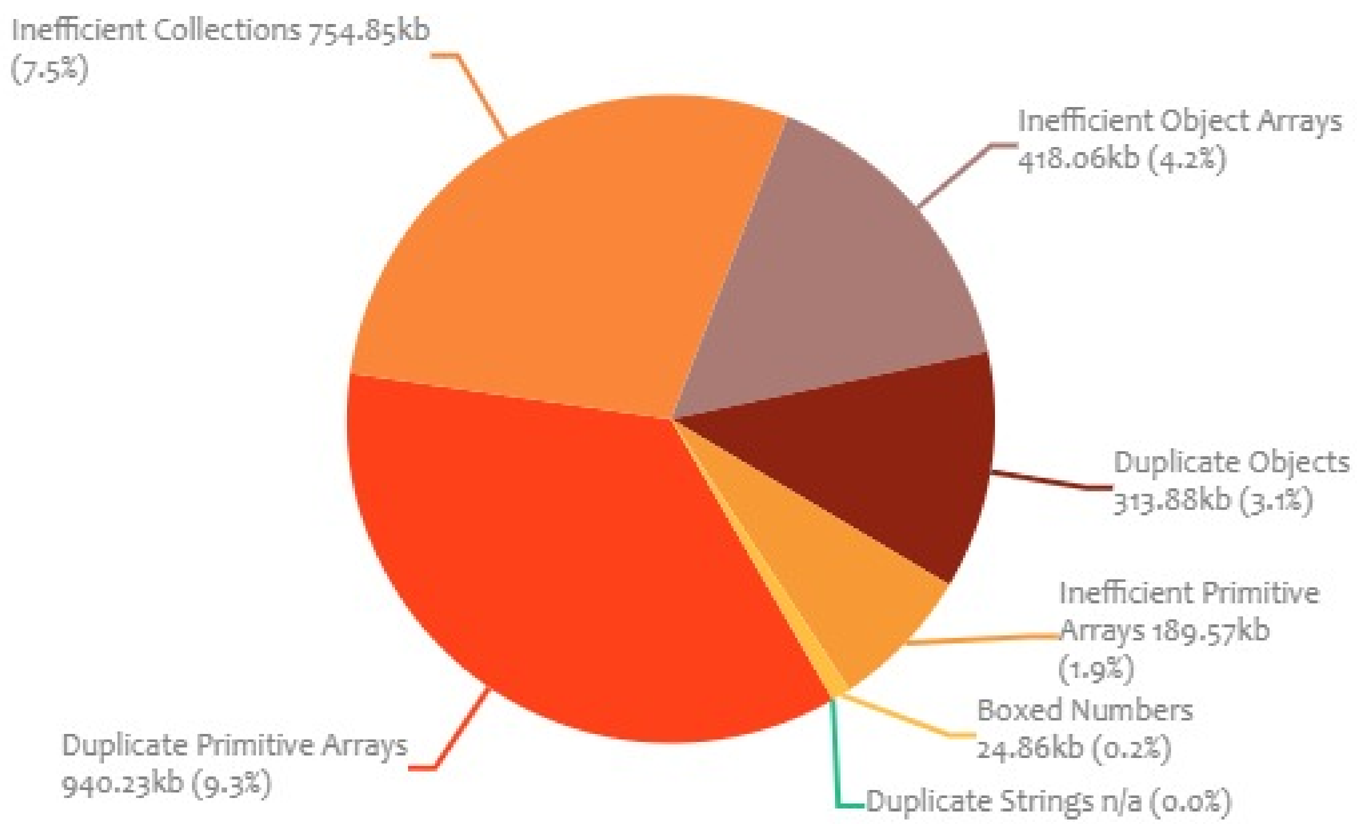
1. Heap Statistics

Learn more about Heap Statistics

Memory wasted: 2.58mb (26%)



Total Size

9.83mb

Object Count

253,967

Class Count

7,840

Thread Count

24

2. What's in your Memory (by class)?

Learn more about What's in Memory

Class	Percentage	Size	Count
String	32.3%	3.18mb	41,814

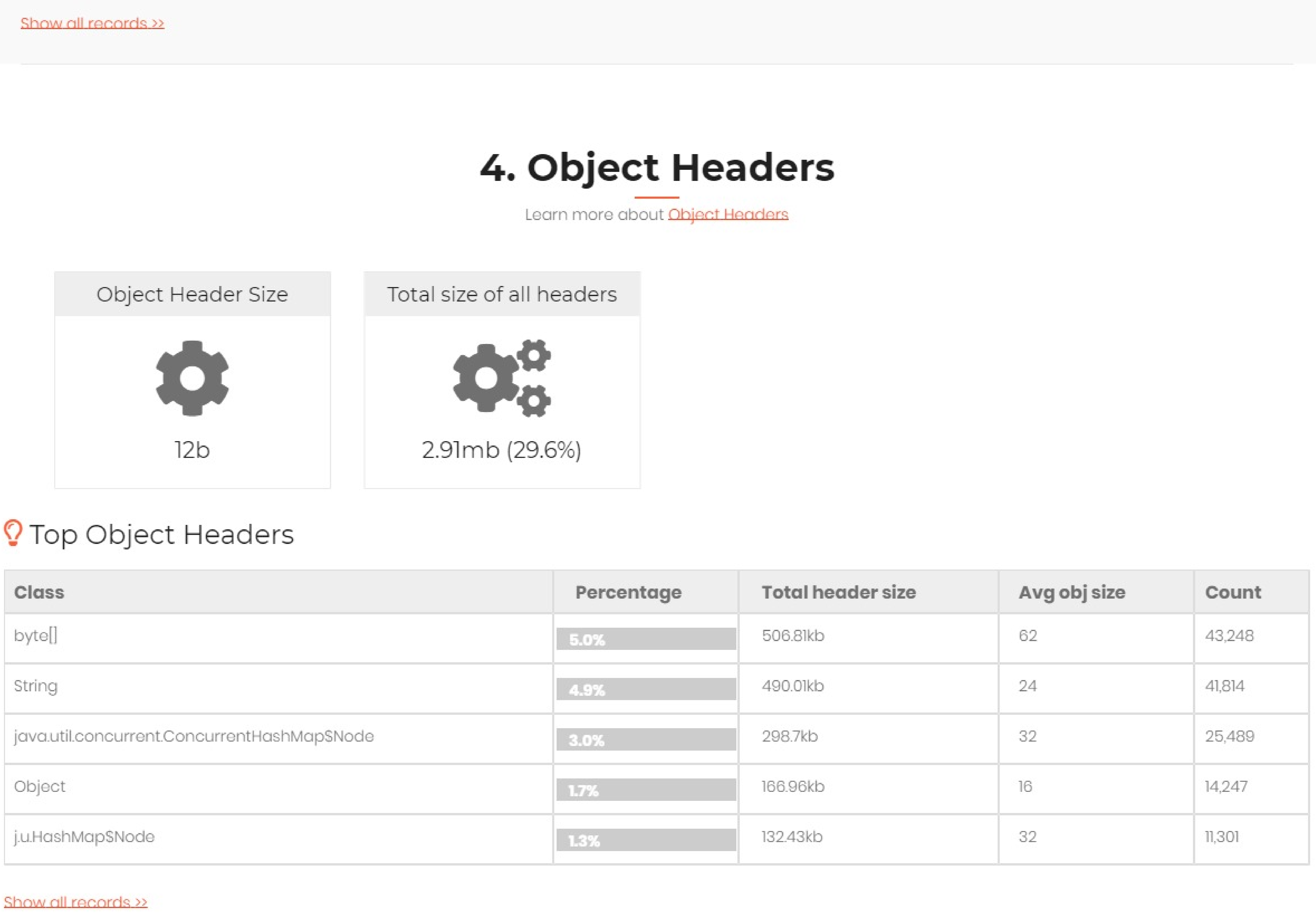
java.util.concurrent.ConcurrentHashMap	10.6%	1.04mb	521
Object[]	3.9%	391.9kb	9,100
byte[]	3.7%	372.74kb	43,248
j.u.HashMap	3.4%	340.24kb	1,227
j.u.LinkedHashMap	3.3%	334.16kb	2,676

Show all records >>

3. Large objects

Learn more about Large Objects

Name	Percentage	Size
Java Static java.lang.ApplicationShutdownHooks.hooks	11.6%	1.14mb
Unreachable (garbage) objects	11.5%	1.13mb
Java Static org.apache.catalina.core.StandardHostValve.MY_CLASSLOADER	5.9%	595.42kb
Java Static jdk.internal.loader.ClassLoaders.PLATFORM_LOADER	5.5%	557.66kb
Java Static org.apache.tomcat.util.modeler.Registry.registry	5.0%	502.88kb
... and 10724 more objects retaining 3.92mb (39.9%)		



Show all records >>

4. Object Headers

Learn more about Object Headers

Object Header Size	Total size of all headers
12b	2.91mb (29.6%)

Top Object Headers

Class	Percentage	Total header size	Avg obj size	Count
byte[]	5.0%	506.81kb	62	43,248
String	4.9%	490.01kb	24	41,814
java.util.concurrent.ConcurrentHashMap$Node	3.0%	298.7kb	32	25,489
Object	1.7%	166.96kb	16	14,247
j.u.HashMap$Node	1.3%	132.43kb	32	11,301

Show all records >>

How to fix excessive Object headers?

Please refer to our recommendations.

5. Duplicate Strings

Learn more about Duplicate Strings

Not Detected

6. Inefficient collections

Learn more about Inefficient Collections

Total Collections	Inefficient collections	Wasted Memory
9,316	7,392	754.85kb (7.5%)

Top inefficient collections

Problem	Percentage	Wasted

20% of j.u.LinkedHashMap contains 1 element only	0.9%	86.05kb
59% of j.u.ArrayList contains 1 element only	0.8%	84.24kb
13% of j.u.LinkedHashMap contains 2 - 4 elements only	0.7%	70.02kb
35% of j.u.LinkedHashSet contains 1 element only	0.6%	61.32kb
9% of java.util.concurrent.ConcurrentHashMap contains half empty elements	0.5%	47.21kb

Show all records >>

Who is holding Inefficient Collections?

Object Tree	Percentage	size
org.springframework.boot.autoconfigure.condition.ConditionEvaluationReport$ConditionAndOutcomes.outcomes	0.4%	36K
{j.u.HashMap}.values	0.3%	25K
j.l.Class$AnnotationData.annotations	0.3%	25K
sun.reflect.generics.tree.ClassTypeSignature.path	0.2%	22K
sun.reflect.generics.tree.ClassTypeSignature.path	0.2%	19K

Show all records >>

How to fix Inefficient Collections?

Please refer to our recommendations.

7. Inefficient Object Arrays

Learn more about Inefficient Object Arrays

Total Object Arrays	Inefficient Object Arrays	Wasted Memory
25,704	13,641	418.06kb (4.2%)

Top inefficient Object Arrays

Problem	Percentage	Wasted
15% of Object[] contains no elements	0.6%	62.04kb
18% of Object[] declared with 1 length	0.4%	38.88kb
35% of j.l.Class[] declared with 1 length	0.4%	36.77kb
2% of Object[] contains half empty elements	0.3%	34.48kb
7% of Object[] contains 1 element only	0.3%	29.51kb

Show all records >>

Who is holding Inefficient Object Arrays?

Object Tree	Percentage	size

Object Tree	Percentage	Size
Unreachable (garbage) objects	0.2%	20K
org.springframework.util.ConcurrentReferenceHashMap$Segment.references	0.2%	18K
java.lang.invoke.MethodTypeForm.lambdaForms	0.2%	15K
java.lang.invoke.LambdaForm$Name.arguments	0.1%	12K

How to fix Inefficient Object Arrays?

Please refer to our recommendations.

8. Inefficient Primitive Arrays

Learn more about Inefficient Primitive Arrays

Total Primitive Arrays

46,766

Inefficient Primi Arrays

2,707

Wasted Memory

189.57kb (1.9%)

Top inefficient Primitive Arrays

Problem	Percentage	Wasted

< 0.1% of byte[] contains no elements	.4%	40.55kb
1% of int[] contains no elements	.3%	34.5kb
69% of int[] declared with 0 length	.3%	32kb
12% of char[] contains no elements	.2%	23.79kb
< 0.1% of byte[] contains lot of 0s	.2%	23.43kb

Show all records >>

Who is holding Inefficient Primitive Arrays?

Object Tree	Percentage	size
java.io.BufferedWriter.cb	0.3%	31K
java.nio.HeapByteBuffer.hb	0.2%	15K
byte[]	0.1%	14K

How to fix Inefficient Primitive Arrays?

Please refer to our recommendations.

9. Boxed Numbers

Learn more about Boxed Numbers

Total Boxed Objects

1,414

Wasted Memory

24.86kb (0.2%)

Who is holding Boxed Numbers?

Object Tree	Percentage	size
j.l.Byte[]	<0.1%	4K
j.l.Long[]	<0.1%	4K
j.l.Short[]	<0.1%	4K

How to fix Boxed Numbers?

Please refer to our recommendations.

10. Duplicate Objects

Learn more about Duplicate Objects

Total Duplicate Objects

16,583

Wasted Memory

313.88kb (3.1%)

Types of Duplicate Objects

Object	Percentage	Wasted	Duplicate Count
Object	2%	222.59kb	14,246
j.l.r.SoftReference	9%	91.29kb	2,337

Top Duplicate Objects

Duplicate Object	Percentage	Wasted	Count
Objec)	.2%	222.59kb	14,247
j.l.r.SoftReference(referent : null, queue : j.l.r.ReferenceQueue$Null@fec1e5a8, next : null, discovered : null, timestamp : 93110730)	.5%	53.16kb	1,362
j.l.r.SoftReference(referent : null, queue : j.l.r.ReferenceQueue$Null@fec1e5a8, next : null, discovered : null, timestamp : 93110707)	0.1%	9.84kb	253
j.l.r.SoftReference(referent : null, queue : j.l.r.ReferenceQueue$Null@fec1e5a8, next : null, discovered : null, timestamp : 93110730)	0.1%	3.36kb	87
j.l.r.SoftReference(referent : null, queue : j.l.r.ReferenceQueue$Null@fec1e5a8, next : null, discovered : null, timestamp : 93110730)	0.1%	1.6kb	42

Show all records >>

Who is holding Duplicate Objects?

Who is holding Duplicate Objects?

Object Tree	Percentage	size
{java.util.concurrent.ConcurrentHashMap}.values	0.7%	70K
{java.util.concurrent.ConcurrentHashMap}.values	0.7%	70K
{java.util.concurrent.ConcurrentHashMap}.values	0.6%	64K
sun.util.locale.BaseLocale$Key.vart	0.3%	28K
sun.util.locale.BaseLocale$Key.scrt	0.3%	27K

Show all records >>

How to fix Duplicate Objects?

Please refer to our recommendations.

11. Duplicate Primitive Arrays

Learn more about Duplicate Primitive Arrays

Total Duplicate Arrays	Wasted Memory
16,441	940.27kb (9.7%)

16,441

940.23kb (9.3%)

Types of Duplicate Arrays

Array Type	Percentage	Wasted	Duplicate Count
byte[]	8.0%	804.9kb	13,920
int[]	0.7%	68.22kb	2,236
char[]	0.6%	55.54kb	192
boolean[]	<0.1%	9.05kb	45
long[]	<0.1%	1.98kb	34

Show all records >>

Top Duplicate Arrays

Duplicate Array	Percentage	Wasted	Count
byte[8192](0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...)		32.06kb	5
int[0]()		31.98kb	2,048
int[1025](0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...)		20.12kb	6
byte[16]('j', 'a', 'v', 'a', '.', 'l', 'a', 'n', 'g', '.', 'O', 'b', 'j', 'e', 'c', 't')		9.94kb	319
int[512](0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...)		6.05kb	4

Show all records >>

Who is holding Duplicate Arrays?

Object Tree	Percentage	size
byte[][][]	0.5%	48K
{java.util.concurrent.ConcurrentHashMap}.values	0.5%	48K
char[][][]	0.2%	24K
char[][][]	0.2%	24K

How to fix Duplicate Arrays?

Please refer to our recommendations.

12. Objects waiting for Finalization

Learn more about Objects waiting for Finalization

Wasted Memory

48b (<0.1%)

What are the objects waiting for finalization?

To see objects waiting for finalization, click here

How to fix objects waiting for finalization?

Please refer to our recommendations.

13. Threads

Learn more about Threads

To view all threads stacktrace click here

14. Heap settings

Learn more about Heap Settings

No major recommendations.

15. System Properties

Learn more about System Properties

Not Report in the Heap dump.