



ReSharper vs Roslyn

Kirill Skrygan

**GO ON. ASK ME AGAIN IF RESHARPER WILL
USE ROSLYN**



I DARE YOU



ReSharper

- Разработка с 2003 года
- Visual Studio plugin + Command Line
- Расширяемый, более 100 плагинов
- C#, VB, JavaScript, Html, Css, Xml, Json, Razor, TypeScript, Xaml, Regexp, ...

Roslyn

- Разработка с 2010 года, в production – с 2015.
- Переписывался два раза 😊
- Visual Studio 2015 + standalone
- Расширяемый + уже встроенный в CodeRush, DuoCode, Scrawl, ...
- C#, VB only

IDE Features

ReSharper

- >1500 code inspections
- ~1000 quick fixes
- ~100 refactorings (50 for C#)
- >1000 feature actions

Roslyn

- <100 code analyzers
- ~20 code fixes

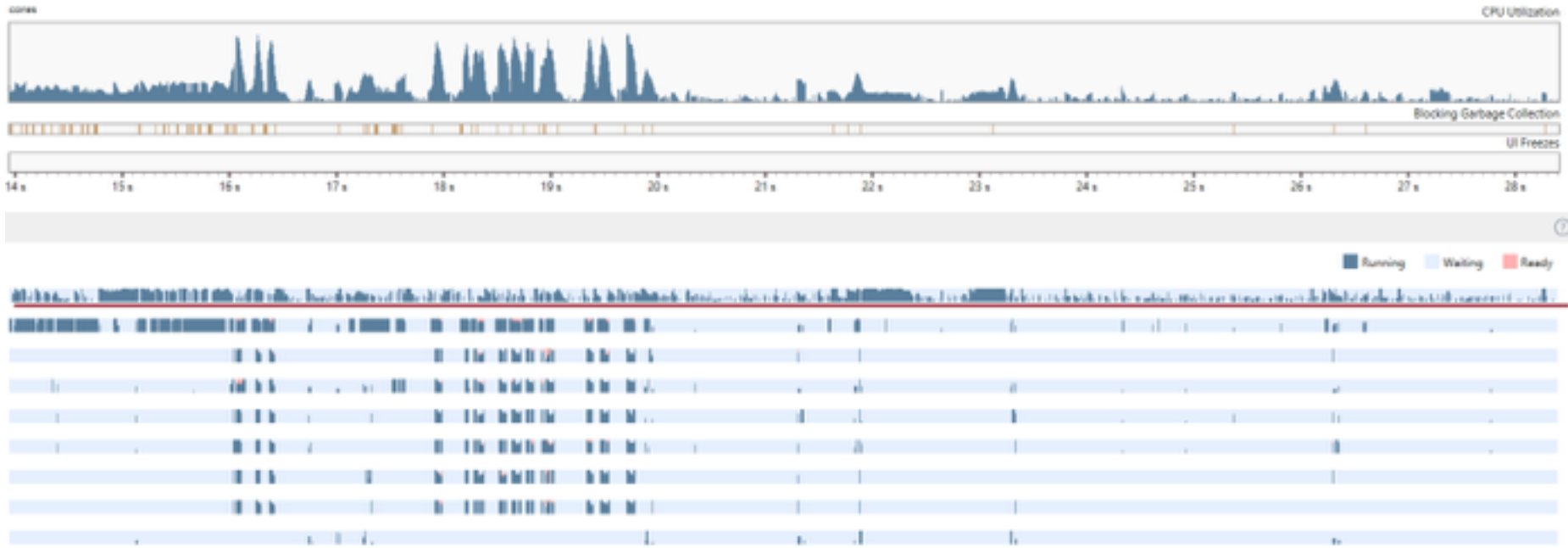
Базовая IDE функциональность

- Go to declaration
- Code completion
- Rename, extract method, introduce variable
- Find Usages

Разные цели

- Roslyn – это прежде всего компилятор (хотя и extensible)...
- ...который обязан прежде всего соблюдать стандарты (ECMA-334, ECMA-335)
- ReSharper – это IDE, и прежде всего IDE
- ReSharper в резолве может позволить себе гораздо больше...
- ...И позволяет 😊

Плавность UI потока



ReSharper Content Model

Project Model

Projects, Files, Folders, Project References,
Assemblies, ...

**Read
Write
Lock**

AST Trees

PSI

Supporting caches

Web symbols, Dependent files, Resources

Symbols

Indexes

Word index, persistentIdIndex, ...

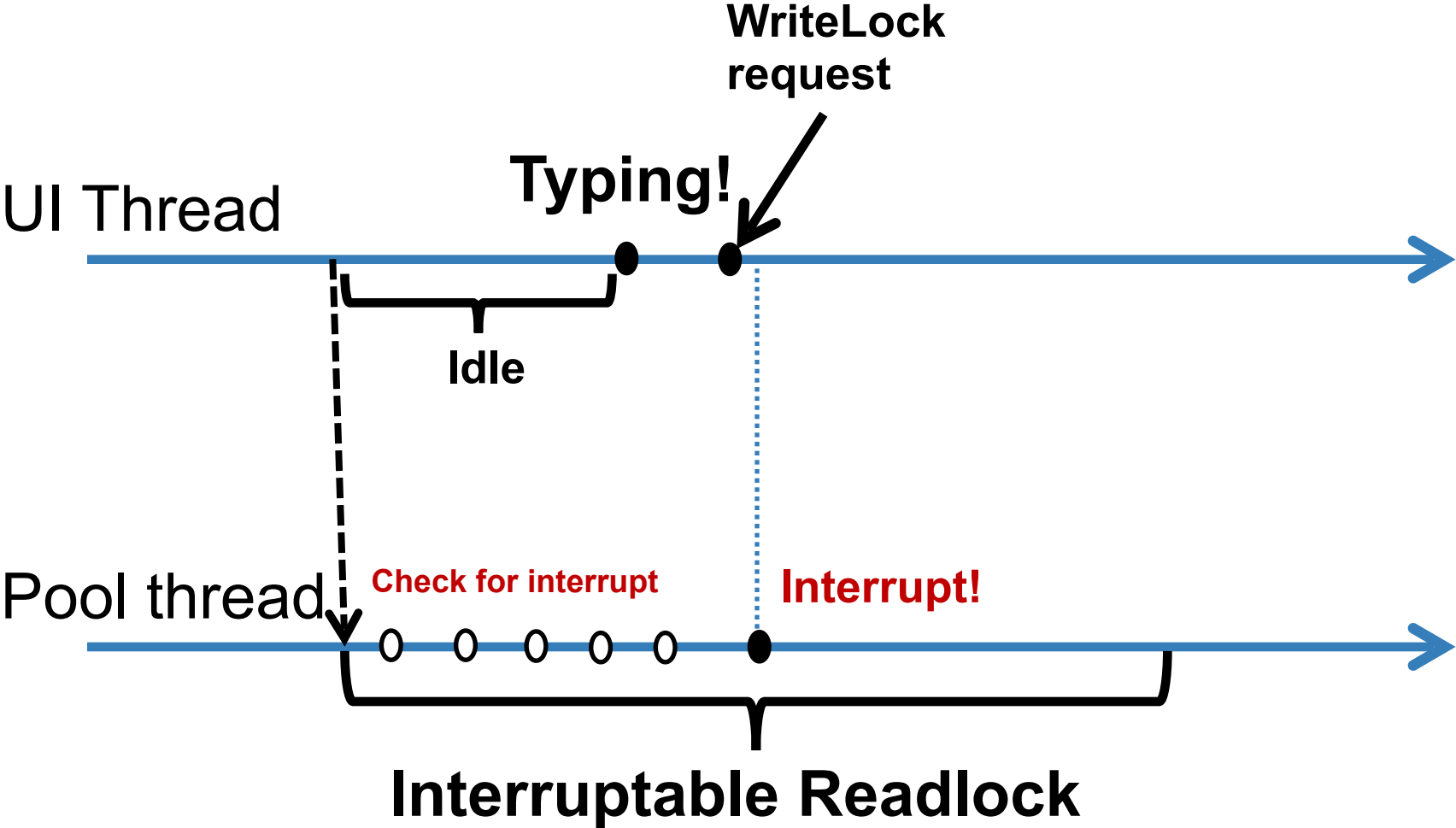
UI thread

Pool thread

Pool thread

Pool thread

Pool thread



ReSharper: contentions

Process Overview ?

CPU Utilization (55,9%)



Blocking Garbage Collection (19,2%)



UI Freeze



**UI Thread ждёт на
WriteLock'е**

Threads ?

ID	Name	Private Bytes	Working Set	Stack
40716	Main	4 425	2,6	
29712	CacheJobService	4 425	2,6	
16692	Finalizer	4 425	2,6	
16408	CLR Worker	4 425	2,6	
23864	CLR Worker	4 425	2,6	
7696	CLR Worker	4 425	2,6	
29616	CLR Worker	4 425	2,6	
43936	CLR Worker	4 425	2,6	
22332	CLR Worker	4 425	2,6	
35526	CLR Worker	4 425	2,6	

**Кэши что-то делают под
непрерываемым Readlock'ом**

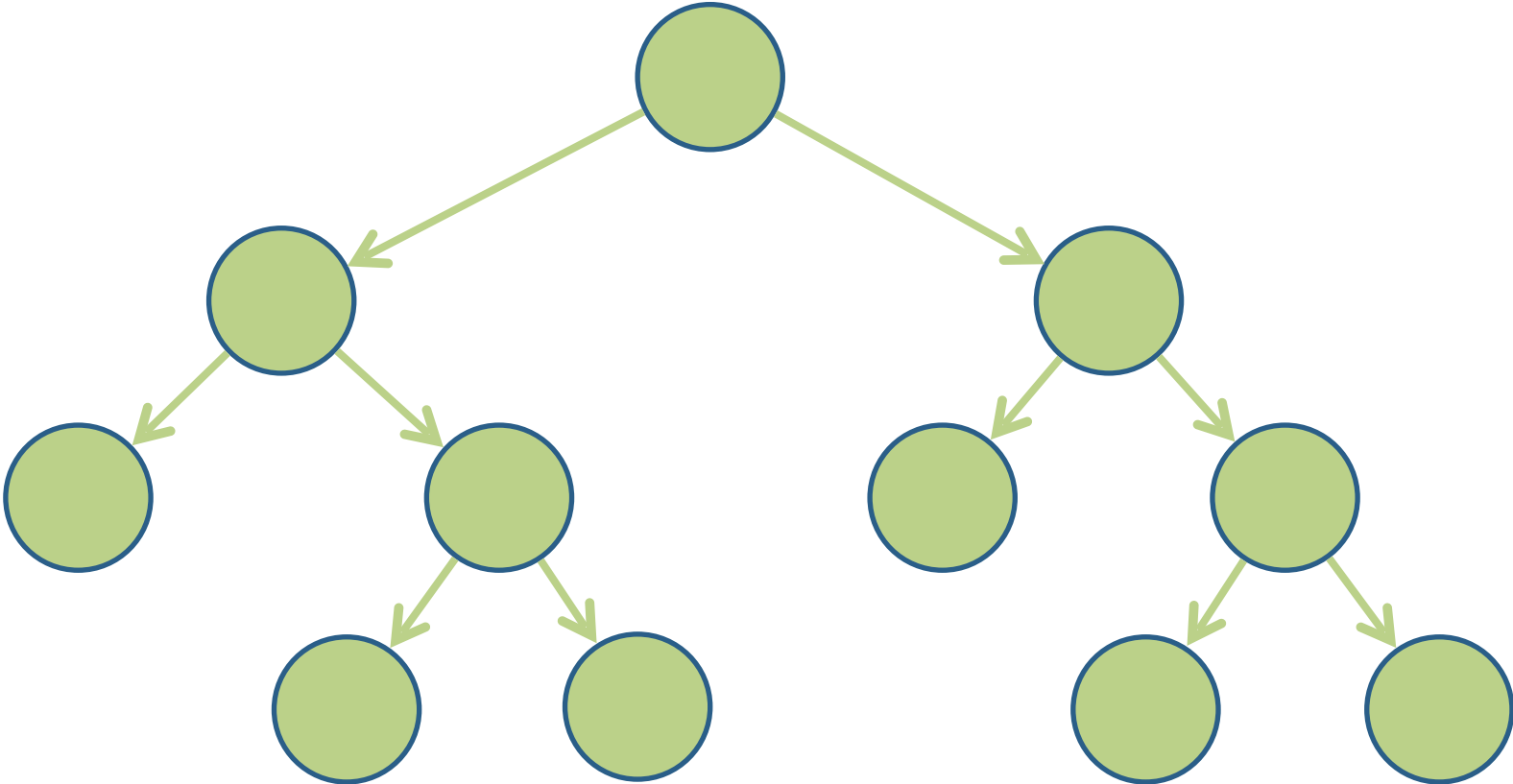
Roslyn

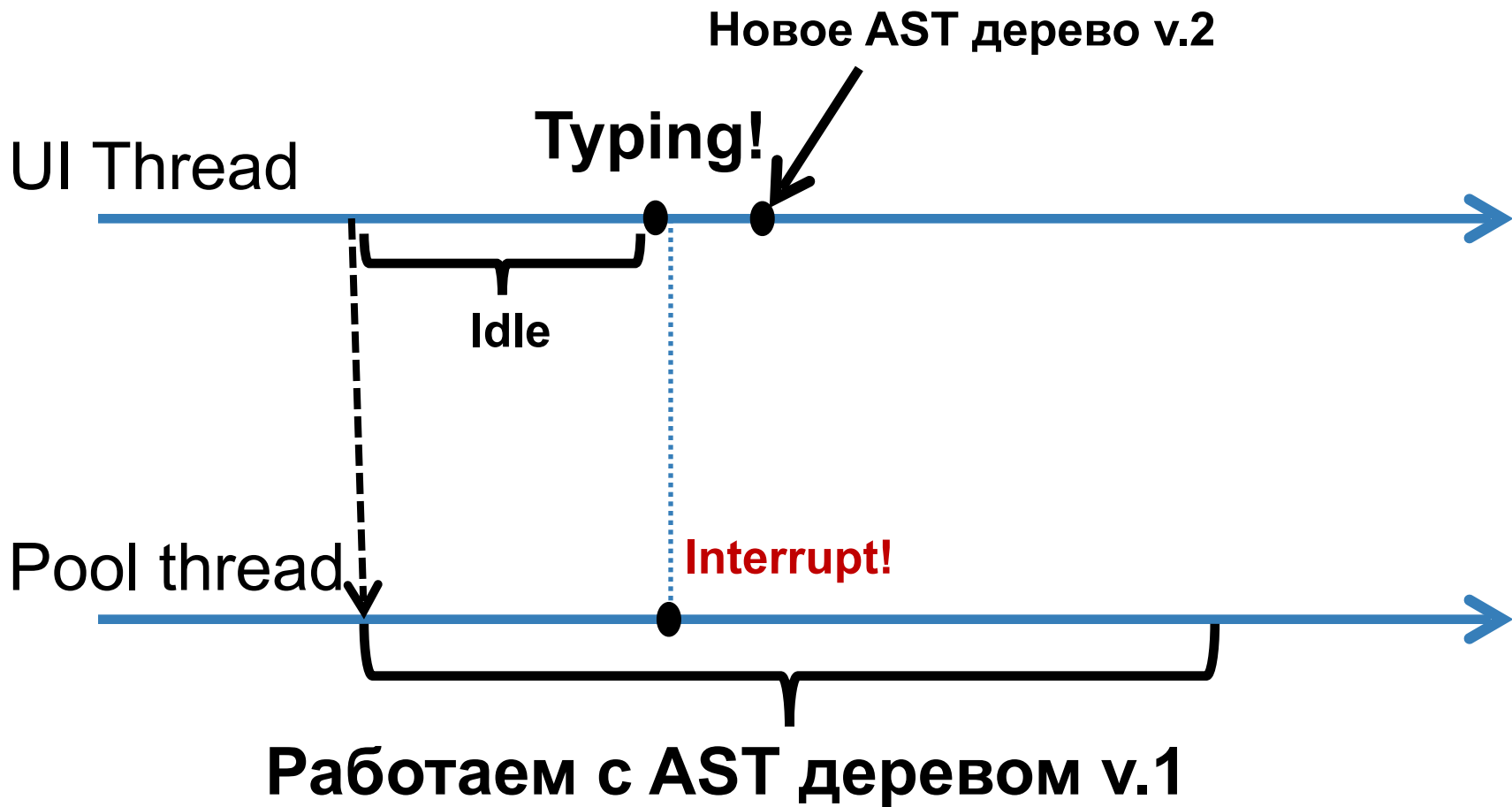
- *Neal M. Gafter, “Efficient immutable syntax representation with incremental change”*
- Полностью Immutable
- Максимально ленивый
- Thread safe

Красно-зелёные деревья

- Зелёные ноды
 1. Не знают своего местоположения в буфере
 2. Знают только про своих детей
 3. Переиспользуемые
 4. Строятся снизу вверх
- Красные ноды
 1. Построены поверх зелёных
 2. Знают про родителей
 3. Знают где находятся
 4. Лениво строятся сверху вниз

Roslyn: обновление зелёных деревьев

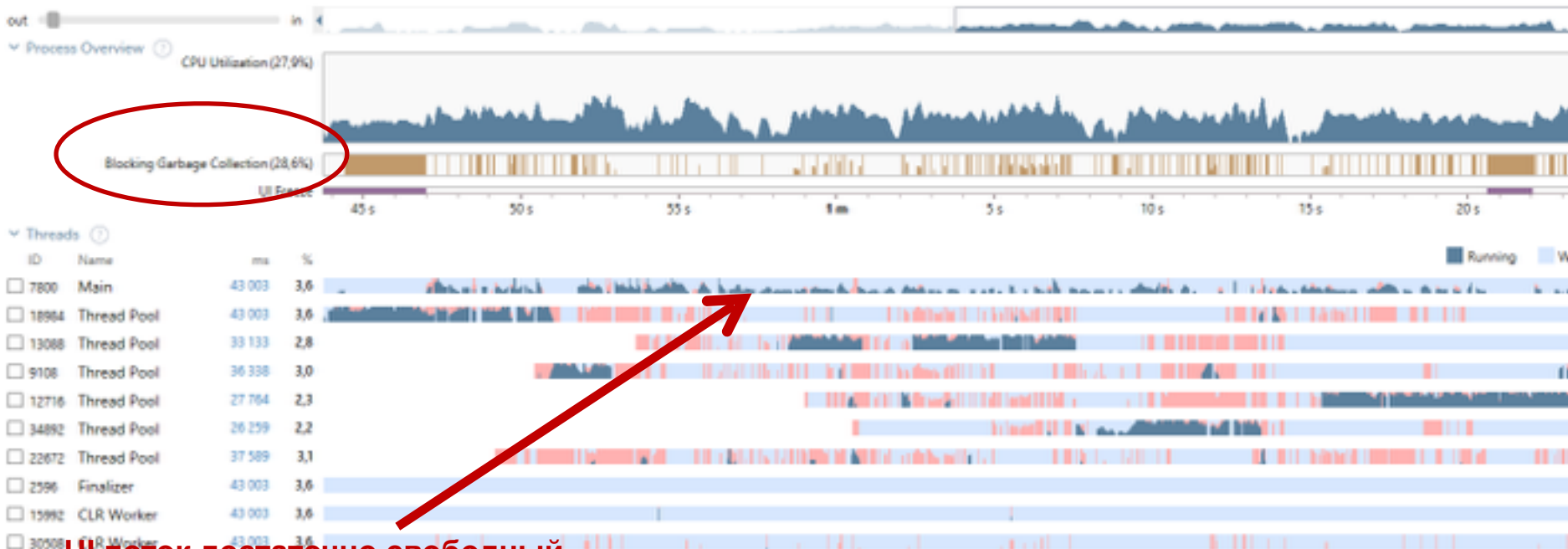




Roslyn: и здесь непрерывабельность

- Binding engine (resolve) переиспользует результаты резолва от разных запросов из разных потоков
- Binder не может понять, что все клиенты уже out-of-date и можно тоже прерваться
- В итоге код на уровне binding-a непрерывабельный => лишнее CPU time, memory traffic...

Roslyn: memory traffic problems

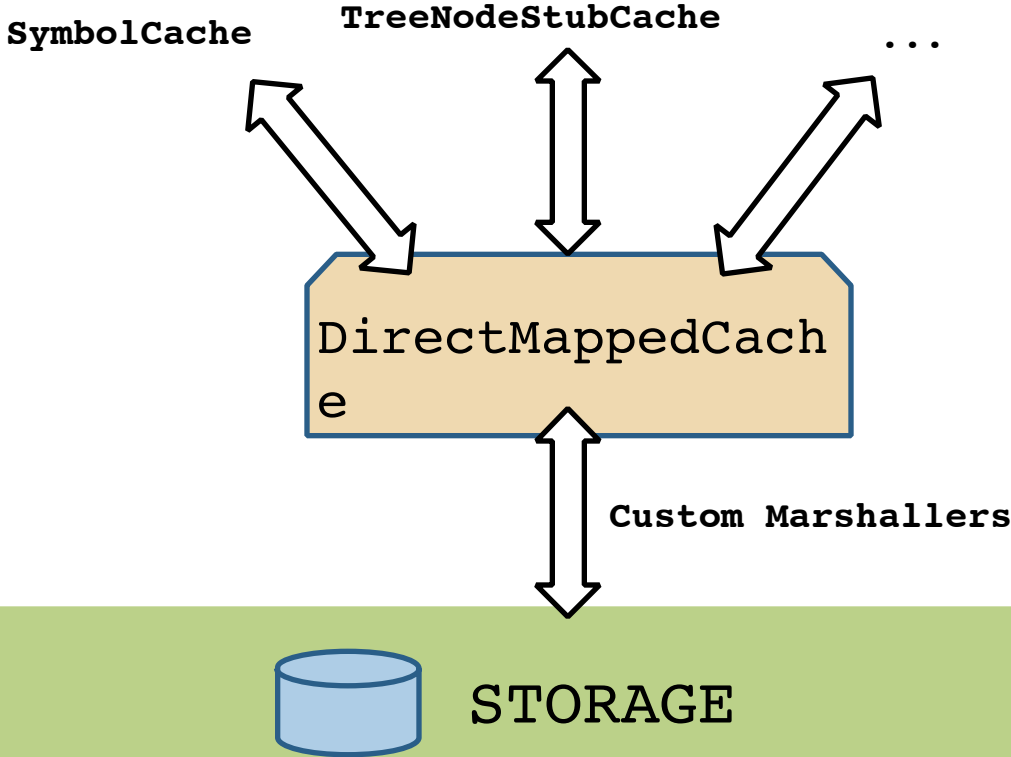


Roslyn: memory footprint

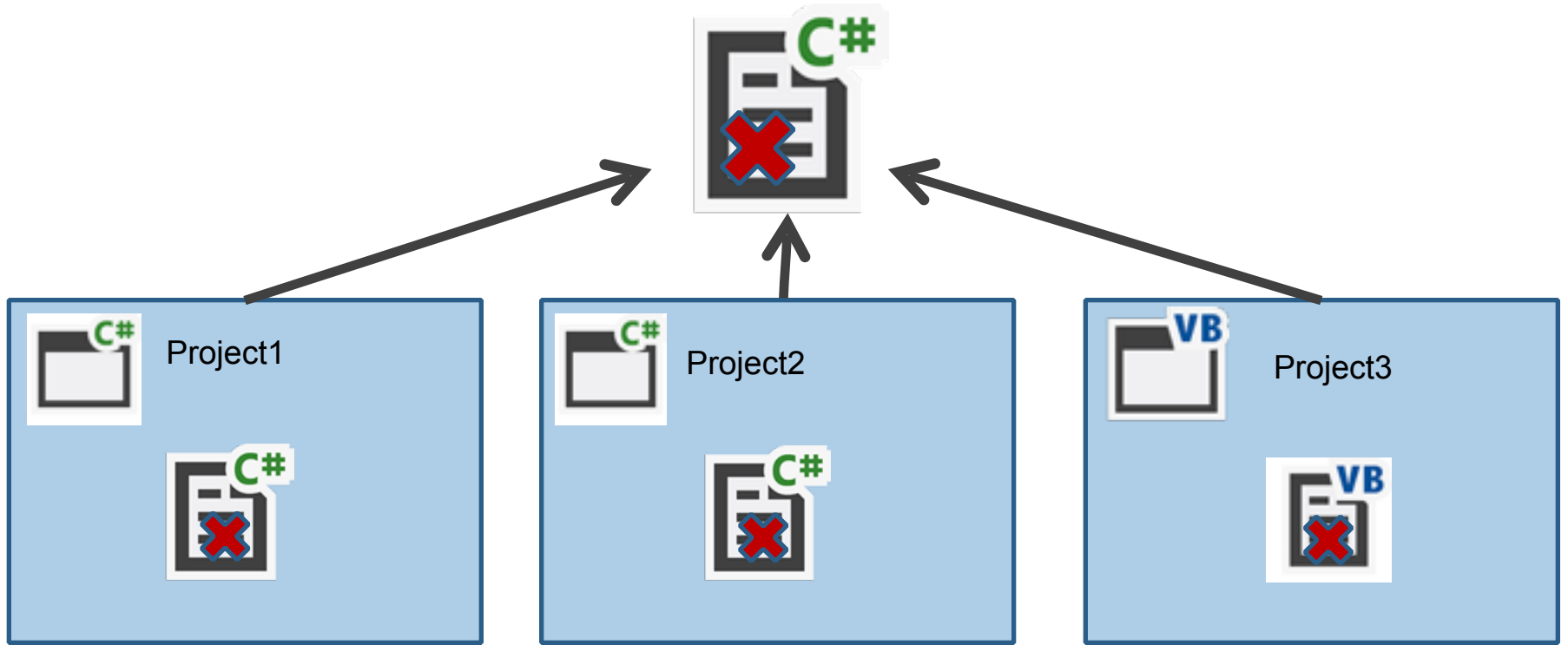
Type	Objects count	Bytes
80 Microsoft.CodeAnalysis.CSharp.CSharpCompilation	3,552,092	112,686,728
8,240 Microsoft.CodeAnalysis.CSharp.CSharpSyntaxTreeFactoryServiceFactory+CSharpSyntaxTreeFactoryService+RecoverableSyntaxTree	2,079,713	65,433,888
20,549 Microsoft.CodeAnalysis.CSharp.Symbols.SourceNamedTypeSymbol	1,985,936	63,921,111
610 Microsoft.Build.Evaluation.Project	1,047,962	53,015,924
80 Microsoft.CodeAnalysis.CSharp.Symbols.SourceAssemblySymbol	1,634,754	52,753,992
654,208 System.String	654,208	47,284,830
28,322 Microsoft.CodeAnalysis.CSharp.Syntax.CompilationUnitSyntax	1,175,671	37,047,356
2,946 Microsoft.CodeAnalysis.CSharp.Symbols.SourceNamespaceSymbol	1,046,708	35,590,721
18 Microsoft.CodeAnalysis.DeferredDocumentationProvider	838,108	28,715,022
446 Microsoft.CodeAnalysis.CSharp.Symbols.Retransforming.RetransformingModuleSymbol	799,141	22,280,119
Microsoft.VisualStudio.Language.QuickSearch.FileNameProvider.Cache	200,347	22,238,750
42,661 Microsoft.CodeAnalysis.CSharp.Symbols.Metadata.PE.PENamedTypeSymbol+PENamedTypeSymbolNonGeneric	395,839	21,466,978
49,338 Microsoft.CodeAnalysis.CSharp.Syntax.InternalSyntax.SyntaxList+WithManyChildren	728,529	20,185,782
68,601 Microsoft.CodeAnalysis.CSharp.Syntax.InternalSyntax.MethodDeclarationSyntax	573,981	18,711,570
37,799 Microsoft.CodeAnalysis.DocumentState	540,549	16,731,140
27,432 Microsoft.CodeAnalysis.CSharp.Syntax.ClassDeclarationSyntax	433,098	14,050,612
43,901 Microsoft.CodeAnalysis.CSharp.Syntax.InternalSyntax.AttributeListSyntax	372,853	11,665,660
168,437 Microsoft.CodeAnalysis.CSharp.SingleNamespaceDeclaration	575,841	10,679,220
29,652 Microsoft.CodeAnalysis.CSharp.Syntax.InternalSyntax.FieldDeclarationSyntax	326,102	9,878,762
71,612 Microsoft.Build.Construction.XmlElementWithLocation	408,263	8,493,656
54,745 Microsoft.CodeAnalysis.CSharp.Syntax.InternalSyntax.BlockSyntax	379,119	11,935,550
38,357 Microsoft.VisualStudio.Services.FileWatcher	237,072	8,116,124
34,755 Microsoft.CodeAnalysis.CSharp.Syntax.MethodDeclarationSyntax	250,207	8,018,828
JetBrains.Application.Parts.AttributeIndexedPartsCatalogue	149,615	7,473,834
24,078 Microsoft.CodeAnalysis.CSharp.Symbols.SourceMemberMethodSymbol	241,169	7,191,457
54,845 Microsoft.CodeAnalysis.CSharp.Syntax.BlockSyntax	291,369	6,947,928
28,320 Microsoft.CodeAnalysis.CSharp.CSharpSyntaxTree+ParsedSyntaxTree	82,682	6,359,402
110,607 Microsoft.CodeAnalysis.CSharp.Syntax.QualifiedNameSyntax	221,195	6,193,536

~1 Gb total (305 projects in .sln)

Memory management



Будет ли в Roslyn'e Solution Wide Analysis?



API

ReSharper

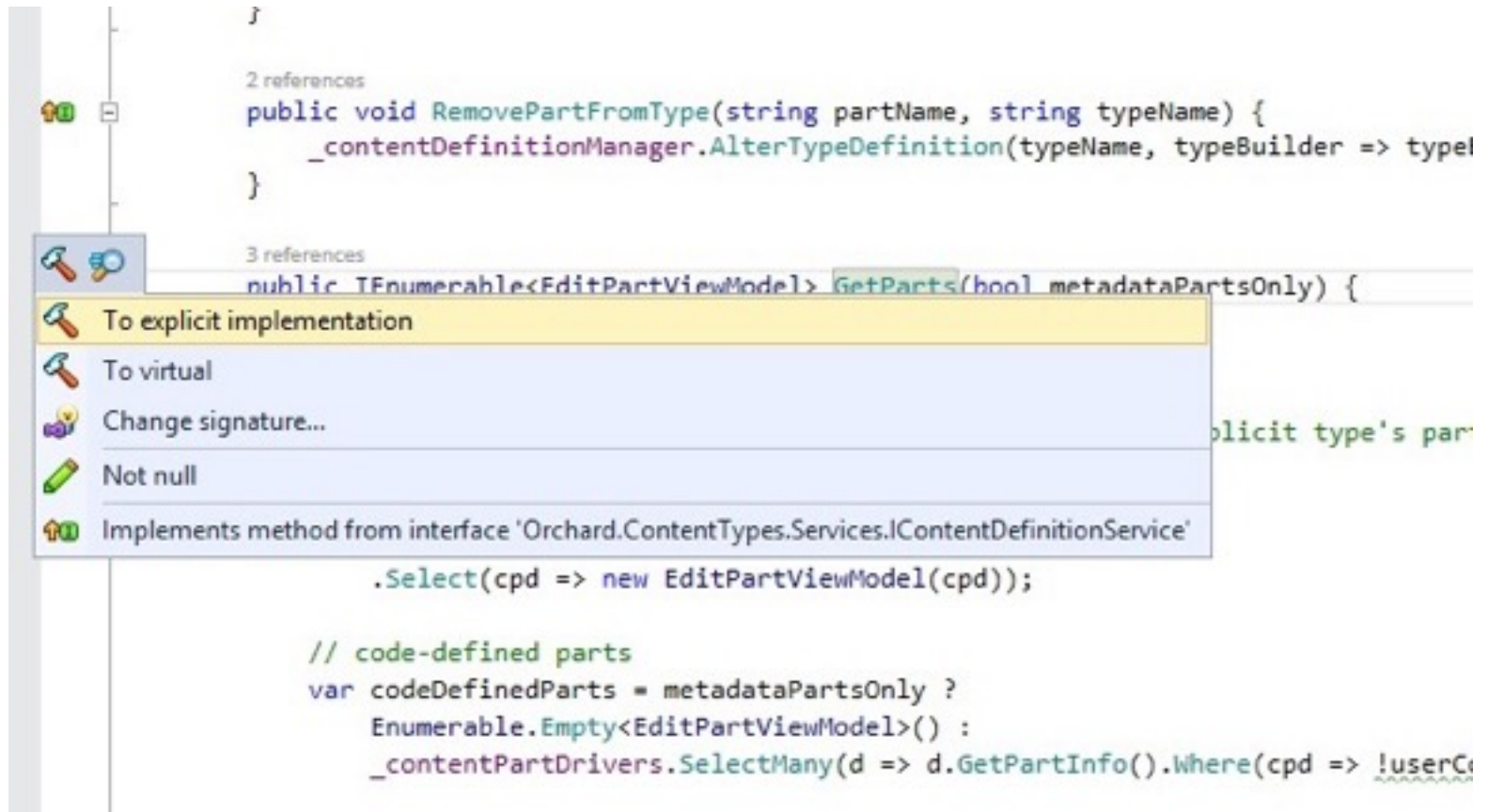
- Не 100% соответствие стандартам
- На руках есть «стэйт всего» – можно писать сложные и «глобальные» анализы
- В случае сложных фич нужно думать о синхронизации
- Уже очень большие и развесистые фреймворки для IDE фич самого разного рода

Roslyn

- CodeAnalyzers – а-ля post compile step
- Удобное API для того чтобы писать простые вещи
- Обработка ошибок размазана по многим уровням (parser, binder, analyzers...)
- Архитектурные ограничения на «глобальные» фичи

Roslyn + ReSharper

```
    }  
  
    2 references  
    public void RemovePartFromType(string partName, string typeName) {  
        _contentDefinitionManager.AlterTypeDefinition(typeName, typeBuilder => typeB  
    }  
  
    3 references  
    public IEnumerable<EditPartViewModel> GetParts(bool metadataPartsOnly) {  
  
        .Select(cpd => new EditPartViewModel(cpd));  
  
        // code-defined parts  
        var codeDefinedParts = metadataPartsOnly ?  
            Enumerable.Empty<EditPartViewModel>() :  
            _contentPartDrivers.SelectMany(d => d.GetPartInfo().Where(cpd => !userC
```



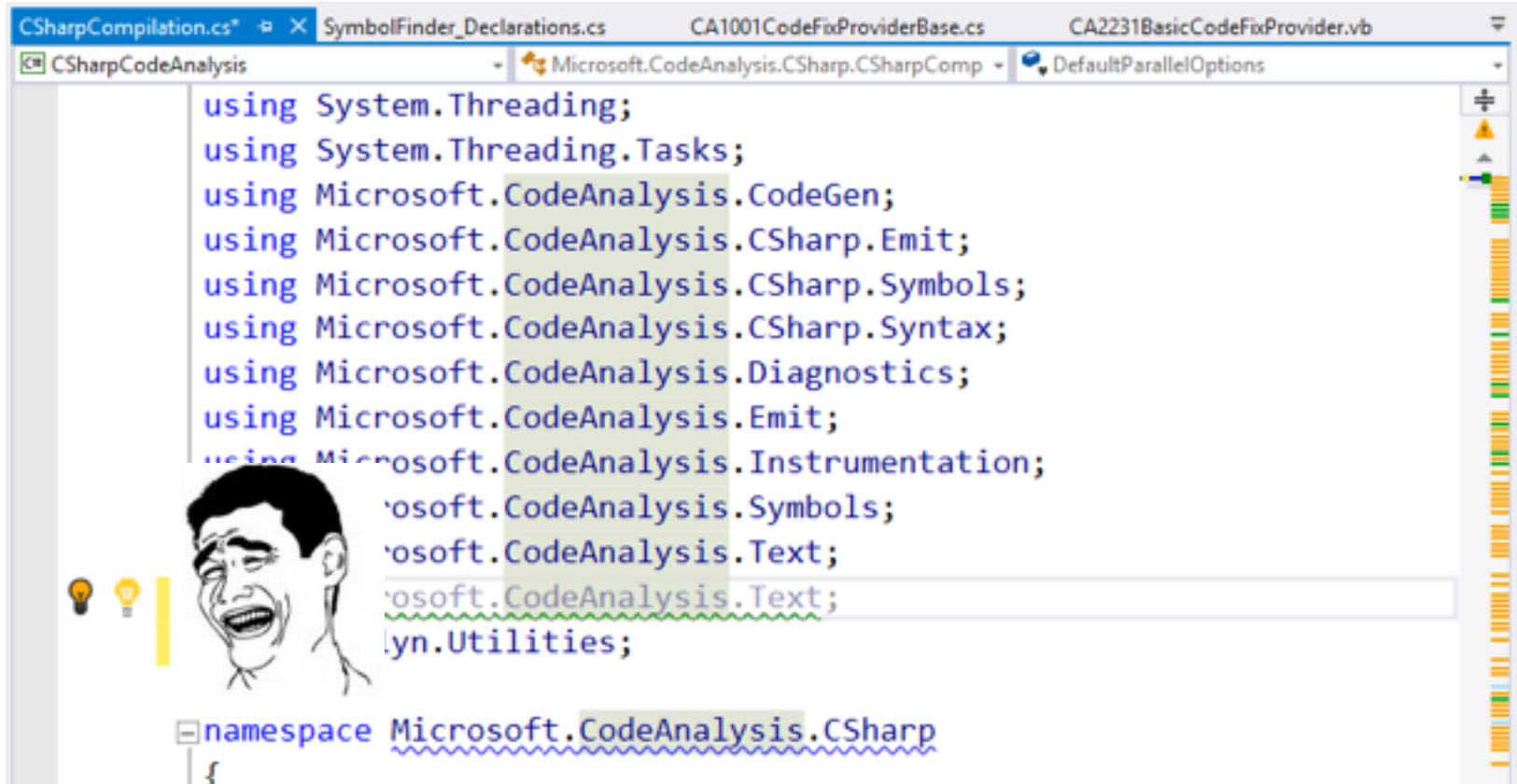
The screenshot shows a code editor with a ReSharper context menu open over the `GetParts` method signature. The menu items are:

- To explicit implementation
- To virtual
- Change signature...
- Not null
- Implements method from interface 'Orchard.ContentTypes.Services.IContentDefinitionService'

The code in the background includes:

```
    }  
  
    2 references  
    public void RemovePartFromType(string partName, string typeName) {  
        _contentDefinitionManager.AlterTypeDefinition(typeName, typeBuilder => typeB  
    }  
  
    3 references  
    public IEnumerable<EditPartViewModel> GetParts(bool metadataPartsOnly) {  
  
        .Select(cpd => new EditPartViewModel(cpd));  
  
        // code-defined parts  
        var codeDefinedParts = metadataPartsOnly ?  
            Enumerable.Empty<EditPartViewModel>() :  
            _contentPartDrivers.SelectMany(d => d.GetPartInfo().Where(cpd => !userC
```

Roslyn + ReSharper vs. Users



The screenshot shows an IDE window with several tabs: CSharpCompilation.cs*, SymbolFinder_Declarations.cs, CA1001CodeFixProviderBase.cs, and CA2231BasicCodeFixProvider.vb. The active file is CSharpCodeAnalysis.cs. The code contains several 'using' statements for the Microsoft.CodeAnalysis.CSharp namespace. A lightbulb icon is visible on the left side of the editor. A meme of a laughing man is overlaid on the code. The code is as follows:

```
using System.Threading;
using System.Threading.Tasks;
using Microsoft.CodeAnalysis.CodeGen;
using Microsoft.CodeAnalysis.CSharp.Emit;
using Microsoft.CodeAnalysis.CSharp.Symbols;
using Microsoft.CodeAnalysis.CSharp.Syntax;
using Microsoft.CodeAnalysis.Diagnostics;
using Microsoft.CodeAnalysis.Emit;
using Microsoft.CodeAnalysis.Instrumentation;
using Microsoft.CodeAnalysis.Symbols;
using Microsoft.CodeAnalysis.Text;
using Microsoft.CodeAnalysis.Text;
using System.Utilities;

namespace Microsoft.CodeAnalysis.CSharp
{
```

Что дальше?

ReSharper

- Ultimate IDE с огромной функциональностью и поддержкой многих языков
- «Сладкая» жизнь вместе с Roslyn в 32-битном процессе Visual Studio

Roslyn

- Открытый, добротный компилятор с небольшим количеством хорошо работающих фич
- Всё же не идеально быстрый из-за проблем с трафиком
- Вангую: будут пытаться выйти из 32-битного процесса Visual Studio

Вопросы?



/kskrygan



Kirill.Skrygan@jetbrains.com



zombieprogramming.blogspot.com

Кирилл Скрыган, JetBrains