

What's new for performance in .NET Core 2.0

Runtime changes, JIT changes
and patterns to get the most from your code

About myself

- Co-founder & CTO of Illyriad Games



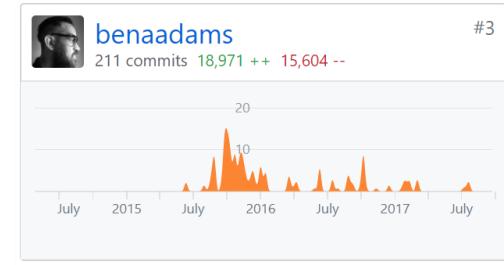
- [Age of Ascent](#) - Ultra-MMO



- Microsoft MVP



- ASP.NET Core Kestrel webserver



- CoreCLR

Avoid unnecessary work
Buffer.BlockCopy #36

Merged jkotas merged 1 commit into `dotnet:master` from `benaadams:patch-1` on Feb 4, 2015

Conversation 9 Commits 1 Files changed 1

benaadams commented on Feb 3, 2015

.NET Blog

A first-hand look from the .NET engineering teams

CoreCLR is now Open Source

February 3, 2015 by The .NET Team // 63 Comments

Member

.NET Core 1.0 & ASP.NET Core 1.0

The degree of improvement is absolutely astonishing ... approximately 85,900% improvement ... That is not a typo, it's 859 times faster!

Microsoft has made C# and ASP.NET one of the most interesting web development platforms available.

We have a brief message to those developers who have avoided Microsoft's web stack thinking it's "slow" or that it's for Windows only: ASP.NET Core is now wicked sick fast

Oh, and of course we're running it on Linux.

You may be thinking about the Microsoft of 10 years ago.

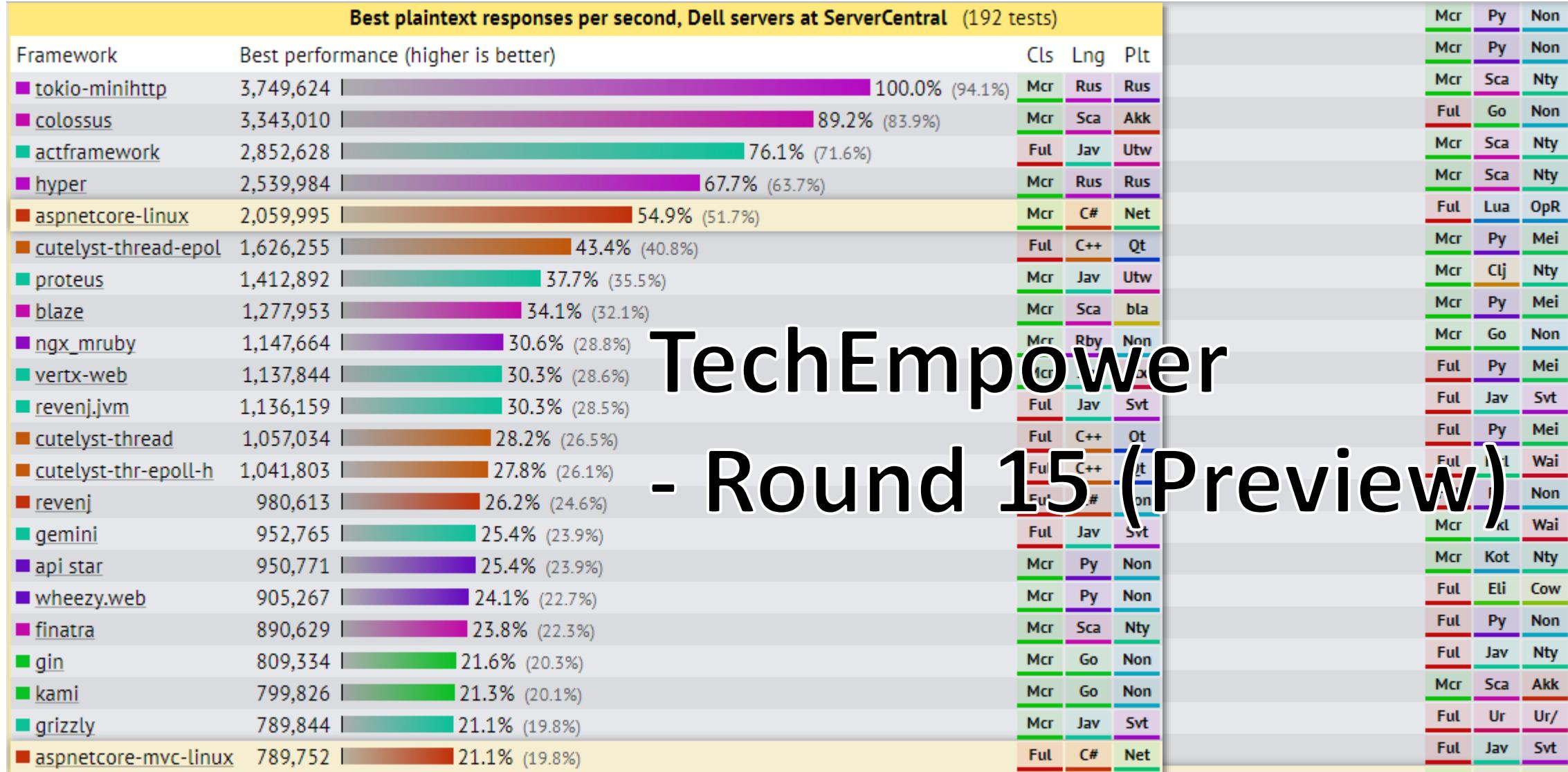
.NET Core 2.0

The Best Yet!

Learn more:
[Announcing .NET Core 2.0](#)
- Rich Lander

.NET Core 2.0

- Even Faster
- Profile-guided Optimized Runtime
- Runtime Store for .NET and ASP.NET
- Even more cross-platform
 - Windows, macOS, Red Hat, CentOS, Fedora, Debian, Ubuntu, Mint, SUSE, ...
- Side-by-side Runtime Installs
 - No upgrade fears - Apps on same server can use different versions
- .NET Standard 2.0
 - Added 19k+ .NET Framework APIs for 32k+ APIs
- Allows a more compact .csproj



TechEmpower

- Round 15 (Preview)

Automatic Gains

Just from using .NET Core 2.0

No code changes

The Framework is Faster

Just by using .NET Core 2.0

Learn more:
[Performance Improvements in .NET Core](#)
- Stephen Toub

Collections are faster...

- **Queue<T>**

Enqueue/Dequeue + 200% faster

- **List<T>**

Add/Remove + 40% faster

Clear (Primitive Types) – Instant!

- **ConcurrentQueue<T>**

Now non-allocating!

Enqueue/TryDequeue + 350% faster

- **ConcurrentBag<T>**

Now non-allocating!

Enqueue/TryDequeue + 30% faster

Text is faster...

- **Guid.ToString** + 69% faster
- **Encoding.UTF8.GetBytes** + 48% faster
- **DateTime.ToString** + 300% faster
- **String**
 - IndexOf + 200% faster
 - StartsWith + 200% faster
- **Enum**
 - Parse + 33% faster
 - ToString + 600% faster

Concurrency is faster...

- `ThreadPool` + 66% faster

- Fail fast contended locks

`SpinLock.TryEnter` + 600% faster

`SemaphoreSlim.WaitAsync(0)` + 140% faster

`SemaphoreSlim.Wait(0)` + 27,515% faster

`Monitor.TryEnter (Contended)` + 4,038,980% faster

- `Lazy<T>` + 500% faster

I/O is faster...

- **FileStream**

CopyToAsync + 46% faster

- **Sockets**

SocketAsyncEventArgs + 360% faster

- **NetworkStream**

CopyToAsync + 500% faster

- **SslStream**

+ 374% faster

Huge allocation reduction

Everything is faster!

- Compression (`DeflateStream`) + 400% faster
- Cryptography
 - Now uses Native OS Implementation
 - `SHA256.ComputeHash` + 200% faster
- Math
 - `BigInteger` + 400% faster
 - `DivRem` + 200% faster
- LINQ
 - Huge rework, up to +300% faster in places
- So much more..!

The Runtime is Faster

...and it gets going faster

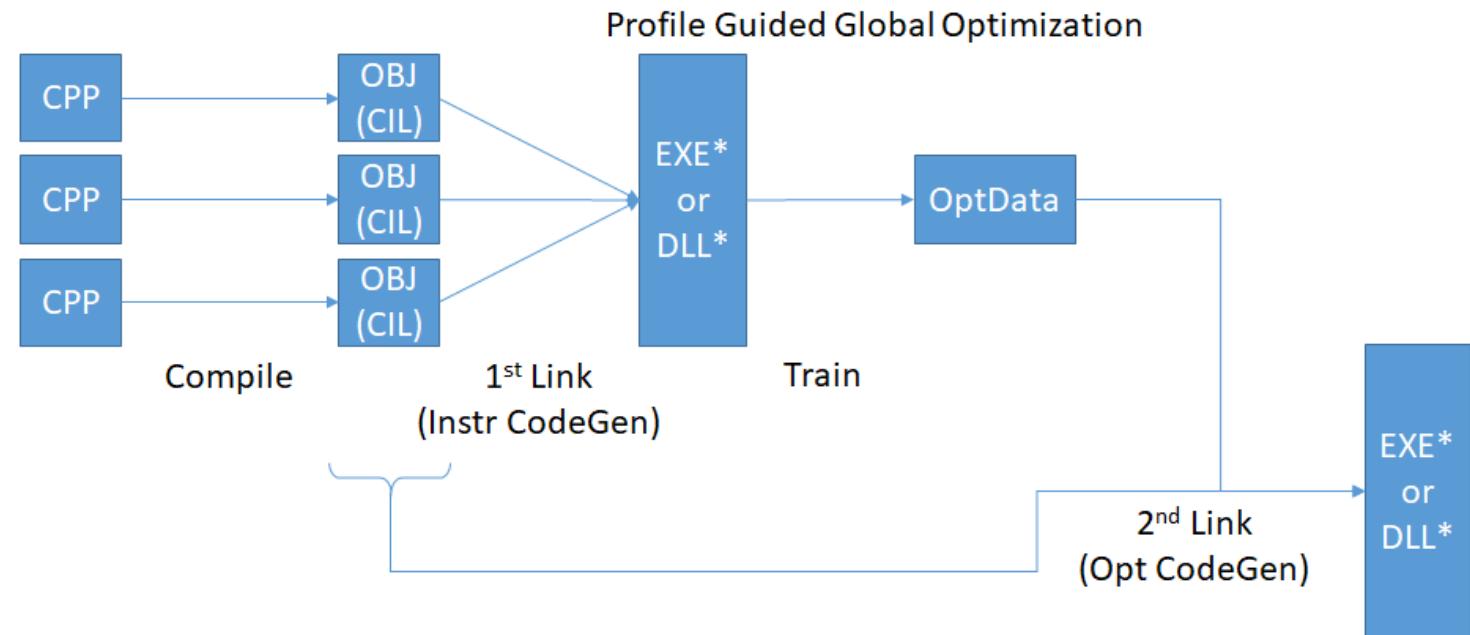
Learn more:

[Profile-guided optimization in .NET Core 2.0](#)

- Daniel Podder and Bertrand Le Roy

Profile Guided Optimization

- Affects the native part of the runtime
 - Faster start-up
 - Faster Jitting
 - Faster GC



The Jit Produces Faster Code

RyuJIT is magic

Learn more:

[Performance Improvements in RyuJIT in .NET Core and .NET Framework](#)

- Joseph Tremoulet

RyuJIT

- Now used for x86 as well as x64 and ARM
- Rotate Patterns
- Devirtualization
(and sealed now means something)
- Enhanced Range Inference
- Shift Count Mask Removal
- Improved Vectors

}

Learn more:

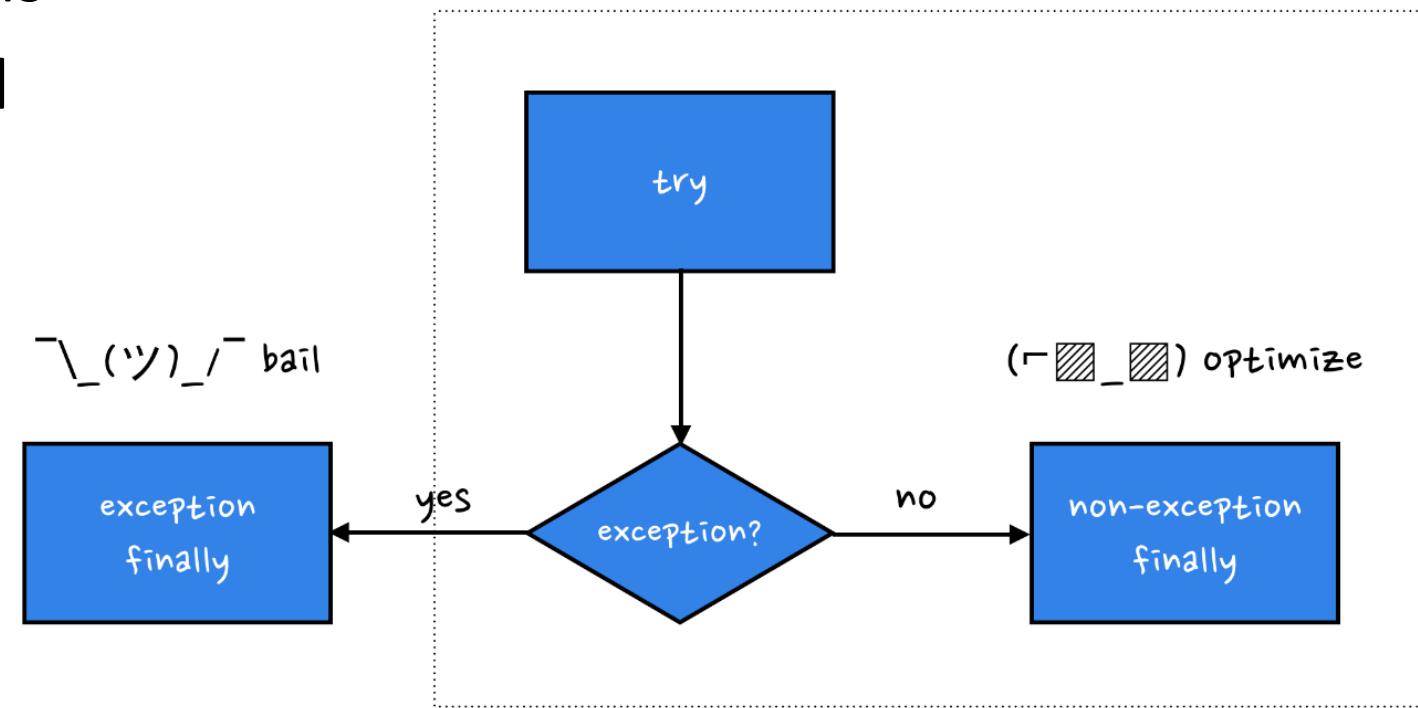
[coreclr#9908 JIT: devirtualization next steps](#)
- Andy Ayers

RyuJIT - Exception Handling

- Enabled Inlining of methods with conditional throws
- Non-returning methods
- Empty Finally Removal
- Finally Cloning

Learn more:
[Do not inline methods that never return](#)
- mikedn

Learn more:
[Finally optimizations](#)
- Andy Ayers



throw is free?

If it never runs..?

But... can miss out on Inlining

```
public Segment(T[] array, int start, int length)
{
    if (array == null)
    {
        throw new ArgumentNullException(nameof(array));
    }
    if (start < 0)
    {
        throw new ArgumentOutOfRangeException(nameof(start));
    }
    if (start >= array.Length)
    {
        throw new ArgumentOutOfRangeException(nameof(start));
    }
    if (length < 0)
    {
        throw new ArgumentOutOfRangeException(nameof(length));
    }
    // subtract start else may overflow
    if (length > array.Length - start)
    {
        throw new ArgumentOutOfRangeException(nameof(length));
    }
    _array = array;
    _start = start;
    _length = length;
}
```

102 bytes cil
367 bytes asm
[FAILED: too many il bytes]

Marking as **NOINLINE** because of too many il bytes

```
public Segment(T[] array, int start, int length)
{
    if (array == null)
    {
        throw new ArgumentNullException(nameof(array));
    }
    if (start < 0 || start >= array.Length)
    {
        throw new ArgumentOutOfRangeException(nameof(start));
    }
    // subtract start else may overflow
    if (length < 0 || length > array.Length - start)
    {
        throw new ArgumentOutOfRangeException(nameof(length));
    }

    _array = array;
    _start = start;
    _length = length;
}
```

80 bytes cil
249 bytes asm
[FAILED: unprofitable inline]

Two less tests

```
if ((uint)start >= (uint)array.Length)
```

Quicker way to test both

```
if (start < 0 ||  
    start >= array.Length)
```

Negative as uint > int.MaxValue

```
public Segment(T[] array, int start, int length)  
{  
    if (array == null)  
    {  
        throw new ArgumentNullException(nameof(array));  
    }  
    if ((uint)start >= (uint)array.Length)  
    {  
        throw new ArgumentOutOfRangeException(nameof(start));  
    }  
    // subtract start else may overflow  
    if ((uint)length > (uint)(array.Length - start))  
    {  
        throw new ArgumentOutOfRangeException(nameof(length));  
    }  
  
    _array = array;  
    _start = start;  
    _length = length;  
}
```

72 bytes cil

237 bytes asm

[FAILED: unprofitable inline]

Non-returning methods

```
public Segment(T[] array, int start, int length)
{
    if (array == null ||
        (uint)start >= (uint)array.Length ||
        (uint)length > (uint)(array.Length - start))
    {
        ThrowHelper.ThrowValidationError(array, start);
    }

    _array = array;
    _start = start;
    _length = length;
}
```

46 bytes cil

72 bytes asm

[profitable inline]

```
internal static class ThrowHelper
{
    1 reference
    public static void ThrowValidationError(Array array, int start)
    {
        throw GetValidationError(array, start);
    }

    // Don't inline on older runtimes
    [MethodImpl(MethodImplOptions.NoInlining)]
    1 reference
    private static Exception GetValidationError(Array array, int start)
    {
        if (array == null)
        {
            throw new ArgumentNullException(nameof(array));
        }
        if ((uint)start >= (uint)array.Length)
        {
            throw new ArgumentOutOfRangeException(nameof(start));
        }
        throw new ArgumentOutOfRangeException("length");
    }
}
```

New terser .csproj (1)

- Huge list of files, gone

```
<!-- the defaults -->
<Compile Include="**\*.cs" />
<EmbeddedResource Include="**\*.resx" />
```

- Project References

Old

```
<ProjectReference Include="..\ClassLibrary1\ClassLibrary1.csproj">
  <Project>{2C7DF870-5B35-49EF-963D-EE1E72C3177E}</Project>
  <Name>ClassLibrary1</Name>
</ProjectReference>
```

New

```
<ProjectReference Include="..\ClassLibrary1\ClassLibrary1.csproj" />
```

Learn more:

[Old csproj to new csproj: Visual Studio 2017 upgrade guide](#)
- Nate McMaster

New terser .csproj (2)

- Removed packages.config and packages directory
- Old

```
<Import Project="..\..\packages\xunit.runner.visualstudio.2.2.0\build\net20\xunit.runner.visualstudio.props" Condition="

<ItemGroup>
  <None Include="packages.config" />

  <Reference Include=" MySql.Data, Version=6.9.9.0, Culture=neutral, PublicKeyToken=c5687fc88969c44d, processorArchitect
    <HintPath>..\..\packages\MySql.Data.6.9.9\lib\net45\MySql.Data.dll</HintPath>
    <Private>True</Private>
  </Reference>
</ItemGroup>
```

- New

```
<ItemGroup>
  <PackageReference Include=" MySql.Data" Version="6.9.9" />
</ItemGroup>
```

Common Performance Issues

Thread-pool starvation

- Isn't "sync vs async"; its blocking vs async
- Blocking calls on Threadpool threads
 - Read -> ReadAsync
 - Write -> WriteAsync
 - task.Result -> await task
 - task.GetAwaiter.GetResult() -> await task
 - task.Wait() -> await task
 - Task.WaitAny(...) -> await Task.WhenAny(...)
 - Task.WaitAll(...) -> await Task.WhenAll(...)
- Can use `.Result` **if you have checked** `.IsCompletedSuccessfully == true`

Detect Blocking Calls

```
TaskBlockingListener  
    .Start();
```

- **.Result**
- **.GetAwaiter.GetResult()**
- **.Wait()**
- **Task.Wait(...)**
- **Task.WaitAll(...)**

```
public sealed class TaskBlockingListener : EventListener  
{  
    private static readonly Guid s_tplGuid = new Guid("2e5dba47-a3d2-4d16-8ee0-6671ffcd7b5");  
    private static readonly Lazy<TaskBlockingListener> s_listener  
        = new Lazy<TaskBlockingListener>(() => new TaskBlockingListener());  
  
    public static void Start() { object ignored = s_listener.Value; }  
    private TaskBlockingListener() { }  
    protected override void OnEventSourceCreated(EventSource eventSource)  
    {  
        if (eventSource.Guid == s_tplGuid)  
        {  
            // 3 == Task|TaskTransfer  
            EnableEvents(eventSource, EventLevel.Informational, (EventKeywords)3);  
        }  
    }  
    protected override void OnEventWritten(EventWrittenEventArgs eventData)  
    {  
        if (eventData.EventId == 10 && // TASKWAITBEGIN_ID  
            eventData.Payload != null &&  
            eventData.Payload.Count > 3 &&  
            eventData.Payload[3] is int value && // Behavior  
            value == 1) // TaskWaitBehavior.Synchronous  
        {  
            Console.WriteLine("Blocking on an incomplete task.\n" + Environment.StackTrace);  
        }  
    }  
}
```

Learn more:

Tips of the Toub [corefx#8931 \(comment\)](#)
- Stephen Toub

Panic Workaround

You are in production and you need a panic workaround; what to do?

Unthrottle the thread pool min threads:

```
using System.Threading;

var threadCount = 2000; // 2GB :-(  
ThreadPool.GetMaxThreads(out _, out var completionThreads);  
ThreadPool.SetMinThreads(threadCount, completionThreads);
```

Then fix the source... go async

HttpClient

- **Use xAsync methods** (on threadpool thread)
Thread-pool starvation
- **Either Dispose**
Keep-alive connections increasing outbound connections
- **Or use static HttpClient**
Less load on server
Don't change DefaultHeaders – Not thread-safe
- **Increase concurrent outbound connections**
`ServicePointManager.DefaultConnectionLimit = 1024`
- **Pipelining**

Garbage Collector

The Garbage Collector is your friend

- Reduces errors
- Easier development
- Allows **very fast** allocation
 - ... **but**, you pay for it later
- Not deterministic
- If you allocate a lot, you can get pauses

What does a GC Pause look like?



([Video](#))



Ben Adams
@ben_a_adams



Blaming perf issues on Garbage Collection is like blaming your hangover on your liver... Its the thing that's saving you from your code

2:41 AM - 21 Aug 2016

578 Retweets 629 Likes



20

578

629





Niall Connaughton @nconnaughton · 3h

Replies to @matthewwarren @ben_a_adams @LeeRyanCampbell

So often the focus with GC is "how do I make it stop hurting me?!", when it should be "how do I stop hurting the GC?"



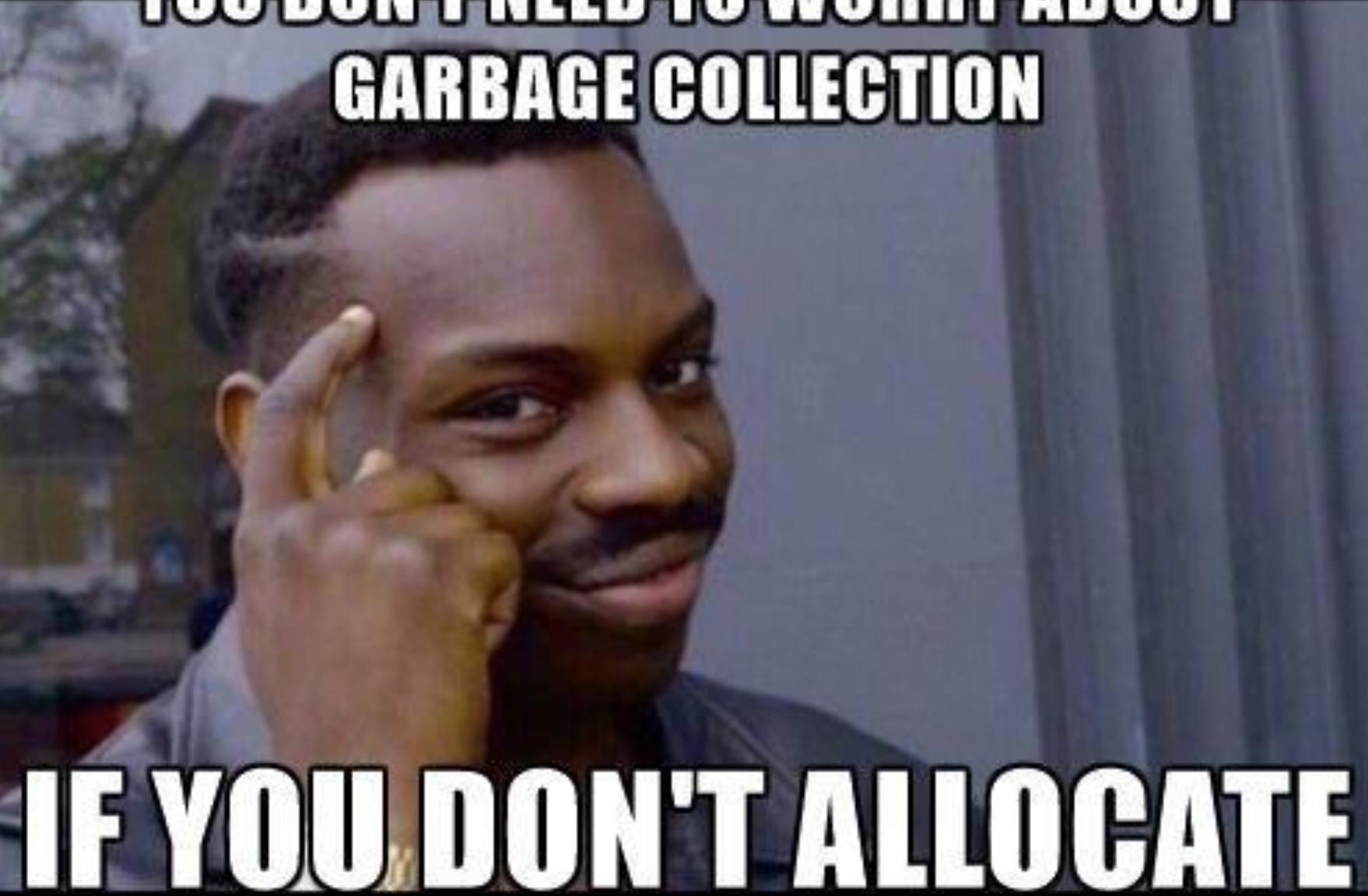
Nick Craver ✅ @Nick_Craver · 34m

Ask not what your garbage collector can do for you — ask what you can do for your garbage collector.

Niall Connaughton @nconnaughton

Replies to @matthewwarren @ben_a_adams @LeeRyanCampbell

So often the focus with GC is "how do I make it stop hurting me?!", when it should be "how do I stop hurting the GC?"



**YOU DON'T NEED TO WORRY ABOUT
GARBAGE COLLECTION**

IF YOU DON'T ALLOCATE

Gains with Changes

Allocate less

- Don't box! (Allocates and is slow)
- Don't capture in Lambdas (Closures)
- Don't use LINQ and `IEnumerable<T>` in hot paths
- Object Pools
- `ArrayPool<T>`
- `Span<T>` & `ReadOnlySpan<T>`
- `ValueTask<T>`
- Cached Tasks (e.g. `Task.CompletedTask`)

Learn more:

[.NET Core: Performance Revolution](#)

- Adam Sitnik

ValueTask<T>

```
[MethodImpl(MethodImplOptions.AggressiveInlining)]
0 references
ValueTask<int>·ReadAsync(byte[]·array)
    =>·IsDataReady
    ... ?·new·ValueTask<int>(result:·GetData(array))·//·INLINEABLE!!!
    ... ::new·ValueTask<int>(task:·GetDataAsync(array));|
```

1 reference

```
int·GetData(byte[]·array)
```

1 reference

```
Task<int>·GetDataAsync(byte[]·array)
```

```
        async·ValueTask<long>·CountDataLength(int·repeats)
{
    long·total·=·0;
    int·current;
    do
    {
        ValueTask<int>·valueTask·=·ReadAsync(_scratchBuffer);·//·INLINEABLE
        current·=·valueTask.IsCompletedSuccessfully
            ?·valueTask.Result·//·hot·path
            :·await·valueTask.AsTask();

        total·+=·current;
    }·while·(current·>·0);
    return·total;
}
```

Learn more:
[.NET Core: Performance Revolution](#)
- Adam Sitnik

Ref returns

```
; Assembly listing for method Dictionary`2::TryGetValue(char,byref):bool:this  
; Emitting BLENDED_CODE for X64 CPU with SSE2  
; Lcl frame size = 40
```

```
G_M28467_IG01:  
    57          push    rdi  
    56          push    rsi  
    4883EC28  sub     rsp, 40  
    488BF1    mov     rsi, rcx  
    498BF8    mov     rdi, r8  
  
G_M28467_IG02:  
    0FB7D2    movzx   rdx, dx  
    488BCE    mov     rcx, rsi  
    E800000000  call    Dictionary`2::FindEntry(char):int:this  
    85C0      test    eax, eax  
    7C29      jl     SHORT G_M28467_IG04  
    488B5610  mov     rdx, gword ptr [rsi+16]  
    3B4208    cmp     eax, dword ptr [rdx+8]  
    732C      jae    SHORT G_M28467_IG06  
    4863C8    movsxsd rcx, eax  
    488D0C49  lea     rcx, [rcx+2*rcx]  
    488B54CA10  mov     rdx, gword ptr [rdx+8*rcx+16]  
    488BCF    mov     rcx, rdi  
    E800000000  call    CORINFO_HELP_CHECKED_ASSIGN_REF  
    B801000000  mov     eax, 1  
  
G_M28467_IG03:  
    4883C428  add     rsp, 40  
    5E          pop    rsi  
    5F          pop    rdi  
    C3          ret  
  
G_M28467_IG04:  
    33C0      xor     rax, rax  
    488907    mov     qword ptr [rdi], rax  
  
G_M28467_IG05:  
    4883C428  add     rsp, 40  
    5E          pop    rsi  
    5F          pop    rdi  
    C3          ret  
  
G_M28467_IG06:  
    E800000000  call    CORINFO_HELP_RNGCHKFAIL  
    CC          int3  
  
; Total bytes of code 86, prolog size 6 for method  
Dictionary`2::TryGetValue(char,byref):bool:this  
=====
```

C#6

```
; Assembly listing for method Dictionary`2::TryGetValue(char,byref):bool:this  
; Emitting BLENDED_CODE for X64 CPU with SSE2  
; Lcl frame size = 48  
  
G_M28467_IG01:  
    56          push    rsi  
    4883EC30  sub     rsp, 48  
    498BF0    mov     rsi, r8  
  
G_M28467_IG02:  
    0FB7D2    movzx   rdx, dx  
    4C8D442428  lea     r8, bword ptr [rsp+28H]  
    E800000000  call    Dictionary`2::FindEntryByRef(char,byref):byref:this :this  
    807C242800  cmp     byte ptr [rsp+28H], 0  
    7507      jne    SHORT G_M28467_IG03  
    488BCE    mov     rcx, rsi  
    33D2      xor     rdx, rdx  
    EB06      jmp    SHORT G_M28467_IG04  
  
G_M28467_IG03:  
    488BCE    mov     rcx, rsi  
    488B10    mov     rdx, gword ptr [rax]  
  
G_M28467_IG04:  
    E800000000  call    CORINFO_HELP_CHECKED_ASSIGN_REF  
    8B442428  mov     eax, dword ptr [rsp+28H]  
    0FB6C0    movzx   rax, al  
  
G_M28467_IG05:  
    4883C430  add     rsp, 48  
    5E          pop    rsi  
    C3          ret  
  
; Total bytes of code 59, prolog size 5 f  
=====
```

C#7

```
public bool TryGetValue(TKey key, out TValue value)  
{  
    int i = FindEntry(key);  
    if (i >= 0)  
    {  
        value = _entries[i].value;  
        return true;  
    }  
    value = default(TValue);  
    return false;  
}
```

```
public bool TryGetValue(TKey key, out TValue value)  
{  
    ref var entry = ref FindEntryByRef(key, out var found);  
    value = found ? entry.value : default(TValue);  
    return found;  
}
```

Runtime Store



Nick Craver ✅ @Nick_Craver · Aug 14

The entire MiniProfiler #AspNetCore sample **upgrade** to 2.0 in pictures:

```
4      - <TargetFramework>netcoreapp1.0</TargetFramework>
4      + <TargetFramework>netcoreapp2.0</TargetFramework>
5      - <PropertyGroup Label="RuntimeType">
6      -   <RuntimeType>aspnetcore</RuntimeType>
7      -   <BuildGenerateCredential Include="Dapper" Version="1.3.0.1" />
8
9
10     - <PackageReference Include="Microsoft.AspNetCore.Diagnostics" Version="1.1.1" />
11     - <PackageReference Include="Microsoft.AspNetCore.Mvc" Version="1.1.2" />
12     - <PackageReference Include="Microsoft.AspNetCore.Routing" Version="1.1.1" />
13     - <PackageReference Include="Microsoft.AspNetCore.Server.IISIntegration" Version="1.1.1" />
14     - <PackageReference Include="Microsoft.AspNetCore.Server.Kestrel" Version="1.1.1" />
15     - <PackageReference Include="Microsoft.AspNetCore.StaticFiles" Version="1.1.1" />
16     - <PackageReference Include="Microsoft.Data.SQLite" Version="1.1.0" />
17     - <PackageReference Include="Microsoft.EntityFrameworkCore.Sqlite" Version="1.1.1" />
18     - <PackageReference Include="Microsoft.Extensions.Configuration.EnvironmentVariables" Version="1.1.1" />
19     - <PackageReference Include="Microsoft.Extensions.Configuration.Json" Version="1.1.1" />
20     - <PackageReference Include="Microsoft.Extensions.Logging" Version="1.1.1" />
21     - <PackageReference Include="Microsoft.Extensions.Logging.Console" Version="1.1.1" />
22     - <PackageReference Include="Microsoft.Extensions.Logging.Debug" Version="1.1.1" />
23     - <PackageReference Include="Microsoft.Extensions.Options.ConfigurationExtensions" Version="1.1.1" />
24     - <PackageReference Include="Microsoft.VisualStudio.Web.BrowserLink.Loader" Version="14.1.0" />
25
26     12  + <PackageReference Include="Microsoft.AspNetCore.All" Version="2.0.0" />
```

Tools

To help you make things faster

Profilers

Discover hot spots in the code

- dotTrace ([JetBrains](#)) (+ memory in Timeline Trace)
- ANTS Performance Profiler ([redgate](#))
- NProfiler www.nprofiler.com
- CodeTrack www.getcodetrack.com

Discover allocations in the code

- dotMemory ([JetBrains](#))
- [.NET Memory Profiler](#)

Do both

- Performance Explorer (Visual Studio)
- [PerfView](#) (watch channel9 videos on how to use!)

Isolate a function and iterate improvements

- [BenchmarkDotNet](#)

Self-contained Apps

Faster deployment

- Own copy of runtime – xcopy deploy on vanilla machine
- Needs to be built for target platform
- Patching (security) needs to be rebuilt and redeployed
 - Standard/Portable Apps only need runtime patched on machine
- .NET IL Linker

Installed runtime

- 307 kB for WebApi + .All

Standalone

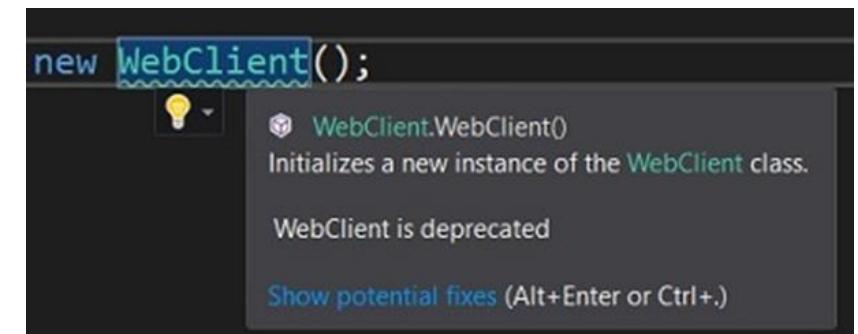
- 68 MB for console & only Kestrel (+deps)
- 38 MB using .NET IL Linker
- 93 MB WebApi + .All

	notlinked	linked
Type:	File folder	File folder
Location:	C:\GitHub\netperf\AspNetCore	C:\GitHub\netperf\AspNetCore
Size:	68.3 MB (71,633,530 bytes)	38.1 MB (40,053,070 bytes)
Size on disk:	69.0 MB (72,454,144 bytes)	38.8 MB (40,734,720 bytes)
Contains:	409 Files, 5 Folders	366 Files, 10 Folders

Learn more:
[ILLinker instructions](#)
(available on myget)

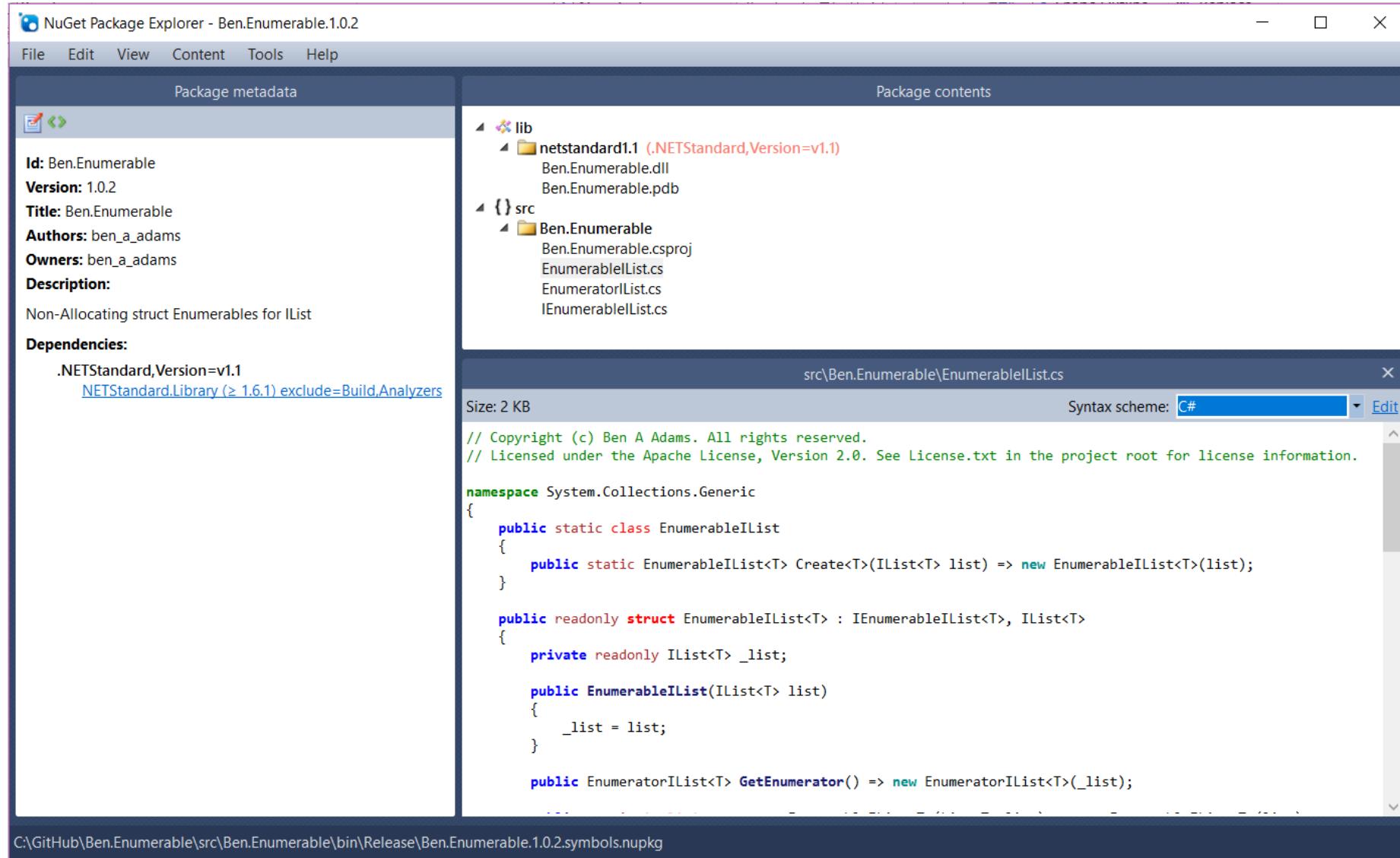
API Compatibility Analyzer

- Use of .NET Standard API that will throw PNSE (cross platform)
- Use of a .NET Standard API not available on .NET Framework 4.6.1
- Use of a native API that doesn't exist in UWP
- Use of an API that is marked as deprecated
- Roslyn analyzer, so works in VS, CLI, CI



Learn more:
[Introducing API Analyzer](#)
- Olia Gavrysh and Immo Landwerth

NuGet Package Explorer



NuGet Package Explorer
- [in Microsoft Store \(free\)](#)

MSBuild Binary and Structured Log Viewer

- Binary logs
- Faster builds
- Smaller logs
- Easier to analyze

The screenshot shows the MSBuild Structured Log Viewer application interface. On the left, there is a search and filtering panel with instructions for searching the log. The main area displays a hierarchical tree view of MSBuild tasks and properties for two projects: "MSBuildStructuredLog.sln" and "StructuredLogViewer.csproj". The "Task FindAppConfigFile" is highlighted in blue. On the right, the XML code for the "Microsoft.Common.CurrentVersion.targets" file is shown, with line numbers from 1043 to 1080. The XML code includes various MSBuild constructs like targets, properties, and tasks, with some parts highlighted in green and red.

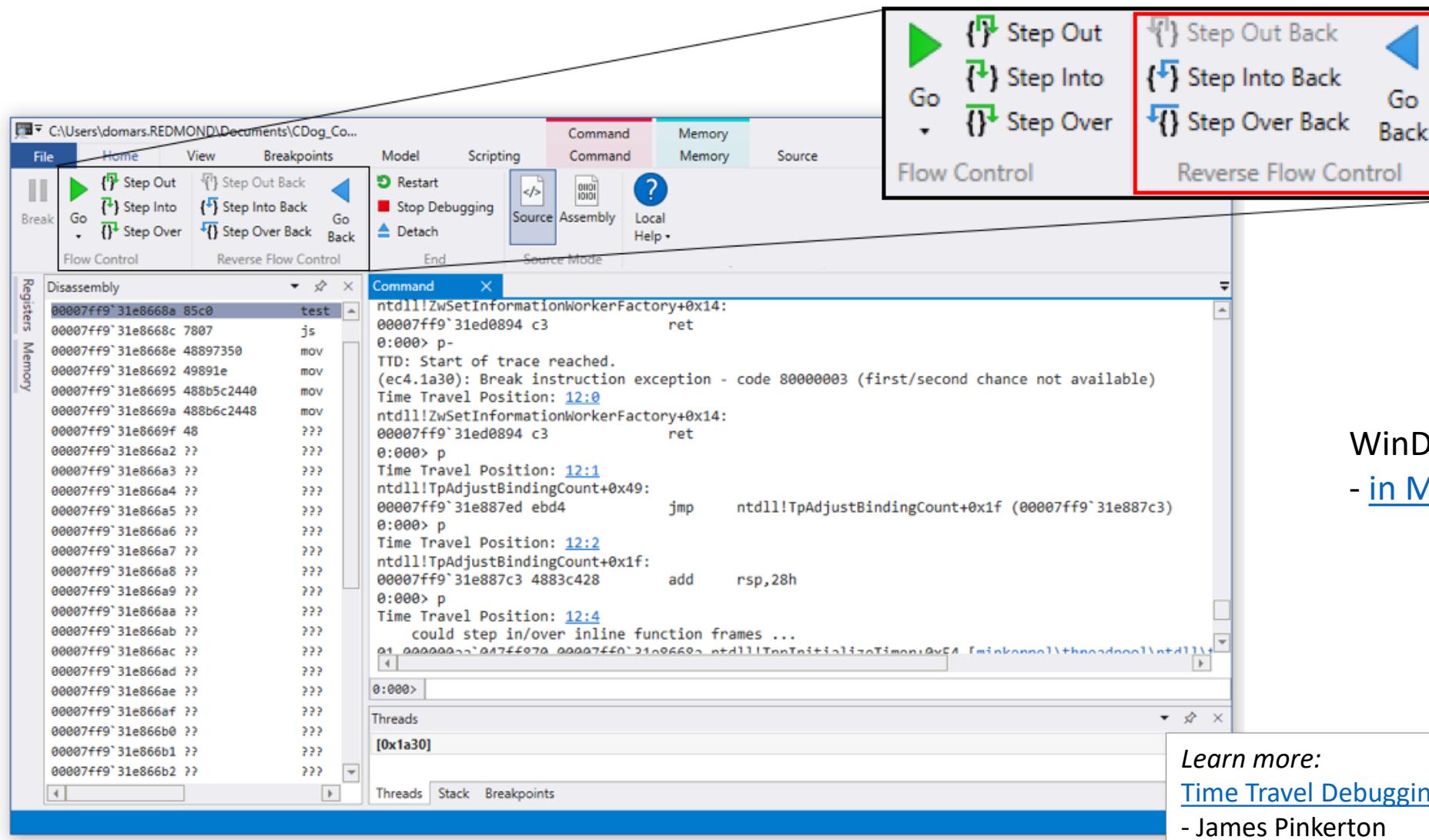
```
C:\MSBuildStructuredLog\msbuild.binlog - MSBuild Structured Log Viewer
File Help
Type in the search box to search. Press Ctrl+F to focus the search box. Results (up to 1000) will display here.
Search for multiple words separated by space (space means AND). Enclose multiple words in double-quotes "" to search for the exact phrase.
Use syntax like '$property Prop' to narrow results down by item kind. Supported kinds: $project $target $task $error $warning $message $property $item $additem $removeitem $metadata $copytask
Use the under(FILTER) clause to filter results to only the nodes where any of the parent nodes in the parent chain matches the FILTER. Examples:
  • $task csc under($project Core)
  • Copying file under(Parent)
Examples:
  • Copying file from
  • Resolved file path is
  • There was a conflict
  • Building target completely
  • is newer than output
  • Property reassignment: ${Importing project}
  • was not imported by
  • out-of-date
  • csc $task
  • ResolveAssemblyReference $task
Search Log Files Find in Files
Project "MSBuildStructuredLog.sln" (default targets): > Project "StructuredLogViewer.csproj" (default targets): > Target PrepareForBuild > Task FindAppConfigFile
Microsoft.Common.CurrentVersion.targets X
Open Copy Path Preprocess Word Wrap C:\Program Files (x86)\Microsoft
1043
1044 <!--
1045 =====
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
=====
Prepare the prerequisites for building.
=====
<PropertyGroup>
<PrepareForBuildDependsOn>GetFrameworkPaths;GetIntermediateDirectories;FindAppConfigFile;CreateAppConfigFile;ResolveReferences;ResolveProjectReferences;PrepareForBuild
</PropertyGroup>
<Target
  Name="PrepareForBuild"
  DependsOnTargets="$(PrepareForBuildDependsOn)">
<ItemGroup>
<AppConfigWithTargetPath Include="$(AppConfigFile)">
<TargetPath>$(TargetFileName).config</TargetPath>
</AppConfigWithTargetPath>
</ItemGroup>
<FindAppConfigFile PrimaryList="@None" SecondaryList="@(AppConfigFile)" ItemName="AppConfigFile" OutputTaskParameter="AppConfigFile" Properties="%(Properties)">
<Output TaskParameter="AppConfigFile" ItemName="AppConfigFile" Properties="%(Properties)">
</FindAppConfigfile>
<!-- Create the directories for intermediate and temporary files
<!-- We are going to continue on error here so we can skip tasks that fail
<MakeDir Directories="$(OutDir);$(IntermediateOutputPath)">
</Target>
<!--
=====
GetFrameworkPaths
GetIntermediateDirectories
GetFrameworkPaths
GetIntermediateDirectories
Get the paths for the .NET Framework installations
-->
```

Learn more:

[MSBuild Binary and Structured Log Viewer](#)

- Kirill Osenkov

WinDBG - Time Travel Debugging



WinDbg Preview
- [in Microsoft Store \(free\)](#)

Learn more:
[Time Travel Debugging is now available in WinDbg Preview](#)
- James Pinkerton

Error Logs and Stack Traces

How can we improve performance here?

What's wrong with Error Logs?

- Constructors

- Iterators/ LINQ

```
<Iterator>d__3.MoveNext()
```

- Lambdas

```
<>c__DisplayClass2_0.<.ctor>b__0(Object state)
```

- Local functions

```
<RefMethod>g__LocalFuncParam|10_0(String val, <>c__DisplayClass10_0&)
```

- Value tuples

```
ValueTuple`2 should be (string val, bool)
```

- Generics

```
RunLambda(Func`1 lambda)
```

Learn more:
[Ben.Demystifier](#)
- Ben Adams

var errorLog = await WrongAsync()?

- What function?

```
at Program.<MethodAsync>d__5`1.MoveNext()
```

- Noise

```
--- End of stack trace from previous location where exception was thrown ---
at System.Runtime.ExceptionServices.ExceptionDispatchInfo.Throw()
at System.Runtime.CompilerServices.TaskAwaiter.ThrowForNonSuccess(Task task)
at System.Runtime.CompilerServices.TaskAwaiter.HandleNonSuccessAndDebuggerNotification(Task task)
at System.Runtime.CompilerServices.TaskAwaiter.ValidateEnd(Task task)
at System.Runtime.CompilerServices.TaskAwaiter.GetResult()
```

Error Logs

```
System.InvalidOperationException: Collection was modified; enumeration operation may not execute.  
  at System.ThrowHelper.ThrowInvalidOperationException_InvalidOperationException_EnumFailedVersion() // ? low value  
  at System.Collections.Generic.List`1.Enumerator.MoveNextRare()  
  at Program.<Iterator>d__3.MoveNext() // which enumerator?  
  at System.Linq.Enumerable.SelectEnumerableIterator`2.MoveNext() // which enumerator?  
  at System.String.Join(String separator, IEnumerable`1 values)  
  at Program.GenericClass`1.GenericMethod[TSubType](TSubType& value)  
  at Program.<MethodAsync>d__4.MoveNext() // which async overload?  
--- End of stack trace from previous location where exception was thrown --- // ? no value  
  at System.Runtime.ExceptionServices.ExceptionDispatchInfo.Throw() // ? no value  
  at System.Runtime.CompilerServices.TaskAwaiter.HandleNonSuccessAndDebuggerNotification(Task task) // ? no value  
  at System.Runtime.CompilerServices.TaskAwaiter`1.GetResult() // ? no value  
  at Program.<MethodAsync>d__5`1.MoveNext() // which async overload?  
--- End of stack trace from previous location where exception was thrown --- // ? no value  
  at System.Runtime.ExceptionServices.ExceptionDispatchInfo.Throw() // ? no value  
  at System.Runtime.CompilerServices.TaskAwaiter.HandleNonSuccessAndDebuggerNotification(Task task) // ? no value  
  at System.Runtime.CompilerServices.TaskAwaiter`1.GetResult() // ? no value  
  at Program.<>c__DisplayClass8_0.<Method>b__0() // ^_^(*)_/_  
  at Program.RunLambda(Func`1 lambda)  
  at Program.Method(String value)  
  at Program.<RefMethod>g__LocalFuncRefReturn|10_1(<>c__DisplayClass10_0& ) // local function  
  at Program.<RefMethod>g__LocalFuncParam|10_0(String val, <>c__DisplayClass10_0& ) // local function  
  at Program.RefMethod(String value)  
  at Program.<>c.<.cctor>b__18_1(String s, Boolean b) // ^_^(*)_/_  
  at Program.<>c.<.cctor>b__18_0(String s, Boolean b) // ^_^(*)_/_  
  at Program.Start(ValueTuple`2 param) // Tuple param?  
  at Program.<Start>g__LocalFunc1|11_0(Int64 l) // local function
```

Ben.Demystifier

High performance understanding for stack traces

```
at bool System.Collections.Generic.List<T>+Enumerator.MoveNextRare()
at bool Program.Iterator(int startAt)+MoveNext() Resolved enumerators
at bool System.Linq.Enumerable+SelectEnumerableIterator<TSource, TResult>.MoveNext()
at string string.Join(string separator, IEnumerable<string> values)
at string Program+GenericClass<TSuperType>.GenericMethod<TSubType>(ref TSubType value)
at async Task<string> Program.MethodAsync(int value) Resolved async
at async Task<string> Program.MethodAsync< TValue >(&TValue value) Method declared
at string Program.Method(string value)+()=>{} lambda
at string Program.RunLambda(Func<string> lambda)
at (string val, bool) Program.Method(string value)
at ref string Program.RefMethod(string value)+LocalFuncRefReturn() ref returning
at int Program.RefMethod(string value)+LocalFuncParam(string val) local function
at string Program.RefMethod(string value)
at (string val, bool) static Program()+(string s, bool b)=>{} static lambda
at void static Program()+(string s, bool b)=>{} returning tuple
at void Program.Start((string val, bool) param) tuple params
at void Program.Start((string val, bool) param)+LocalFunc1(long l)
at bool Program.Start((string val, bool) param)+LocalFunc2(bool b1, bool b2)
at string Program.Start()
at void Program()+()=>{} ctor declared lambdas
at void Program(Action action)+(object state)=>{} Constructors
at void Program.RunAction(Action<object> lambda, object state)
at new Program(Action action)
at new Program()
at void Program.Main(String[] args)
```

Coming soon at:
GitHub [Ben.Demystifier](#)

.NET Core Futures

Its still getting better!

(Also flows into Desktop Framework, but slower...)

Futures (.NET Core 2.1+)

- Even more cross-platform (Alpine Linux)
- `Enum.HasFlag` (Intrinsic)
- `EqualityComparer<T>.Default` (Intrinsic)
- Block Layout for Search Loops

Method	Toolchain	Mean
LoopReturn	.NET Core 2.0	61.97 ns
LoopGoto	.NET Core 2.0	53.63 ns
LoopReturn	.NET Core 2.1.0-preview1-25719-04	53.75 ns
LoopGoto	.NET Core 2.1.0-preview1-25719-04	53.52 ns

- SSE/SSE2/SSE3/SSSE3/SSE41/SSE42/AVX/AVX2
AES/BMI1/BMI2/FMA/LZCNT/PCLMULQDQ
CRC/POPCNT Intrinsics

RyuJIT x64 .NET Core 2.0

```
mov rcx, [[AttributeTargets type]]
call [[box]]
mov rbx,rax
mov rcx, [[AttributeTargets type]]
call [[box]]
mov ecx,dword ptr [rsi+8]
mov dword ptr [rbx+8],ecx
mov rcx,rbx
mov dword ptr [rax+8],0Ch
mov rdx,rax
call [[System.Enum.HasFlag]]
mov byte ptr [rsi+0Ch],al
```

RyuJIT x64 .NET Core 2.1.0-preview1-25719-04

```
mov r8d,edx
and r8d,0Ch
cmp r8d,0Ch
sete r8b
mov byte ptr [rcx+0Ch],r8b
```

Enum.HasFlag

Learn more:

[Add Intel hardware intrinsic APIs to CoreFX](#)

- Fei Peng (Intel)

Learn more:

[RyuJIT Just-in-Time Compiler Optimization Enhancements](#)

- Joseph Tremoulet

Futures (.NET Core 2.1+)

- Faster Tasks and async
- Faster AsyncLocal
Still don't use it, its evil 😊
- Better Concurrency for Timers
- Faster SslStream - Though its fast now

ASP.NET Core 2.0 Kestrel - plaintext

Windows Server 2016 - Azure Standard L32s

Kestrel	Requests per second		
(Connections)	256	512	1024
http	1,797,487	1,914,983	1,893,890
https	1,114,302	1,154,201	1,131,808



benaadams commented an hour ago

With the new design, those are eliminated, such that even if a non-c



stephentoub commented an hour ago

You keep weakening my AysncLocal is evil generalization 😊

Mwahahaha

Faster Tasks and async:

coreclr [#2718](#), [#3132](#), [#3139](#), [#4918](#), [#5131](#), [#8567](#), [#8846](#), [#9342](#), [#9343](#), [#9471](#), [#12819](#), [#13105](#), [#13270](#), [#13390](#), [#14178](#), [#14541](#), [#14759](#) wow!
- Stephen Toub

Futures (.NET Core 2.1+)

Epic Performance

Performance Epic #24747

ⓘ Open

danmosemsft opened this issue 16 days ago · 2 comments



danmosemsft commented 16 days ago • edited

Member

This Epic is to track work we are doing in CoreFX (and corelib) to monitor and improve performance for the next release - excepting Networking performance, which is likely in the [Networking Epic](#).



danmosemsft added the **Epic** label 16 days ago



danmosemsft added the **tenet-performance** label 16 days ago

CORESTART 2.0

Děkuji!

Thank you!