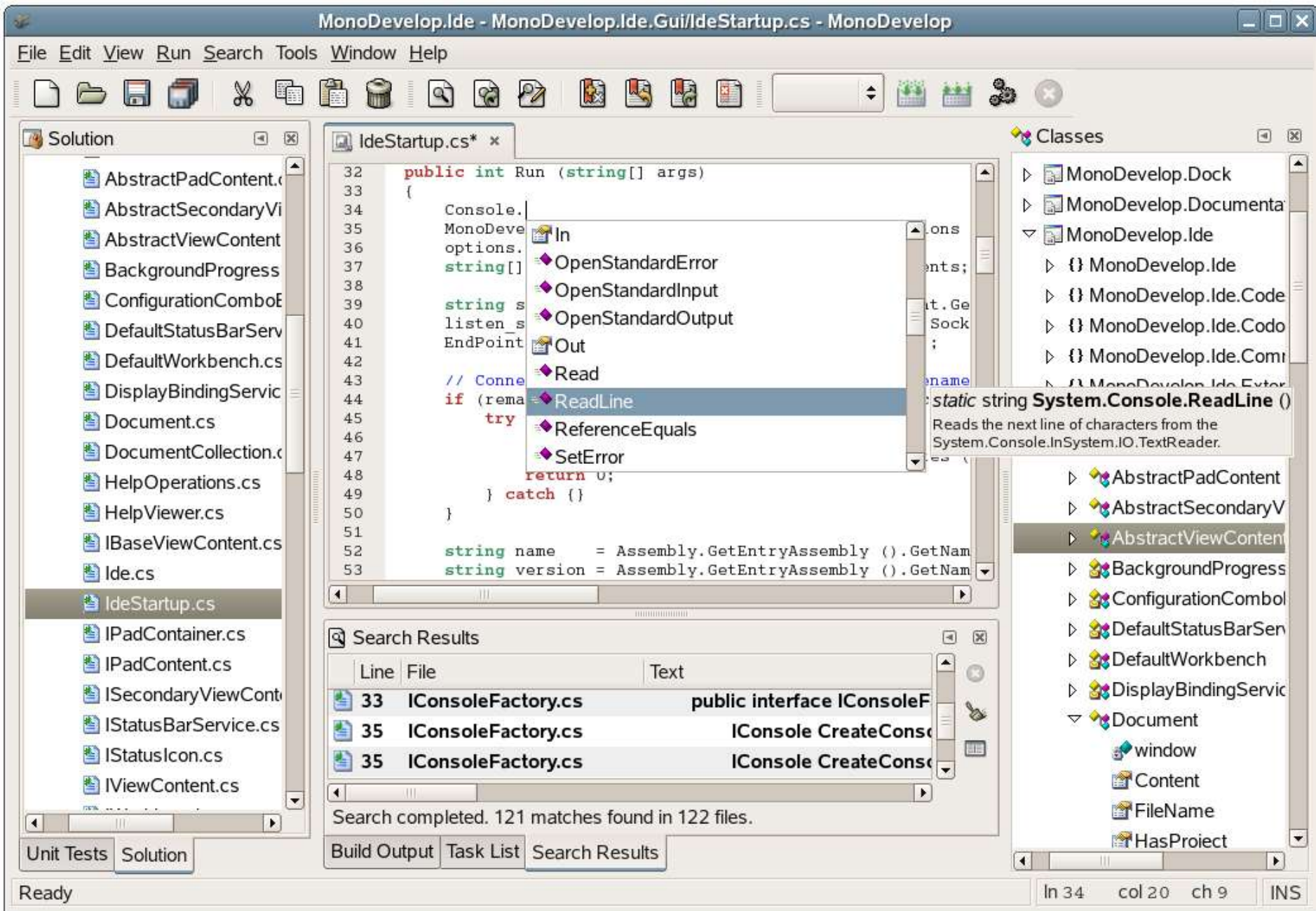
        // method line 1982
        .method public virtual hidebysig specialname
            instance default string get_Name () cil managed
        {
            // Method begins at RVA 0xcc78
            // Code size 17 (0x11)
            .maxstack 8
            IL_0000: ldarg.0
            IL_0001: call instance string class Mono.Cecil.TypeSpecification::get_Name()
            IL_0006: ldstr "*"
            IL_000b: call string string::Concat(string, string)
            IL_0010: ret
        } // end of method PointerType::instance default string get_Name ()

        // method line 1983
        .method public virtual hidebysig specialname
            instance default string get_FullName () cil managed
        {
            // Method begins at RVA 0xcc8c
            // Code size 17 (0x11)
            .maxstack 8

```

4.

IDE : MonoDevelop



architecture
ouverte

plugins

ttest2 - MonoDevelop

File Edit View Run Search Tools Window Help

Unit Tests

- ttest
 - AnotherTestNamespace
 - MainClass Test
 - SomethingElse
 - Test3
 - Test4
 - Test5
 - Test7
 - Test8
 - TestNamespace
 - MainClass Test

ttest

Summary Regressions Failed tests

	Green	Red	Yellow	Test
13	0	3		TestNamespace
3	2	1		AnotherTestNamespace

Main.cs

```
1 using System;
2 using System.Xml.Serialization;
3
4 using NUnit.Framework;
5
6 namespace TestNamespace
7 {
8     [TestFixture]
9     public class MainClassTest
10     {
11         [Test]
12         [Category ("special")]
13         public void TestSomethings33 ()
14         {
15             Console.WriteLine("Hello World!");
16         }
17
18         [Test]
19         public void SomethingElases ()
20         {
21             Console.WriteLine("Hello World!");
22         }
23     }
24 }
```

Test results

Successful Tests Failed Tests Ignored Tests Output

Running tests for **ttest** configuration **Release**

- ttest.ttest.TestNamespace.MainClass Test.Something
- ttest.ttest.TestNamespace.MainClass Test.SimpleTes
- ttest.ttest.TestNamespace.MainClass Test.SimpleTes
- ttest.ttest.AnotherTestNamespace.MainClass Test.Sc
- ttest.ttest.AnotherTestNamespace.MainClass Test.Te

19 / 22 Running ttest.ttest.AnotherTestNames Failed: 2 Ignored: 3

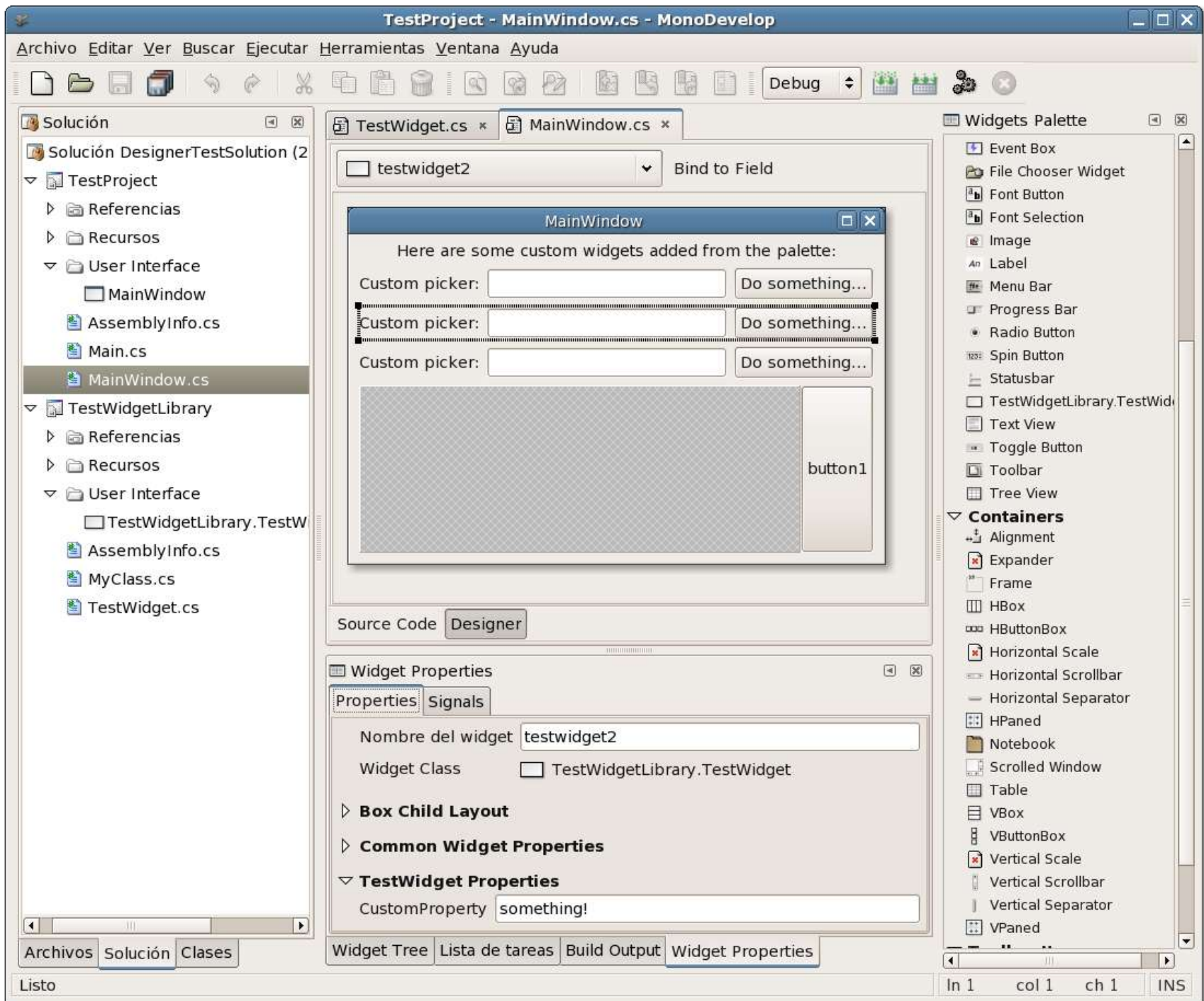
Task List Build Output Boo Shell Test results

Files Help Solution Unit Tests

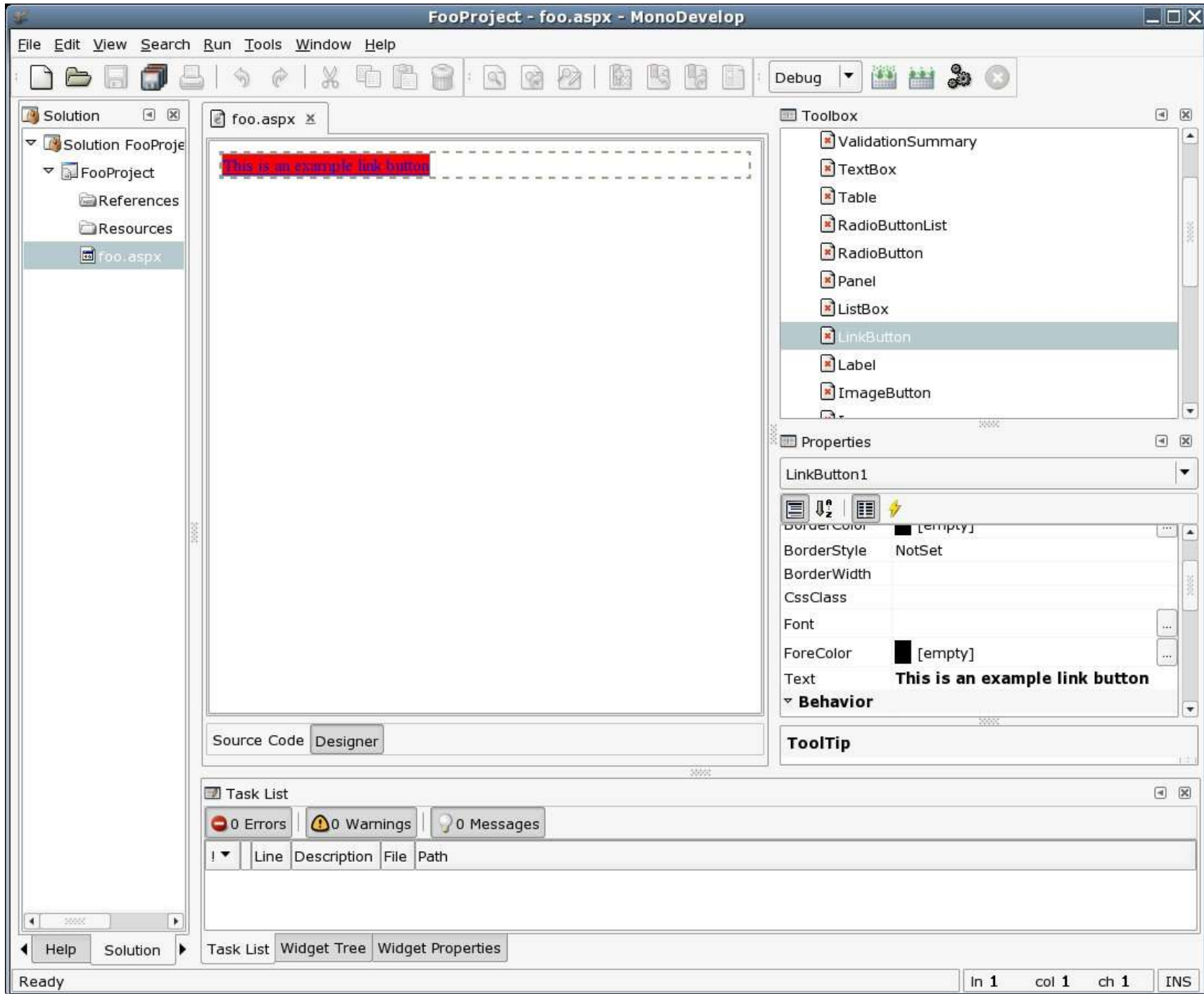
Build: 0 errors, 1 warnings.

ln 1 col 1 ch 1 INS

GUI: Gtk#



Web : ASP.NET



5.

Gendarme

gendarme :

- Équivalent de FxCop
- Vérificateur de règles
- Mais aussi détecteur de bugs

qui dit
open source

dit licences

runtime:

LGPL

libraries:

MIT/X11

compilateur C# :
GPL

+ licences
commerciales



3. les alternatives à Mono ?



VS



plateformes avec
de fortes
similitudes

code intermédiaire

ramasse-miettes

sécurité

orientée objets

introspection

gestion des
exceptions

métabolismes



quelles
différences ?

assembly / class

CIL / bytecode

pensé pour le
multi langages

facilité
d'intégration avec
le code natif

des _ vrais _
génériques

et puis

pourquoi vouloir
utiliser

Mono ou Java

quand on peut
utiliser

Java avec Mono

IKVM

ikvm:

- JVM fonctionnant sous .net
- Traduction du bytecode vers CIL
 - JIT
 - AOT
- Permet de faire tourner eclipse sous Mono



VS

Shared Source CLI

ROTOR

rotor ?

sous - ensemble

.net framework

shared-source

crée et documenté

pour la recherche
et l'enseignement

disent-ils

complexe

COM/C++

incomplet

non optimisé

et qui par
définition

n'a pas
d'application
industrielle

cela dit

chapeau

Microsoft®

pour son ouverture
sur ce projet

brief

4. ce qu'on peut
faire de Mono

idées

1.

languages de programmation

Langages :

- JScript.NET
- Boo
 - Python-like
- Nemerle
 - C# + Programmation fonctionnelle
- Mais aussi

ADA

Eiffel

Forth

Haskell

Mercury

Oberon

Pascal

Prolog

Scheme

Smalltalk



votre langage ?

2.

programmation
orientée aspect

aop :

- Au niveau du langage
- Au niveau du runtime
- Au niveau du code CIL

3.

gestion de la
mémoire

mémoire:

- Remplacer le GC actuel
- En cours de développement:
 - Précis
 - Compact
- A implémenter:
 - Concurrent
 - Incrémental

4.

JIT

JIT:

- Formidable terrain d'essais
- Transformation de code intermédiaire
- JIT managé

5.

en vrac

idées :

- Concurrency
- Sécurité
- Services distribués
- ...

voilà

toutes les marques

tous les logos

appartiennent à
leur propriétaire
respectif

jb at evain.net

<http://evain.net>



bonus

historique

2001

30 Juin:
Lancement du
projet Mono

28 août :

« Hello World ! »

5 Septembre:
mcs compile un
hello world.

2002

12 Mars:

mcs compile mcs

30 Juin:

mcs compile

mscorlib.dll



30 Juin 2004:

Mono 1.0

~Septembre 2006 :

Mono 1.2