CUSTIS® ИТ-РЕШЕНИЯ ДЛЯ РАЗВИТИЯ

Тайна динамических сборок

Игорь Шаталкин

Георгий Минашин

DotNext
21 апреля 2021

Системнов

Системнов

Пуаро, здравствуйте!

Системнов

Пуаро, здравствуйте!

Добрый день👋

Системнов

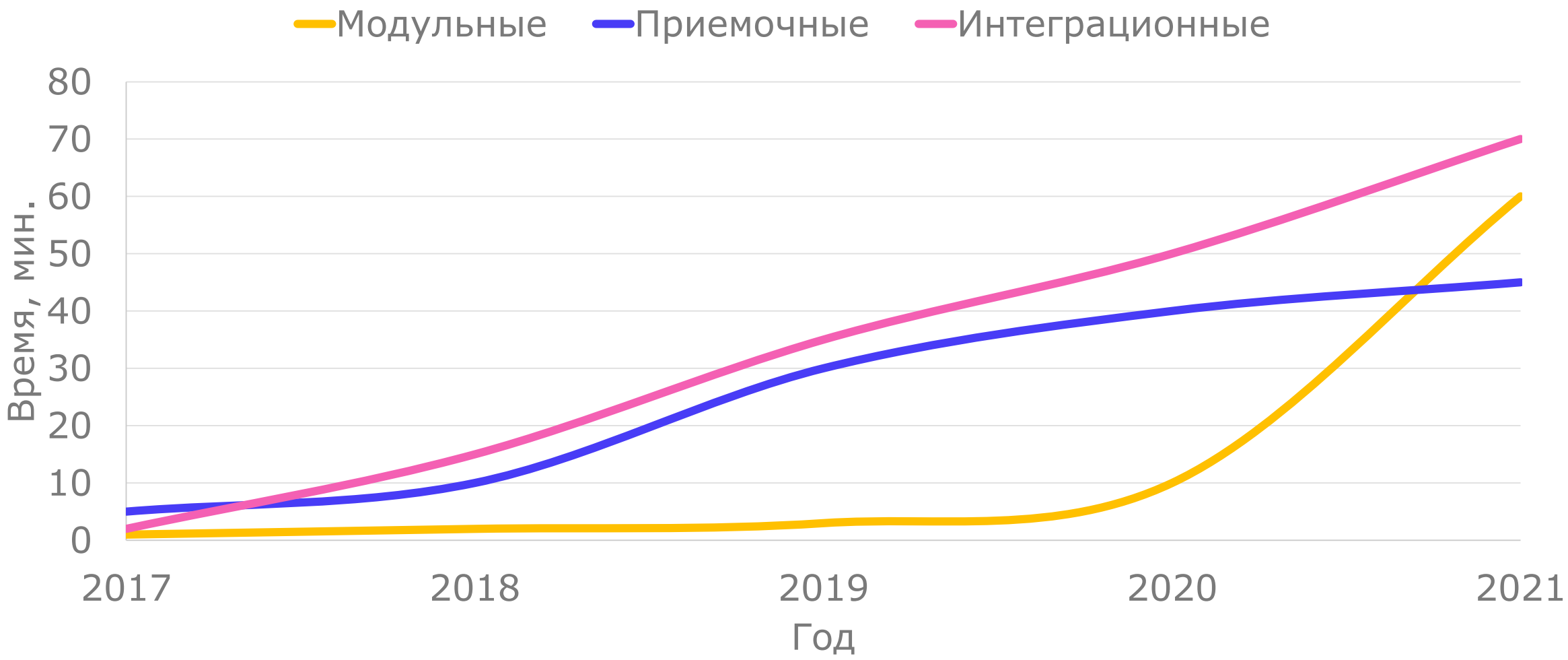
Пуаро, здравствуйте!

Добрый день👋

Возникла проблема, не можем решить 😖
Модульные тесты очень долгие, по 60 минут...

CUSTIS

Время работы тестов



Системнов

Пуаро, здравствуйте!

Добрый день👋

Возникла проблема, не можем решить😖
Модульные тесты очень долгие, по 60 минут...

Системнов

Пуаро, здравствуйте!

Добрый день👋

Возникла проблема, не можем решить😖
Модульные тесты очень долгие, по 60 минут...

А что используете при разработке?

Системнов

Пуаро, здравствуйте!

Добрый день👋

Возникла проблема, не можем решить 😖
Модульные тесты очень долгие, по 60 минут...

А что используете при разработке?

C#, .NET 5, NUnit, Moq, React

Системнов

Добрый день👋

Возникла проблема, не можем решить😖
Модульные тесты очень долгие, по 60 минут...

А что используете при разработке?

C#, .NET 5, NUnit, Moq, React

TDD?

Системнов

Возникла проблема, не можем решить 😖
Модульные тесты очень долгие, по 60 минут...

А что используете при разработке?

C#, .NET 5, NUnit, Moq, React

TDD?

Да, внедрили пару лет назад

Системнов

Модульные тесты очень долгие, по 60 минут...

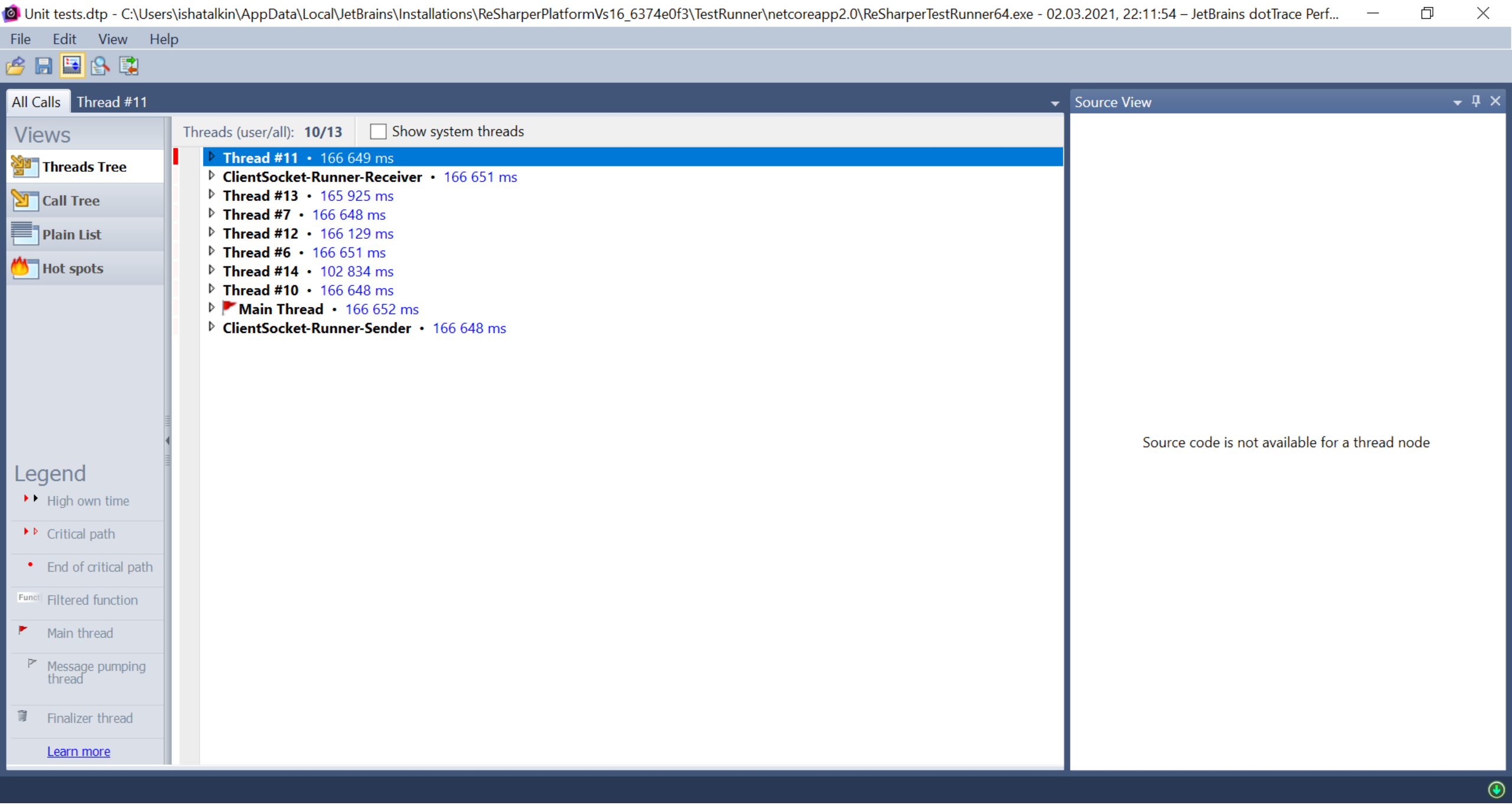
А что используете при разработке?

C#, .NET 5, NUnit, Moq, React

TDD?

Да, внедрили пару лет назад

Профайлером смотрели?


Unit tests.dtp - C:\Users\ishatalkin\AppData\Local\JetBrains\Installations\ReSharperPlatformVs16_6374e0f3\TestRunner\netcoreapp2.0\ReSharperTestRunner64.exe - 02.03.2021, 22:11:54 – JetBrains dotTrace Perf...
File
Edit
View
Help
All Calls
Thread #11
Source View
Views
Threads Tree
Call Tree
Plain List
Hot spots
Threads (user/all): 10/13
Show system threads
Thread #11 • 166 649 ms
ClientSocket-Runner-Receiver • 166 651 ms
Thread #13 • 165 925 ms
Thread #7 • 166 648 ms
Thread #12 • 166 129 ms
Thread #6 • 166 651 ms
Thread #14 • 102 834 ms
Thread #10 • 166 648 ms
Main Thread • 166 652 ms
ClientSocket-Runner-Sender • 166 648 ms
Source code is not available for a thread node
Legend
High own time
Critical path
End of critical path
Filtered function
Main thread
Message pumping thread
Finalizer thread
Learn more

Системнов

Модульные тесты очень долгие, по 60 минут...

А что используете при разработке?

C#, .NET 5, NUnit, Moq, React

TDD?

Да, внедрили пару лет назад

Профайлером смотрели?

Системнов

А что используете при разработке?

C#, .NET 5, NUnit, Moq, React

TDD?

Да, внедрили пару лет назад

Профайлером смотрели?

Пробовали заменить Moq на самописные заглушки?

Системнов

Да, внедрили пару лет назад

Профайлером смотрели?

Пробовали заменить Moq
на самописные заглушки?

Да. Написал утилиту для
генерации классов-заглушек,
тесты стали работать шустро.
НО!!!

Системнов

Пробовали заменить Moq на самописные заглушки?

Да. Написал утилиту для генерации классов-заглушек, тесты стали работать шустро. НО!!!

Мы используем фишки Moq, а их сложно реализовать через генератор...

moq / moq4

Watch 189 Star 4k Fork 595

Code Issues 21 Pull requests Actions Projects Wiki Security Insights

Filters is:issue is:open performance Labels 21 Milestones 1 New issue

Clear current search query, filters, and sorts

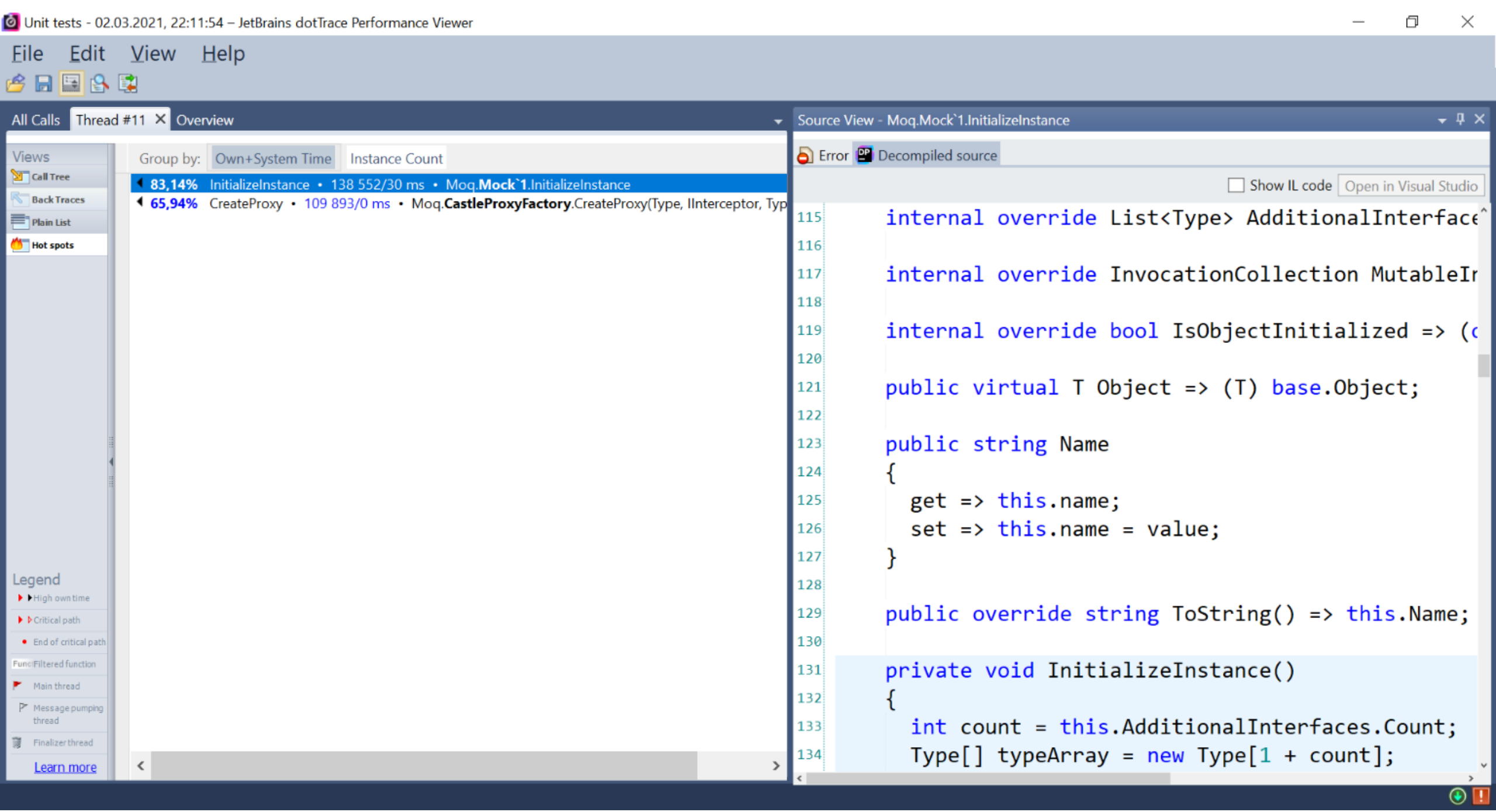
3 Open 26 Closed Author Label Projects Milestones Assignee Sort

DefaultValue.Mock lazy initialization needs-repro 1
#1149 opened 7 days ago by informatorius

Provide non-generic API to support runtime type mocking discussion enhancement 9
#887 opened on 11 Aug 2019 by bclothier

MockSequence and VerifyAll discussion enhancement up-for-grabs 32
#75 opened on 19 Dec 2013 by henriquemotaesteves

ProTip! Type g i on any issue or pull request to go back to the issue listing page.


Unit tests - 02.03.2021, 22:11:54 – JetBrains dotTrace Performance Viewer
File Edit View Help
All Calls Thread #11 Overview
Views
Call Tree
Back Traces
Plain List
Hot spots
Group by: Own+System Time Instance Count
83,14% InitializeInstance • 138 552/30 ms • Moq.Mock`1.InitializeInstance
65,94% CreateProxy • 109 893/0 ms • Moq.CastleProxyFactory.CreateProxy(Type, IInterceptor, Typ
Legend
High own time
Critical path
End of critical path
Filtered function
Main thread
Message pumping thread
Finalizer thread
Learn more
Source View - Moq.Mock`1.InitializeInstance
Error
Decompiled source
Show IL code
Open in Visual Studio
115 internal override List<Type> AdditionalInterface
116
117 internal override InvocationCollection MutableIr
118
119 internal override bool IsObjectInitialized => (
120
121 public virtual T Object => (T) base.Object;
122
123 public string Name
124 {
125 get => this.name;
126 set => this.name = value;
127 }
128
129 public override string ToString() => this.Name;
130
131 private void InitializeInstance()
132 {
133 int count = this.AdditionalInterfaces.Count;
134 Type[] typeArray = new Type[1 + count];

Системнов

Пробовали заменить Moq на самописные заглушки?

Да. Написал утилиту для генерации классов-заглушек, тесты стали работать шустро. НО!!!

Мы используем фишки Moq, а их сложно реализовать через генератор...

Системнов

тесты стали работать шустро. НО!!!

Мы используем фишки Moq, а их сложно реализовать через генератор...

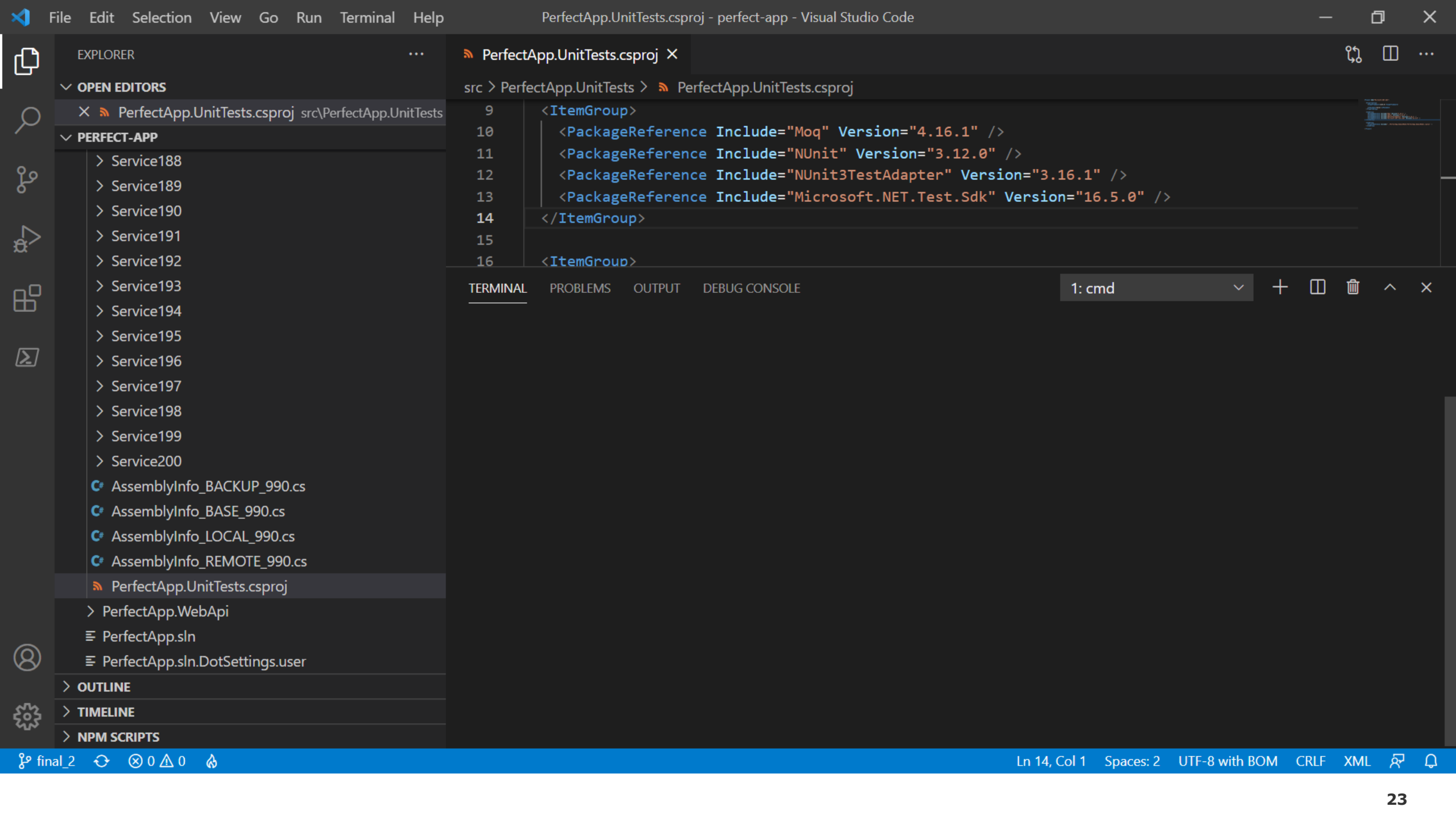
Погуглите, может, кто-то еще сталкивался с тормозами Mock.InitializeInstace или CastleProxyFactory.CreateInstance?

Системнов

Мы используем фишки Moq, а их сложно реализовать через генератор...

Погуглите, может, кто-то еще сталкивался с тормозами Mock.InitializeInstace или CastleProxyFactory.CreateInstance?

Искал, ничего не нашел 🤷


File Edit Selection View Go Run Terminal Help
PerfectApp.UnitTests.csproj - perfect-app - Visual Studio Code
EXPLORER
OPEN EDITORS
PerfectApp.UnitTests.csproj src\PerfectApp.UnitTests
PERFECT-APP
Service188
Service189
Service190
Service191
Service192
Service193
Service194
Service195
Service196
Service197
Service198
Service199
Service200
AssemblyInfo_BACKUP_990.cs
AssemblyInfo_BASE_990.cs
AssemblyInfo_LOCAL_990.cs
AssemblyInfo_REMOTE_990.cs
PerfectApp.UnitTests.csproj
PerfectApp.WebApi
PerfectApp.sln
PerfectApp.sln.DotSettings.user
OUTLINE
TIMELINE
NPM SCRIPTS
PerfectApp.UnitTests.csproj
src > PerfectApp.UnitTests > PerfectApp.UnitTests.csproj
9 <ItemGroup>
10 <PackageReference Include="Moq" Version="4.16.1" />
11 <PackageReference Include="NUnit" Version="3.12.0" />
12 <PackageReference Include="NUnit3TestAdapter" Version="3.16.1" />
13 <PackageReference Include="Microsoft.NET.Test.Sdk" Version="16.5.0" />
14 </ItemGroup>
15
16 <ItemGroup>
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE
1: cmd
final_2 0 0
Ln 14, Col 1 Spaces: 2 UTF-8 with BOM CRLF XML

Значимые факты

- Рост времени прогона модульных тестов
- Использование Moq
- Внедрение TDD
- Много времени занимает создание объектов-заглушек
- Время UAT и интеграционных тестов растет не так быстро, как время модульных

Подозреваемые

- Оборудование
- Moq
- NUnit

Системнов

Мы используем фишки Moq, а их сложно реализовать через генератор...

Погуглите, может, кто-то еще сталкивался с тормозами Mock.InitializeInstace или CastleProxyFactory.CreateInstance?

Искал, ничего не нашел 🤷

Системнов

а их сложно реализовать через генератор...

Погуглите, может, кто-то еще сталкивался с тормозами Mock.InitializeInstace или CastleProxyFactory.CreateInstance?

Искал, ничего не нашел 🤷

Меняли оборудование для CI?

Системнов

Погуглите, может, кто-то еще сталкивался с тормозами Mock.InitializeInstace или CastleProxyFactory.CreateInstance?

Искал, ничего не нашел 🤷

Меняли оборудование для CI?

Нет

Демонстрация

- GitExtensions
- NSubstitute
- xUnit

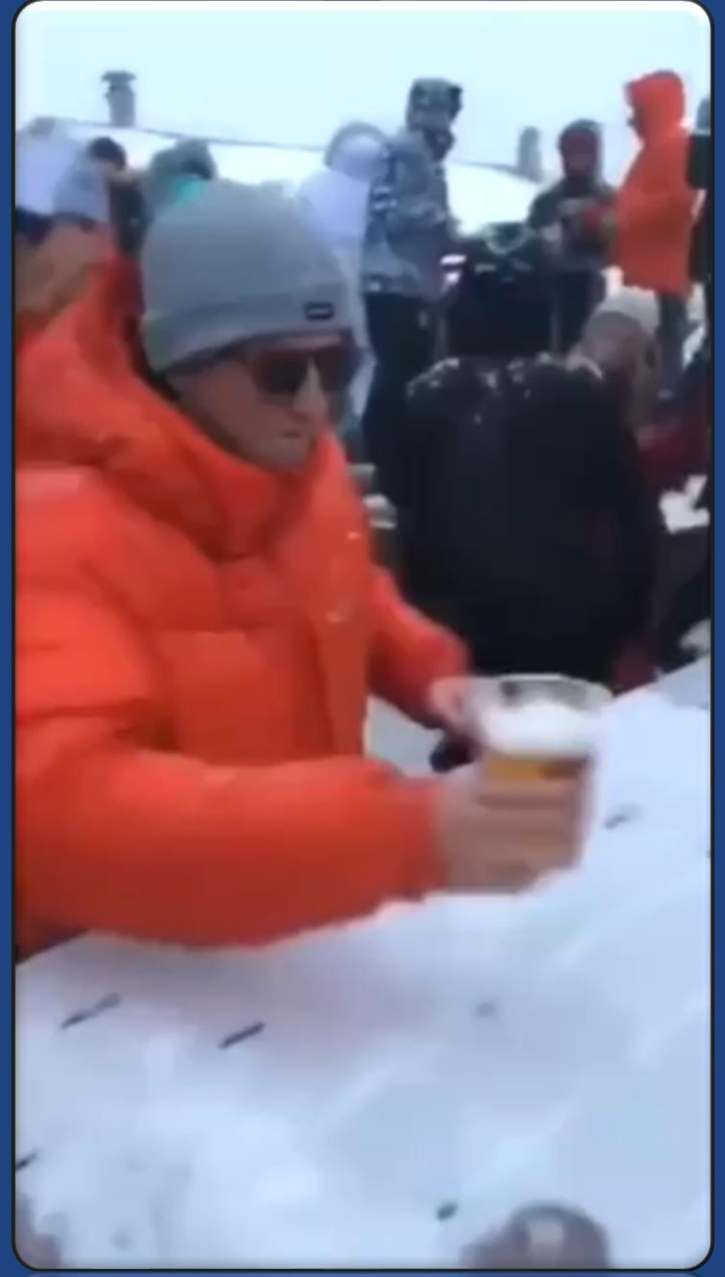
CUSTIS

Время работы одного теста

- Всего тестов: 7400
- Общее время работы: 2,5 мин = 150 с
- Среднее время теста: 20 мс

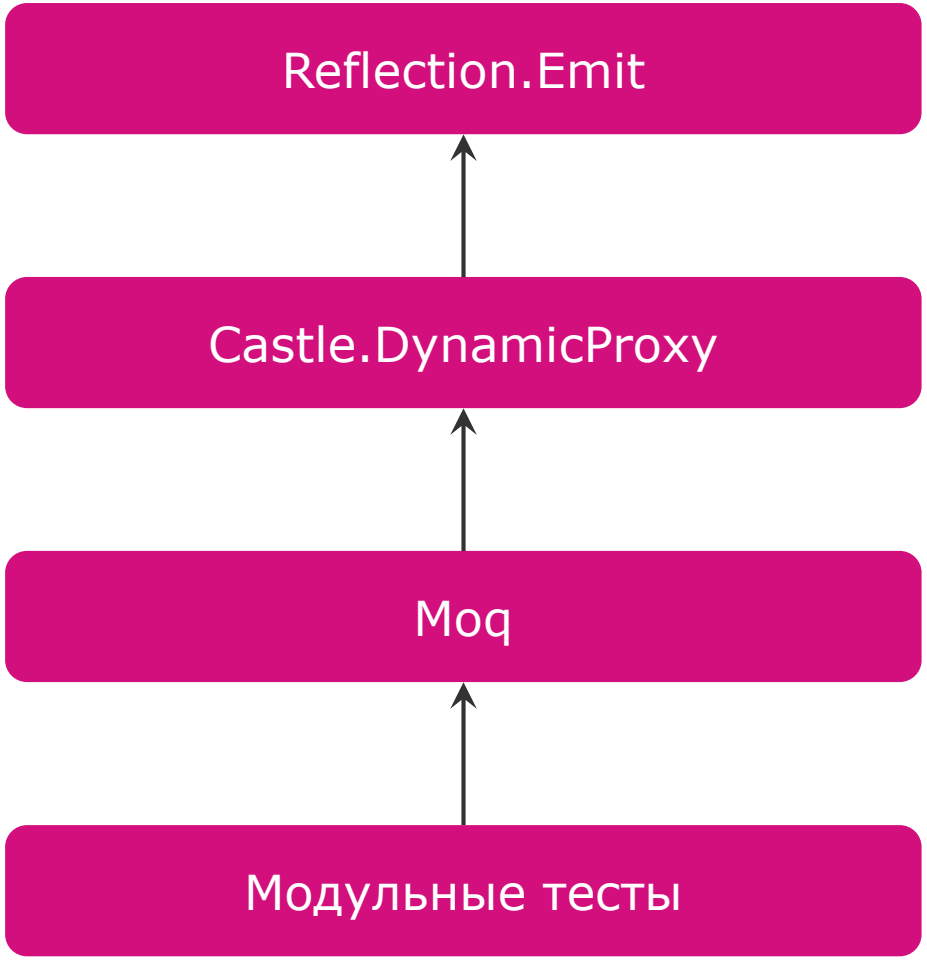
dotTrace

Демонстрация Show System Functions



Источник: https://t.me/loldev/816

Reflection.Emit



Реальные и динамические типы

	Реальные типы	Динамические типы
Источник кода	Написание кода	Генерация кода
Момент	Design Time	Runtime
Сборка	Реальная сборка	Динамическая сборка

ORM

Что пишет программист

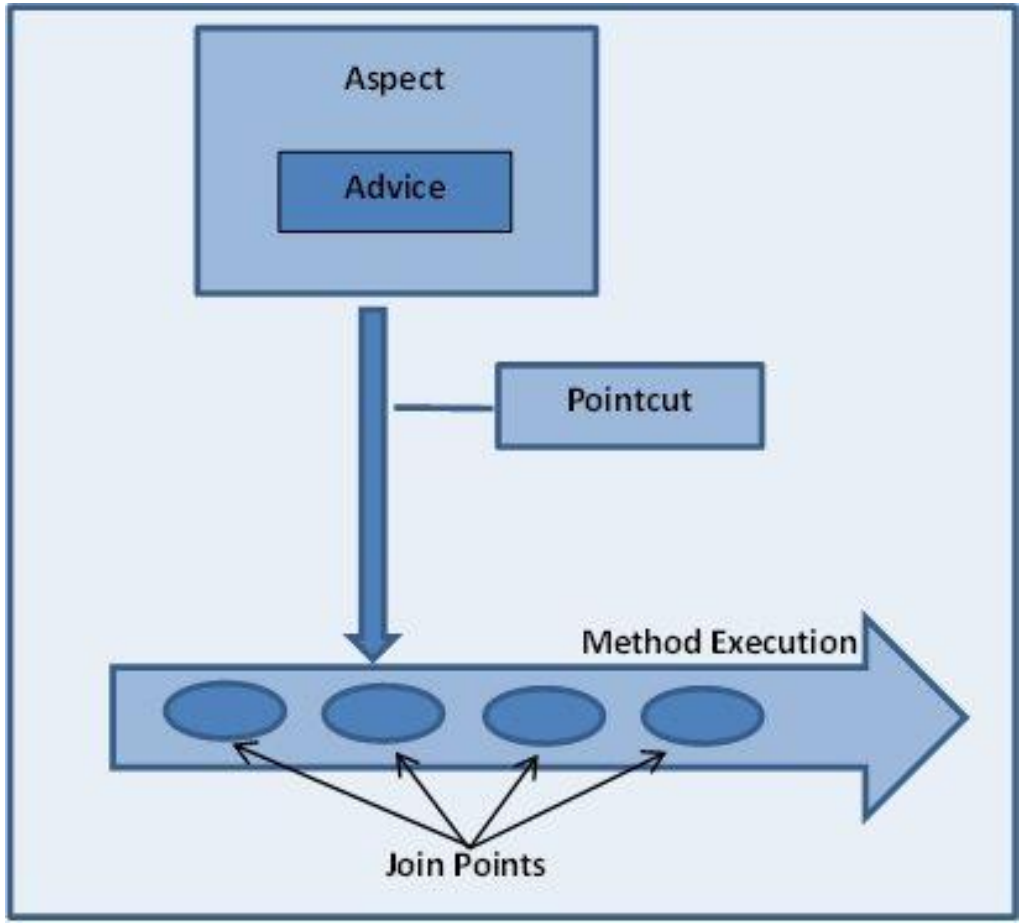
```
public class Foo
{
    public string Name { get; set; }
}
```

Что нужно ORM

```
public class FooProxy
{
    private string _name;
    public string Name
    {
        get => _name;
        set
        {
            if (_name != value)
            {
                Changed = true;
                _name = value;
            }
        }
    }

    public bool Changed { get; set; }
}
```

АОП



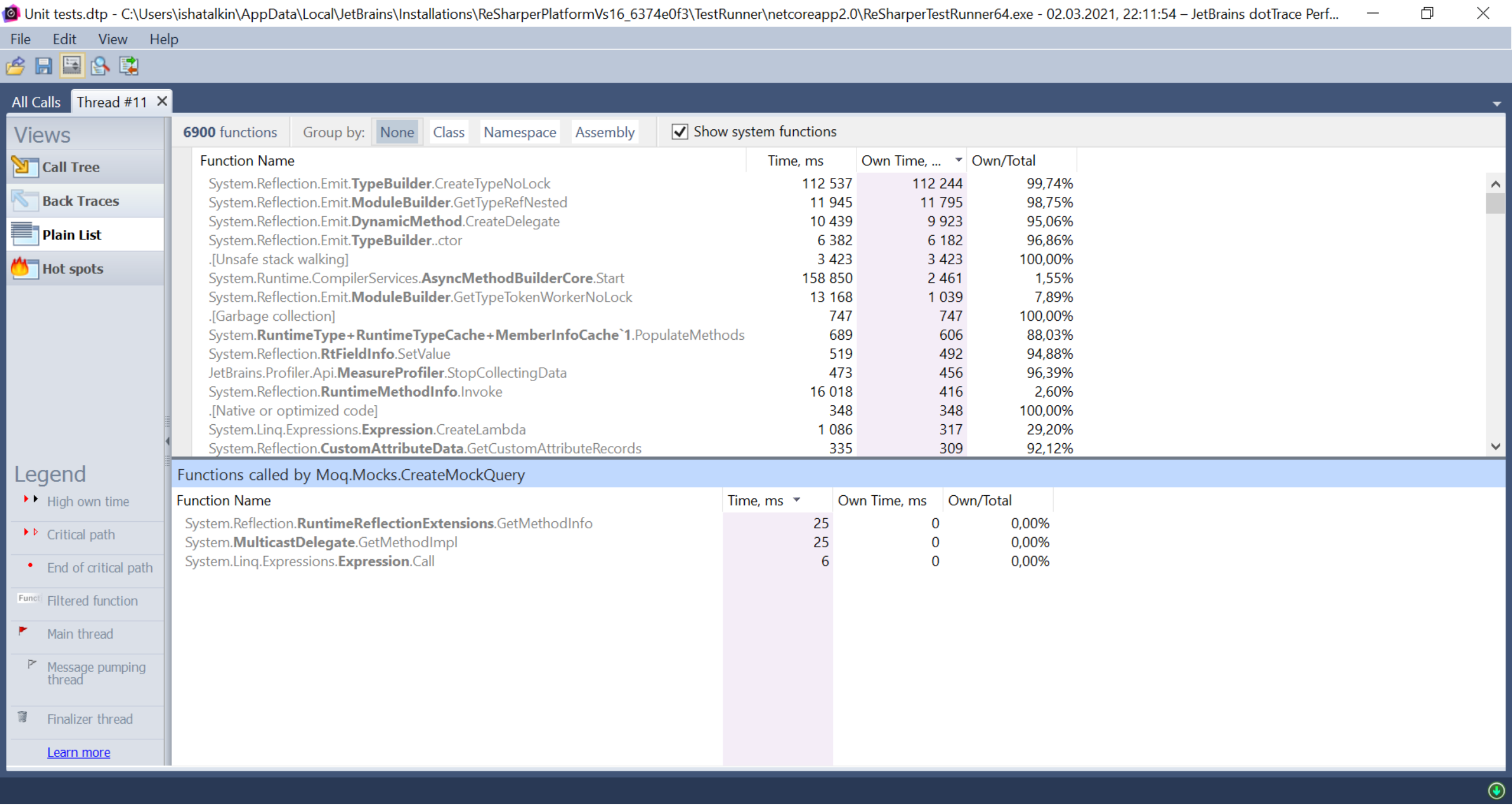
Источник: https://howtodoinjava.com/spring-aop/spring-aop-aspectj-example-tutorial-using-annotation-config/

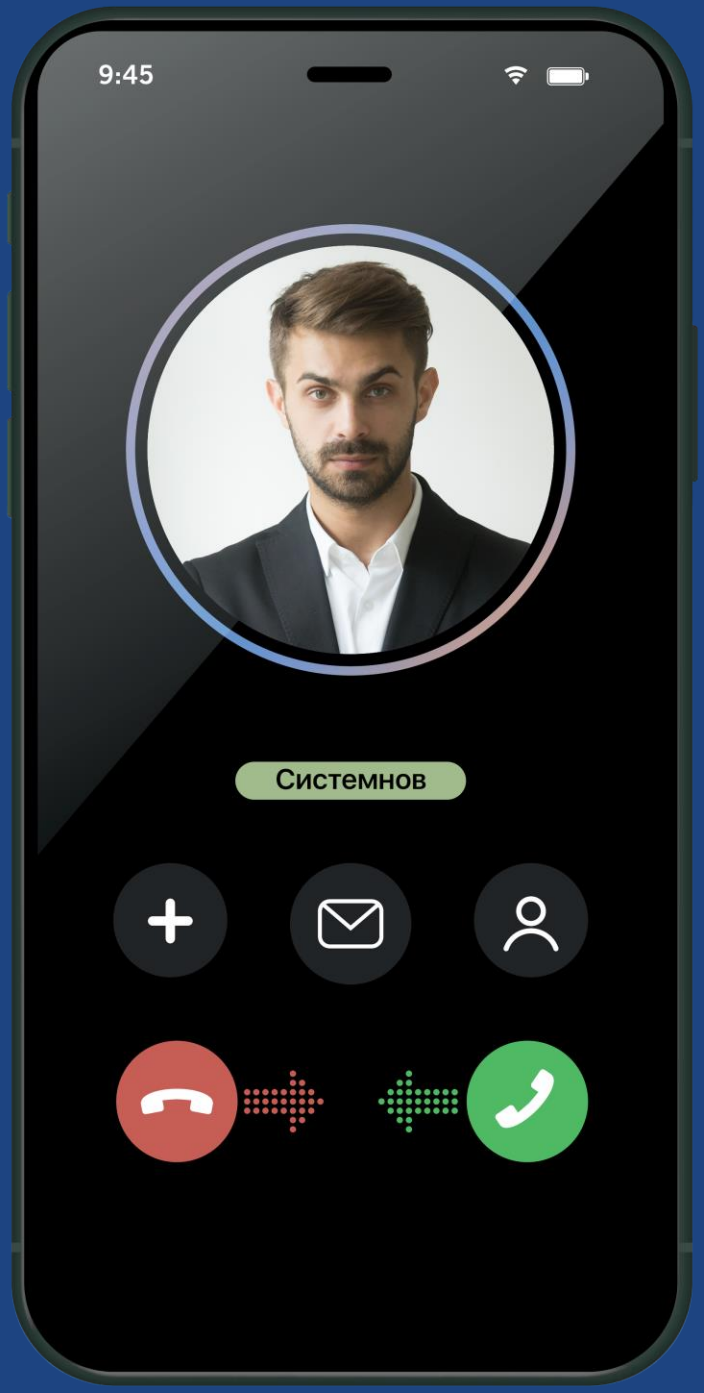
Демонстрация

- Reflection.Emit
- Castle.DynamicProxy

Локализация проблемы

- Прогон одного теста проходит быстро
- Прогон того же теста вместе с другими проходит медленно
- Самый медленный метод — `TypeBuilder.CreateTypeNoLock`


Unit tests.dtp - C:\Users\ishatalkin\AppData\Local\JetBrains\Installations\ReSharperPlatformVs16_6374e0f3\TestRunner\netcoreapp2.0\ReSharperTestRunner64.exe - 02.03.2021, 22:11:54 – JetBrains dotTrace Perf...
File Edit View Help
All Calls
Thread #11
Views
Call Tree
Back Traces
Plain List
Hot spots
6900 functions
Group by: None Class Namespace Assembly
Show system functions
Function Name
Time, ms
Own Time, ...
Own/Total
System.Reflection.Emit.TypeBuilder.CreateTypeNoLock 112 537 112 244 99,74%
System.Reflection.Emit.ModuleBuilder.GetTypeRefNested 11 945 11 795 98,75%
System.Reflection.Emit.DynamicMethod.CreateDelegate 10 439 9 923 95,06%
System.Reflection.Emit.TypeBuilder..ctor 6 382 6 182 96,86%
.[Unsafe stack walking] 3 423 3 423 100,00%
System.Runtime.CompilerServices.AsyncMethodBuilderCore.Start 158 850 2 461 1,55%
System.Reflection.Emit.ModuleBuilder.GetTypeTokenWorkerNoLock 13 168 1 039 7,89%
.[Garbage collection] 747 747 100,00%
System.RuntimeType+RuntimeTypeCache+MemberInfoCache`1.PopulateMethods 689 606 88,03%
System.Reflection.RtFieldInfo.SetValue 519 492 94,88%
JetBrains.Profiler.Api.MeasureProfiler.StopCollectingData 473 456 96,39%
System.Reflection.RuntimeMethodInfo.Invoke 16 018 416 2,60%
.[Native or optimized code] 348 348 100,00%
System.Linq.Expressions.Expression.CreateLambda 1 086 317 29,20%
System.Reflection.CustomAttributeData.GetCustomAttributeRecords 335 309 92,12%
Functions called by Moq.Mocks.CreateMockQuery
Function Name
Time, ms
Own Time, ms
Own/Total
System.Reflection.RuntimeReflectionExtensions.GetMethodInfo 25 0 0,00%
System.MulticastDelegate.GetMethodImpl 25 0 0,00%
System.Linq.Expressions.Expression.Call 6 0 0,00%
Legend
High own time
Critical path
End of critical path
Filtered function
Main thread
Message pumping thread
Finalizer thread
Learn more


9:45
Системнов

CUSTIS

Источник: https://t.me/loldev/176

Демонстрация

- Размещение типов в разных сборках
- Minidump
- Встройка в Moq

Документация CLI

II.6 An assembly is a set of one or more files deployed as a unit.

II.6 A module is a single file containing executable content in the format specified here. If the module contains a manifest then it also specifies the modules (including itself) that constitute the assembly. An assembly shall contain only one manifest amongst all its constituent files.

II.6.5 When an item is in the current assembly, but is part of a module other than the one containing the manifest, the defining module shall be declared in the manifest of the assembly using the .module extern directive.

Источник: ECMA-335 Common Language Infrastructure (CLI)

Документация CLI (вольный перевод)

II.6 Сборка — это один или несколько файлов, публикуемых как единое целое

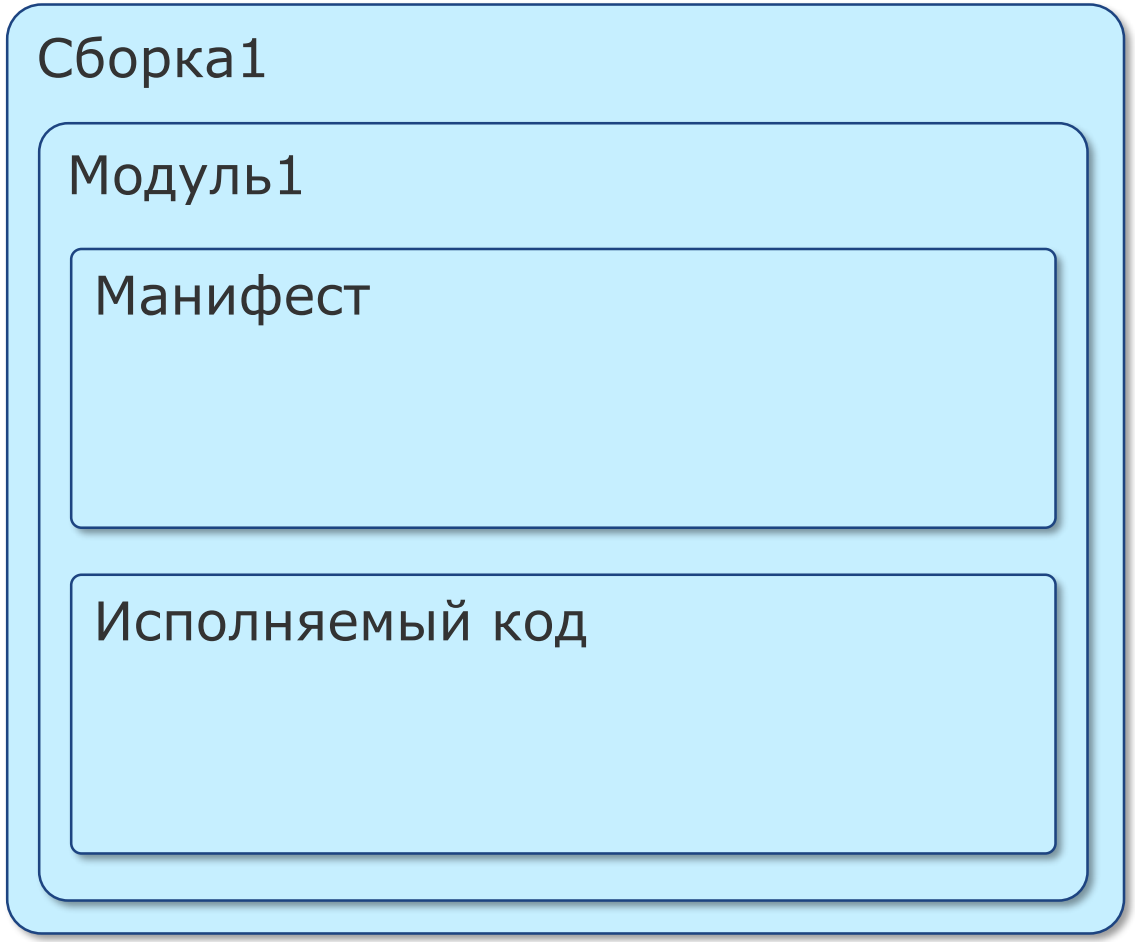
II.6 Модуль — это файл, содержащий исполняемый код. Один (и только один) из модулей содержит манифест, в котором описаны все модули, включая модуль с манифестом

II.6.5 Если элемент объявлен в модуле без манифеста, то такой модуль должен быть указан в манифесте

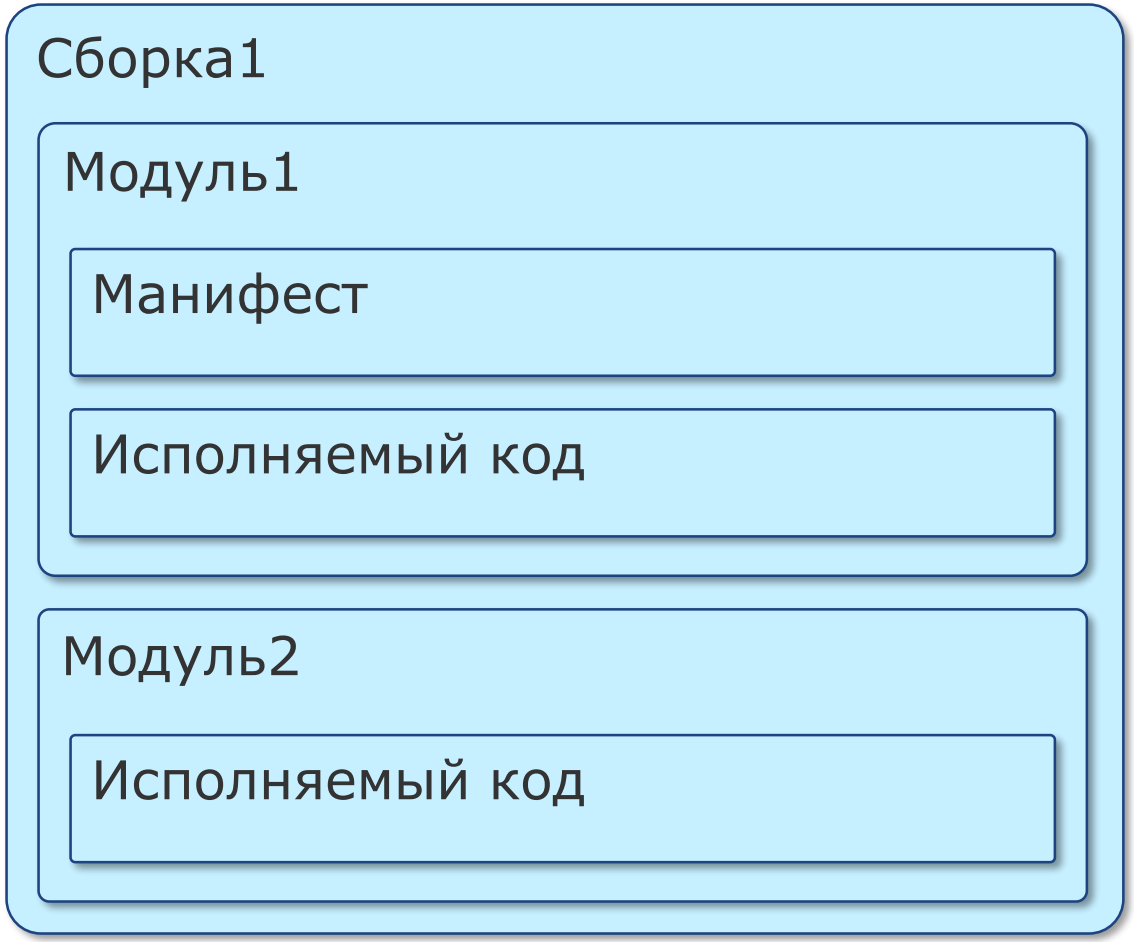
Источник: ECMA-335 Common Language Infrastructure (CLI)

Структура сборки

В сборке один тип



В сборке множество типов



Результаты расследования

Причина: способ размещения типов в сборках clr

Потенциально уязвимые места:

- Заглушки
- ORM
- AOP
- Reflection.Emit

Решение: размещение каждого динамического типа в своей сборке

- При использовании Castle.DynamicProxy — создание выделенного ProxyGenerator для каждого типа

CUSTIS

Расследование долгой работы

- Профилировщик
- Изучение инфраструктуры
- Локализация проблемы
- Гипотезы и их проверка
- Google / StackOverFlow

CUSTIS® ИТ-РЕШЕНИЯ ДЛЯ РАЗВИТИЯ

Спасибо за внимание!

Пуаро — Игорь Шаталкин, ishatalkin@gmail.com

Гастингс — Георгий Минашин, hoborg91@gmail.com

CUSTIS

Случай из жизни

- Общее число модульных тестов: 2200
- Время работы до ускорения: 19 мин.
- Время работы после ускорения: 2 мин.

Ссылки:

- Обсуждение вопросов производительности, связанных с размещением динамических типов в одной или нескольких сборках: https://stackoverflow.com/questions/47295780/multiple-types-in-one-dynamic-assembly-is-way-slower-than-multiple-dynamic-assem/47297416

- PR в Moq на использование множества динамических сборок вместо одной: https://github.com/moq/moq4/pull/938