VS Performance Profiling and Diagnostic Tools

Tools in Visual Studio to help you find and fix performance issues in your code.

Why Profiling?

- Find memory leaks, and general performance issues
- Detect and Fix hard to find bugs and performance issues
- Fine-Tune and Improve app performance
- Find issues related to external libraries and dependencies

Common Peformance Issues

- Mishandling CPU time
  - Misusing async patterns and threading - I/O bound vs CPU bound
  - Caching issues specially when dealing with apis
- Memory Issues
  - Memory leaks - not disposing of objects
  - Using too much memory specially when dealing with Lists
  - Issues around GC - may happen too often
- External dependencies
  - Network or File system issues
  - Database issues
  - 3rd party libraries

Performance Collection Methods

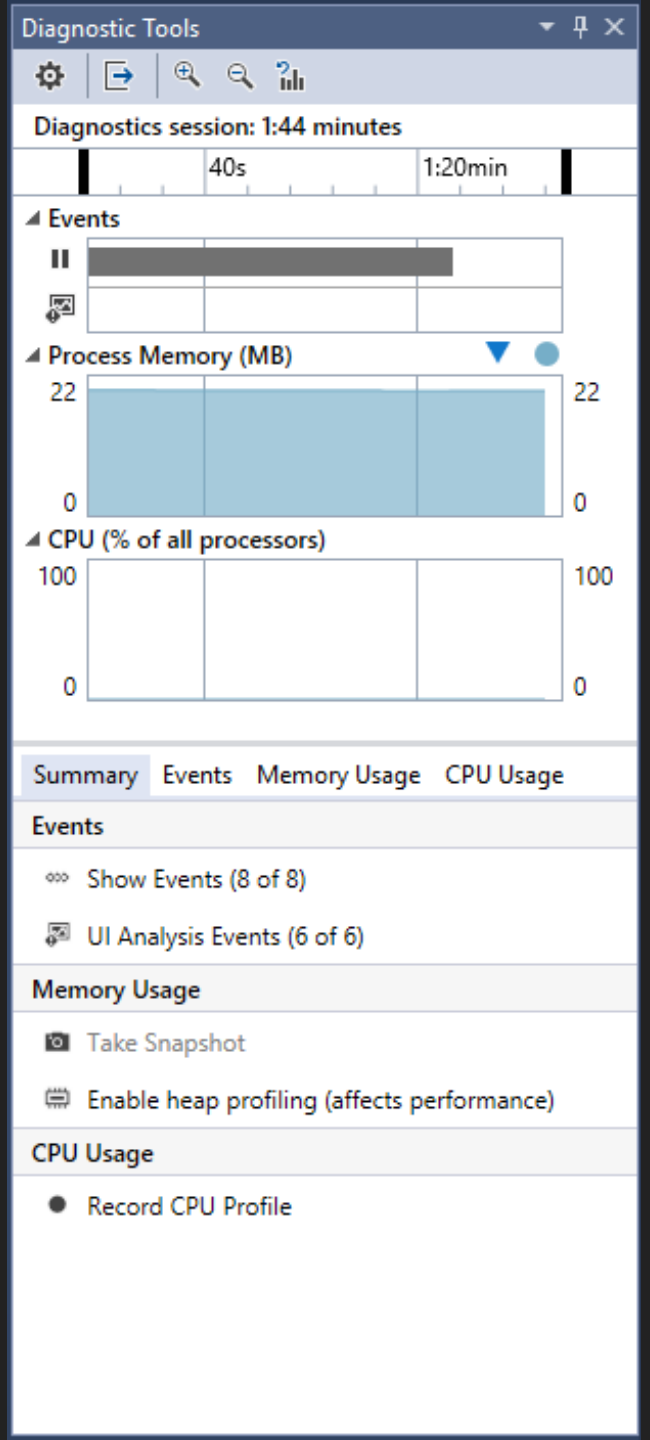
- **Sampling**:
  Collects statistical data about the work that is performed by an application during profiling. Least impact on performance
- **Tracing**:
  provides better information on how often a method was executed. If you need accurate measures of call numbers, use tracing. Huge Impact on performance
- **Instrumentation**
  Collects detailed information about the work that is performed by an application during a profiling run. Data collection is done by tools that either inject code into a binary file that captures timing information or by using callback hooks to collect and emit exact timing and call count information while an application runs. high overhead when compared to sampling-based approaches

Tools in Visual Studio

- Visual Studio Diagnostics Tools and PerfTips (While Debugging)
- Visual Studio Performance Profiler (Standalone Suite Alt+F2 in VS or CMD)
  - CPU Usage
  - Memory Profiler
  - Async Profiler
  - Databases
  - .NET performance counters
  - Event Viewer

Diagnostic Tools

Breakpoints and associated timing data gets recorded in the Diagnostic Tools window.



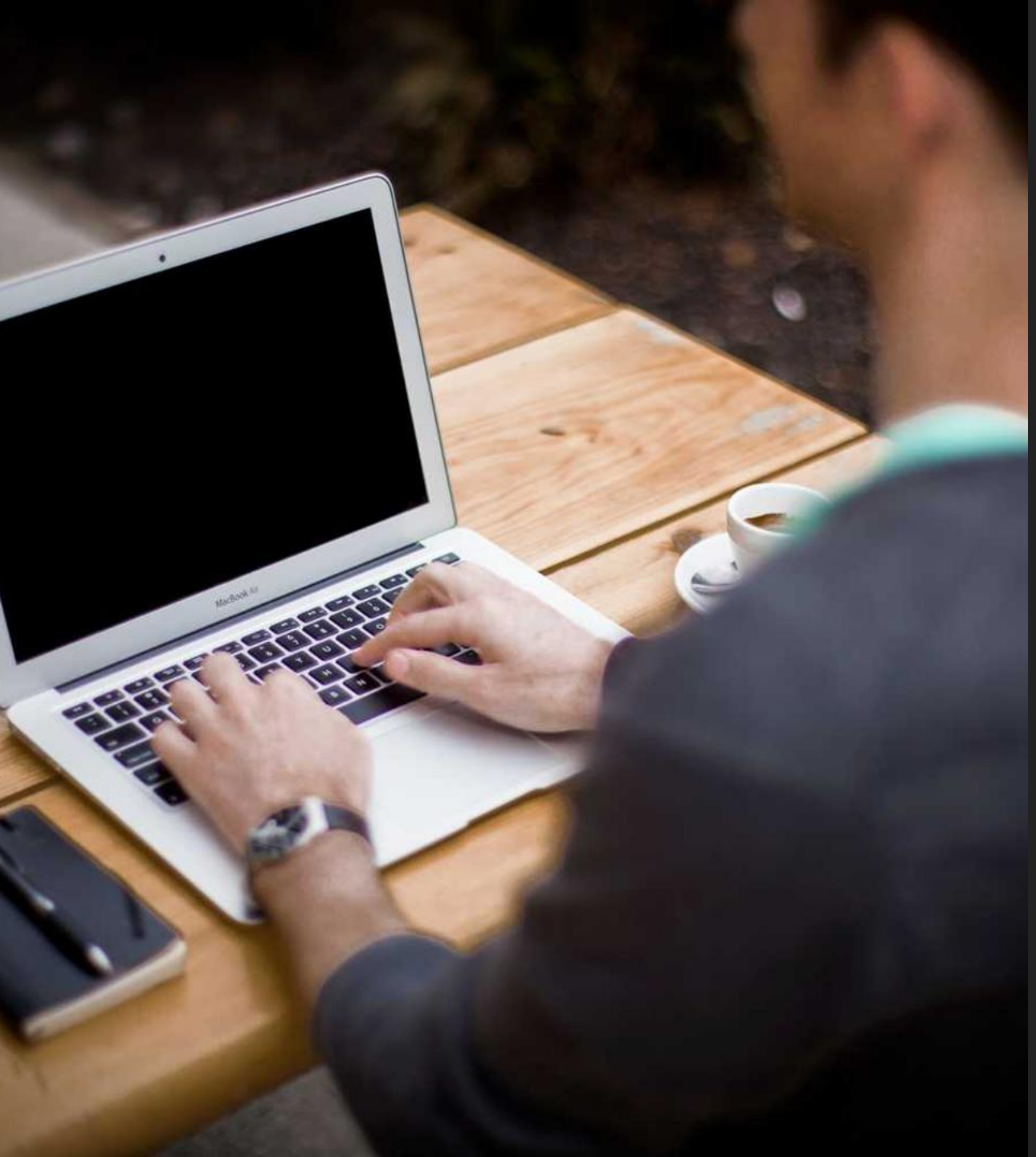
PerfTips

When the debugger stops execution at a breakpoint or stepping operation, the elapsed time between the break and the previous breakpoint appears as a tip

MainWindow.xaml.cs

WpfMemory | WpfMemory.MainWindow | Button1_Click(object sender, Routed...

```
19  /// <summary>
20  /// Interaction logic for MainWindow.xaml
21  /// </summary>
    2 references
22  public partial class MainWindow : Window
23  {
24      List<int> list = new List<int>();
     0 references
25      public MainWindow()
26      {
27          InitializeComponent();
28      } 2
    1 reference
29  private void Button1_Click(object sender, RoutedEventArgs e)
30  {
31      Debugger.Break();
32      TextBox1.Text += "Using managed memory\n";
33      UseManagedMemory();
34      TextBox1.Text += "Done using managed memory\n"; ≤864ms elapsed
35      Debugger.Break();
36  }
37
    1 reference
38  private void UseManagedMemory()
39  {
```

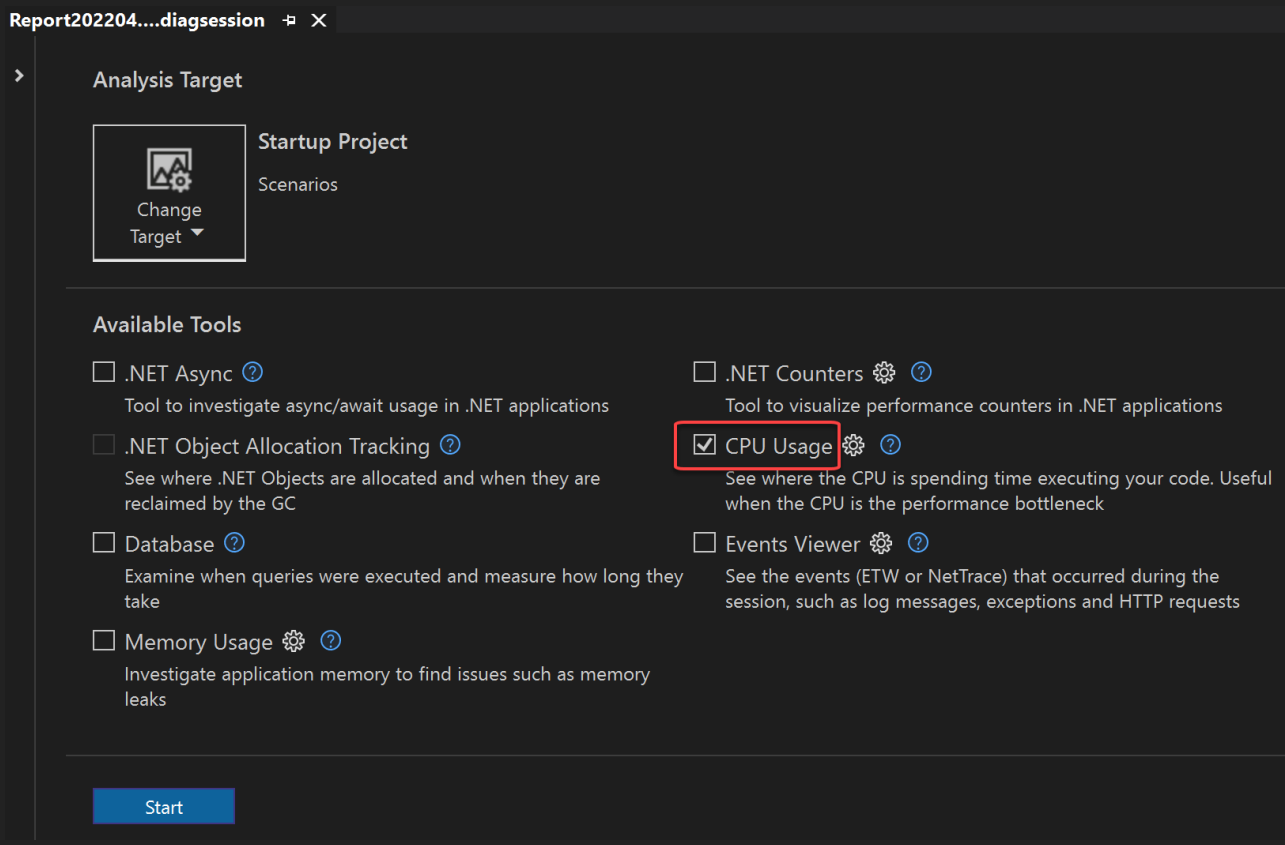


Demo

Demonstrate Diagnostic Tools vs Performance Profiler

CPU Usage

One of the main tools in Performance profiler suite.



The CPU Usage tool

- Diagnose a slow-down or a process hang in your team’s codebase.
- Identify in timeline where CPU should be working and isn't or vice versa
- Identify performance issues in DevOps scenarios, such as when a customer reports that some requests or orders are not getting through to the retail website during peak season.
- If your latency issue isn’t within an API request, then you can check for high CPU utilization and other related issues with the CPU Usage tool. The CPU Usage tool can help you identify bottlenecks so that you can narrow down where to optimize.

Output | Zoom In | Reset Zoom | Clear Selection

Diagnostics session: 1:57 minutes

12.5s 25s 37.5s 50s 1:02.5min 1:15min 1:27.5min 1:40min 1:52.5min

CPU (% of all processors)

100 0 100 0

Open details... Categories Filter

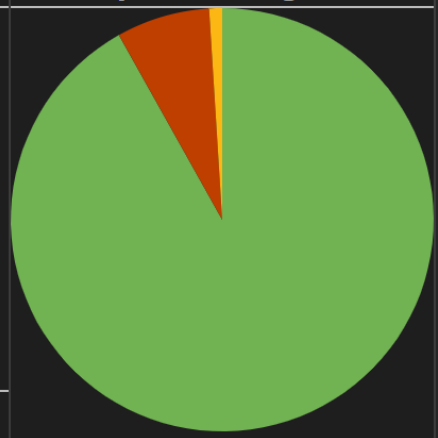
Top Functions

Function Name	Total CPU [unit, %]	Self CPU [unit, %]
[External Call] ntdll.dll!0x00007ff85d342651	1570 (97.27%)	680 (42.13%)
[External Call] Microsoft.EntityFrameworkCore.Infrastructure.DatabaseFacade.EnsureCreated()	507 (31.41%)	371 (22.99%)
[External Call] Newtonsoft.Json.JsonSerializer.Deserialize<T>(Newtonsoft.Json.JsonReader)	111 (6.88%)	111 (6.88%)
[External Call] microsoft.extensions.hosting.abstractions.dll!0x00007ff854ed6573	737 (45.66%)	104 (6.44%)
[External Call] Microsoft.EntityFrameworkCore.DbContext.ctor(Microsoft.EntityFrameworkCore.DbContextOptions)	71 (4.40%)	71 (4.40%)

Hot Path

Function Name	Total CPU [unit, %]	Self CPU [unit, %]
Scenarios (PID: 19176)	1614 (100.00%)	3 (0.19%)
[System Code] ntdll.dll!0x00007ff85d342651	1570 (97.27%)	680 (42.13%)
Scenarios.Program.Main(System.String[])	890 (55.14%)	0 (0.00%)

Top Five Categories



Kernel : 91.9% (1483)

Name	Description
Total CPU	$\text{Total CPU\%} = \frac{\textit{total method activity}}{\textit{app activity}} \times 100$ The milliseconds and CPU percentage used by calls to the function, and functions called by the function, in the selected time range.
Self CPU	$\text{Self CPU\%} = \frac{\textit{self method activity}}{\textit{app activity}} \times 100$ The milliseconds and CPU percentage used by calls to the function in the selected time range, excluding functions called by the function.
Module	The name of the module containing the function.

CPU Usage -...8.diagsession | Report202204...8.diagsession* | Startup.cs

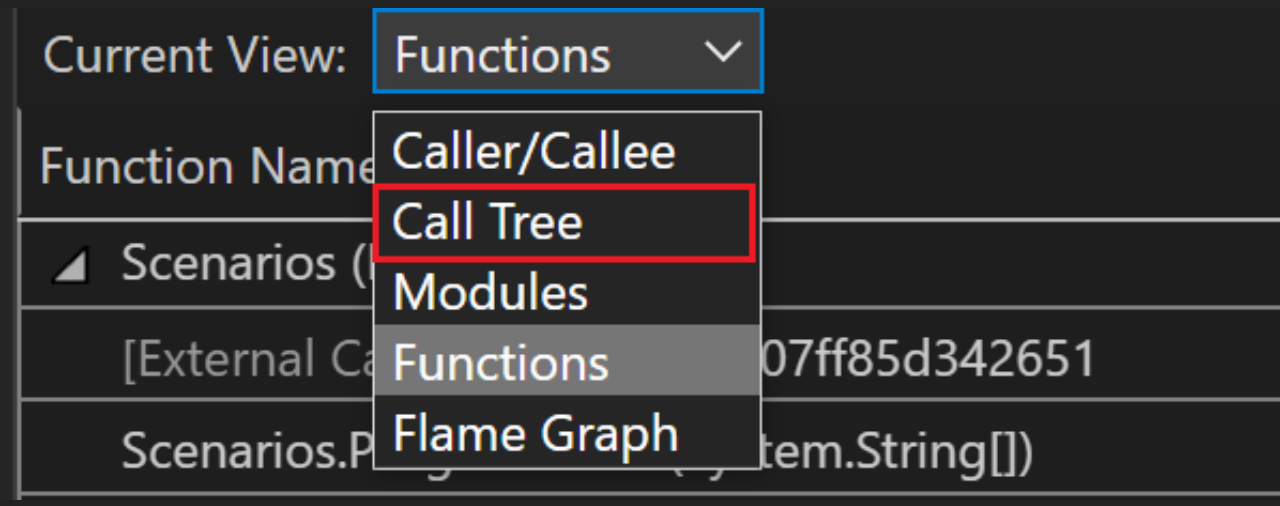
Current View: Call Tree | Expand Hot Path | Show Hot Path | Reset Root

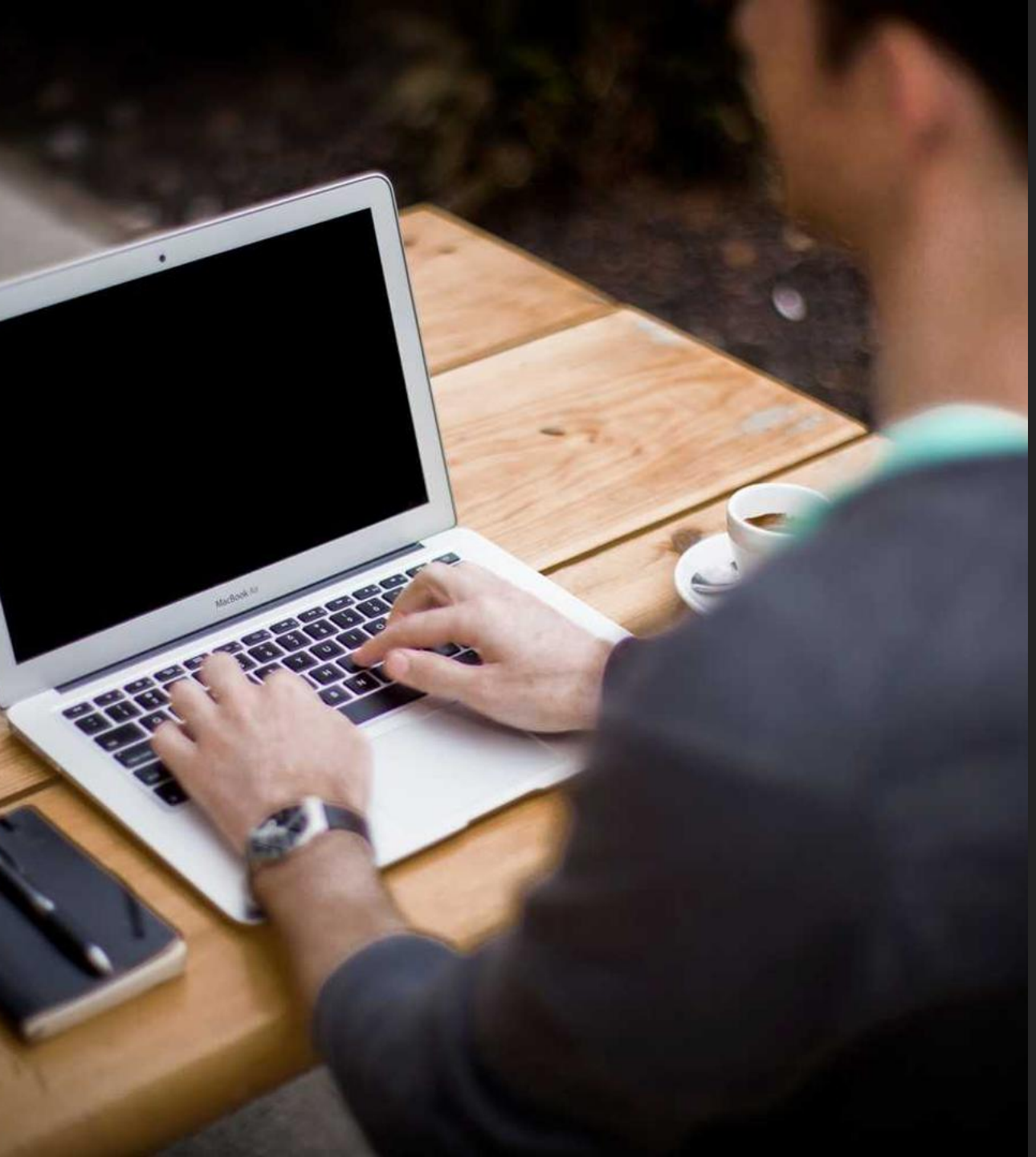
Function Name	Total CPU [unit, %]	Self CPU [unit, %]	Module
Scenarios (PID: 34232) 1	2200 (100.00%)	5 (0.23%)	Multiple modules
[External Call] ntdll.dll!0x00007ff85d34265 2	2155 (97.95%)	1005 (45.68%)	ntdll
Scenarios.Program.Main(System.String[3	1150 (52.27%)	0 (0.00%)	scenarios
[External Call] microsoft.extensions.hosti 4	929 (42.23%)	154 (7.00%)	microsoft.extensio...
Scenarios.Startup.Configure(Microsoft....	664 (30.18%)	0 (0.00%)	scenarios
[External Call] Microsoft.EntityFrame...	580 (26.36%)	417 (18.95%)	microsoft.entityfra...
[External Call] microsoft.aspnetcore.r...	82 (3.73%)	0 (0.00%)	microsoft.aspnetc...
[External Call] coreclr.dll!0x00007fff8...	1 (0.05%)	1 (0.05%)	coreclr
[External Call] microsoft.aspnetcore.r...	1 (0.05%)	1 (0.05%)	microsoft.aspnetc...
Scenarios.PokemonDbContext.ctor(Mic...	107 (4.86%)	0 (0.00%)	scenarios
Scenarios.Startup.ConfigureServices.An...	4 (0.18%)	0 (0.00%)	scenarios

Image	Description
1	The top-level node in CPU Usage call trees is a pseudo-node.
2	In most apps, when the **Show External Code** option is disabled, the second-level node is an **[External Code]** node. The node contains the system and framework code that starts and stops the app, draws the UI, controls thread scheduling, and provides other low-level services to the app.
3	The children of the second-level node are the user-code methods and asynchronous routines that are called or created by the second-level system and framework code.
4	Child nodes of a method have data only for the calls of the parent method.

Detailed CPU Usage View

- Caller/Callee
- Call Tree
- Modules
- Functions
- Flame Graph





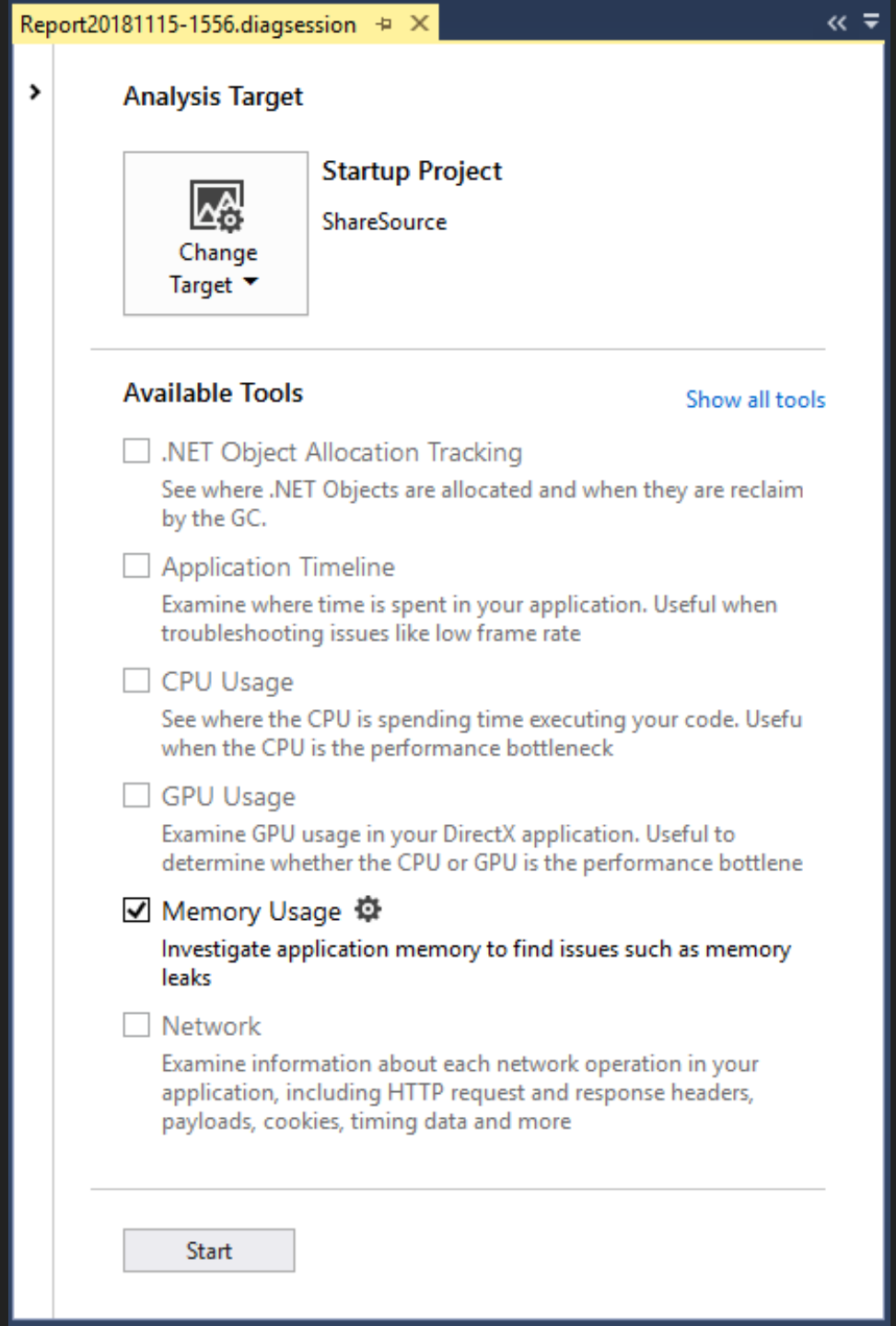
Demo

Demonstrate Performance Profiler

CPU Usage

Memory Usage

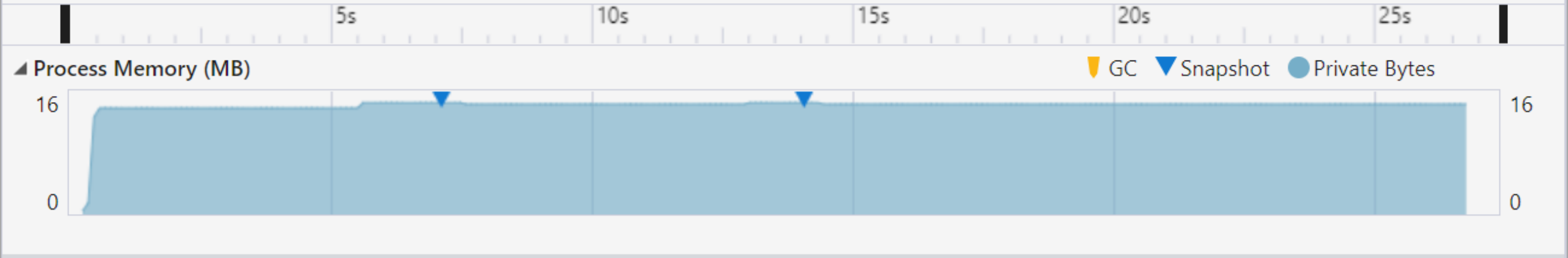
You can use the tool to study the real-time memory effects of scenarios you're actively developing in Visual Studio



Report202201....diagsession* | Object Browser | Form1.cs [Design]

Output | Zoom In | Reset Zoom | Clear Selection

Diagnostics session: 27.407 seconds



View Heap

ID	Time	Objects (Diff)	Heap Size (Diff)
1	7.12s	2,213 (n/a)	181.73 KB (n/a)
2	14.07s	2,201 (-12)	181.36 KB (-0.37 KB)

View Heap

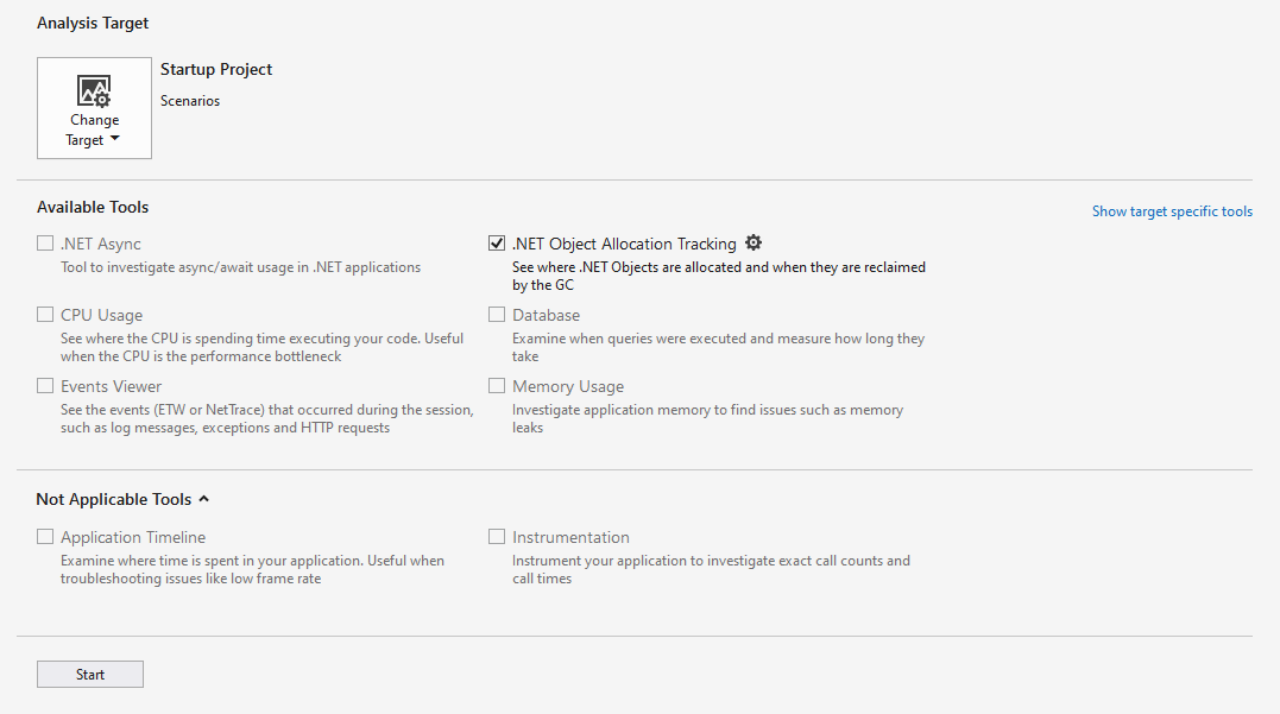
ID	Time	Objects (Diff)	Heap Size (Diff)
1	7.12s	2,213 (n/a)	181.73 KB (n/a)
2	14.07s	2,201 (-12)	181.36 KB (-0.37 KB)

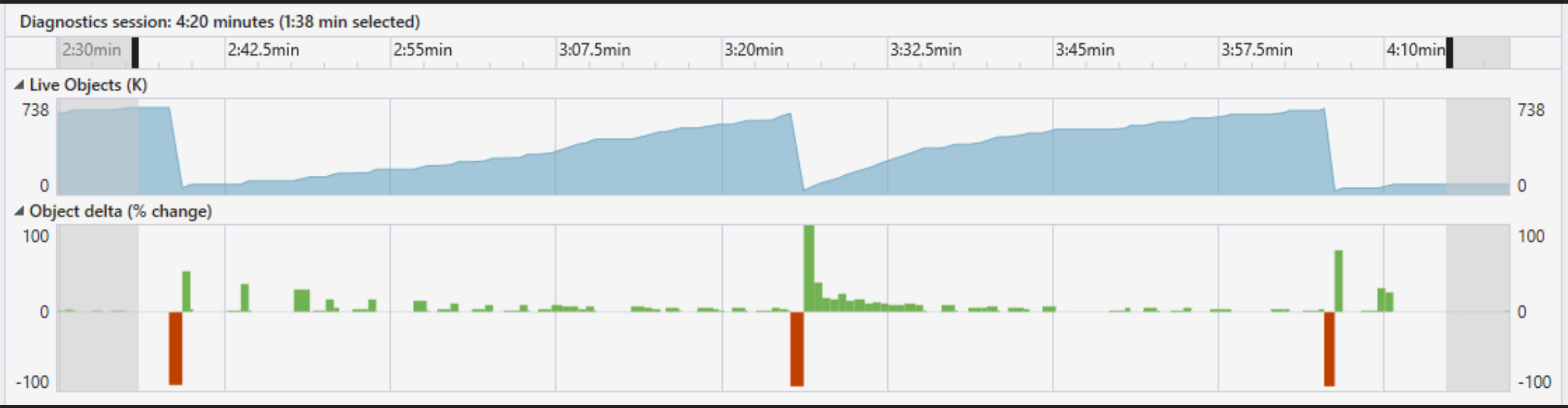
1 2 3 4

Image	Description
1	Total number of objects in memory when the snapshot was taken. Select this link to display a snapshot details report sorted by the count of instances
2	Difference between the total number of memory objects in this snapshot and the previous snapshot. Select this link to display a snapshot diff report sorted by the difference in the total count of instances
3	Total number of bytes in memory when the snapshot was taken. Select this link to display a snapshot details report sorted by the total size
4	Difference between the total size of memory objects in this snapshot and the previous snapshot. positive or negative number means the memory size of this snapshot is larger or smaller than the previous one.

Object Allocation Tool

You can see how much memory your app uses and what code paths allocate the most memory by using the .NET Object Allocation tool.




Diagnostics session: 4:20 minutes (1:38 min selected)
2:30min
2:42.5min
2:55min
3:07.5min
3:20min
3:32.5min
3:45min
3:57.5min
4:10min
Live Objects (K)
738
0
Object delta (% change)
100
0
-100

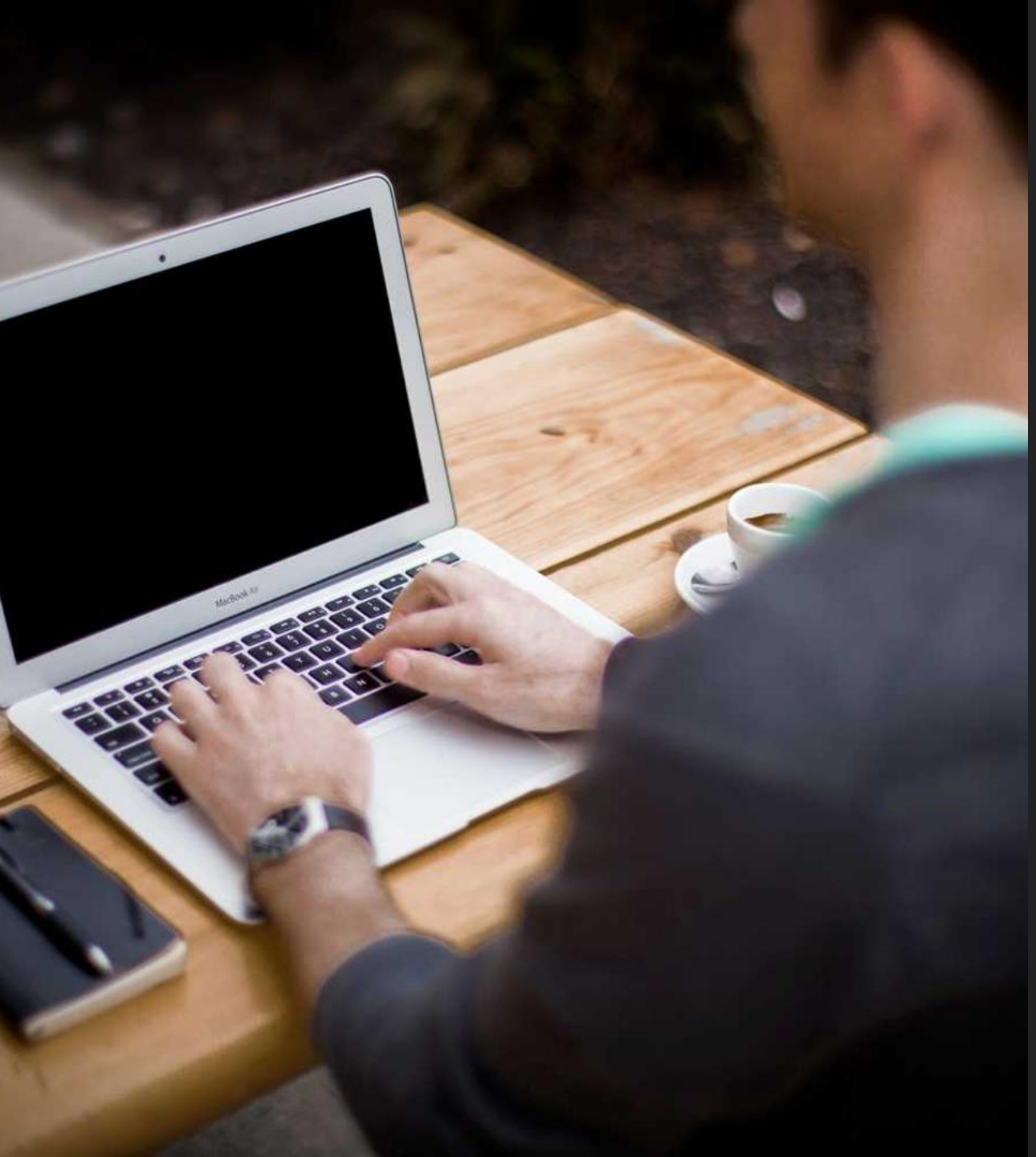
Allocation | Call Tree | Functions | Collection

Show Just My Code | Show Native Code | Filter types

Type	Allocations	Bytes	Average Size (Bytes)
System.String	12,540	960,864	76.62
System.SByte[]	8,459	519,656	61.43
System.Reflection.RuntimeMet...	5,260	547,040	104
System.Object[]	4,381	563,480	128.62
System.Reflection.CustomAttrib...	4,212	223,872	53.15
System.Int32[]	3,715	203,568	54.8
System.RuntimeMethodInfoStub	3,424	301,312	88
System.SZArrayEnumerator	2,658	85,056	32
System.RuntimeType[]	2,566	91,176	35.53
System.Reflection.ParameterInf...	2,269	80,736	35.58
System.Single	2,134	51,216	24
System.RuntimeType	1,907	76,280	40
System.String[]	1,838	101,424	55.18
System.Type[]	1,732	69,672	40.23
System.Int32	1,687	40,488	24
System.Byte[]	1,483	664,253	447.91
System.Signature	1,368	109,440	80
System.Object	1,356	32,544	24
System.Reflection.RuntimePara...	1,271	122,016	96
System.Reflection.RuntimeMet...	1,223	209,912	171.64
System.Collections.Generic.Stac...	1,125	36,000	32
System.Collections.Generic.Sort...	1,125	95,672	85.04

Backtrace: System.String

Function Name	Allocations	Bytes	Average Size (Bytes)	Module Name
[External Code]	12,540	960,864	76.62	Multiple modules
Scenarios.PokemonDbContext.OnModelCreat...	2,802	113,004	40.33	Scenarios.dll
[External Call] Microsoft.EntityFrameworkC...	2,802	113,004	40.33	[External Code]
Scenarios.Startup.Configure(Microsoft.As...	2,802	113,004	40.33	Scenarios.dll
[External Call] coreclr.dll	2,802	113,004	40.33	coreclr.dll
Scenarios.Program.Main(System.Stri...	2,802	113,004	40.33	Scenarios.dll
[External Code]	2,802	113,004	40.33	Multiple modules
Scenarios.Services.PokemonService.GetPoke...	2,789	111,260	39.89	Scenarios.dll
[External Call] System.Runtime.CompilerSer...	2,617	100,444	38.38	Multiple modules
[External Call] System.Runtime.CompilerSer...	172	10,816	62.88	[External Code]
Scenarios.Services.PokemonService.GetP...	172	10,816	62.88	Scenarios.dll
Scenarios.Controllers.BigJsonOutputC...	172	10,816	62.88	Scenarios.dll
[External Call] System.Runtime.Com...	172	10,816	62.88	[External Code]
Scenarios.Controllers.BigJsonOutp...	172	10,816	62.88	Scenarios.dll
[External Call] dynamicClass.lam...	172	10,816	62.88	Multiple modules
Scenarios.Startup.Configure.AnonymousMeth...	1,448	60,832	42.01	Scenarios.dll
[External Call] Microsoft.AspNetCore.Builde...	1,448	60,832	42.01	[External Code]
Scenarios.Startup.Configure(Microsoft.As...	1,448	60,832	42.01	Scenarios.dll
[External Call] coreclr.dll	1,448	60,832	42.01	coreclr.dll
Scenarios.Program.Main(System.Stri...	1,448	60,832	42.01	Scenarios.dll
[External Code]	1,448	60,832	42.01	Multiple modules
Scenarios.Program.Main(System.String[])	1,321	88,400	66.92	Scenarios.dll

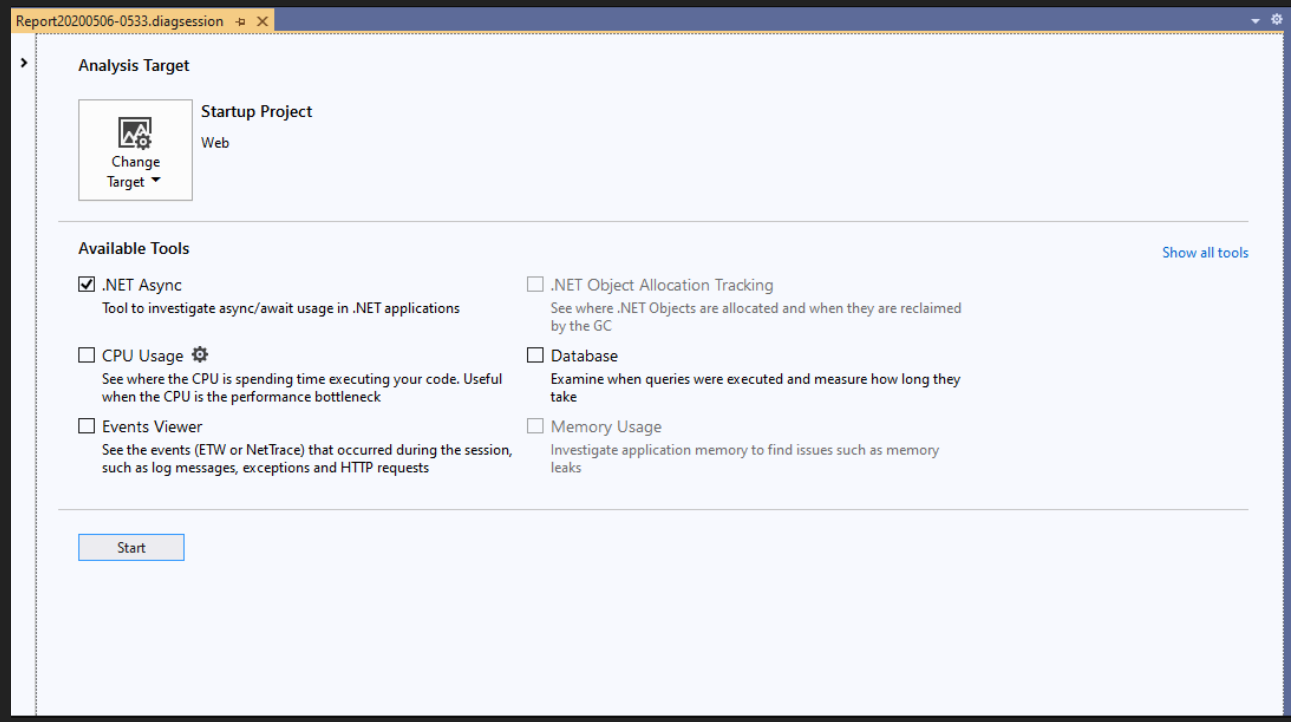


Demo

Demonstrate Memory Usage and Object Allocation Tracking

.NET Async Tool

Use the .NET Async tool to analyze the performance of asynchronous code in your app.

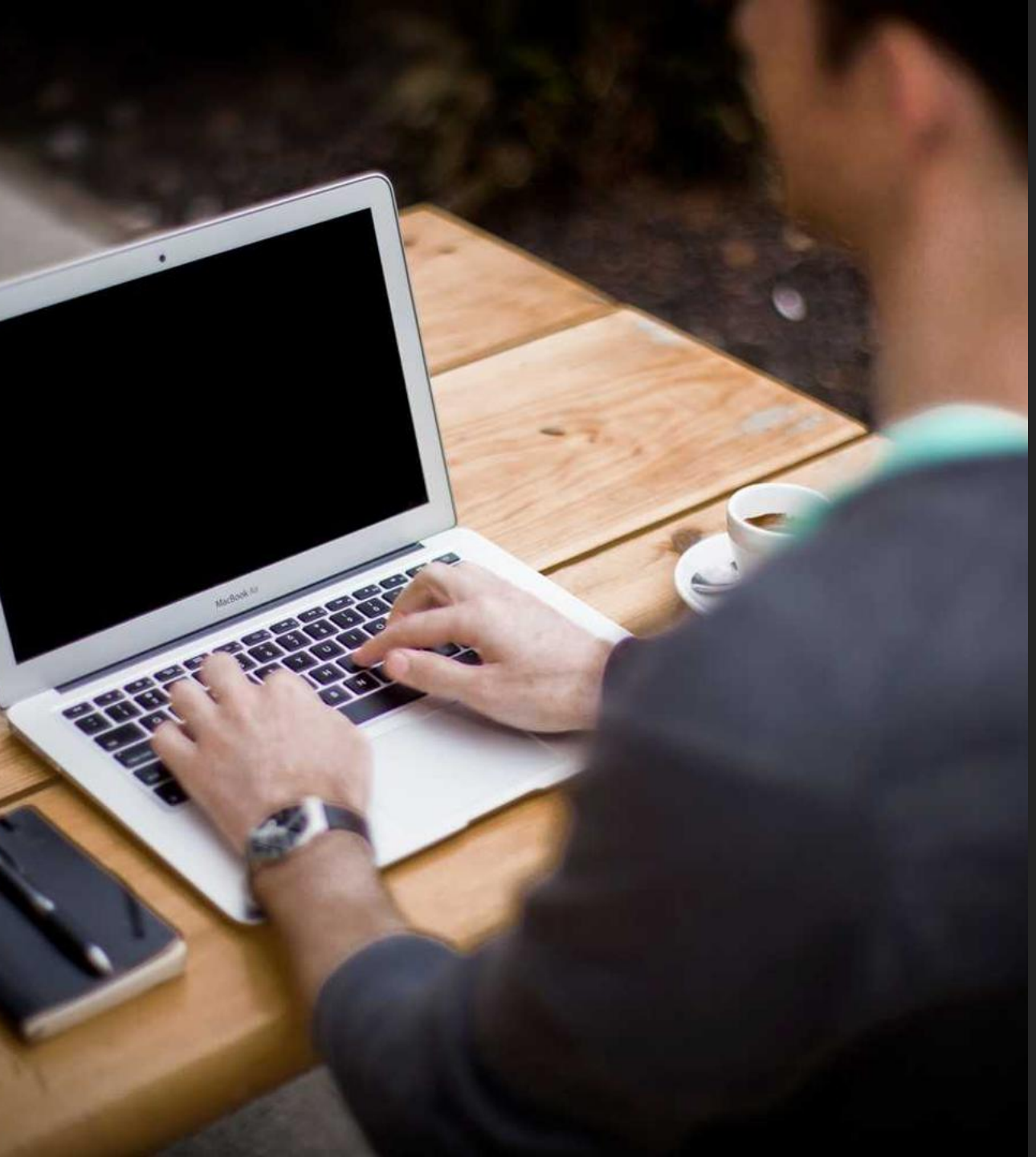


Report20200506-1310.diagsession | Report20200506-0542.diagsession* | Report20200506-0541.diagsession* | Program.cs

Name	Start Time (ms)	End Time (ms)	Total Time (ms)
AsyncActivities.Program.Run1Async()	323.14	430.44	107.3
AsyncActivities.Program.Run2Async()	324.58	430.58	106
[Task] AsyncActivities.Program.Run2Async.AnonymousMethod__2_0 ()	324.09	430.02	105.93
AsyncActivities.Program.Run2Async.AnonymousMethod__2_0()	325.22	429.41	104.19
AsyncActivities.Program.DoWorkAsync()	324.71	328.32	3.6
AsyncActivities.Program.FinishWorkAsync()		429.4	0.13
AsyncActivities.Program.Run3Async()		330.57	4.75
AsyncActivities.Program.DoWorkAsync()	325.27	328.3	3.04
AsyncActivities.Program.FinishWorkAsync()	329.88	330.41	0.53
AsyncActivities.Program.RunLongAsync()	431.78	10432.15	10000.37
[Task] AsyncActivities.Program.RunLongAsync.AnonymousMethod__4_0 ()	431.22	10431.65	10000.43

Go To Source File

Copy

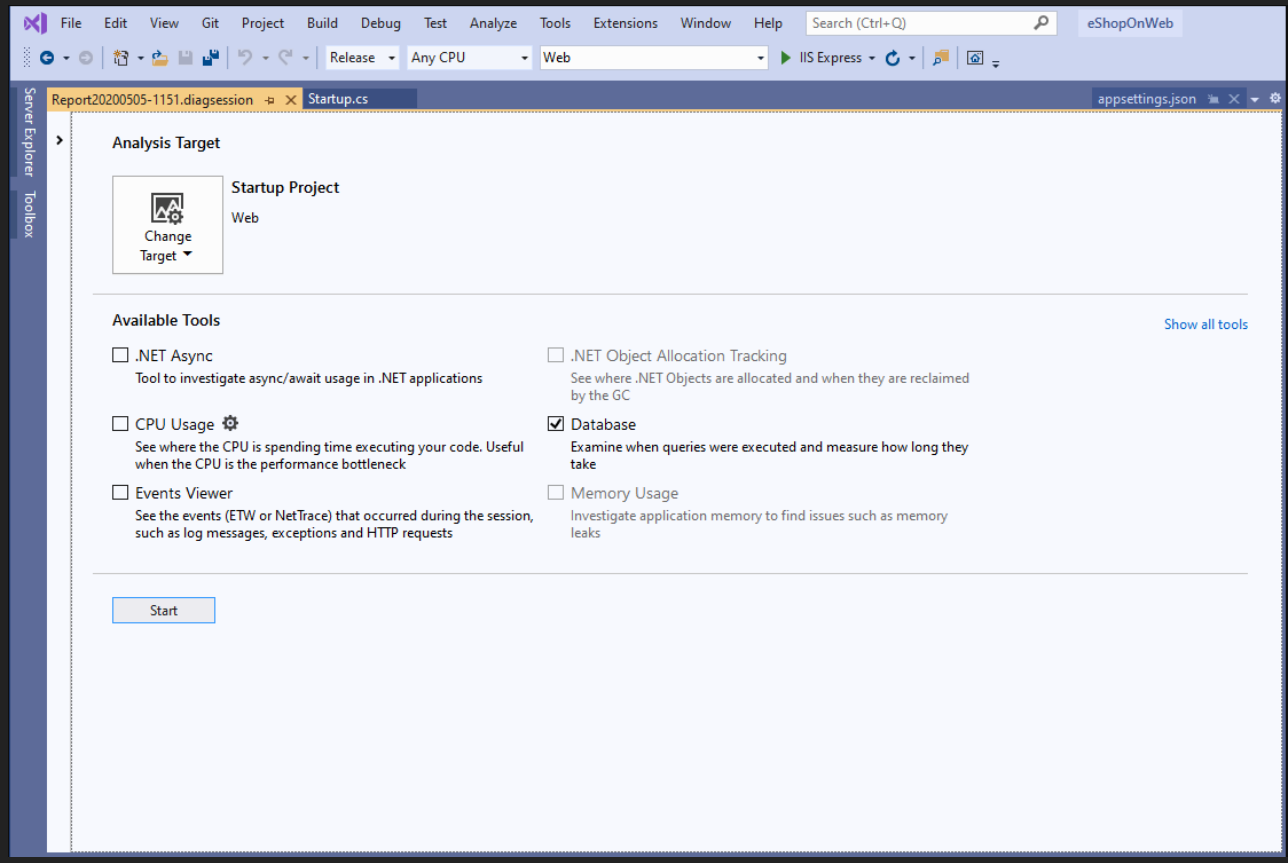


Demo

Demonstrate .NET Async Tool

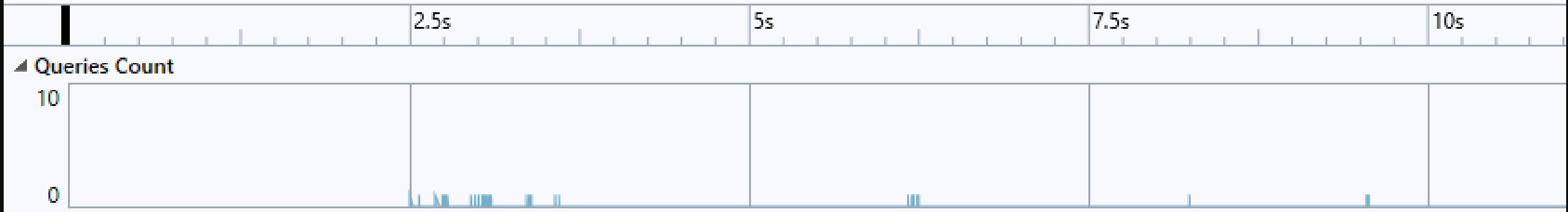
Database Tool

Use the Database tool to record the database queries that your app makes during a diagnostic session. You can then analyze information about individual queries to find places to improve your app's performance.



Output | Zoom In | Reset Zoom | Clear Selection

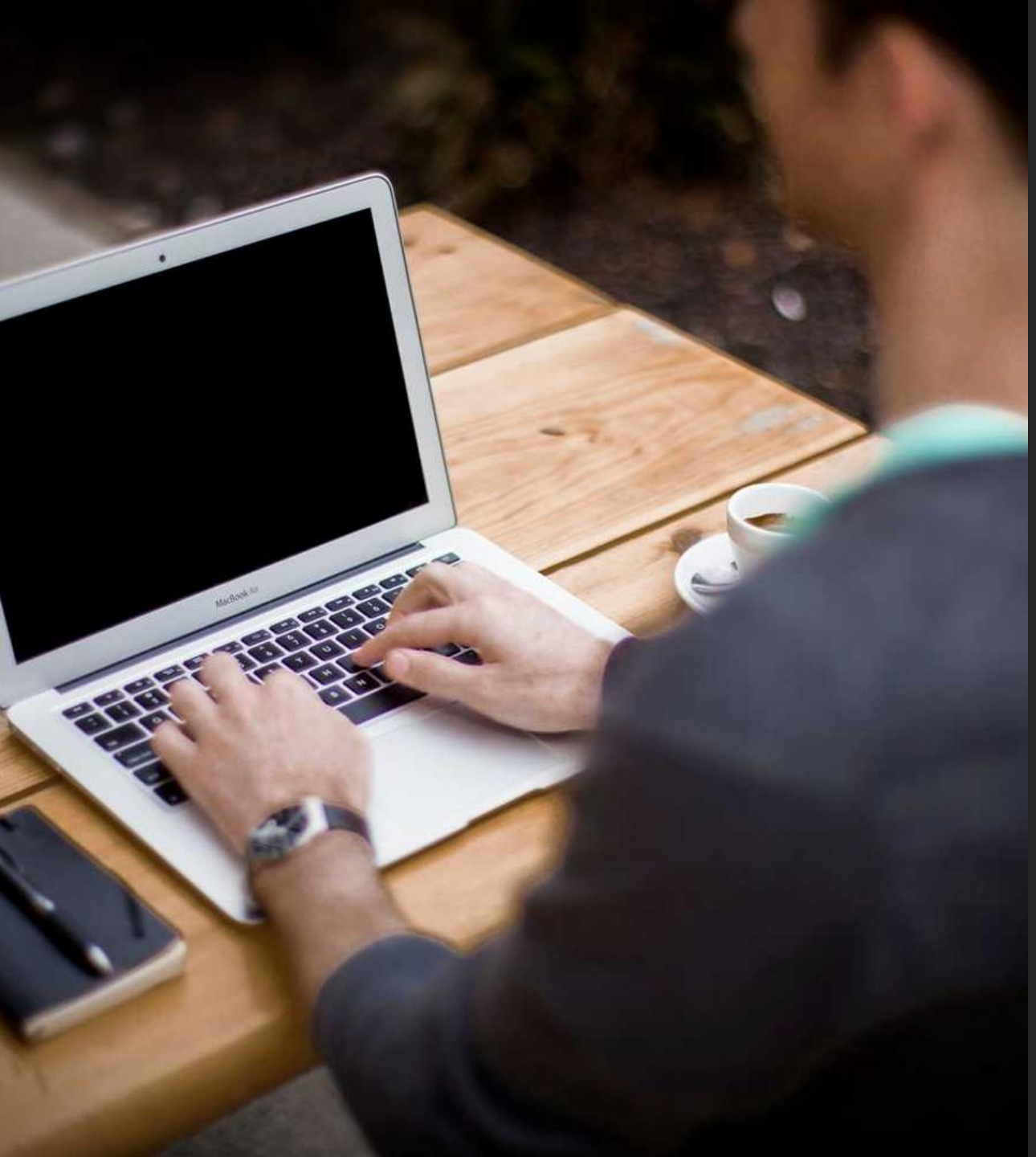
Diagnostics session: 18.633 seconds



Start Time (s)	Query	Duration (ms)
2.501	SELECT CASE WHEN EXISTS (SELECT 1 FROM [CatalogBrands] AS [c]) THEN CAST(1 AS bit) ELSE CAST(0 AS bit)...	27.691
2.572	SELECT NEXT VALUE FOR [catalog_brand_hilo]	3.5
2.686	SET NOCOUNT ON; INSERT INTO [CatalogBrands] ([Id], [Brand]) VALUES (@p0, @p1), (@p2, @p3), (@p4, @p5),...	37.395
2.744	SELECT CASE WHEN EXISTS (SELECT 1 FROM [CatalogTypes] AS [c]) THEN CAST(1 AS bit) ELSE CAST(0 AS bit) E...	0.953
2.746	SELECT NEXT VALUE FOR [catalog_type_hilo]	0.715
2.754	SET NOCOUNT ON; INSERT INTO [CatalogTypes] ([Id], [Type]) VALUES (@p0, @p1), (@p2, @p3), (@p4, @p5), (...	1.419
2.757	SELECT CASE WHEN EXISTS (SELECT 1 FROM [Catalog] AS [c]) THE ND	1.303
2.76	SELECT NEXT VALUE FOR [catalog_hilo]	0.696
2.772	SELECT NEXT VALUE FOR [catalog_hilo]	0.271
2.778	SET NOCOUNT ON; INSERT INTO [Catalog] ([Id], [CatalogBrandId], [CatalogTypeId], [Description], [Name], [Pictu...	7.701
2.954	SELECT TOP(1) [a].[Id], [a].[ConcurrencyStamp], [a].[Name], [a].[NormalizedName] FROM [AspNetRoles] AS [a] W...	6.036
2.984	SET NOCOUNT ON; INSERT INTO [AspNetRoles] ([Id], [ConcurrencyStamp], [Name], [NormalizedName]) VALUE...	1.818
3.009	SELECT TOP(1) [a].[Id], [a].[AccessFailedCount], [a].[ConcurrencyStamp], [a].[Email], [a].[EmailConfirmed], [a].[Lock...	1.754

Go To Source File

Copy

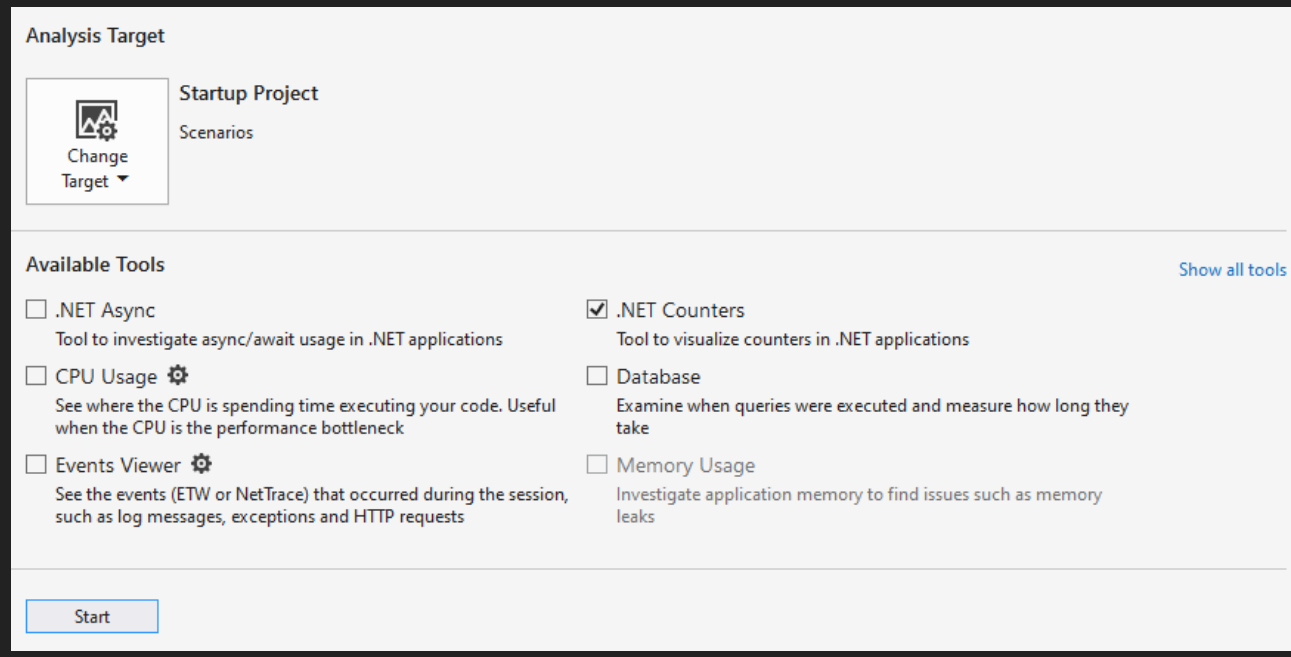


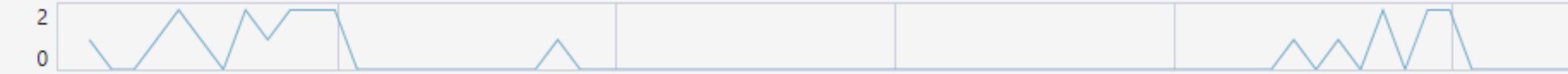
Demo

Demonstrate Database Tool

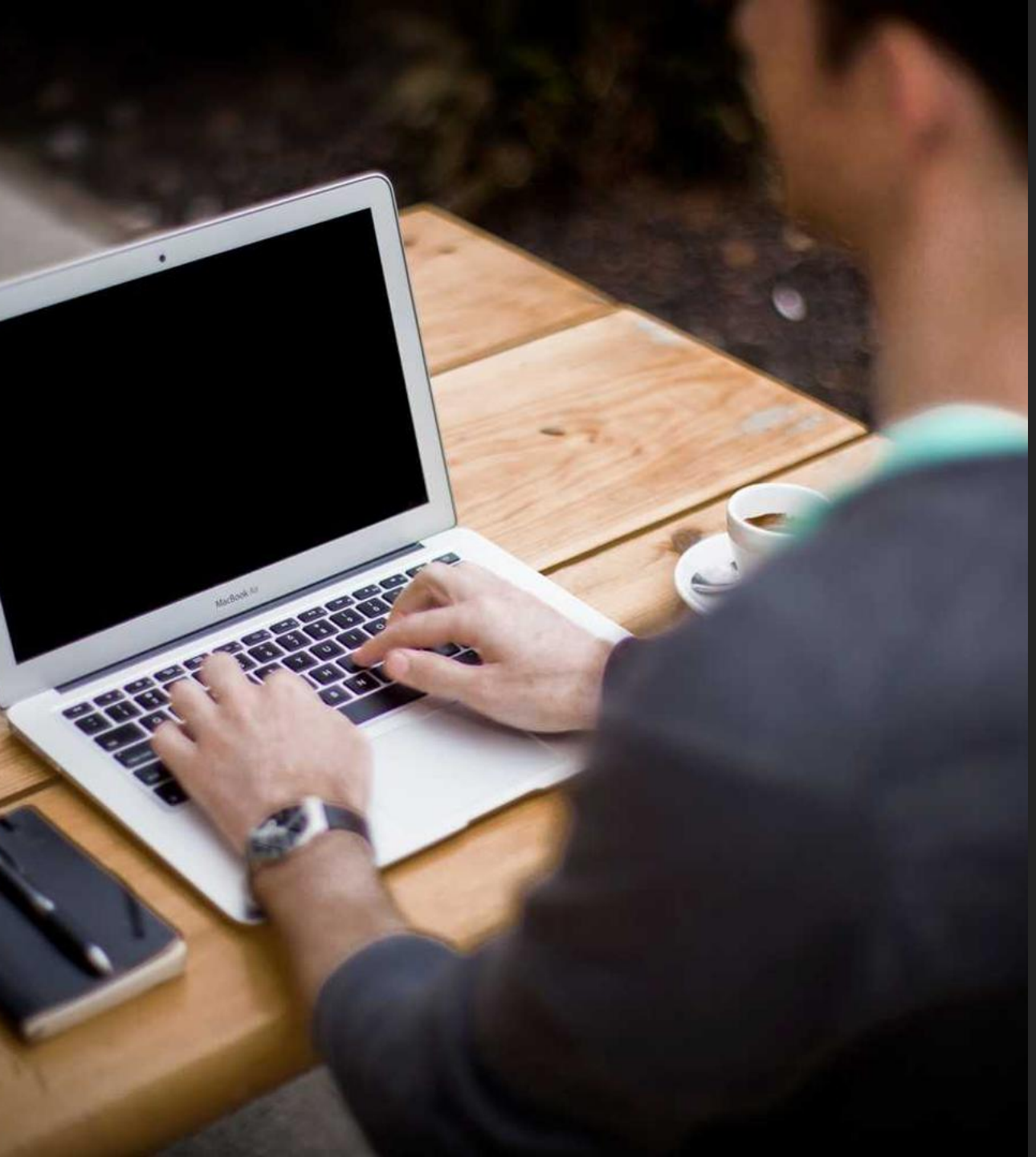
.NET Counters

The .NET Counters tool allows you to visualize dotnet counters over time right from within the Visual Studio profiler.





Name	Min	Max	Average
Microsoft.AspNetCore.Hosting			
Current Requests	0	0	0
Failed Requests	0	0	0
Request Rate	0	2	0.21
Total Requests	1	23	17.71
System.Runtime			
% Time in GC since last GC	0%	21%	3.32%
Allocation Rate	0 B	110.50 MiB	4.19 MiB
CPU Usage	0%	2%	0.04%
Exception Count	0	6	0.05
GC Heap Size	19 MiB	121 MiB	112.59 MiB
Gen 0 GC Count	0	3	0.06
Gen 0 Size	0 B	15.85 MiB	15.15 MiB
Gen 1 GC Count	0	3	0.06
Gen 1 Size	0 B	384 B	370 B
Gen 2 GC Count	0	3	0.06
Gen 2 Size	0 B	384 B	370 B
LOH Size	0 B	161.36 MiB	150.81 MiB
Monitor Lock Contention Count	0	7	0.08

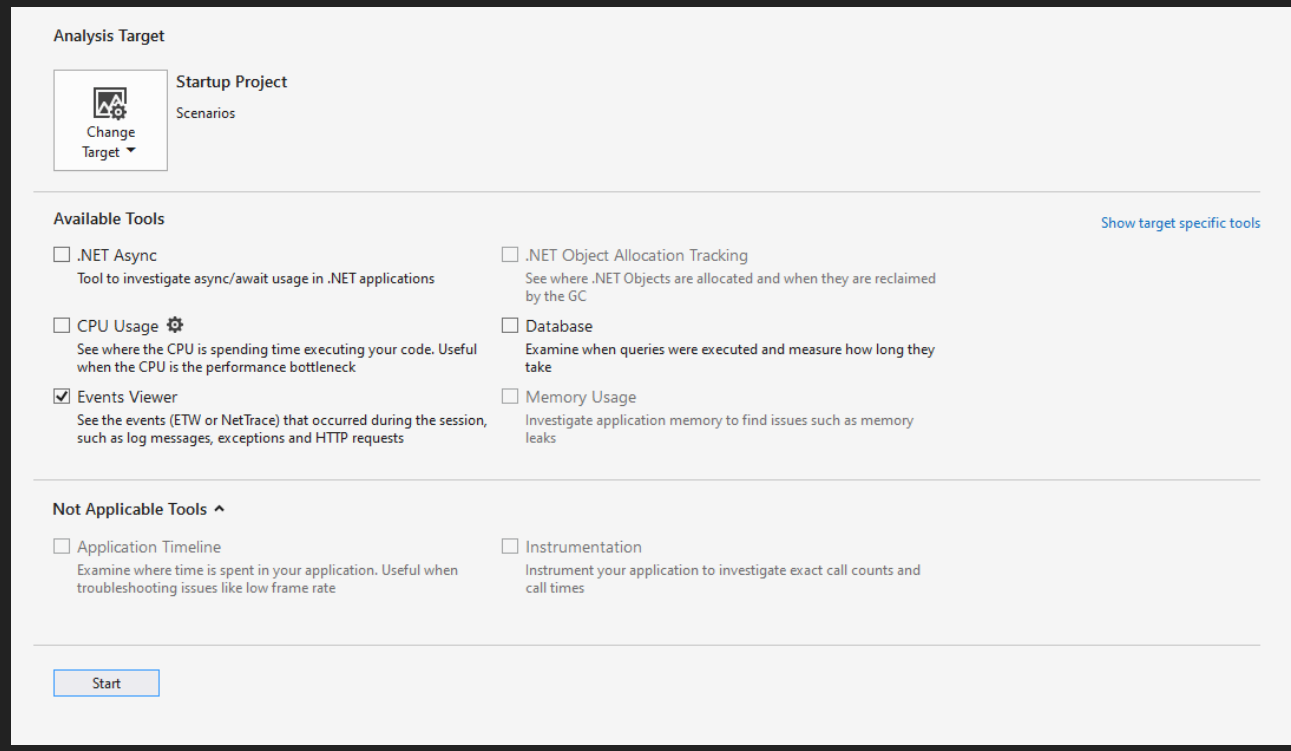


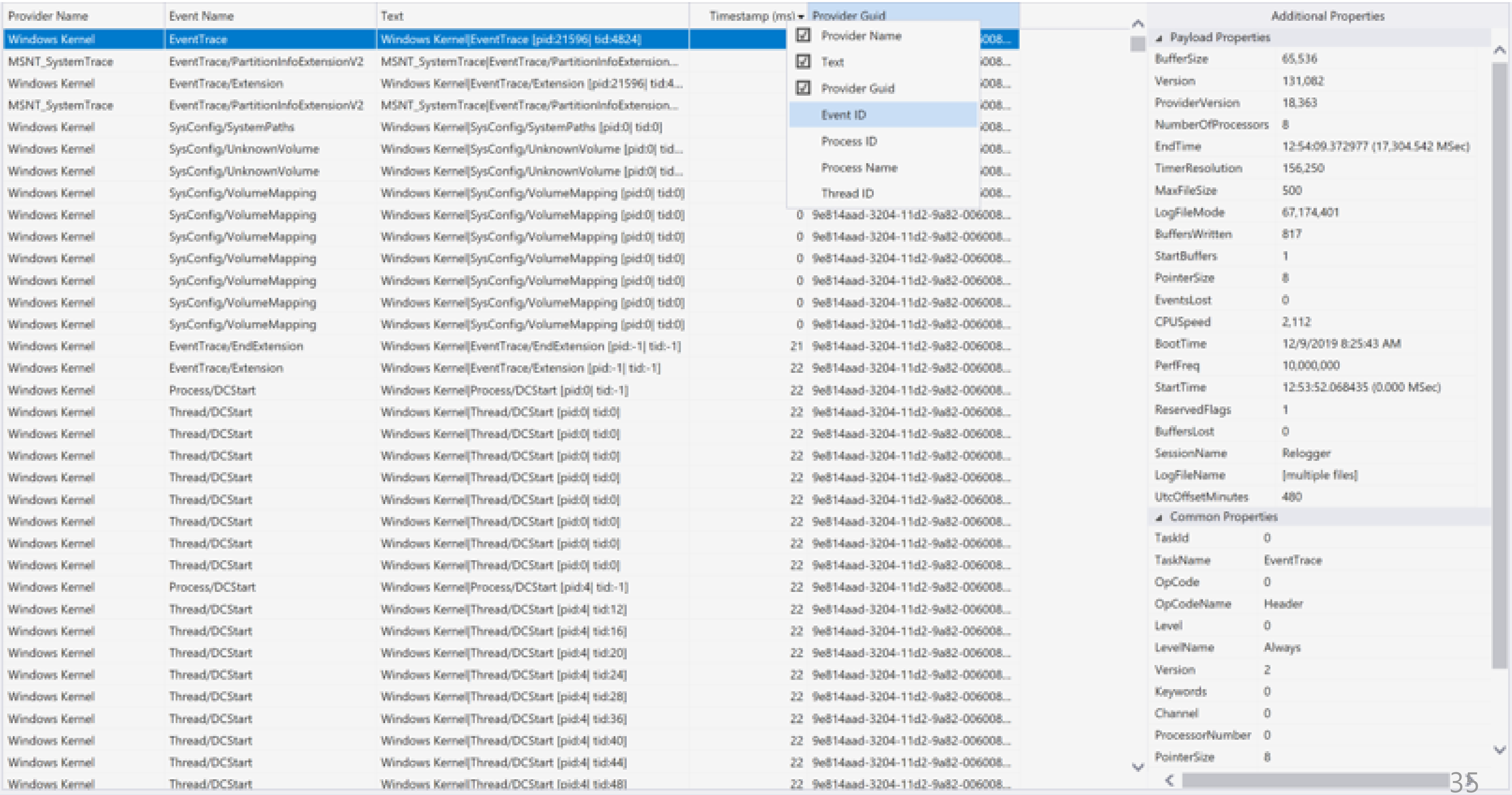
Demo

Demonstrate .NET Counters

Event Viewer

In the Performance Profiler, you can collect diagnostic info while your app is running, and then examine the collected information after the app stops like a post-mortem analysis.





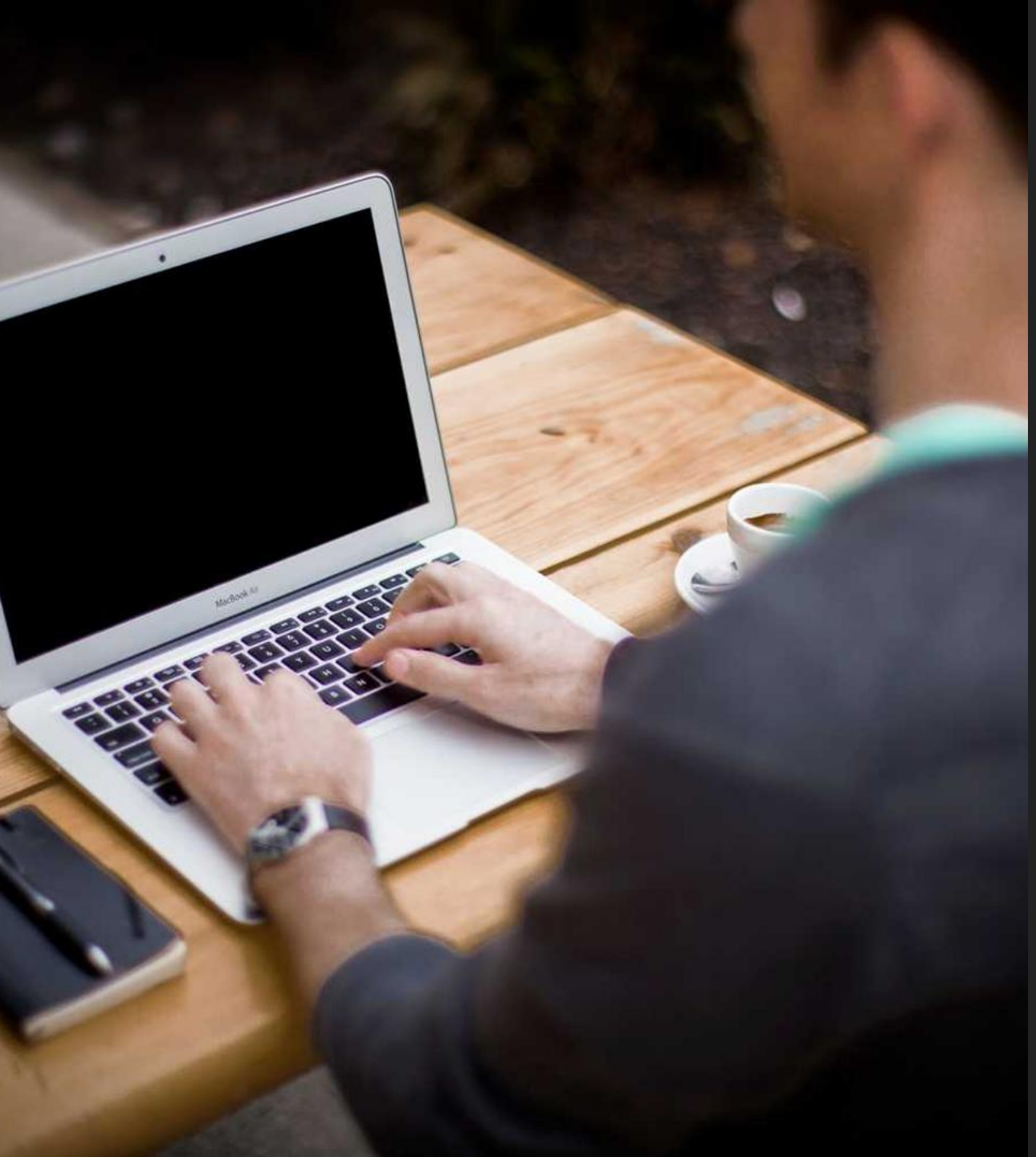
Provider Name	Event Name	Text	Timestamp (ms)	Provider Guid
Windows Kernel	EventTrace	Windows Kernel|EventTrace [pid:21596| tid:4824]		008...
MSNT_SystemTrace	EventTrace/PartitionInfoExtensionV2	MSNT_SystemTrace|EventTrace/PartitionInfoExtension...		008...
Windows Kernel	EventTrace/Extension	Windows Kernel|EventTrace/Extension [pid:21596| tid:4...		008...
MSNT_SystemTrace	EventTrace/PartitionInfoExtensionV2	MSNT_SystemTrace|EventTrace/PartitionInfoExtension...		008...
Windows Kernel	SysConfig/SystemPaths	Windows Kernel|SysConfig/SystemPaths [pid:0| tid:0]		008...
Windows Kernel	SysConfig/UnknownVolume	Windows Kernel|SysConfig/UnknownVolume [pid:0| tid...		008...
Windows Kernel	SysConfig/UnknownVolume	Windows Kernel|SysConfig/UnknownVolume [pid:0| tid...		008...
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel|SysConfig/VolumeMapping [pid:0| tid:0]		008...
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel|SysConfig/VolumeMapping [pid:0| tid:0]	0	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel|SysConfig/VolumeMapping [pid:0| tid:0]	0	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel|SysConfig/VolumeMapping [pid:0| tid:0]	0	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel|SysConfig/VolumeMapping [pid:0| tid:0]	0	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel|SysConfig/VolumeMapping [pid:0| tid:0]	0	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	SysConfig/VolumeMapping	Windows Kernel|SysConfig/VolumeMapping [pid:0| tid:0]	0	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	EventTrace/EndExtension	Windows Kernel|EventTrace/EndExtension [pid:-1| tid:-1]	21	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	EventTrace/Extension	Windows Kernel|EventTrace/Extension [pid:-1| tid:-1]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Process/DCStart	Windows Kernel|Process/DCStart [pid:0| tid:-1]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:0| tid:0]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:0| tid:0]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:0| tid:0]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:0| tid:0]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:0| tid:0]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:0| tid:0]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:0| tid:0]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:0| tid:0]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Process/DCStart	Windows Kernel|Process/DCStart [pid:4| tid:-1]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:12]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:16]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:20]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:24]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:28]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:36]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:40]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:44]	22	9e814aad-3204-11d2-9a82-006008...
Windows Kernel	Thread/DCStart	Windows Kernel|Thread/DCStart [pid:4| tid:48]	22	9e814aad-3204-11d2-9a82-006008...

Additional Properties

Payload Properties	
BufferSize	65,536
Version	131,082
ProviderVersion	18,363
NumberOfProcessors	8
EndTime	12:54:09.372977 (17,304.542 MSec)
TimerResolution	156,250
MaxFileSize	500
LogFileMode	67,174,401
BuffersWritten	817
StartBuffers	1
PointerSize	8
EventsLost	0
CPUSpeed	2,112
BootTime	12/9/2019 8:25:43 AM
PerfFreq	10,000,000
StartTime	12:53:52.068435 (0.000 MSec)
ReservedFlags	1
BuffersLost	0
SessionName	Relogger
LogFileName	[multiple files]
UtcOffsetMinutes	480

Common Properties	
TaskId	0
TaskName	EventTrace
OpCode	0
OpCodeName	Header
Level	0
LevelName	Always
Version	2
Keywords	0
Channel	0
ProcessorNumber	0
PointerSize	8

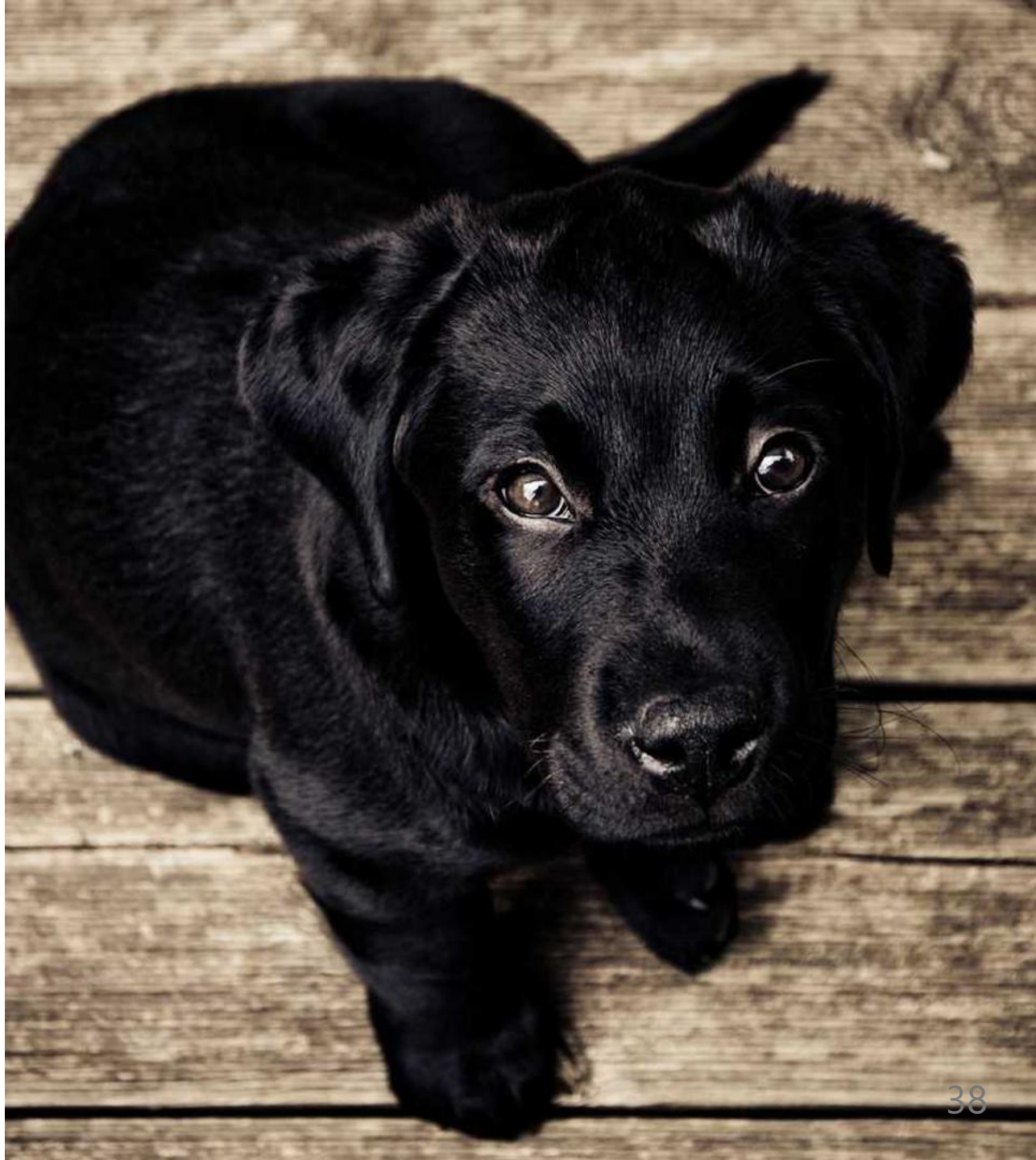
Column name	Description
Provider Name	The event source
Event Name	The event as specified by its provider
Text	Descriptions of the provider, event name, and ID for the event
Timestamp (ms)	When the event took place
Provider Guid	The ID of the event provider
Event ID	The ID of the event
Process ID	The process from which the event occurred (if known)
Process Name	The name of the process if it's actively running
Thread ID	The ID of the thread from which the event occurred (if known)



Demo

Demonstrate Event Viewer

Any Questions?



Links

- Youtube Series (aka.ms/vsprofilerseries)
- GitHub Repo (github.com/davidfowl/AspNetCoreDiagnosticScenarios)
- Documentation (learn.microsoft.com/en-us/visualstudio/profiling)
- Sysinternals for Windows (docs.microsoft.com/en-us/sysinternals)
- Sysinternals for Linux (github.com/Sysinternals)