

# Профайлинг .Net приложений на Linux

# INTRODUCTION



**Nikita Starichenko**

6+ Years of Experience:

**STO solutions, San Francisco USA**

**01/2020 -**

**now**

- Designing and Developing SPA application for healthcare risk adjustment automation
- Designed architecture of the entire system from scratch

**Dodo Pizza, Oxford USA / Moscow Russia**

**03/2018 –**

**04/2019**

- Developing and maintaining DODO IS
- Reduced release time by 40% by fixing more than 50 UI tests that led to a decrease of manual testing
- Piloted first microservice on .Net Core and GRPC that is composing by Docker that uses a full CI/CD including integration tests and run in Kubernetes
- Piloted integration React to Angular.js and add ability to step by step rewrite frontend from Angular to React that speeded up front development by 2 times
- Got rid of the need to restart the system by eliminating the daily memory leak of 100mb by finding that leak in .Net Core application using memory snapshot tools on Linux in runtime

# TRAINING ROADMAP: OVERVIEW

■ Обзор функций и примеров профайлинга	6
■ Обзор утилит профайлинга на Windows	15
■ Сравнение удобства Windows утилит	26
■ Обзор утилит профайлинга на Linux	27
■ Сравнение удобства Linux утилит	34
■ Практическая часть	

**Семинар рассказывает о функциях профайлинга и способах работы с ним на платформе Linux. Целью семинара является показать способы профайлинга на Linux. Семинар подходит как начинающим, так и опытным разработчикам.**

**Понадобятся:**

- **Visual studio**
- **Linux**

# TRAINING ROADMAP: STRUCTURE

- **1 час теории**
- **40 минут практики**
- **20 минут ответов на вопросы**



# Теория





## Что такое профайлинг?

# Профилирование

Форма **анализа программ**, которая измеряет, память, скорость работы, частоту и продолжительность вызовов функций. Чаще всего, профайлинг используется для **оптимизации программы**.



# Что можно профилировать?

1. Оперативную память
2. Использование процессора
3. Различные эвенты
4. Вызовы методов (последовательность, время выполнения, расход ресурсов)





Что можно найти с помощью профайлинга?

# Утечка памяти на эвентах

```
public class MyClass
{
    public MyClass(WiFiManager wiFiManager)
    {
        wiFiManager.WiFiSignalChanged += OnWiFiChanged;
    }

    private void OnWiFiChanged(object sender, WifiEventArgs e)
    {
        // do something
    }
}
```

1. Подписаться на эвент
2. Забыть отписаться
3. Утечка

# Утечка памяти на замыканиях

```
public class MyClass
{
    private JobQueue _jobQueue;
    private int _id;

    public MyClass(JobQueue jobQueue)
    {
        _jobQueue = jobQueue;
    }

    public void Foo()
    {
        _jobQueue.EnqueueJob(() =>
        {
            Logger.Log($"Executing job with ID {_id}");
            // do stuff
        });
    }
}
```

1. Передать в замыкание член класса
2. Утечка

# Утечка памяти на коллекциях

```
public class ProfilePicExtractor
{
    private Dictionary<int, byte[]> PictureCache { get; set; } =
        new Dictionary<int, byte[]>();

    public byte[] GetProfilePicByID(int id)
    {
        // A lock mechanism should be added here, but let's stay on point
        if (!PictureCache.ContainsKey(id))
        {
            var picture = GetPictureFromDatabase(id);
            PictureCache[id] = picture;
        }
        return PictureCache[id];
    }

    private byte[] GetPictureFromDatabase(int id)
    {
        // ...
    }
}
```

1. Положить что-то в коллекцию и забыть об этом
2. Утечка

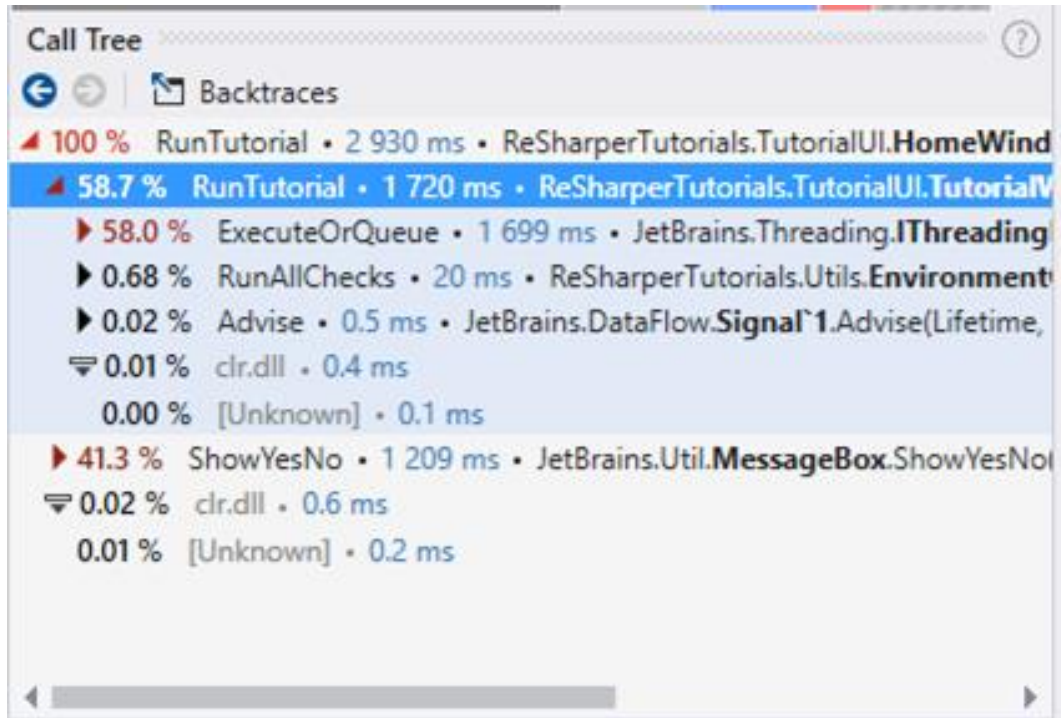
# Утечка памяти на дефрагментации LON

```
int i = 85000;  
while (i < 1000000)  
{  
    var items = new byte[i];  
    var test = System.Text.Encoding.UTF8.GetString(items);  
    Console.WriteLine(test.Length);  
    i++;  
}
```

1. Создать объект занимающий больше чем 85000 байт
2. Дефрагментация
3. Утечка



# Поиск узких мест



1. Запустить профилировщик
2. Посмотреть какие части кода занимают всех больше времени

# Лишнее использование ЦПУ

```
0 references
static void Main(string[] args)
{
    for (int i = 0; i < 1000; i++)
    {
        var service = new SomeService();
        service.IsMatch(test: "test");
    }
}

1 reference
class SomeService
{
    private readonly Regex _regex = new Regex(pattern: ".*");

    1 reference
    public bool IsMatch(string test)
    {
        return _regex.IsMatch(test);
    }
}
```

1. Инициализировать сервис при каждом использовании
2. Инициализация всех членов сервиса
3. Сборка мусора после каждого использования



# Обзор утилит профайлинга на Windows

## Утилиты профайлинга на Windows

1. JetBrains dotTrace
2. Redgate ANTS Performance Profiler
3. Visual Studio's performance profiler
4. PerfView



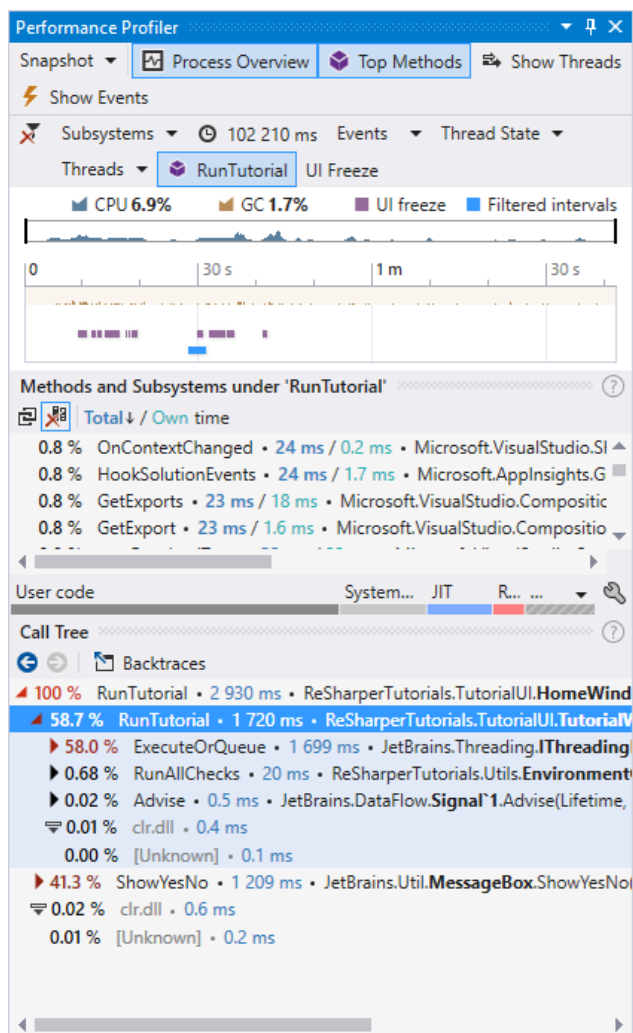
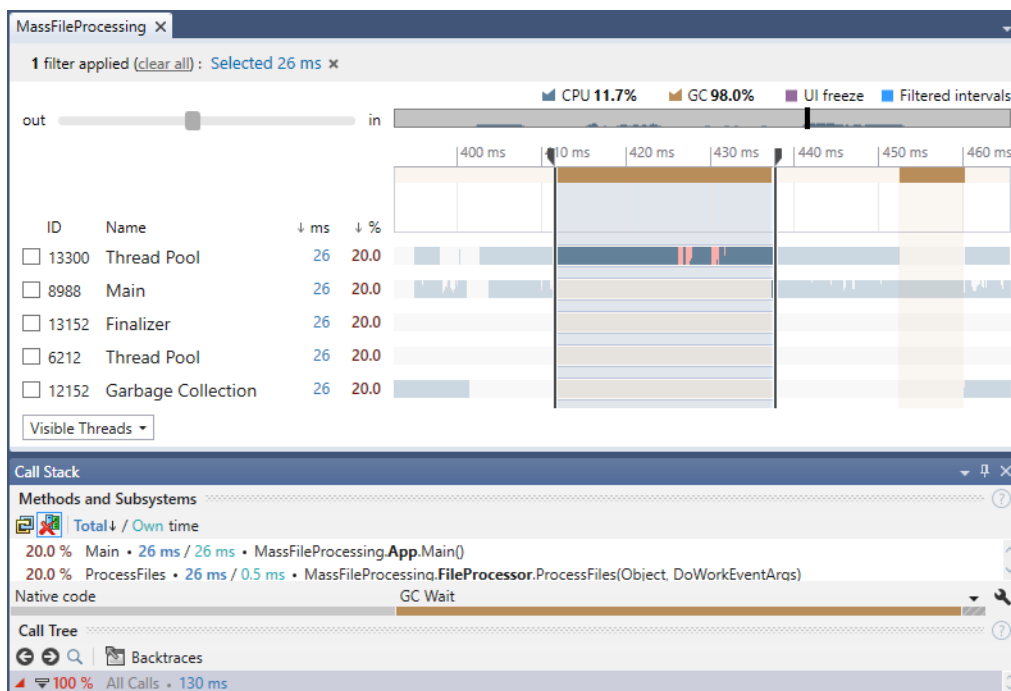


# JetBrains dotTrace

1. Задержки в отрисовке интерфейса
2. Избыточная сборка мусора
3. Неравномерное распределение рабочей нагрузки
4. Неэффективный файловый ввод-вывод
5. Профилирование SQL и HTTP запросов
6. Хронологический режим профилирования и асинхронный
7. Профилирование удаленных приложений
8. Сравнение снимков
9. Профилирование и анализ результатов прямо в Visual Studio



# JetBrains dotTrace



TutorialWindowManager.cs SolutionCopyHelper.cs GlobalSettings.cs

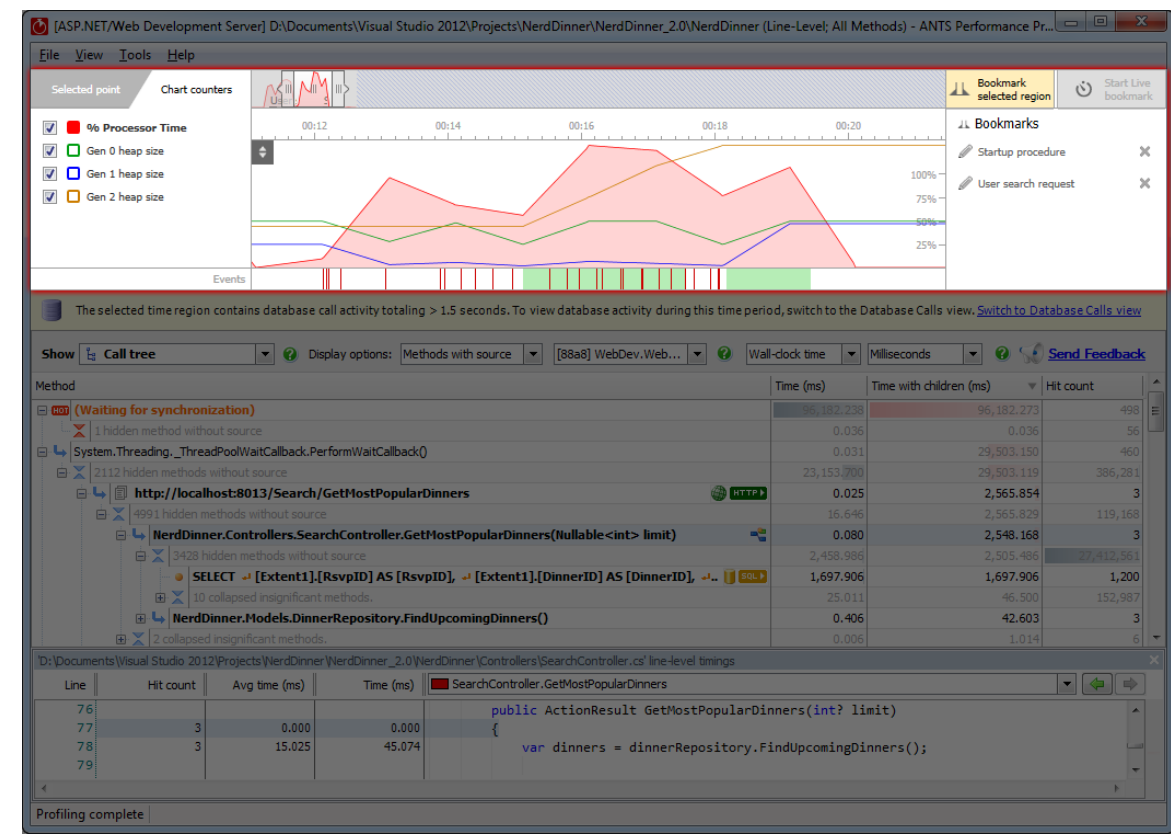
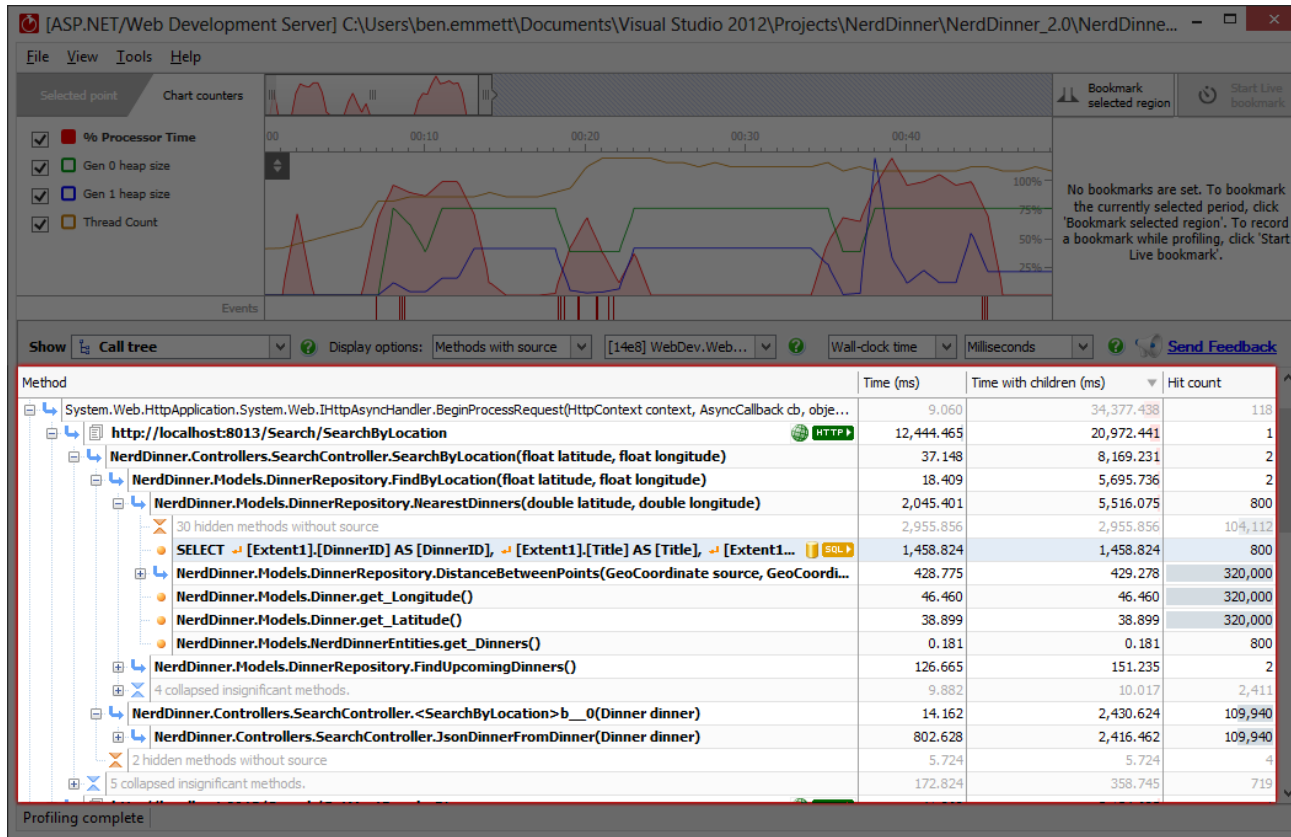
ReSharperTutorials ReSharperTutorials.TutorialL RunTutorial(string htmlTuto

```
115 {
116     if (_homeWindow != null)
117     {
118         _homeWindow.Show();
119         return;
120     }
121
122     _threading.ExecuteOrQueue("RunMainWindow", () =>
123     {
124         _homeWindow = new HomeWindow(_shellLifetime, th
125         _actionManager, _toolWindowClass, _windowsh
126         {
127             PageText = HtmlGenerator.GetResource("HomeF
128         });
129         _homeWindow.Show();
130     });
131 }
132
133
134 public void RunTutorial(string htmlTutorialId)
135 {
136     try
137     {
138         EnvironmentChecker.RunAllChecks();
139     }
140     catch (Exception e)
141     {
142         if (e is NoShortCutsAssignedException)
143         {
144             MessageBox.ShowError(
145                 "ReSharper shortcuts are not assigned!
146                 \"ReSharper | Options... | Environment |
147                 \"ReSharper Tutorials\");
148             return;
149         }
150     }
151
152     var result =
153     MessageBox.ShowYesNo(
```

# Redgate ANTS Performance Profiler

1. Профилирование SQL и HTTP запросов
2. Отслеживание узких мест в коде
3. Хронологический режим профилирования
4. Графический режим
5. Группировка методов по HTTP запросам
6. Группировка методов по асинк методам

# Redgate ANTS Performance Profiler



# Visual Studio's performance profiler

1. Анализ данных об использовании памяти
2. Анализ использования ЦП
3. PerfTips - подсказки рядом с кодом во время отладки
4. Изучение событий приложения
5. Анализ асинхронного кода
6. Анализ производительности базы данных
7. Проверка производительности и доступности пользовательского интерфейса
8. Анализ использования GPU

# Visual Studio's performance profiler

Summary Events Memory Usage CPU Usage	
CPU Profiling	
Function Name	Total CPU [ms, %]
Profiling App.exe (PID: 14232)	2369 (100.00 %)
Profiling_App.MainWindow::GetMaxNumberButton_Click	2308 (97.43 %)
[External Call] System.Windows.Application.Run()\$##6000...	2308 (97.43 %)
Profiling_App.App::Main	2308 (97.43 %)
[External Call] System.Random.Next()\$##6000FF7	1255 (52.98 %)
Profiling_App.MainWindow::GetNumber	1010 (42.63 %)
[External Call] System.Random.InternalSample()\$##6000FF6	38 (1.60 %)
[External Call] System.Windows.Controls.TextBox.set_Text(...	3 (0.13 %)

Snapshot #2 Heap P...g App.exe (15.80s) X

MainWindow.xaml

MainWindow.xaml.cs

Managed Memory (Profiling App.exe)

Compare to: Snapshot #1

Object Type	Count Diff.	Size Diff. (Bytes)	Inclusive Size Diff. (Bytes)	Count	Size (Bytes)	Inclusive Size (Bytes)
ClassHandlersStore	0	+3,492	+3,428	32	9,296	23,580
EventRoute	+3	+1,368	+1,524	4	1,808	2,016
StylusPointPropertyInfo	+30	+1,320	+1,320	30	1,320	1,320
StylusTouchDevice	+5	+520	+520	5	520	520
DispatcherOperation	+5	+420	+1,780	8	672	2,528
List<Int32>	+1	+292	+292	1	292	292
TextTreeTextBlock	+1	+244	+244	1	244	244
ExecutionContext	+5	+220	+320	12	528	744
Task<Object>	+5	+220	+280	8	352	448
Hashtable	+2	+216	+1,184	79	64,264	193,276
PriorityItem<DispatcherOpe...	+5	+160	+1,152	8	256	1,272

Event Filter										Provider Filter
The count of displayed events is limited to 20,000. This session has 276,474 total events, which have been clipped in the view. Please add more filters to reduce the event count. Don't show again										X
Provider Name	Event Name	Text	Timestamp (ms)	Provider Guid	Event ID	Process ID	Process Name	Thread ID	Additional Properties	
Windows Kernel	EventTrace	Windows Kernel[EventTrace [pid:21596] tid:4824]	0	9e814aad-3204-11d2-9a82-006008...	65535	21596	PerfView	4824	Payload Properties	
MSNT_SystemTrace	EventTrace/PartitionInfoExtensionV2	MSNT_SystemTrace[EventTrace/PartitionInfoExtension...	0	9e814aad-3204-11d2-9a82-006008...	65535	21596	PerfView	4824	BufferSize 65,536	
Windows Kernel	EventTrace/Extension	Windows Kernel[EventTrace/Extension [pid:21596] tid:4...	0	9e814aad-3204-11d2-9a82-006008...	65535	21596	PerfView	4824	Version 131,082	
MSNT_SystemTrace	EventTrace/PartitionInfoExtensionV2	MSNT_SystemTrace[EventTrace/PartitionInfoExtension...	0	9e814aad-3204-11d2-9a82-006008...	65535	21596	PerfView	4824	ProviderVersion 18,363	
Windows Kernel	SysConfig/SystemPaths	Windows Kernel[SysConfig/SystemPaths [pid:0] tid:0]	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	NumberOfProcessors 8	
Windows Kernel	SysConfig/UnknownVolume	Windows Kernel[SysConfig/UnknownVolume [pid:0] tid:...	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	EndTime 12:54:09.372977 (17,304.542 MSec)	
Windows Kernel	SysConfig/UnknownVolume	Windows Kernel[SysConfig/UnknownVolume [pid:0] tid:...	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	TimerResolution 156,250	
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel[SysConfig/VolumeMapping [pid:0] tid:0]	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	MaxFileSize 500	
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel[SysConfig/VolumeMapping [pid:0] tid:0]	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	LogFileMode 67,174,401	
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel[SysConfig/VolumeMapping [pid:0] tid:0]	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	BuffersWritten 817	
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel[SysConfig/VolumeMapping [pid:0] tid:0]	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	StartBuffers 1	
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel[SysConfig/VolumeMapping [pid:0] tid:0]	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	PointerSize 8	
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel[SysConfig/VolumeMapping [pid:0] tid:0]	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	EventsLost 0	
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel[SysConfig/VolumeMapping [pid:0] tid:0]	0	9e814aad-3204-11d2-9a82-006008...	65535	0	Idle	0	CPUspeed 2,112	
Windows Kernel	EventTrace/EndExtension	Windows Kernel[EventTrace/EndExtension [pid:-1] tid:-1]	21	9e814aad-3204-11d2-9a82-006008...	65535	-1		-1	BootTime 12/9/2019 8:25:43 AM	
Windows Kernel	EventTrace/Extension	Windows Kernel[EventTrace/Extension [pid:-1] tid:-1]	22	9e814aad-3204-11d2-9a82-006008...	65535	-1		-1	PerfFreq 10,000,000	

22



# PerfView

1. Хронологический режим профилирования
2. Анализ использования GC
3. Анализ использования ЦП
4. Анализ использования памяти
5. Анализ скорости работы

# PerfView

PerfView Stacks

File Help Tutorial Understanding Perf Data Starting an Analysis Troubleshooting FAQ Tips

Back Forward Totals Metric: 5,326.0 Count: 5,326.0 First: 666.160ms Last: 6,975.3055ms Duration: 6,309.145ms Metric/msec: 0.84 TimeBucket: 197.2ms

Start: 0 End: 9214.377 Find: tutorial!Program\SpinForASecond()\

GroupPats: [just my code] \PerfView\=> Fold%: 1.0 FoldPats: EXTERNAL <<ntdll?>> IncPats: ^Process tutorial (45) Expats: ^Process Idle\EEShut

By Name Caller-Callee CallTree

Name	Inc. %	Inc	Exc. %	Exc	When	First	Last
✓ ROOT	100.0	5,326	0.0	0	0888999979999AA99A61661	666.160	6,975.30
+✓ Process tutorial (+520)	100.0	5,326	0.0	0	0888999979999AA99A61661	666.160	6,975.30
-✓ Thread (6000) [1-15 frames]	100.0	5,326	0.2	9	0888999979999AA99A61661	666.160	6,975.30
+✓ EXTERNAL <<mscorlib?>> [8-41 frames]	97.7	5,201	8.0	427	0787999979999AA99A61661	859.290	6,970.30
+ tutorial!Program.Main(class System.String[])	88.2	4,698	0.0	0	787999979999AA99A61661	894.291	5,899.40
+ tutorial!Program.RecSpin(int32)	88.2	4,698	0.5	29	787999979999AA99A61661	894.291	5,899.40
+ tutorial!Program.RecSpinHelper(int32)	73.0	3,887	1.1	57	499979999AA99A61661	1,896.330	5,899.40
+ tutorial!Program.RecSpin(int32)	54.9	2,926	0.7	35	79999AA99A61661	2,897.368	5,899.40
+ tutorial!Program.RecSpinHelper(int32)	37.4	1,993	0.5	28	4A9A999AA99A61661	3,898.404	5,899.40
+ tutorial!Program.RecSpin(int32)	18.7	997	0.6	30	9999	4,899.442	5,899.40
+ tutorial!Program.SpinForASecond()	18.2	967	2.2	116	9999	4,899.442	5,899.40
+ EXTERNAL <<mscorlib?>> [1-17 frames]	16.0	851	16.0	851	8888	4,899.442	5,899.40
+ tutorial!Program.SpinForASecond()	18.2	968	2.1	112	49990	3,899.404	4,898.40
+ EXTERNAL <<mscorlib?>> [1-17 frames]	16.1	856	16.1	856	48880	3,899.404	4,898.40
+ tutorial!Program.SpinForASecond()	16.9	898	1.9	102	7999	2,897.368	3,897.40
+ EXTERNAL <<mscorlib?>> [1-18 frames]	14.9	796	14.9	796	8888	2,898.369	3,897.40
+ tutorial!Program.SpinForASecond()	17.0	904	2.1	112	99990	1,896.330	2,896.30
+ EXTERNAL <<mscorlib?>> [1-19 frames]	14.9	792	14.9	792	98770	1,896.330	2,896.30
+ tutorial!Program.SpinForASecond()	14.7	782	2.0	105	7874	895.293	1,895.30
+ EXTERNAL <<mscorlib?>> [1-25 frames]	12.7	677	12.7	677	6760	895.293	1,895.30
+ EXTERNAL <<mscorlib?>> [1-3 frames]	1.4	76	1.4	76	0	866.290	6,964.30
+ BROKEN [1-15 frames]	2.2	116	2.2	116	00011_0_00_00_0_000000	667.160	6,975.30

Ready Log Cancel

[illegible]

# Сравнение

## JetBrains dotTrace

1. Неэффективный файловый ввод-вывод
2. Профилирование удаленных приложений
3. Профилирование и анализ результатов прямо в Visual Studio

## Redgate ANTS Performance Profiler

1. Группировка методов по HTTP запросам
2. Группировка методов по асинк методам

## Visual Studio's performance profiler

1. PerfTips - подсказки рядом с кодом во время отладки
2. Изучение событий приложения
3. Проверка производительности и доступности пользовательского интерфейса
4. Анализ использования GPU



## Обзор утилит профайлинга на Linux



## Утилиты профайлинга на Linux

1. gcore (core dump)
2. lldb debugger
3. libsosplugin
4. perf
5. dotnet trace
6. dotTrace + Rider





# gcore

## 1. Снятие дампа процесса и запись его в файл

```
1  sudo gcore 4058
2  # ...
3  # Saved corefile core.4058
```

# lldb debugger

1. Анализ снятого дампа
2. Анализ используемой памяти
3. Можно использовать как полноценный дебаггер, но не для .Net приложений

```
1 $ lldb-3.6 `which dotnet` -c core.4058
2 # (lldb) target create "/usr/bin/dotnet" --core "core.4058"
3 # Core file '/home/ubuntu/core.4058' (x86_64) was loaded.
4 # (lldb) plugin load /usr/share/dotnet/shared/Microsoft.NETCore.App/2.0.0/libse
5 # (lldb)
```

```
1 (lldb) memory read 00007f6d0e8810f0+0xc -f s -c 13
2 #0x7f6d0e8810fc: "R"
3 #0x7f6d0e8810fe: "a"
4 #0x7f6d0e881100: "n"
5 #0x7f6d0e881102: "d"
6 #0x7f6d0e881104: "o"
7 #0x7f6d0e881106: "m"
8 #0x7f6d0e881108: " "
9 #0x7f6d0e88110a: "s"
10 #0x7f6d0e88110c: "t"
11 #0x7f6d0e88110e: "r"
12 #0x7f6d0e881110: "i"
13 #0x7f6d0e881112: "n"
14 #0x7f6d0e881114: "g"
```

# libsosplugin

1. Добавляет к lldb поддержку .Net типов и помогает анализировать дампы снятый с .Net приложения

```
1 (lldb) sos DumpHeap -stat
2 #Statistics:
3 # MT Count TotalSize Class Name
4 #00007f6d32992aa8 1 24 UNKNOWN
5 #00007f6d329911d8 1 24 UNKNOWN
6 #....
7 #00007f6d323defd8 4 17528 System.Object[]
8 #00007f6d323e08a8 25 40644 System.Int32[]
9 #00007f6d323e0168 29 82664 System.String[]
10 #00007f6d323e3440 335 952398 System.Char[]
11 #000000000223b860 10092 6083604 Free
12 #00007f6d3242b460 150846 204845172 System.String
13 #Total 161886 objects
14 (lldb)
```

```
1 (lldb) sos DumpHeap -type System.String
2 # Address MT Size
3 #00007f6d0bfff3f0 00007f6d3242b460 26
4 #00007f6d0bfff4c0 00007f6d3242b460 42
5 #...
6 #00007f6d0c099ab0 00007f6d3242b460 20056
7 #00007f6d0c09e920 00007f6d3242b460 20056
8 #...
9 #00007f6d323e0168 29 82664 System.String[]
10 #00007f6d3242b460 150846 204845172 System.String
11 #Total 150895 objects
```

```
1 (lldb) sos DumpObj 00007f6d0e8810f0
2 #Name: System.String
3 #MethodTable: 00007f6d3242b460
4 #EEClass: 00007f6d31c49eb8
5 #Size: 20056(0x4e58) bytes
6 #File: /usr/share/dotnet/shared/Microsoft.NETCore.App/2.0.0/System.Private.CoreLib.dll
7 #String:
8 #Fields:
9 # MT Field Offset Type VT Attr
10 #00007f6d3244b020 40001c9 8 System.Int32 1 instance
11 #00007f6d3242f420 40001ca c System.Char 1 instance
12 #00007f6d3242b460 40001cb 38 System.String 0 shared
13 # >> Domain:Value 00000000022ab050:NotInit
```

# perf

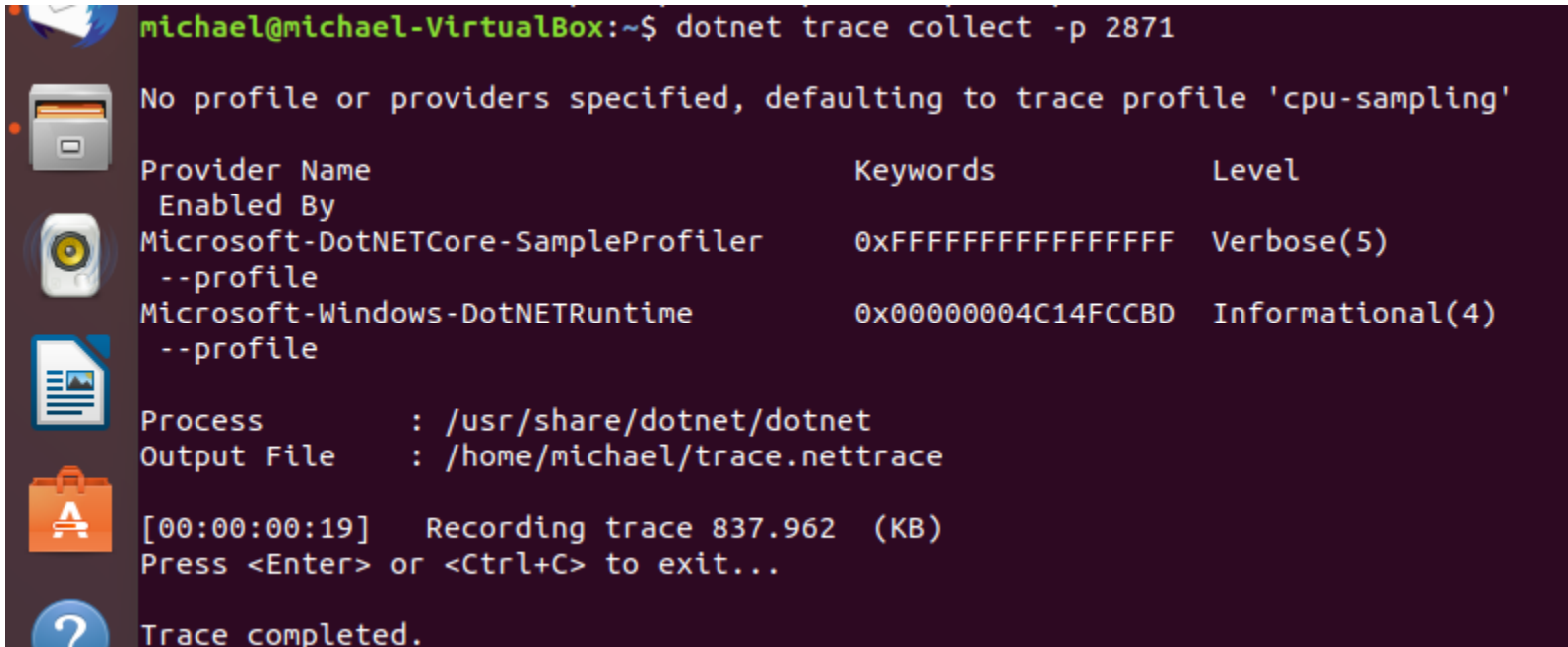
1. Анализ производительности ЦП

2. Анализ стека вызовов методов

```
Samples: 14K of event 'cpu-clock', Event count (approx.): 3703000000
Children      Self    Command  Shared Object      Symbol
+ 99.97%      0.00% dotnet    libc-2.23.so       [.] __libc_start_main
+ 99.97%      0.00% dotnet    [unknown]          [.] 0x41d589495541f689
+ 99.97%      0.00% dotnet    perf-13723.map     [.] void [calc] calc.Program::Main(string[])
+ 99.94%      0.00% dotnet    perf-13723.map     [.] void [calc] calc.Program::DoSomeMath()
+ 77.59%      37.72% dotnet    perf-13723.map     [.] !!0 [System.Linq] System.Linq.Enumerable::Aggregate(class [System.Runtime]
+ 51.69%      0.00% dotnet    perf-13723.map     [.] int32 [calc] calc.Program::Sum()
+ 48.26%      0.00% dotnet    perf-13723.map     [.] int32 [calc] calc.Program::Mul()
- 18.16%      18.15% dotnet    perf-13723.map     [.] instance int32 [calc] calc.Program+<>c::<Sum>b__3_0(int32,int32)
- 18.15% 0x41d589495541f689
  __libc_start_main
  main
  run
  hostfxr_main
  fx_muxer_t::execute
  fx_muxer_t::parse_args_and_execute
  fx_muxer_t::read_config_and_execute
  execute_app
  corehost_main
  run
  coreclr::execute_assembly
  coreclr_execute_assembly
  0xaa53b
  0x279063
  0x278dc3
  0x167e70
  0x257097
  void [calc] calc.Program::Main(string[])
  void [calc] calc.Program::DoSomeMath()
- int32 [calc] calc.Program::Sum()
- 16.43% !!0 [System.Linq] System.Linq.Enumerable::Aggregate(class [System.Runtime]System.Collections.Generic.IEnumerabl
  instance int32 [calc] calc.Program+<>c::<Sum>b__3_0(int32,int32)
  1.71% instance int32 [calc] calc.Program+<>c::<Sum>b__3_0(int32,int32)
+ 0.01% instance int32 [calc] calc.Program+<>c::<Sum>b__3_0(int32,int32)
+ 16.88%      16.79% dotnet    perf-13723.map     [.] instance int32 [calc] calc.Program+<>c::<Mul>b__4_0(int32,int32)
+ 3.53%      0.00% dotnet    System.Linq.dll     [.] 0xffff8046e9489891
```

# dotnet trace

1. Снятие дампа с запущенного .Net приложения
2. Дамп можно затем изучить с помощью PerfView

A terminal window with a dark purple background and a sidebar on the left containing icons for a file, a folder, a CD, a document, a briefcase, and a question mark. The terminal text is as follows:

```
michael@michael-VirtualBox:~$ dotnet trace collect -p 2871  
No profile or providers specified, defaulting to trace profile 'cpu-sampling'  

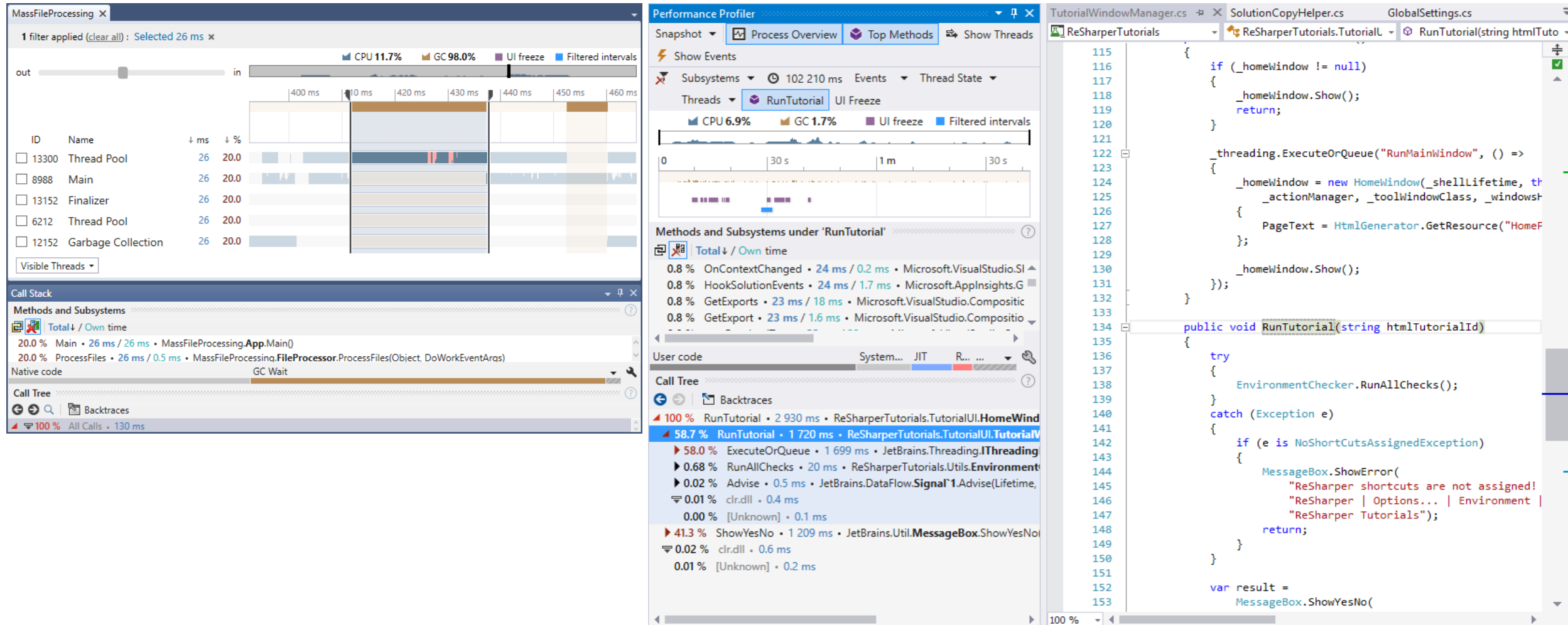

| Provider Name                       | Keywords           | Level            |
|-------------------------------------|--------------------|------------------|
| Enabled By                          |                    |                  |
| Microsoft-DotNETCore-SampleProfiler | 0xFFFFFFFFFFFFFFFF | Verbose(5)       |
| --profile                           |                    |                  |
| Microsoft-Windows-DotNETRuntime     | 0x00000004C14FCCBD | Informational(4) |
| --profile                           |                    |                  |

  
Process      : /usr/share/dotnet/dotnet  
Output File  : /home/michael/trace.nettrace  
  
[00:00:00:19] Recording trace 837.962 (KB)  
Press <Enter> or <Ctrl+C> to exit...  
  
Trace completed.
```



# dotTrace + Rider

## 1. Всё тоже самое, что и на Windows



# Сравнение

## **gcore lldb sosplugin**

1. Анализ снятого дампа в консоли

## **dotnet trace + PerfView**

1. Снятие дампа с запущенного приложения из консоли
2. Все возможности PerfView на Windows

## **dotTrace + Rider**

1. Профайлинг здорового человека с графическим интерфейсом
2. Необходим Rider
3. Необходим графический интерфейс



# Практика



# Вопросы?

# Thank You!

think.  
create.  
accelerate.

**Пройди опрос**  
- поделись своим мнением!



[https://ru.surveymonkey.com/r/seminar\\_03\\_september](https://ru.surveymonkey.com/r/seminar_03_september)

**Luxoft** | training  
A DXC Technology Company