

Точечная переработка драйвера MongoDB

Для многократного увеличения производительности



Андрей Гурин



Сидристый Станислав



Станислав Сидристый

- R&D Team Gear
- C#, C/C++, C++/CLI когда необходимо
- Книга: <https://github.com/sidristij/dotnetbook>
- telegram: @sidristij
- sunex.development@gmail.com

Андрей Гурин

- R&D Team Gear
- telegram: @andreygurinspb
- sunex.development@gmail.com

File Edit Selection View Go Debug Terminal Help

EXPLORER

OPEN EDITORS

DOTNETBOOK

readme.md 1-Exceptions-Intro.md

book > en > ExceptionalFlow > 1-Exceptions-Intro.md

in turn, means that using `try/catch` can lead to stack unrolling despite the security settings of a subdomain.

321

322 > We should pay additional attention to the `FullTrust` way. CLR fully trusts everything that happens in serialization. Admit, we do not have any protection. Each domain loader it creates copies of all types. You will get the same object type. In other words, if we cast `(Boo)boo`` won't be a problem as the object will be the original object are called.

323

324 By transferring a serialized object in another one while keeping the original is used only for those types non-serializable as an exception.

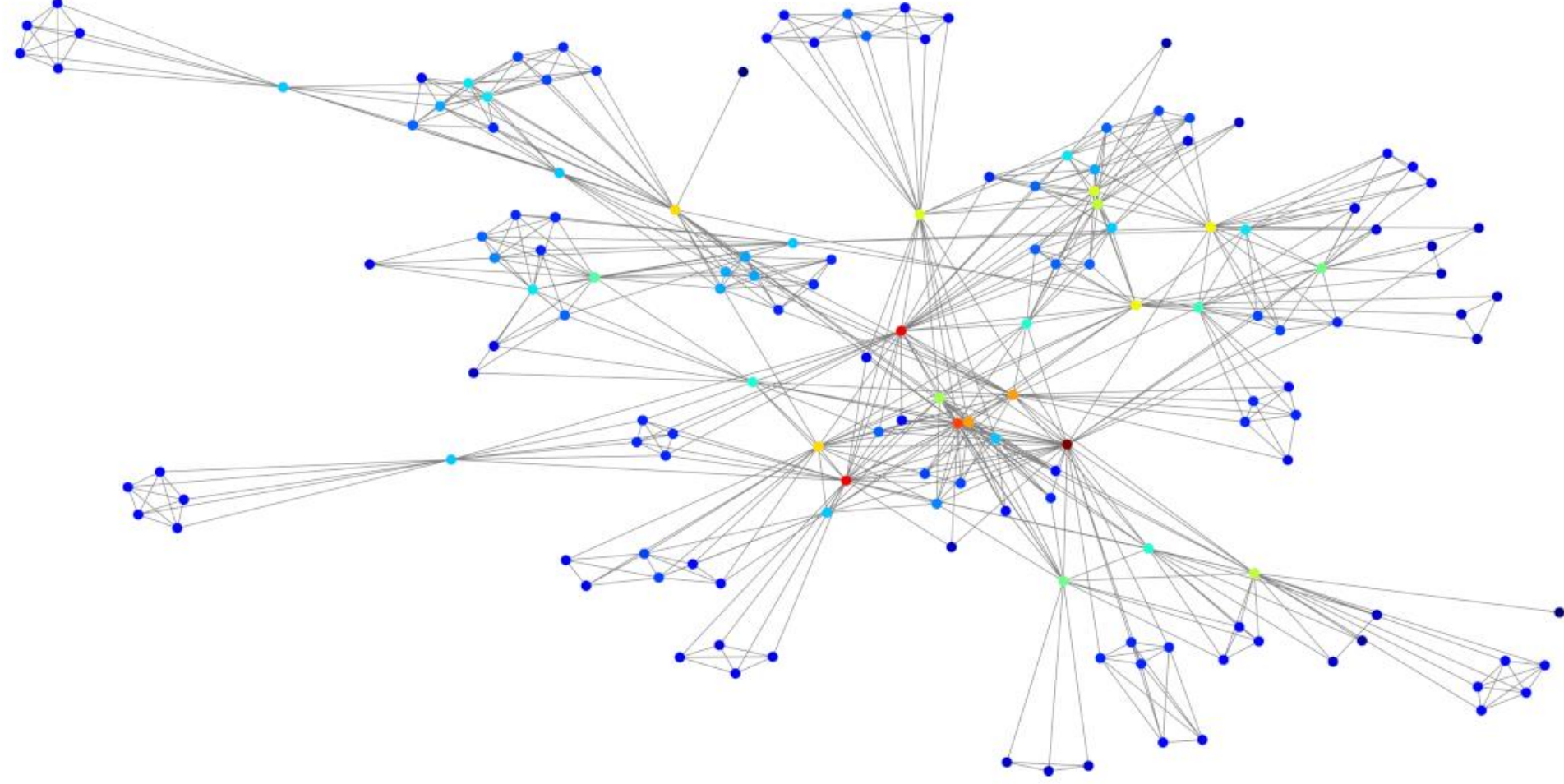
PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL

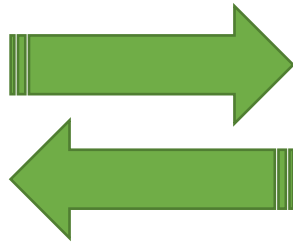
Integrity Check succeeded.
Installing package 'OmniSharp for .NET Core...'
Downloading package '.NET Core Debug Tools...'
Validating download...
Integrity Check succeeded.



второй босс

Первый акт — это презентация основных действующих лиц, зов к странствию, встреча с наставником, принятие зова





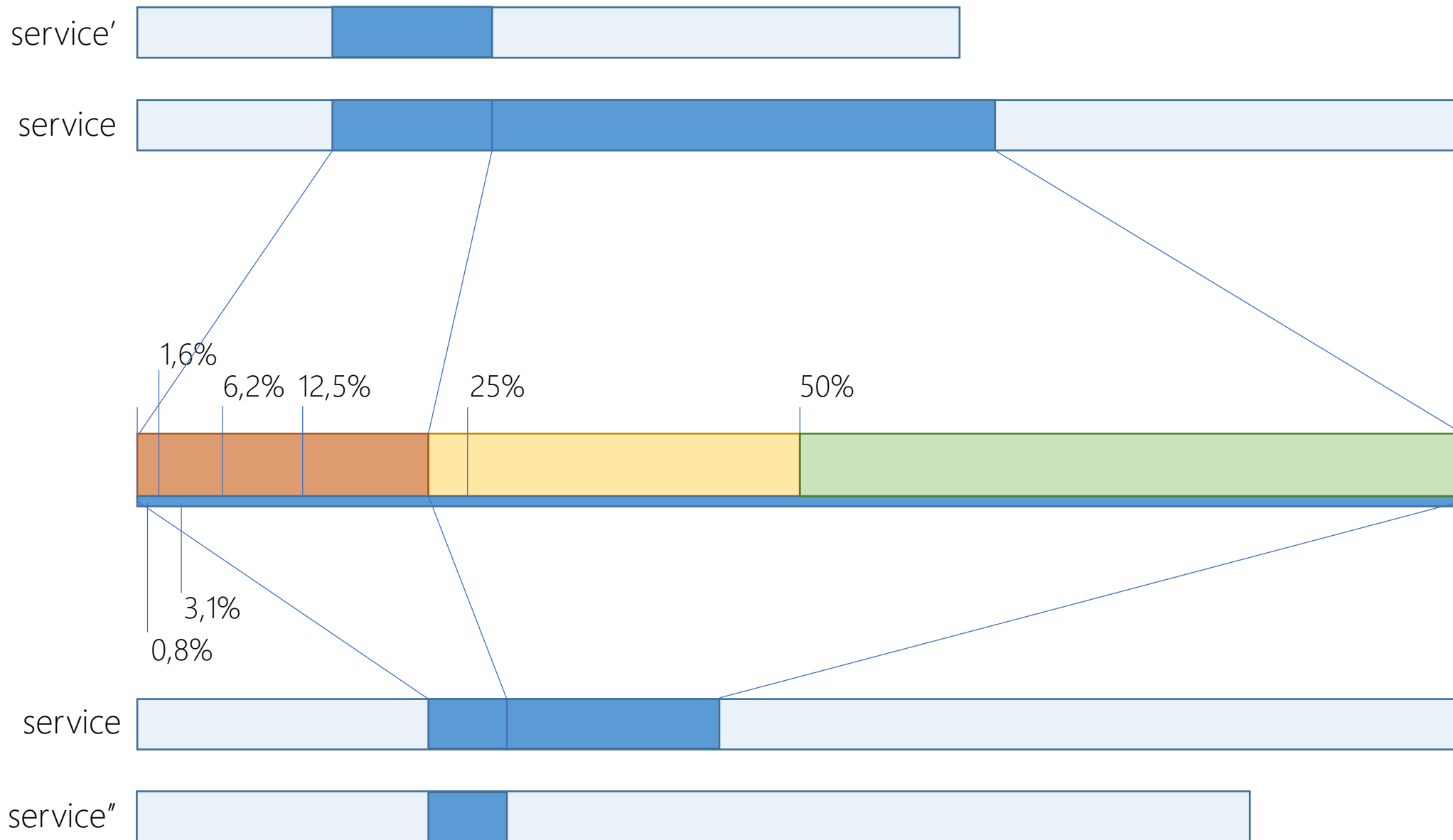
SMB_{2.0+}, SMB_{3.0}

При высокой нагрузке на приложение – очень плотная работа с серверами

Єсть ли вообще проблема?

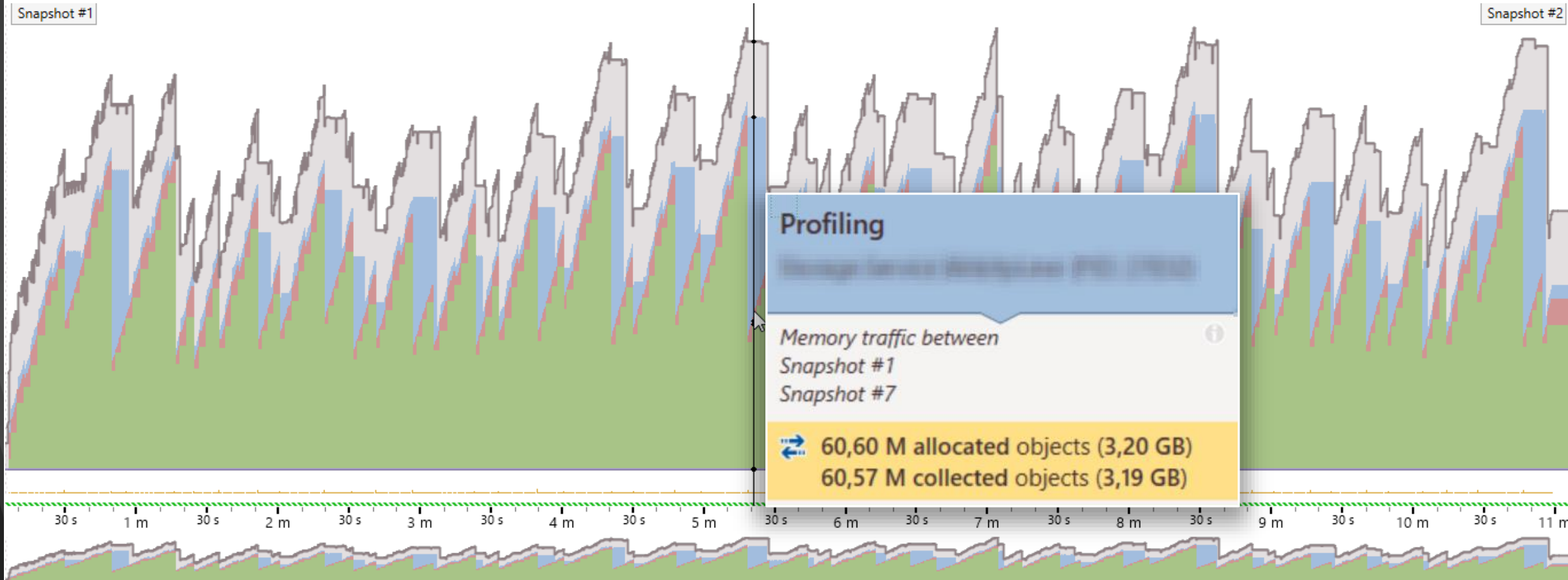


+ счётчики производительности
+ Stopwatch :)



В чём может быть проблема?

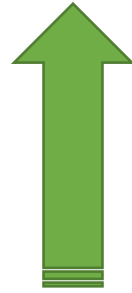
new T(), new T[N];



Plain List	Group by: Namespace	Assembly	Interface	Allocated bytes	Allocated objects	Collected bytes	Collected objects
▷ { }	System			2 308 571 057	35 873 142	2 303 583 569	35 844 492
▷ { }	MongoDB			651 774 952	13 911 474	651 760 632	13 911 165
▷ { }	Serilog			213 775 008	6 245 632	213 671 816	6 242 886
▷ { }	Microsoft			114 429 080	2 141 545	114 328 760	2 140 002
▷ { }	Unity			53 850 040	839 443	53 690 056	837 401
▷ { }	Jaeger			31 799 864	566 282	31 798 736	566 261
▷ { }	StackExchange			15 001 216	221 299	15 000 832	221 291
▷ { }	Newtonsoft			11 024 448	84 608	11 007 328	84 397
▷ { }	DataStorageService			10 893 000	224 030	10 892 952	224 028
▷ { }	RedLockNet			5 273 480	85 393	5 273 280	85 392
▷ { }	Swashbuckle			4 358 224	113 431	4 356 936	113 401
▷ { }	Prometheus			3 833 796	104 779	3 828 988	104 704
▷ { }	OpenTracing			3 113 280	80 615	3 111 952	80 571
▷ { }	DevTools			2 673 648	45 317	2 673 536	45 316
▷ { }	global::			1 210 192	27 729	1 209 936	27 725
▷ { }	Storage			1 187 624	11 463	1 187 520	11 462
▷ { }	Authentication			643 088	25 394	642 824	25 388
▷ { }	Quartz			43 936	694	43 328	686
▷ { }	MS			11 560	20	1 600	4
▷ { }	Internal			568	11	568	11
▷ { }	Windows			48	2	48	2

Function	Allocated bytes	Allocated objects	Collected bytes	Collected objects	Namespace
Byte[] (System)	174 396 671	1 194 096	171 947 067	1 192 242	
Func<IElementNameValidator> (System)	113 168 128	1 768 252	113 168 128	1 768 252	
ByteBufferStream.ctor(IByteBuffer, Boolean)	72 152 896	767 584	72 152 896	767 584	MongoDB.Bson.IO
ByteArrayChunk.CreateByteArray(Int32)	61 303 389	80 224	61 292 154	80 214	MongoDB.Bson.IO
GC.AllocateUninitializedArray<T>(Int32)	24 924 072	295	22 913 288	267	System
BsonStreamExtensions.ReadBytes(BsonStream, Int32)	3 531 135	80 251	3 531 003	80 248	MongoDB.Bson.IO
BitConverter.GetBytes(Int64)	3 254 176	101 693	3 254 176	101 693	System
ByteBufferStream.WriteInt64(Int64)	3 254 176	101 693	3 254 176	101 693	MongoDB.Bson.IO
Encoding.GetBytes(String)	2 543 813	35 535	2 519 117	35 286	System.Text
FileStream.GetBuffer()	1 343 120	326	1 343 120	326	System.IO
Utils.UniqueId()	1 128 128	35 254	1 128 128	35 254	Jaeger.Util
BinaryConnection+ <ReceiveBufferAsync> d_54.MoveNext()	1 123 472	40 124	1 123 472	40 124	MongoDB.Driver.Core.Core
HashProviderCng.FinalizeHashAndReset()	840 280	15 005	840 280	15 005	Internal.Cryptography
HashAlgorithm.CaptureHashCodeAndReinitialize()	838 880	14 980	838 880	14 980	System.Security.Cryptogr
SocketAddress.ctor(AddressFamily, Int32)	546 000	8 535	546 000	8 535	System.Net.Internals
RuntimeAssembly.GetPublicKey()	272 208	11 342	272 208	11 342	System.Reflection
HttpConnection.ctor(HttpConnectionPool, Socket, Stream, TransportContext)	115 360	28	107 120	26	System.Net.Http
DateHeaderValueManager.SetDateValues(DateTimeOffset)	55 998	918	55 693	913	Microsoft.AspNetCore.Ser
SignatureHelper.Init(Module)	13 552	242	8 064	144	System.Reflection.Emit
ILGenerator.IncreaseCapacity(Int32)	12 216	29	5 224	15	System.Reflection.Emit
BufferExtensions.CreateNumericBytesScratch()	4 356	99	3 960	90	System.Buffers

Function	Allocated bytes	Allocated objects	Collected bytes	Collected objects	Namespace
Func<IElementNameValidator> (System)	113 168 128	1 768 252	113 168 128	1 768 252	
Func<Type, IBsonSerializer> (System)	85 476 800	1 335 575	85 476 800	1 335 575	
Char[] (System)	83 032 812	835 260	82 925 614	835 220	
Int32[] (System)	50 254 500	1 281 606	50 222 102	1 281 526	
‣ BsonWriter.WriteName(String)	51 882 496	810 664	51 882 496	810 664	MongoDB.Bson.IO
‣ BsonWriter.PushElementNameValidator(IElementNameValidator)	30 642 816	478 794	30 642 816	478 794	MongoDB.Bson.IO
‣ BsonWriter.PopElementNameValidator()	30 642 816	478 794	30 642 816	478 794	MongoDB.Bson.IO



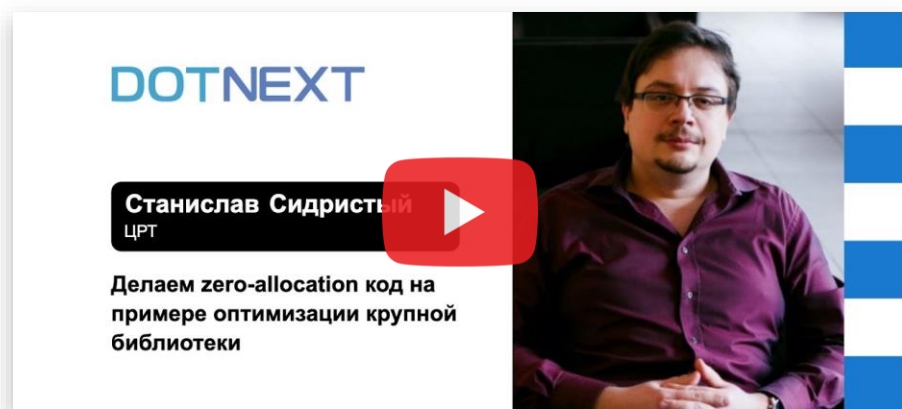
MongoDB.Driver by: MongoDB

↓ 42,205,643 total downloads ⌚ last updated 19 days ago 📄 Latest version: 2.13.0-beta1

Official .NET driver for MongoDB.

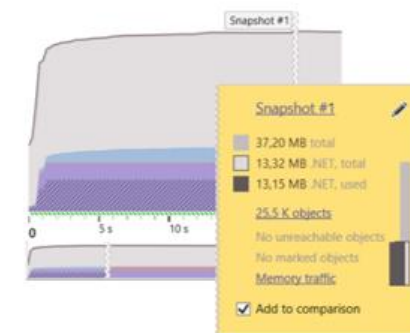
SMB_{2.0+}, SMB_{3.0}

Скачать 21,000
файлов со сложной структуре каталогов



255,222 сек.
25,15 Gb

4,000+ GC₀



11 сек.
0,021 Mb

0 GC₀

Так что же делать?


Строим план решения!








mongodb / mongo-csharp-driver Watch

[Code](#) [Pull requests 32](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)

master 24 branches 97 tags [Go to file](#) [Add file](#) [Code](#)

 **BorisDog** [CSHARP-3305: Rate limit new connection creations \(maxConnecting\)](#) 59a1268 4 days ago 3,793 commits

 Docs	Update releases.toml. (#490)	14 days ago
 Release Notes	Added release notes for 2.12.2. (#491)	14 days ago
 evergreen	CSHARP-3371: Support AWS authentication with temporary credenti...	2 months ago
 src	CSHARP-3305: Rate limit new connection creations (maxConnecting)	4 days ago
 tests	CSHARP-3305: Rate limit new connection creations (maxConnecting)	4 days ago

Взять другое?..

mongodb / mongo-c-driver

Watch 89 Star 648 Fork 377

Code Pull requests 6 Actions Security Insights

master 27 branches 159 tags Go to file Add file Code

jmikola CDRIVER-3958 add trailing space after base64 JSON field (#779) c64caa4 4 hours ago 6,528 commits

.evergreen	CDRIVER-3917 Run versioned API tests in Evergreen (#773)	8 days ago
build	CDRIVER-3917 Run versioned API tests in Evergreen (#773)	8 days ago
debian	(Debian packaging) mark ready for release	yesterday
generate_uninstall	CDRIVER-2849 name uninstall program based on components built	2 years ago
orchestration_configs	CDRIVER-3917 Run versioned API tests in Evergreen (#773)	8 days ago
src	CDRIVER-3958 add trailing space after base64 JSON field (#779)	4 hours ago
.clang-format	CDRIVER-1167 Add clang-format file, and remove uncrustify	4 years ago

About

A high-performance MongoDB driver for C

mongoc.org

Readme

Apache-2.0 License

Releases 159

mongo-c-driver 1.18.0-alpha **Latest**
12 days ago

+ 158 releases



Замена на mongo-c-driver

Второй акт — это самая большая, основная часть истории. Здесь вызов, брошенный антагонистом, заставляет героя действовать. К середине второго акта он уже не может повернуть назад, как бы ему этого ни хотелось. Во второй половине второго акта многократно возрастают ставки и риски. И к концу второго акта герой терпит большое поражение, оказываясь в максимальной опасности, практически в безвыходной ситуации.

Что будем делать?

- Оценка используемого API
- Оценка реализации используемого API
- Proof Of Concept реализации с использованием mongo-c-driver

```
13  /// managed instances.
14  /// </summary>
15  [23 usages] [Stanislav Sidristij +1*] [5 ext methods] [11 exposing APIs]
16  public sealed class CountdownMemoryOwner<T> : IMemoryOwner<T>
17  {
18      private int _owners;
19      private T[] _arr;
20      private CountdownMemoryOwner<T> _parent;
21      private JetStack<T[]> _queue;
22
23      [8 usages]
24      public Memory<T> Memory
25      {
26          [MethodImpl(MethodImplOptions.AggressiveInlining)]
27          get; private set;
28      }
29
30      [MethodImpl(MethodImplOptions.AggressiveInlining)]
31      [3 usages] [Stanislav Sidristij +1*]
32      internal CountdownMemoryOwner<T> Init(CountdownMemoryOwner<T> parent, int offset, int length, bool defaultOwner)
33      {
34          _owners = defaultOwner?1:0;
35          _parent = parent;
36          _parent.AddOwner();
37      }
38  }
```

mongo-c-driver

Profiling
DevExperiments.MongoDriversCompariso...

Memory traffic between
Snapshot #3
Snapshot #4

209.90 M allocated objects (7.96 GB)
209.89 M collected objects (7.95 GB)

Profiling
DevExperiments.MongoDriversCompariso...

Memory traffic between
Snapshot #1
Snapshot #2

75.82 M allocated objects (2.92 GB)
75.82 M collected objects (2.92 GB)

Profiling
DevExperiments.MongoDriversCompariso...

Memory traffic between
Snapshot #2
Snapshot #3

51.3 K allocated objects (5.91 MB)
426 collected objects (483.5 KB)

Group by: **Types** Back Traces

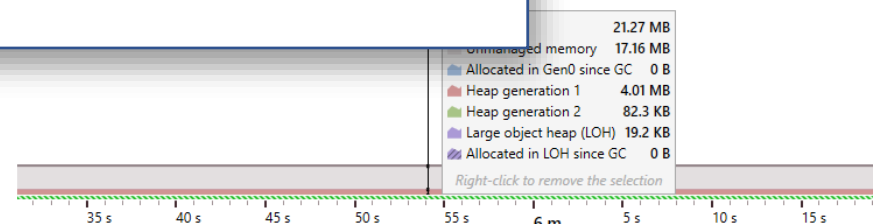
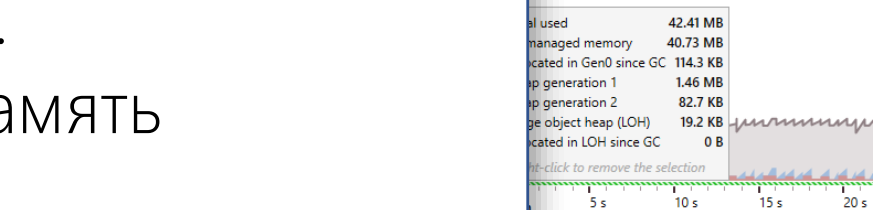
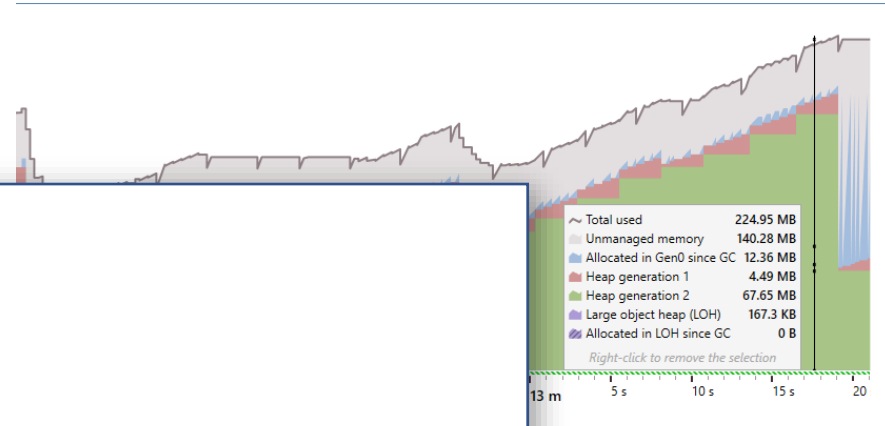
Here you can view what objects were created/collected during the selected timeframe as well as back traces of functions that created these objects. ⓘ

Filter: Filter by namespaces, types, arrays, and generic arguments. Type lg to exclude generic args from matching or !a to exclude arrays.

Plain List Group by: Namespace Assembly Interface

	Allocated bytes	Allocated objects	Collected bytes
...	1,575,168,000	40,324,000	1,575,168,000

Type	Allocated bytes	Allocated objects	Collected bytes
ThirdLevelDocument (DevExperiments.MongoDriversComparison.Models.Docs)	350,000	6,250	
MemoryOwnerArrayLike<Char> (DevExperiments.MongoDriversComparison.MongoC.Factories)	302,120	7,553	
CountdownMemoryOwner<Char>[] (DevTools.MemoryPools.Memory)	262,448	14	131,328
BucketsBasedCrossThreadsMemoryPool+BufferHolder<Char> (DevTools.MemoryPools.Memory.Pooling)	241,696	7,553	
Char[][] (System)	131,352	13	65,768
BucketsBasedCrossThreadsMemoryPool+BufferHolder<Char>[] (DevTools.MemoryPools.Memory.Pooling)	131,352	13	65,768
MemoryOwnerArrayLike<Char>[] (DevExperiments.MongoDriversComparison.MongoC.Factories)	130,216	7	64,656



Результаты:
Ускорение: x3 в макс.
Снижение давления на память

Дополнительно:

1. Запрет на использование интерфейсов (прямой вызов вместо вызова через VMT)
2. Сначала забери Span, потом – используй его. Нельзя в цикле: `memoryOwner.Memory.Span[i];`
3. Все ви
4. Если м
рамка

ountdown в

Результаты:

Ускорение: **x5** в макс.

A grayscale image of a landscape shrouded in thick fog. In the center, a single, dark tree stands out against the misty background. The ground is dark and textured, possibly covered in grass or low-lying vegetation. The overall atmosphere is somber and mysterious.

Но есть проблема
поддерживать API сложно

Так что же делать?

Новый план:

- Делаем fork MongoDB.Driver
- Перерабатываем всю сериализацию: избавляемся от траффика массивов, делаем максимально-быструю реализацию тех частей, что не будут переработаны
- Пишем ручные сериализаторы, какими они должны быть после source generators
- Получаем результат, пишем тесты на BenchmarkDotNet
- Максимально ускоряем
- Делаем по образу и подобию Source Generators
- Profit!

```
13  /// managed instances.
14  /// </summary>
15  [23 usages] [Stanislav Sidristij +1*] [5 ext methods] [11 exposing APIs]
16  public sealed class CountdownMemoryOwner<T> : IMemoryOwner<T>
17  {
18      private int _owners;
19      private T[] _arr;
20      private CountdownMemoryOwner<T> _parent;
21      private JetStack<T[]> _queue;
22
23      [8 usages]
24      public Memory<T> Memory
25      {
26          [MethodImpl(MethodImplOptions.AggressiveInlining)]
27          get; private set;
28      }
29
30      [MethodImpl(MethodImplOptions.AggressiveInlining)]
31      [3 usages] [Stanislav Sidristij +1*]
32      internal CountdownMemoryOwner<T> Init(CountdownMemoryOwner<T> parent, int offset, int length, bool defaultOwner)
33      {
34          _owners = defaultOwner?1:0;
35          _parent = parent;
36          _parent.AddOwner();
37      }
38  }
```

Build succeeded at 19:05:28 (today 19:05) .NETStandard 2.0 MemoryPools 51:10 CRLF UTF-8 Tab release 113 warnings in 58 files

mongo-csharp-driver (наш)

Profiling
DevExperiments.MongoDriversCompariso...

Memory traffic between
Snapshot #3
Snapshot #4

209.90 M allocated objects (7.96 GB)
209.89 M collected objects (7.95 GB)

Memory traffic between
Snapshot #1
Snapshot #2

52.53 M allocated objects (2.19 GB)
52.53 M collected objects (2.19 GB)

Profiling
DevExperiments.MongoDriversCompariso...

Memory traffic between
Snapshot #2
Snapshot #3

51.3 K allocated objects (5.91 MB)
426 collected objects (483.5 KB)

Group by: **Types** Back Traces

Here you can view what objects were created/collected during the selected timeframe as well as back traces of functions that created these objects. ⓘ

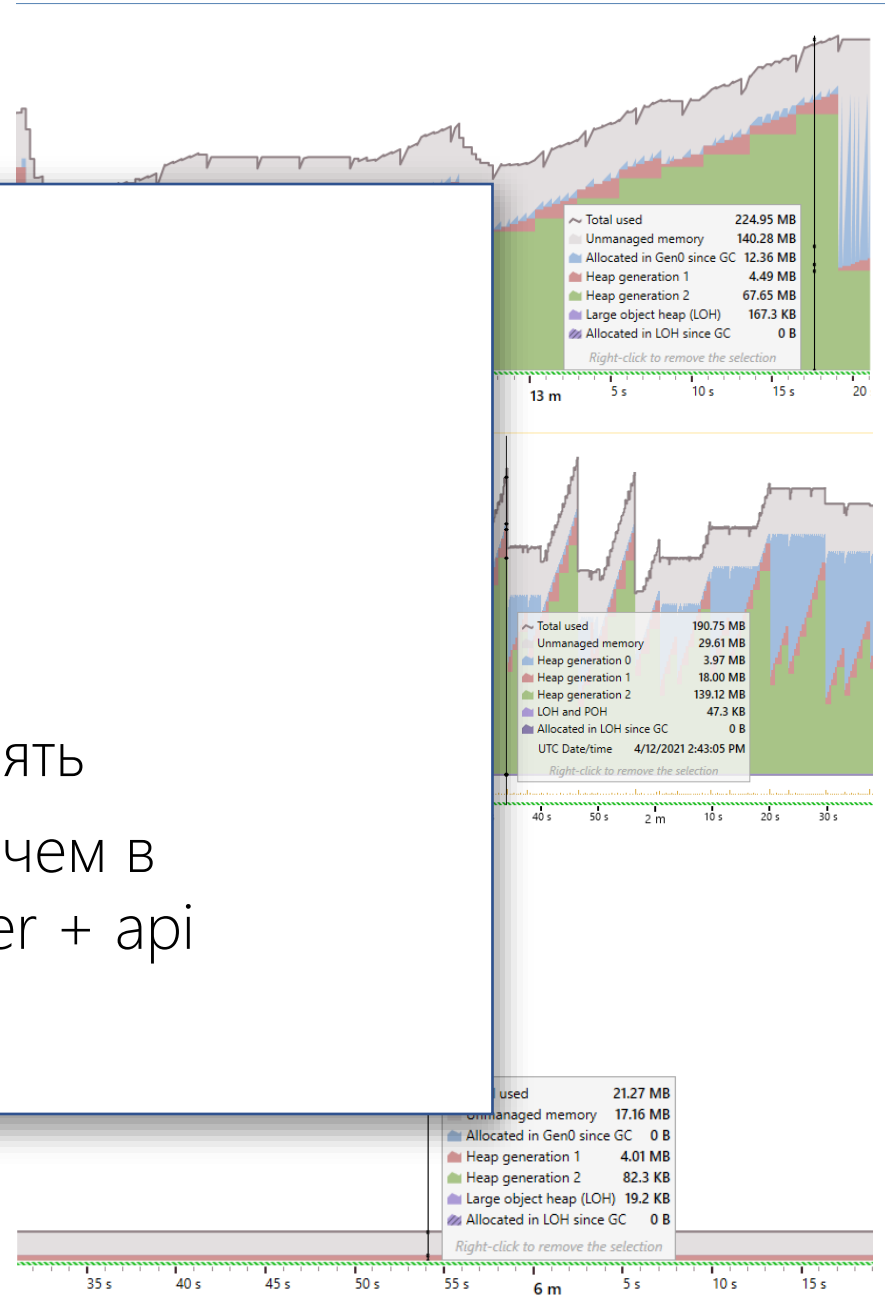
Filter: Filter by namespaces, types, arrays, and generic arguments. Type lg to exclude generic args from matching or la to exclude arrays.

Plain List Group by: Namespace Assembly Interface

	Allocated bytes	Allocated objects	Collected bytes
ThirdLevelDocument (DevExperiments.MongoDriversComparison.Models.Docs)	1,575,168,000	40,324,000	1,575,168,000

ThirdLevelDocument (DevExperiments.MongoDriversComparison.Models.Docs)	350,000	6,250	
MemoryOwnerArrayLike<Char> (DevExperiments.MongoDriversComparison.MongoC.Factories)	302,120	7,553	
CountdownMemoryOwner<Char[]> (DevTools.MemoryPools.Memory)	262,448	14	131,328
BucketsBasedCrossThreadsMemoryPool+BufferHolder<Char> (DevTools.MemoryPools.Memory.Pooling)	241,696	7,553	
Char[] (System)	131,352	13	65,768
BucketsBasedCrossThreadsMemoryPool+BufferHolder<Char[]> (DevTools.MemoryPools.Memory.Pooling)	131,352	13	65,768
MemoryOwnerArrayLike<Char[]> (DevExperiments.MongoDriversComparison.MongoC.Factories)	130,216	7	64,656

Результаты:
Ускорение: x5 в макс.
Снижение давления на память
Использование памяти ниже, чем в
связке mongo-c-driver + wrapper + api

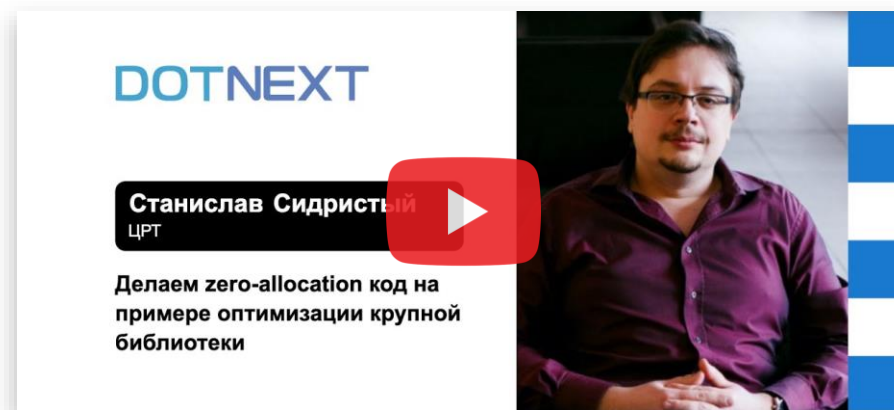




Выводы

Выводы

- Для начала надо понять, стоит ли овчинка выделки
 - Какой % планируемого к оптимизации кода будет работать в конечном сервисе
 - Как он повлияет на систему?
 - Какой процент времени освободит? Может там 1 запрос в минуту
- Убедиться, что проблема не в железе
- Выбрать методы оптимизации: что недозагружено? Что перегружено?
- Желательно идти через proof of concept



A decorative graphic consisting of several overlapping keys in a dark red color, arranged in a cluster behind the text.

QA