

Отчёт по нагрузочному тестированию сервиса “book”

Оглавление

Цели и результаты тестирования.....	3
Цели тестирования.....	3
Результаты тестирования	3
Проведенные тесты.....	3
Максимальная производительность	3
Подтверждение максимальной производительности.....	5
Тест надёжности.....	6
Промежуточные выводы.....	7
Дополнительные тесты	7
Локализация утечки памяти.....	8
Тест без утечки памяти	9
Поиск блокировок в сервисе.....	9

Цели и результаты тестирования

Цели тестирования

- 1) Определить максимальную производительность.
- 2) Определить надёжность на 100% профиля.

№	Операция	Интенсивность	%
1	Информация о книге	30 в секунду.	75%
2	Изменить цену	6 в секунду	15%
3	Список всех книг	2 в секунду	5%
4	Удалить книгу	1 в секунду	2,5%
5	Добавить книгу	1 в секунду	2,5%

- 3) Определить, где в сервисе возникают блокировки.
- 4) Локализовать утечку памяти, определить, что в ней хранится.

Результаты тестирования

- 1) Максимальная производительность составляет, около 1040 запросов в секунду.

№	Операция	Интенсивность	%
1	Информация о книге	780 в секунду.	75%
2	Изменить цену	156 в секунду	15%
3	Список всех книг	52 в секунду	5%
4	Удалить книгу	26 в секунду	2,5%
5	Добавить книгу	26 в секунду	2,5%

- 2) Постоянный рост занимаемой памяти, при длительном тесте.
- 3) В сервисе присутствует утечка памяти(`oldBooksPrices`
`java.util.concurrent.ConcurrentHashMap`) таблица с данными книг.
- 4) Блокировки возникают на 352 строке кода.

-

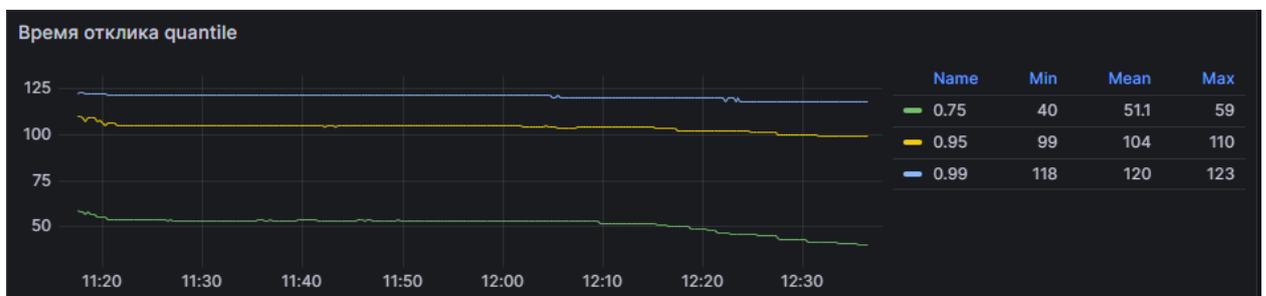
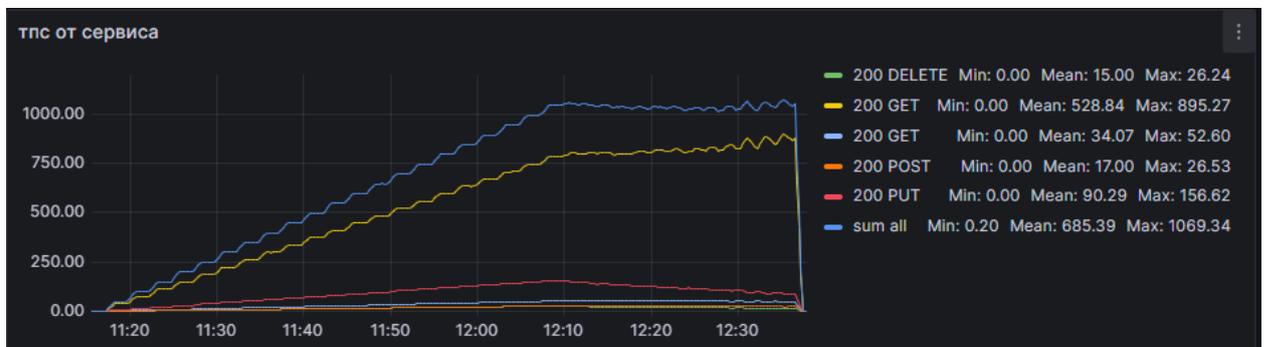
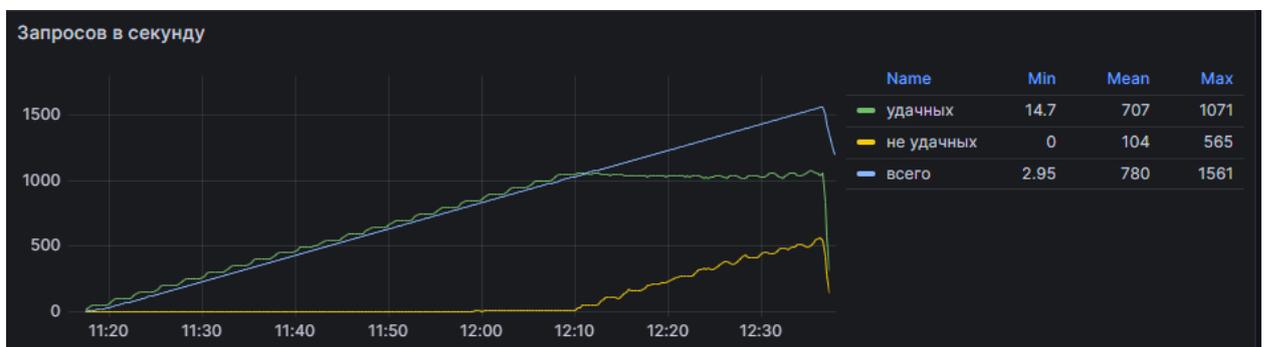
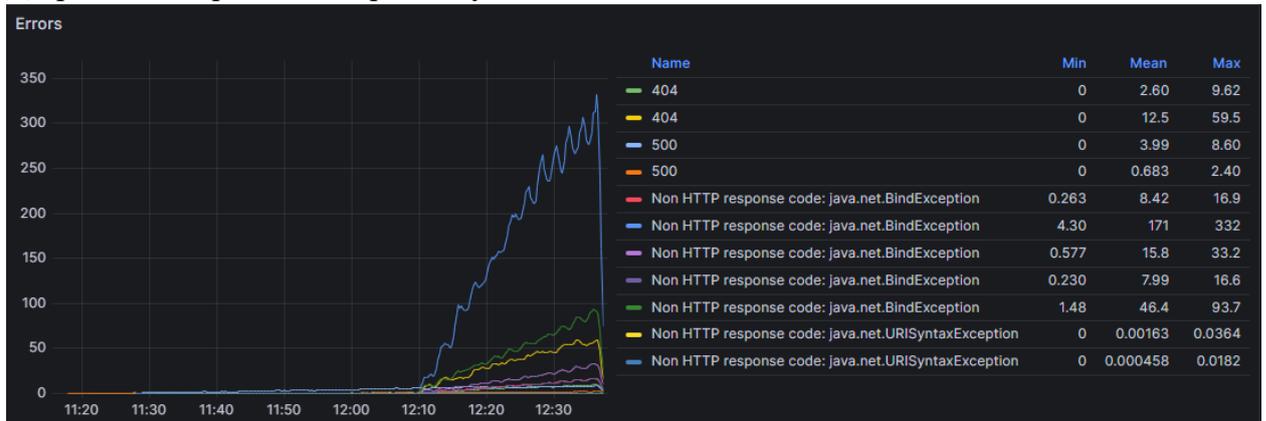
Проведенные тесты

Максимальная производительность

При проведении серии тестов с автоматическим управлением количества потоков и интенсивностью операций эквивалентной профилю нагрузки умноженной на коэффициент

N, установлено начало деградации системы при коэффициенте N=26(1040 запросов в секунду)

Деградация выражалась в резком увеличении количества ошибок.



Подтверждение максимальной производительности

Плавное повышение нагрузки до максимальной(N=26) и фиксация на 1 час.



При выполнении теста с нагрузкой N=26 профиля(1040 запросов в секунду), деградации сервиса не наблюдалось.

Тест надёжности

Тест проводился при нагрузке 40 запросов в секунду в соответствии с профилем длительное время.



Промежуточные выводы

Тестируемый сервис при тестовой нагрузке с автоматическим управлением количеством потоков показал стабильную работу до интенсивности нагрузки в N26(1040 запросов в секунду) включительно

(N = 100% профиля нагрузки)

Тест надёжности показал равномерный рост(без падений) объёма памяти используемого для хранения долгоживущих объектов и массивов

Дополнительные тесты

Для выявления утечки памяти был повторён тест надёжности с выделенной сервису памятью 20Мб



Иногда, при срабатывании сборки мусора в старом поколении количество памяти, не уменьшается. Что возможно свидетельствует об утечке памяти.

После заполнения памяти, сервис перестал отвечать на запросы.



Локализация утечки памяти

С помощью Eclipse Memory Analyzer и снимка “heap_dump” был найден объект создающий утечку.

➤ ❌ Problem Suspect 1

One instance of «**ru.neoflex.skills.performance.service.BookService**» loaded by «**jdk.internal.loader.ClassLoaders\$AppClassLoader @ 0xfed852a8**» occupies **5 586 032 (31,92 %)** bytes. The memory is accumulated in one instance of «**java.util.concurrent.ConcurrentHashMap\$Node[]**», loaded by «**<system class loader>**», which occupies **5 549 040 (31,71 %)** bytes.

Thread «**org.springframework.boot.web.embedded.netty.NettyWebServer\$1 @ 0xff5311e8 server**» has a local variable or reference to «**org.springframework.boot.web.embedded.netty.NettyWebServer @ 0xff4aaa48**» which is on the shortest path to «**java.util.concurrent.ConcurrentHashMap\$Node[65536] @ 0xffc6abb8**». The thread **org.springframework.boot.web.embedded.netty.NettyWebServer\$1 @ 0xff5311e8 server** keeps local variables with total size **256 (0,00 %)** bytes.

The stacktrace of this Thread is available. [See stacktrace.](#) [See stacktrace with involved local variables.](#)

Keywords

ru.neoflex.skills.performance.service.BookService
jdk.internal.loader.ClassLoaders\$AppClassLoader
java.util.concurrent.ConcurrentHashMap\$Node[]

[Details »](#)

Class Name	Shallow Heap	Retained Heap	Percentage
ru.neoflex.skills.performance.service.BookService @ 0xf007300	32	5 586 032	31,92 %
oldBooksPrices java.util.concurrent.ConcurrentHashMap @ 0xff007b98	64	5 549 104	31,71 %
table java.util.concurrent.ConcurrentHashMap\$Node[65536] @ 0xffc6abb8	262 160	5 549 040	31,71 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff8e4e38	32	752	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff827950	32	672	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff83c9e8	32	672	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff893ca0	32	672	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff8d3368	32	672	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xffc0f1f8	32	672	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff891668	32	640	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xffaf3378	32	640	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xffc0198	32	640	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff84c8c8	32	624	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff8e7940	32	616	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff845318	32	616	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xffaf0398	32	600	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xffc4808	32	600	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xffe4800	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xffe5f40	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff73ab30	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff788358	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff788358	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff799f40	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff7905f8	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff7c4c38	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff7cc0c0	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff7db0b0	32	560	0,00 %
java.util.concurrent.ConcurrentHashMap\$Node @ 0xff818958	32	560	0,00 %
Total: 25 of 29 433 entries: 33 408 more			
books java.util.Collections\$SynchronizedSet @ 0xf007320	24	36 840	0,21 %
rnd java.util.Random @ 0xf007b48	32	56	0,00 %
Total: 3 entries			
jdk.internal.loader.ClassLoaders\$AppClassLoader @ 0xfed852a8 JINI Global	96	1 517 536	8,67 %
org.springframework.beans.factory.support.DefaultListableBeanFactory @ 0xf293b78	232	1 214 744	6,94 %
jdk.internal.loader.ClassLoaders\$PlatformClassLoader @ 0xfed90a80 JINI Global	96	331 496	1,89 %
class java.lang.invoke.MethodType @ 0xfed6c00 System Class	40	232 216	1,36 %
org.springframework.boot.autoconfigure.condition.ConditionEvaluationReport @ 0xf2	32	230 584	1,32 %
class java.time.zone.ZoneRulesProvider @ 0xf216798 System Class	16	204 088	1,17 %
java.util.zip.ZipFileSource @ 0xfec48b0	80	198 272	1,13 %
java.util.zip.ZipFileSource @ 0xfec4010	80	175 360	1,00 %
class sun.util.locale.BaseLocaleCache @ 0xf4f79488 System Class	8	158 888	0,91 %
java.util.zip.ZipFileSource @ 0xfec3e30	80	150 312	0,86 %
class sun.util.calendar.ZoneInfoFile @ 0xfec7c10 System Class	120	149 976	0,86 %
java.util.zip.ZipFileSource @ 0xfec4e30	80	138 072	0,79 %
java.util.zip.ZipFileSource @ 0xfec17638	80	134 456	0,77 %
java.util.zip.ZipFileSource @ 0xfec4e968	80	123 976	0,71 %
java.util.zip.ZipFileSource @ 0xfec3c3e8	80	123 528	0,71 %

ru.neoflex.skills.performance.service.BookService @ 0xf007300	32	5 586 032	31,92 %
oldBooksPrices java.util.concurrent.ConcurrentHashMap @ 0xff007b98	64	5 549 104	31,71 %
table java.util.concurrent.ConcurrentHashMap\$Node[65536] @ 0xffc6abb8	262 160	5 549 040	31,71 %

Поток был заблокирован на 352 строке кода

Он ждёт освобождения объекта <0x00000000c16de0d0>

Занятого в строке 294

The screenshot shows an IDE window with a Java stack trace and search results. The stack trace is as follows:

```
96 at org.apache.logging.log4j.core.util.CloseShieldOutputStream.write(CloseShieldOutputStream.java:53)
97 at org.apache.logging.log4j.core.appender.OutputStreamManager.writeToDestination(OutputStreamManager.java:25)
98 - eliminated <0x00000000c16de0d0> (a org.apache.logging.log4j.core.appender.OutputStreamManager)
99 at org.apache.logging.log4j.core.appender.OutputStreamManager.flushBuffer(OutputStreamManager.java:283)
100 - eliminated <0x00000000c16de0d0> (a org.apache.logging.log4j.core.appender.OutputStreamManager)
101 at org.apache.logging.log4j.core.appender.OutputStreamManager.flush(OutputStreamManager.java:294)
102 - locked <0x00000000c16de0d0> (a org.apache.logging.log4j.core.appender.OutputStreamManager)
103 at org.apache.logging.log4j.core.appender.AbstractOutputStreamAppender.directEncodeEvent(AbstractOutputStreamAppender.java:179)
104 at org.apache.logging.log4j.core.appender.AbstractOutputStreamAppender.tryAppend(AbstractOutputStreamAppender.java:174)
105 at org.apache.logging.log4j.core.appender.AbstractOutputStreamAppender.append(AbstractOutputStreamAppender.java:164)
106 at org.apache.logging.log4j.core.config.AppenderControl.tryCallAppender(AppenderControl.java:156)
```

Below the stack trace, the search results are displayed:

Результаты поиска - (совпадений: 5)
Поиск "<0x00000000c16de0d0>" (найдено 5 совпадений в 1 файлах из 1) [Обычный]
новый 1 (совпадений: 5)
Строка 98: - eliminated <0x00000000c16de0d0> (a org.apache.logging.log4j.core.appender.OutputStreamManager)
Строка 100: - eliminated <0x00000000c16de0d0> (a org.apache.logging.log4j.core.appender.OutputStreamManager)
Строка 102: - locked <0x00000000c16de0d0> (a org.apache.logging.log4j.core.appender.OutputStreamManager)
Строка 248: - waiting to lock <0x00000000c16de0d0> (a org.apache.logging.log4j.core.appender.OutputStreamManager)
Строка 405: - waiting to lock <0x00000000c16de0d0> (a org.apache.logging.log4j.core.appender.OutputStreamManager)