VMStructs: зачем приложению знать о внутренностях JVM

2018

Андрей Паньгин
Одноклассники

Кто здесь?

@AndreiPangin

Ведущий разработчик

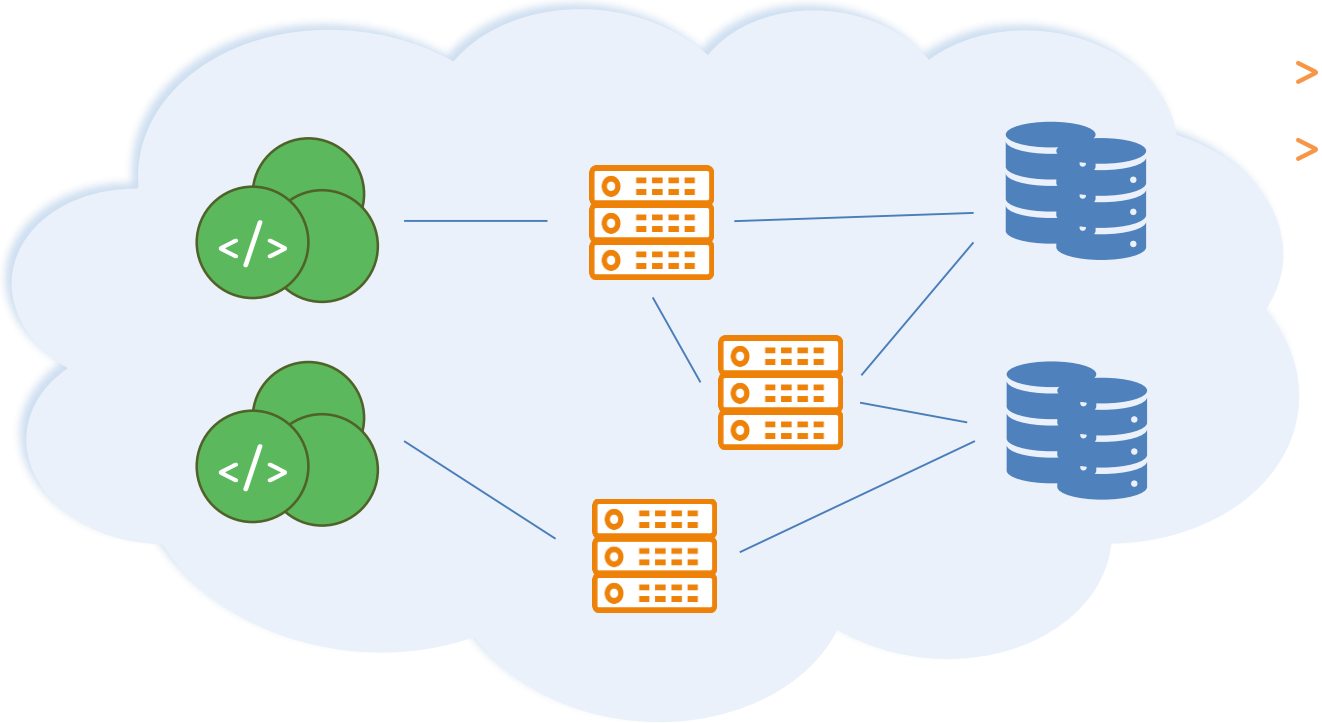
Специалист по HotSpot JVM

Автор one-nio и async-profiler

- Виртуальные машины – сложно

- У меня обычное Java приложение

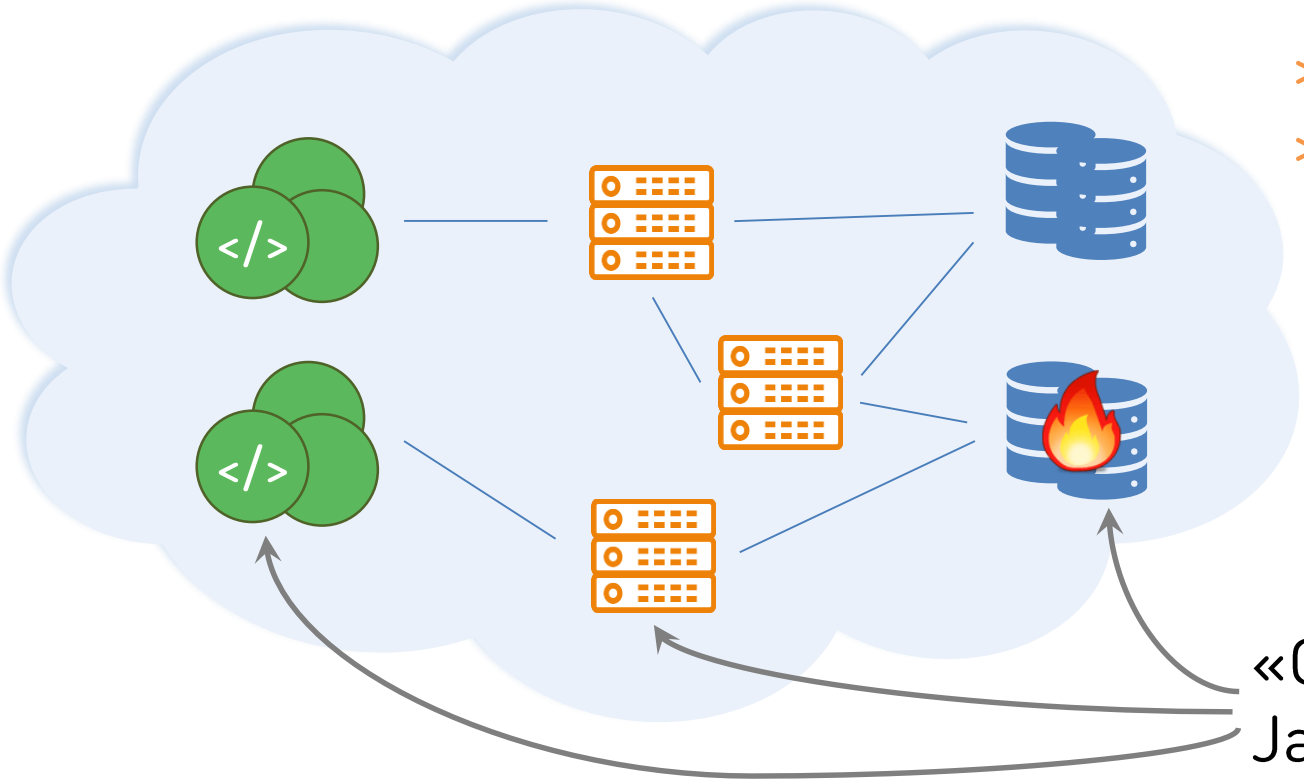
В Одноклассниках



> 200 приложений
> 8000 серверов

4 ЦОД
one-cloud

В Одноклассниках



> 200 приложений
> 8000 серверов

4 ЦОД
one-cloud

«Обычные»
Java приложения

Что-то сломалось

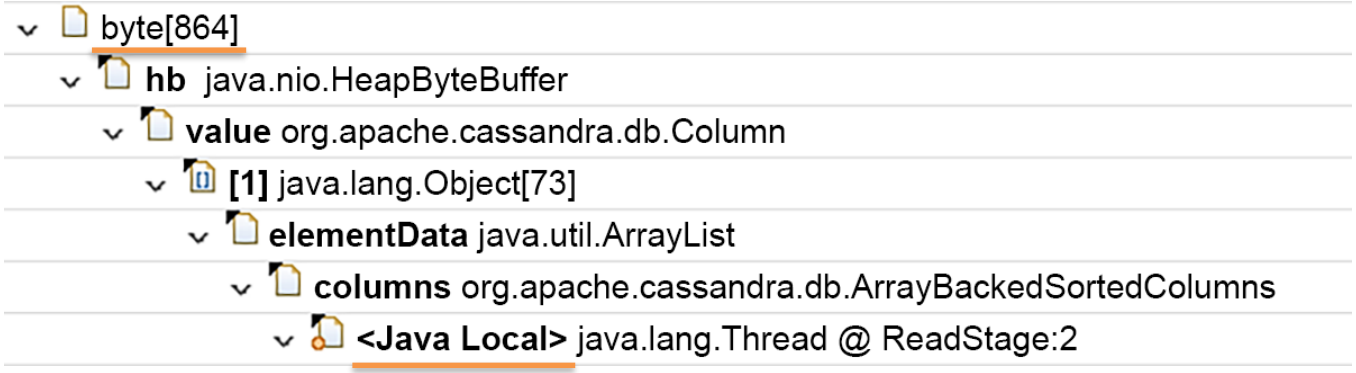
Cassandra для фотосервиса*

```
[CMS (concurrent mode failure)
[Full GC (Allocation Failure) : 23275437K->4191725K(24117248K), 27.4027 secs]
```

* Олег Анастасьев: NewSQL на Cassandra
  https://youtu.be/qyTj09e-EM0

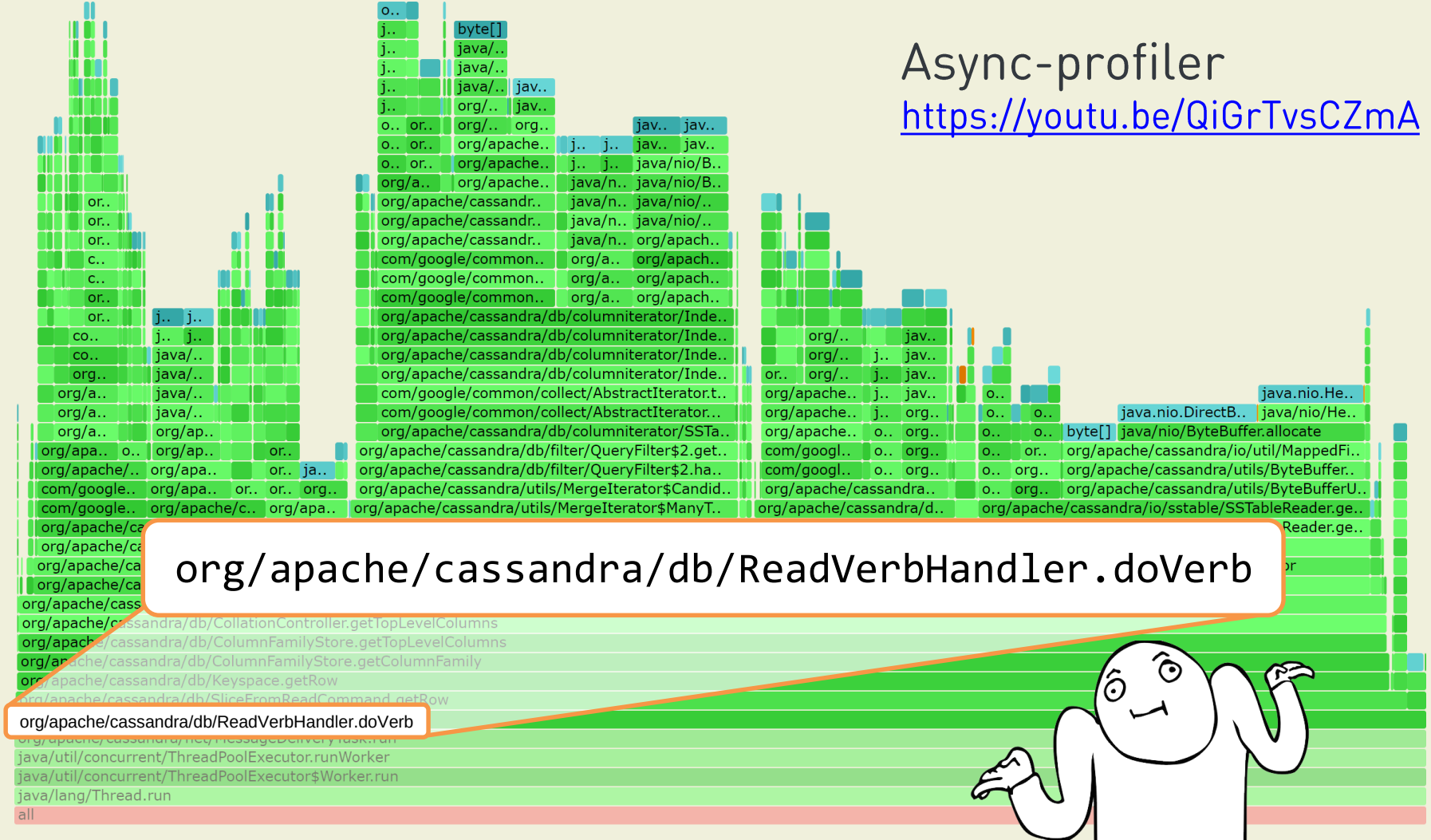
Heap Dump

- Долго, дорого, бесполезно
  - 24G heap
  - 10K запросов/сек
  - не утечка!



Async-profiler

https://youtu.be/QiGrTvsCZmA



Аллокации

- Зависят от запроса и его параметров
- На любом уровне
  - Frontend → Business logic → DB
- Какой запрос тянет много данных?

Инструментируем запросы

```
public void doVerb(ReadCommand command) {
    long before = ...

    Row row = command.getRow(keyspace);

    sendResponse(new ReadResponse(row));

    long after = ...
    if (after - before > MAX) 
}
```

JMX в помощь

```
interface com.sun.management.ThreadMXBean {

    public long getThreadAllocatedBytes(long id);
```

Per thread allocations

Deltas

Threads: 5 126

Thread Name	Allocated Bytes		**Allocated Bytes/sec**
CompactionExecutor:1881422	148 847 951 792	(0.0%)	131 241 075
ReadStage:3	25 321 405 411 496	(0.9%)	15 431 135
ReadStage:79	25 300 953 948 384	(0.9%)	15 025 117
MemoryMeter:1	34 375 120 552 608	(1.2%)	12 535 985

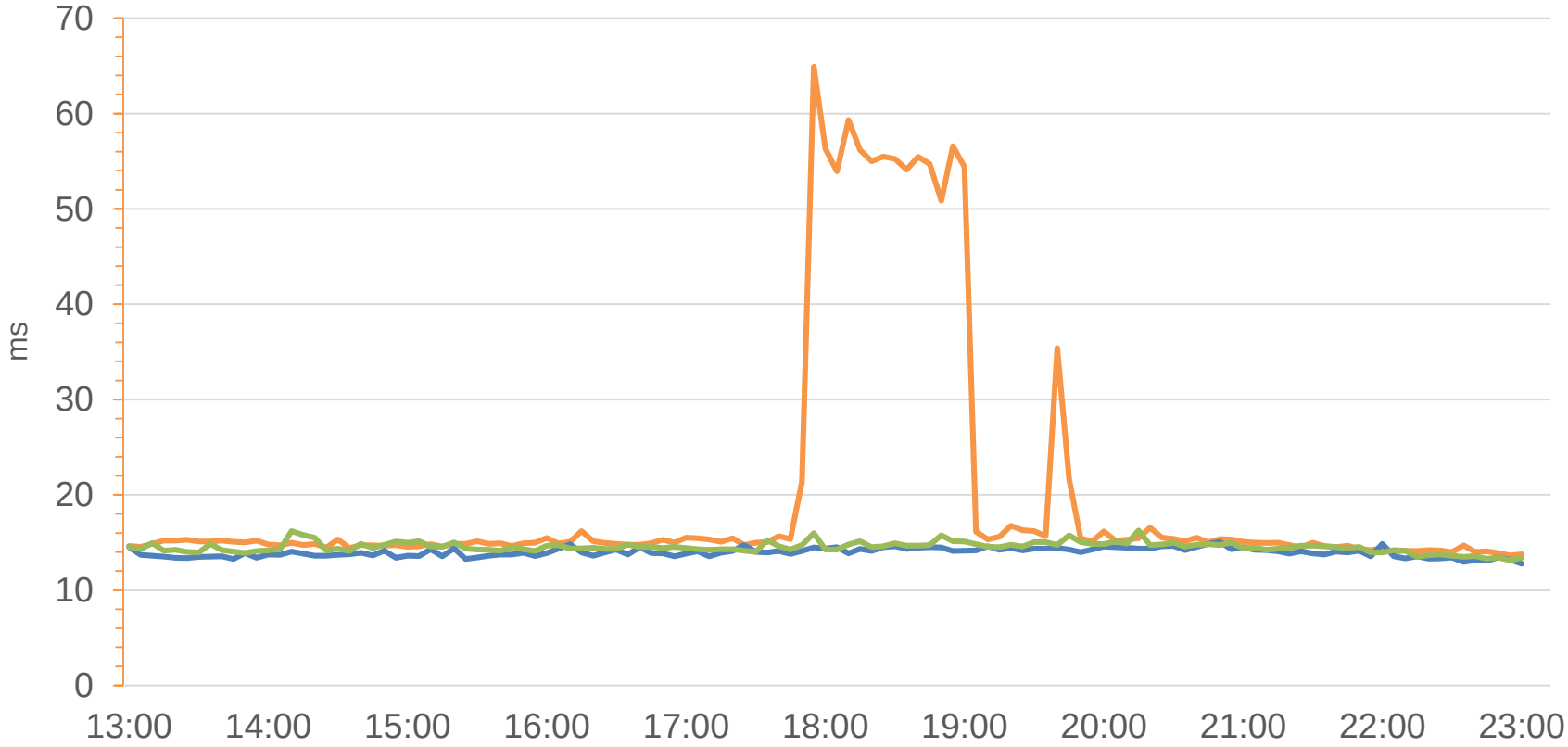
ThreadMXBean

```
static final ThreadMXBean threadMXBean =
        (ThreadMXBean) ManagementFactory.getThreadMXBean();
```

ThreadMXBean

```
static final ThreadMXBean threadMXBean =
        (ThreadMXBean) ManagementFactory.getThreadMXBean();

public void doVerb(ReadCommand command) {
    long tid = Thread.currentThread().getId();
    long before = threadMXBean.getThreadAllocatedBytes(tid);

    Row row = command.getRow(keyspace);

    sendResponse(new ReadResponse(row));

    long after = threadMXBean.getThreadAllocatedBytes(tid);
```

Среднее время запроса



Измерим с JMH

```
@Benchmark
@Threads(N)
public long allocatedBytes() {
    return threadMXBean.getThreadAllocatedBytes(tid);
}
```

threads	ns / op
1	77 ±1%
4	959 ±2%
40	27 441 ±6%
400	80 533 720 ±50%

Исходники OpenJDK

- http://hg.openjdk.java.net/jdk9

  jdk9

  - jdk
  - hotspot

Внутри JDK

```
protected long getThreadAllocatedBytes(long id)
```

```
private static native void getThreadAllocatedMemory1(long[] ids,
                                                      long[] result);
```

java.management/share/classes/sun/management/ThreadImpl.java

Внутри HotSpot

```
void jmm_GetThreadAllocatedMemory(...) {

   MutexLockerEx ml(Threads_lock);

   for (int i = 0; i < num_threads; i++) {
      JavaThread* java_thread =
          Threads::find_java_thread_from_java_tid(ids_ah->long_at(i));

      if (java_thread != NULL) {
         sizeArray_h->long_at_put(i, java_thread->cooked_allocated_bytes());
      }
   }
```

src/share/vm/services/management.cpp

Внутри HotSpot

```
void jmm_GetThreadAllocatedMemory(...) {

   MutexLockerEx ml(Threads_lock);

   for (int i = 0; i < num_threads; i++) {
      JavaThread* java_thread =
          Threads::find_java_thread_from_java_tid(ids_ah->long_at(i));

      if (java_thread != NULL) {
         sizeArray_h->long_at_put(i, java_thread->cooked_allocated_bytes());
      }
   }
```

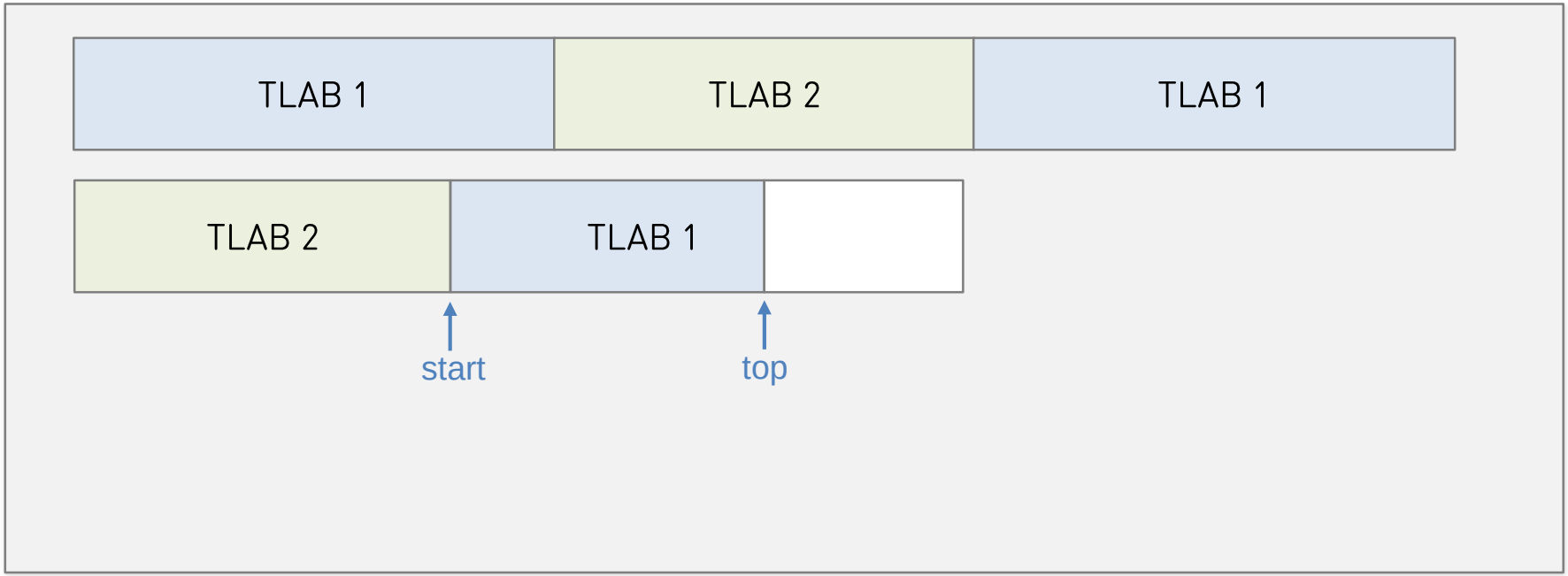
src/share/vm/services/management.cpp

Внутри HotSpot

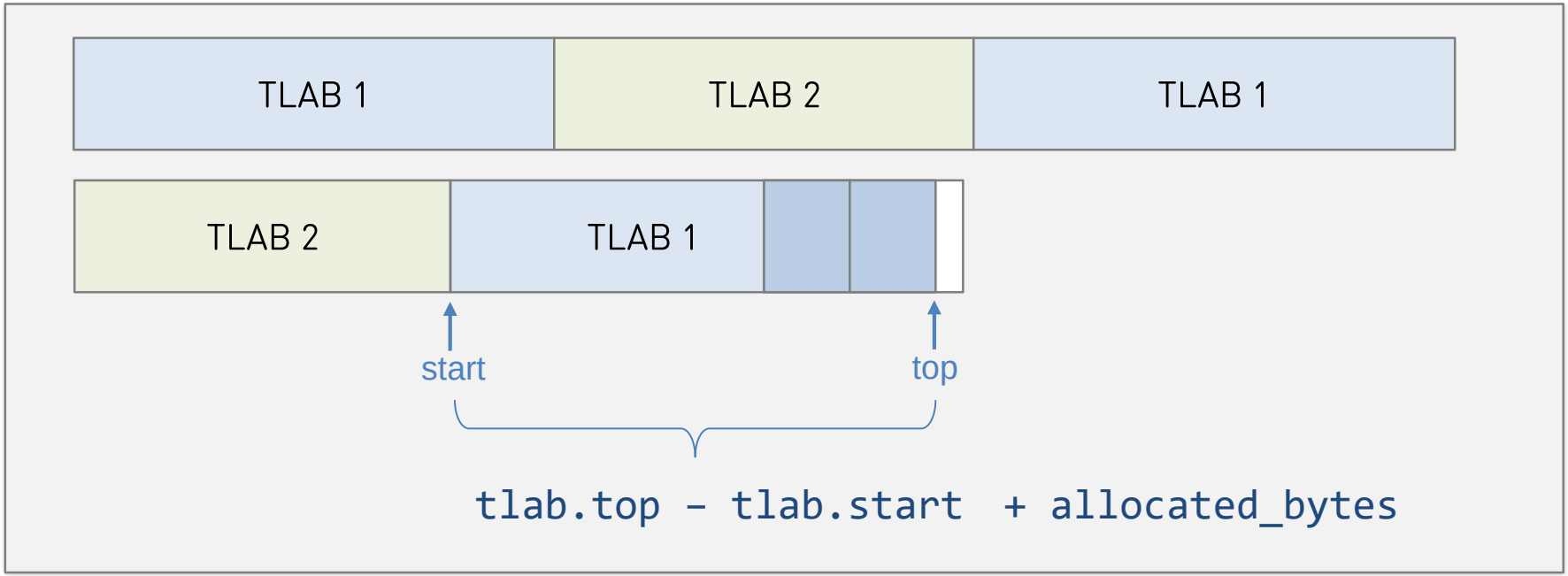
```
JavaThread* Threads::find_java_thread_from_java_tid(jlong java_tid) {

    // Sequential search for now.  Need to do better optimization later.
    for (JavaThread* thread = Threads::first();
         thread != NULL;
         thread = thread->next()) {

        ...
    }
```

- RFE: Optimize ThreadMXBean.getThreadAllocatedBytes()

Аллокация памяти



Аллокация памяти

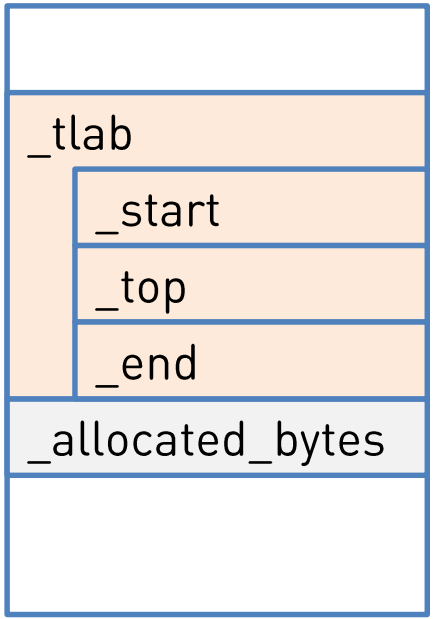


java.lang.Thread

```
class Thread {
  String    name;
  long      tid;
  int       priority;
  ...
  long      eetop;
```

Java Heap

VMThread



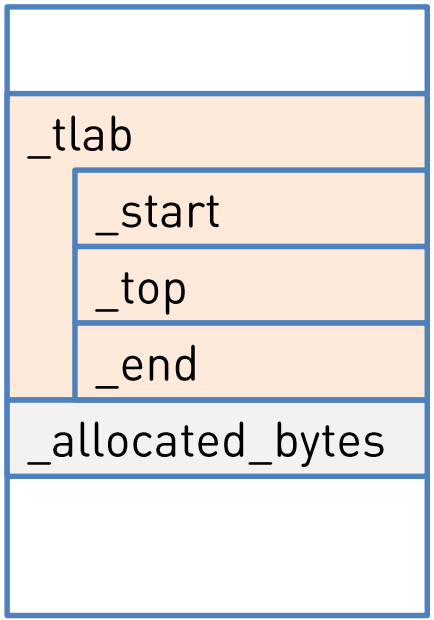
JVM memory

java.lang.Thread

```
class Thread {
  String    name;
  long      tid;
  int       priority;
  ...
  long      eetop;
```

через Reflection

VMThread



Java Heap

JVM memory

Чтение памяти JVM

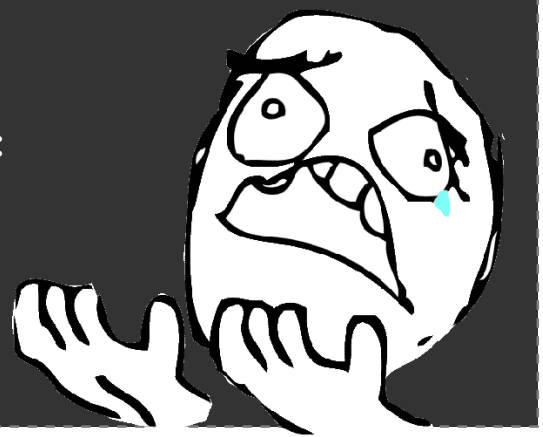
```
long vmThread = eetopField.getLong(Thread.currentThread());

long tlabStart = unsafe.getAddress(vmThread + off_tlab_start);
long tlabTop   = unsafe.getAddress(vmThread + off_tlab_top);
long allocated = unsafe.getLong(vmThread + off_allocated_bytes);

return (tlabTop - tlabStart) + allocated;
```

Чтение памяти JVM

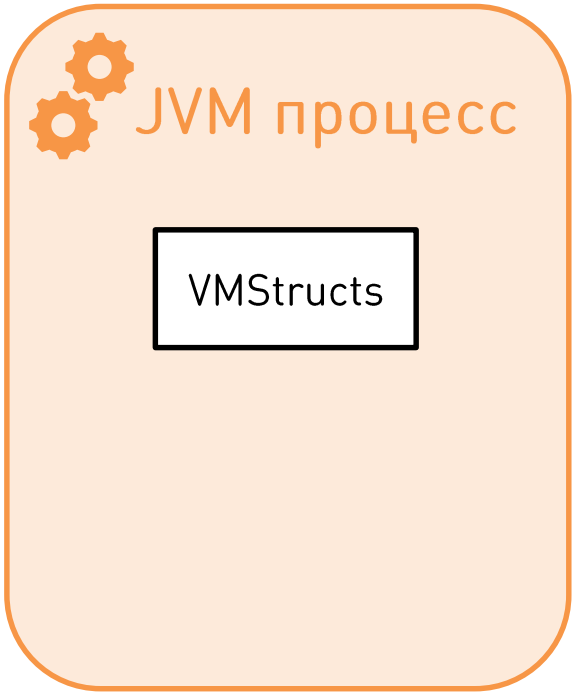
```
long vmThread = eetopField.getLong(Thread.currentThread());

long tlabStart = unsafe.getAddress(vmThread + off_tlab_start);
long tlabTop   = unsafe.getAddress(vmThread + off_tlab_top);
long allocated = unsafe.getLong(vmThread + off_allocated_bytes);

return (tlabTop - tlabStart) + allocated;
```

Поменяли чуть-чуть оффсеты

```
#
# A fatal error has been detected by the Java Runtime Environment:
#
#  SIGSEGV (0xb) at pc=0x00007f6440cc3941, pid=2358, tid=0x00007f64235e0700
#
# JRE version: OpenJDK Runtime Environment (8.0_151-b12)
# Java VM: OpenJDK 64-Bit Server VM (25.151-b12 mixed mode linux-amd64)
# Problematic frame:
# V  [libjvm.so+0xa33941]
#
# An error report file with more information is saved as:
# /home/apangin/hs_err_pid2358.log
#
```

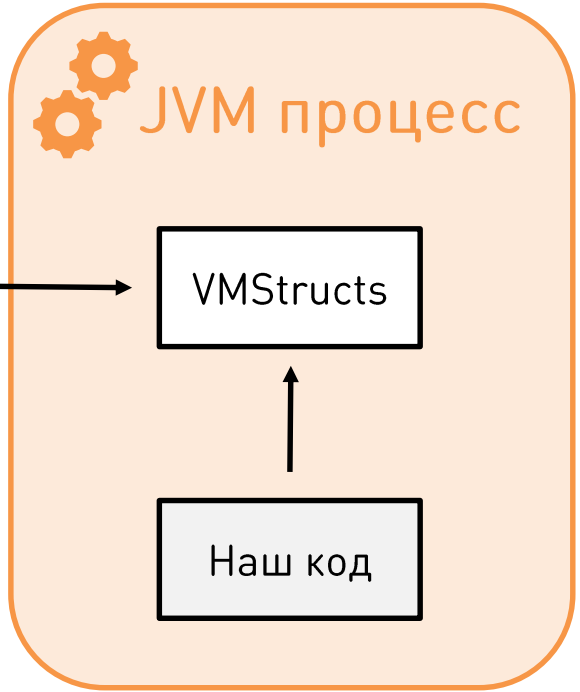


Как найти оффсеты?



Как найти оффсеты?

Serviceability Agent
jstack -F, jmap -F

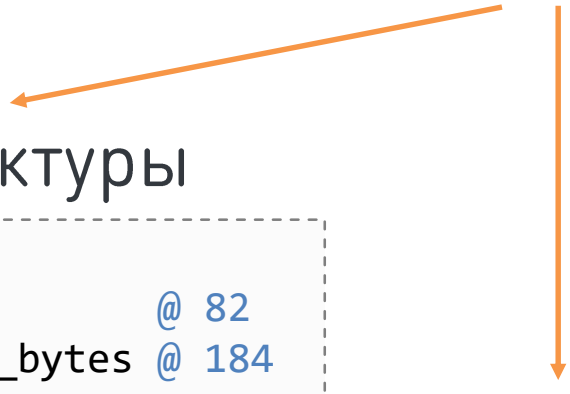


VM Structs

Структуры

```
Thread
  _tlab             @ 82
  _allocated_bytes  @ 184
```

VM Structs



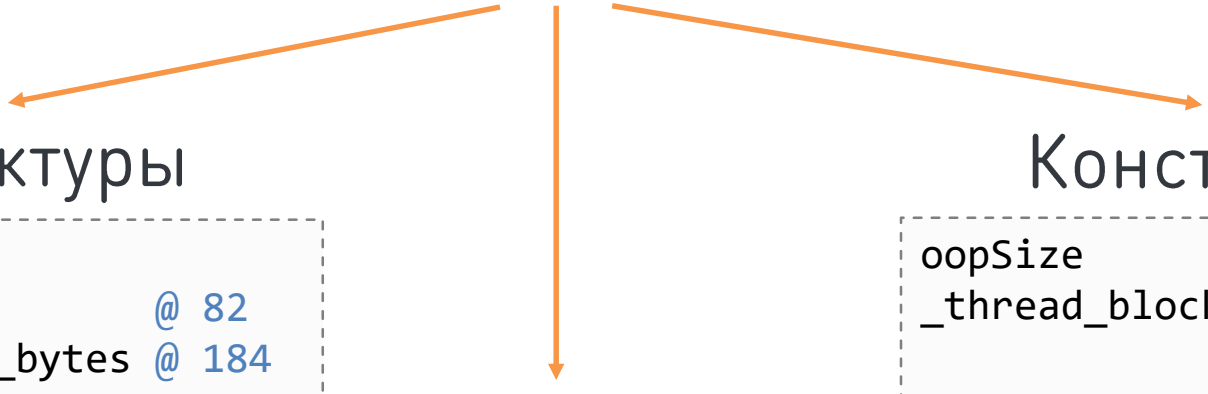
Структуры

```
Thread
  _tlab              @ 82
  _allocated_bytes   @ 184
```

Типы

```
nmethod extends CodeBlob (size=256)

ArrayKlass extends Klass (size=232)
```

VM Structs



Структуры

```
Thread
  _tlab              @ 82
  _allocated_bytes   @ 184
```

Константы

```
oopSize          = 8
_thread_blocked  = 10
```

Типы

```
nmethod extends CodeBlob (size=256)

ArrayKlass extends Klass (size=232)
```

src/share/vm/runtime/vmStructs.cpp

Структуры

Type name	Field name	Offset	Is static
Thread	_tlab	88	0
Thread	_allocated_bytes	184	0
ThreadLocalAllocBuffer	_start	0	0
ThreadLocalAllocBuffer	_top	8	0

Структуры

Type name	Field name	Offset	Is static
Thread	_tlab	88	0
Thread	_allocated_bytes	184	0
ThreadLocalAllocBuffer	_start	0	0
ThreadLocalAllocBuffer	_top	8	0

gHotSpotVMStructs

gHotSpotVMStructEntryFieldNameOffset

gHotSpotVMStructEntryArrayStride

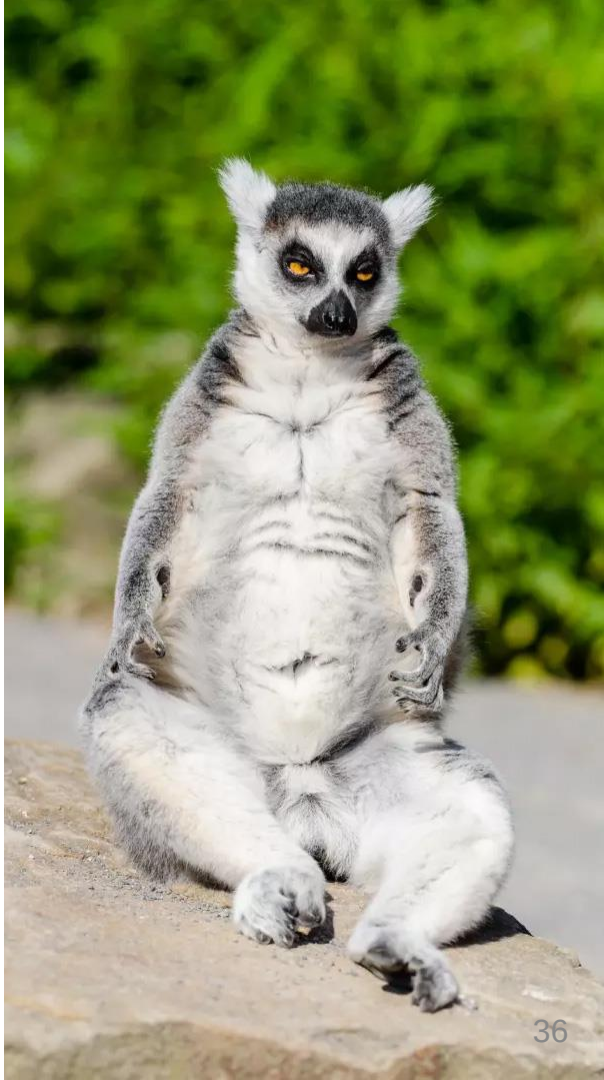
Как прочитать таблицу?

1. Найти libjvm.so
2. Распарсить ELF / PE / Mach-O формат
3. Найти адреса gHotSpotVMStructs*
4. Прочитать их через JNI / Unsafe

... или ...

Helfy

- https://github.com/apangin/helfy
  - парсит VM Structs
  - Java API
  - весь Unsafe спрятан



Helfy: пример

```
JVM jvm = new JVM();
```

Helfy: пример

```
JVM jvm = new JVM();

Type thread = jvm.type("Thread");
Type tlab = jvm.type("ThreadLocalAllocBuffer");
```

Helfy: пример

```
JVM jvm = new JVM();

Type thread = jvm.type("Thread");
Type tlab = jvm.type("ThreadLocalAllocBuffer");

off_allocated = thread.offset("_allocated_bytes");

off_tlab_start = thread.offset("_tlab")
               + tlab.offset("_start");
```

А что с памятью?

```
public long getThreadAllocatedBytes() {

    long vmThread = VMThread.current();




}
```

А что с памятью?

```
public long getThreadAllocatedBytes() {

    long vmThread = VMThread.current();

    long tlabStart = jvm.getAddress(vmThread + off_tlab_start);
    long tlabTop   = jvm.getAddress(vmThread + off_tlab_top);
    long allocated = jvm.getLong(vmThread + off_allocated);


}
```

А что с памятью?

```
public long getThreadAllocatedBytes() {

    long vmThread = VMThread.current();

    long tlabStart = jvm.getAddress(vmThread + off_tlab_start);
    long tlabTop   = jvm.getAddress(vmThread + off_tlab_top);
    long allocated = jvm.getLong(vmThread + off_allocated);

    return (tlabTop - tlabStart) + allocated;
}
```

А перформанс?

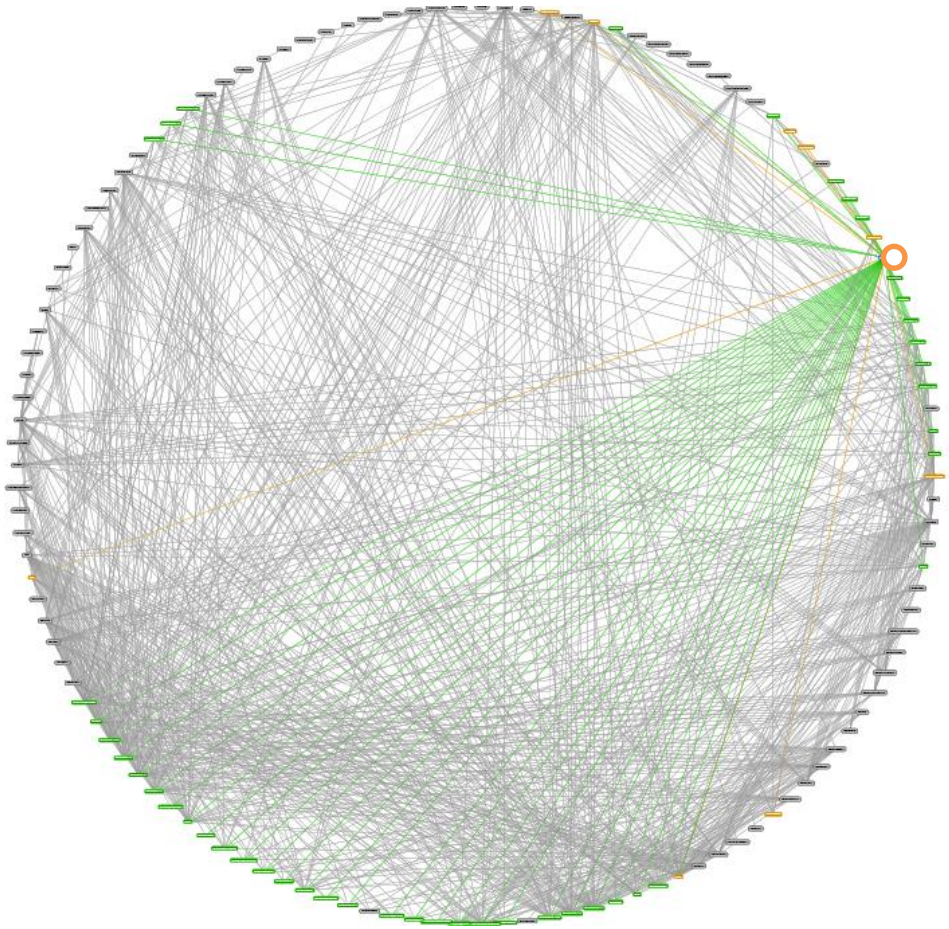
threads	JMX	VMStructs
1	77 ±1%	6 ±2%
4	959 ±2%	7 ±2%
40	27 441 ±6%	10 ±5%
400	80 533 720 ±50%	106 ±5%

nanosec/op

Что получили?

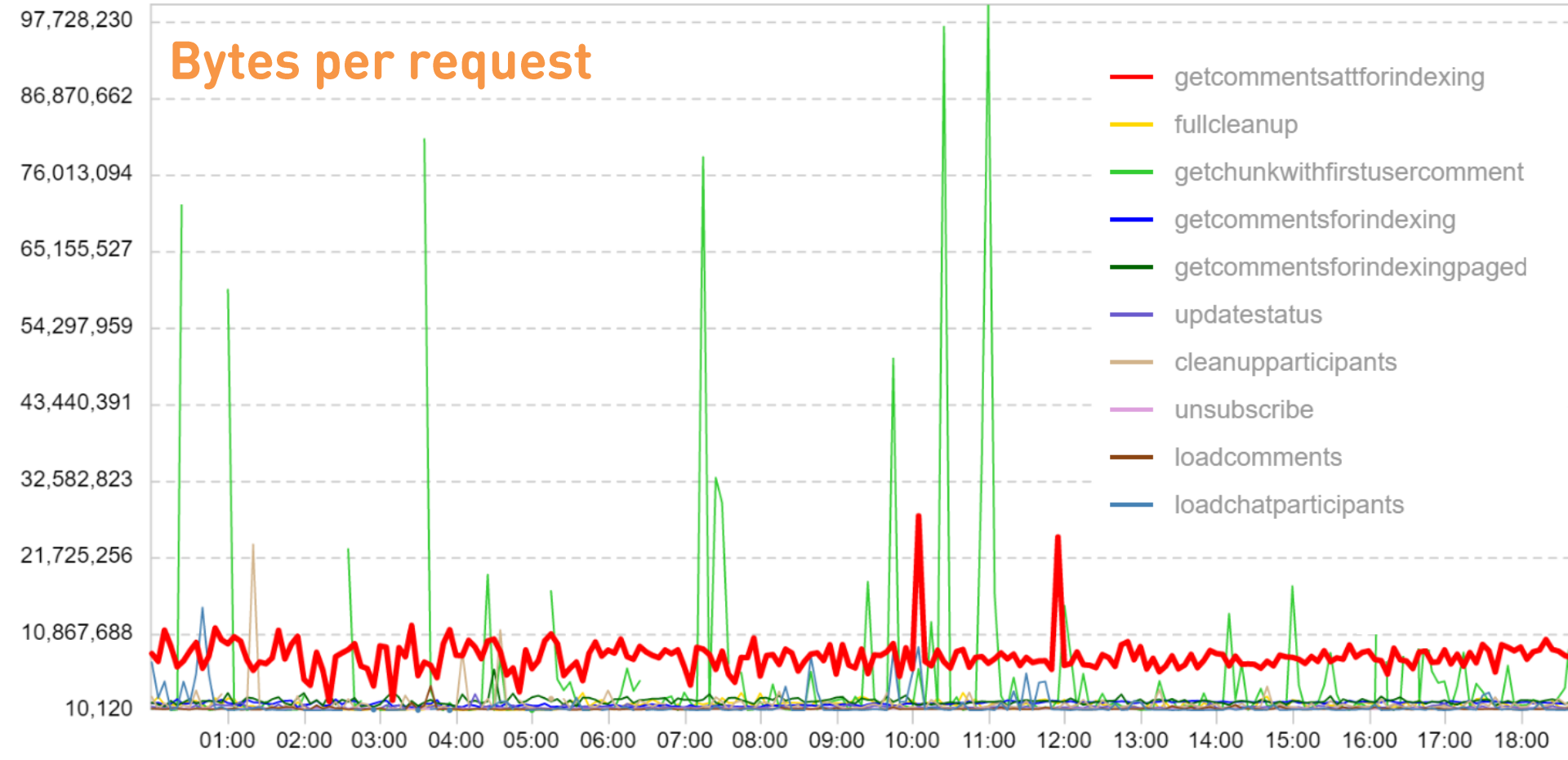
- [WARN] Too greedy request - 42 MB allocated

```
SliceFromReadCommand(
     keyspace='photos',
     cfName='photo_album',
     key='002f345674',
     filter='SliceQueryFilter [slices=… count=1026]’
)
```

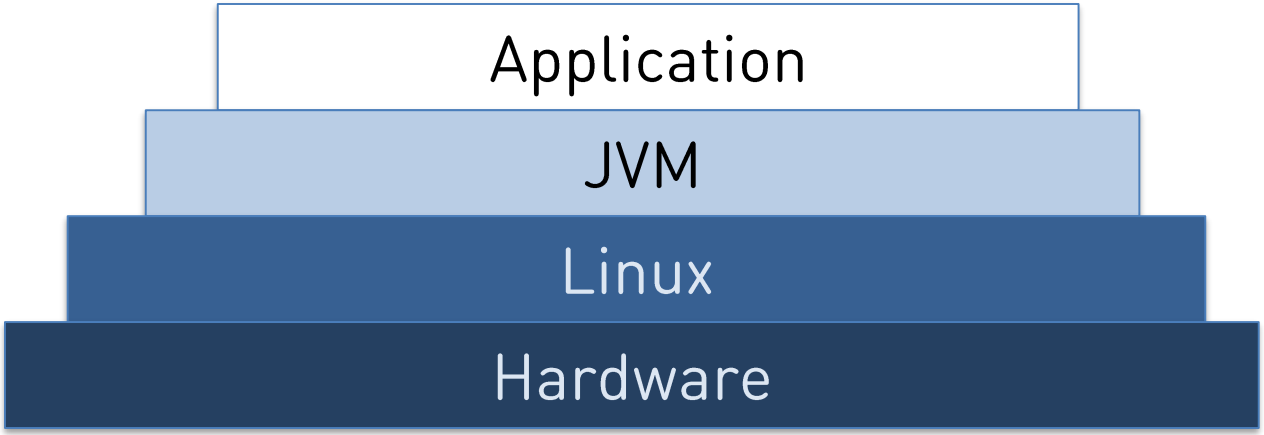


Allocation Tracker на уровне протокола

1. Инструментируем вызовы
2. Собираем статистику
3. Пишем в DWH

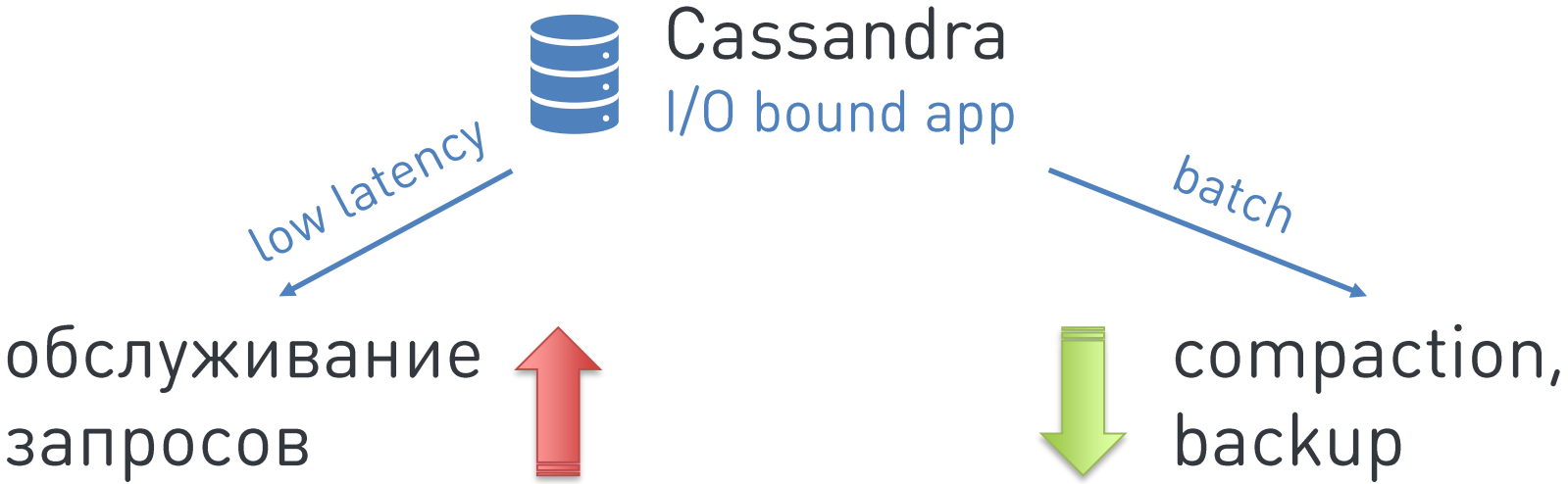

Bytes per request
97,728,230
86,870,662
76,013,094
65,155,527
54,297,959
43,440,391
32,582,823
21,725,256
10,867,688
10,120
01:00
02:00
03:00
04:00
05:00
06:00
07:00
08:00
09:00
10:00
11:00
12:00
13:00
14:00
15:00
16:00
17:00
18:00
getcommentsattforindexing
fullcleanup
getchunkwithfirstusercomment
getcommentsforindexing
getcommentsforindexingpaged
updatestatus
cleanupparticipants
unsubscribe
loadcomments
loadchatparticipants

JVM – не только про память



- File system
- Network
- Threads

Пример: фоновое I/O



Про классы изоляции задач в облаке Одноклассников
https://habrahabr.ru/company/odnoklassniki/blog/346868/

Пример: фоновое I/O

```
while (!eof()) {
    long t0 = System.nanoTime();
    int bytes = channel1.read(buf);
    long t1 = System.nanoTime();
    ...
```

	Throughput	Latency 99%
Одна задача	98 MB/s	11.4 ms

Пример: фоновое I/O

```
while (!eof()) {
    long t0 = System.nanoTime();
    int bytes = channel1.read(buf);
    long t1 = System.nanoTime();
    ...
```

Thread.setPriority()

```
while (!eof()) {
    long t0 = System.nanoTime();
    int bytes = channel2.read(buf);
    long t1 = System.nanoTime();
    ...
```

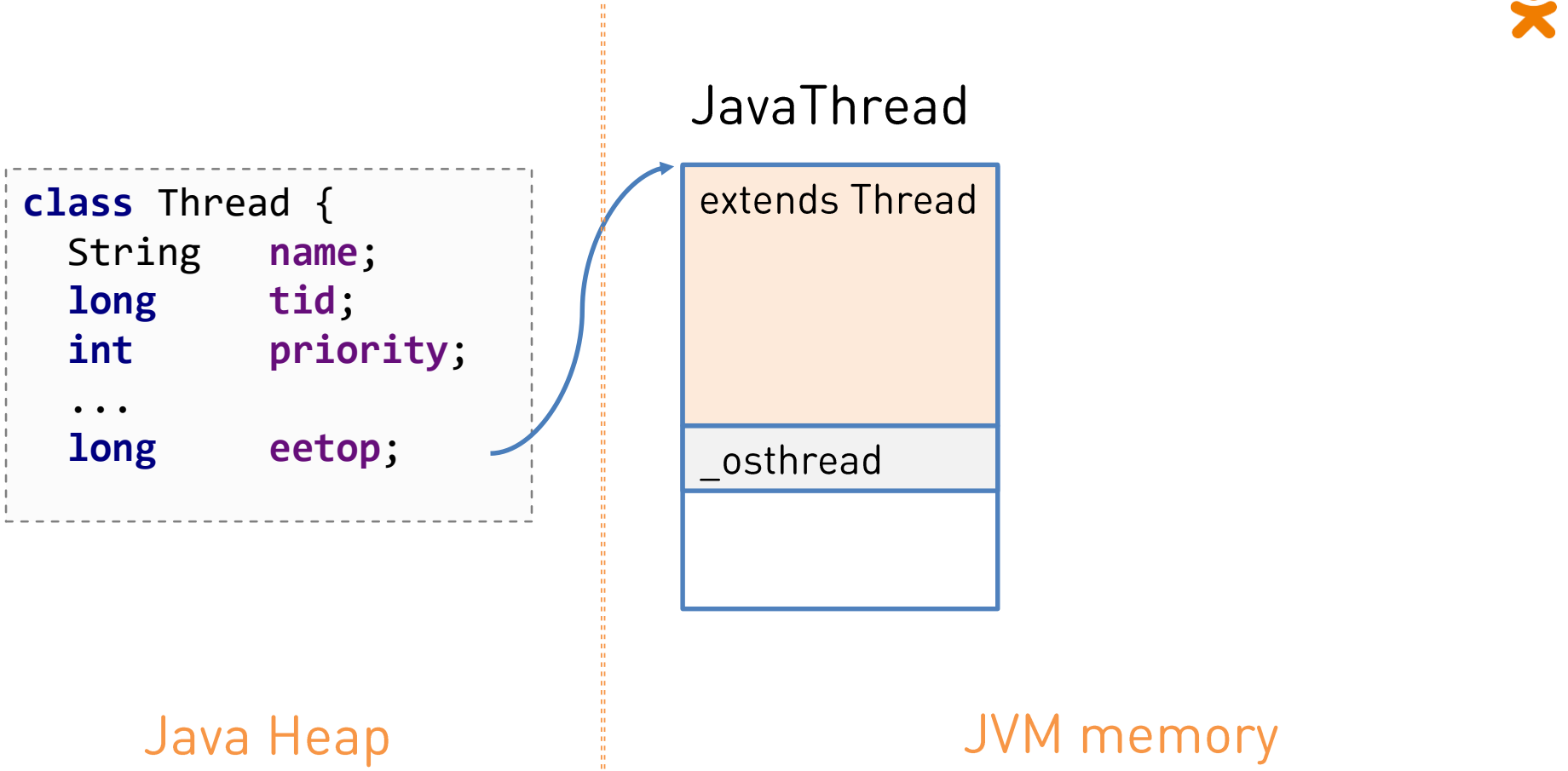
	Throughput	Latency 99%
Одна задача	98 MB/s	11.4 ms
С фоновым потоком	75 MB/s	19.2 ms

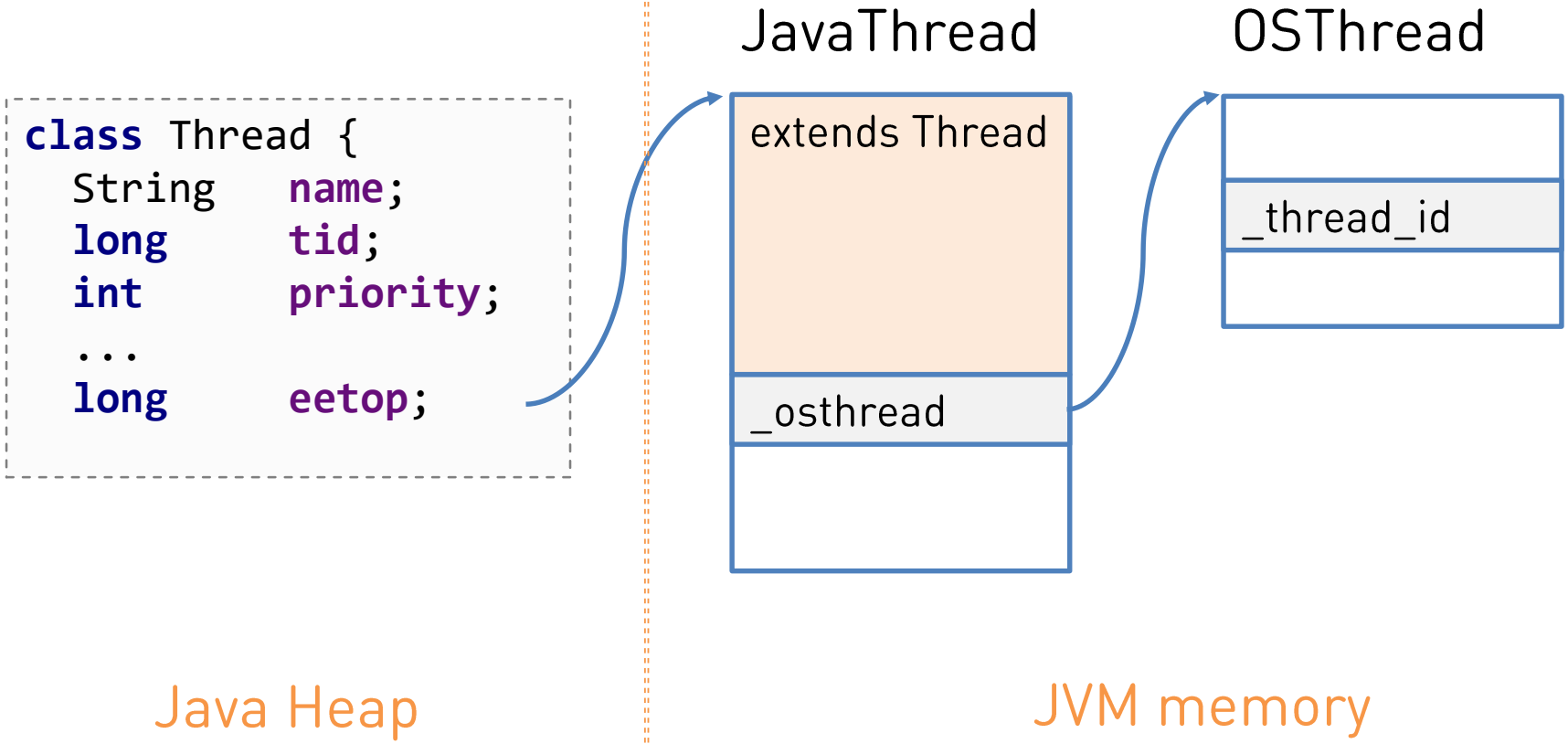
I/O приоритеты в Linux

```
int ioprio_set(int which, int who, int ioprio);
```

системный thread id

а у нас только java.lang.Thread ☹


class Thread {
String name;
long tid;
int priority;
...
long eetop;
JavaThread
extends Thread
_osthread
Java Heap
JVM memory


class Thread {
String name;
long tid;
int priority;
...
long eetop;
JavaThread
OSThread
extends Thread
_osthread
_thread_id
Java Heap
JVM memory

Получение ID потока

```
off_osthread  = jvm.type("JavaThread").offset("_osthread");
off_thread_id = jvm.type("OSThread").offset("_thread_id");
```

Получение ID потока

```
off_osthread  = jvm.type("JavaThread").offset("_osthread");
off_thread_id = jvm.type("OSThread").offset("_thread_id");
```

```
public int getThreadId(Thread javaThread) {
    long vmThread = VMThread.of(javaThread);




}
```

Получение ID потока

```
off_osthread  = jvm.type("JavaThread").offset("_osthread");
off_thread_id = jvm.type("OSThread").offset("_thread_id");
```

```
public int getThreadId(Thread javaThread) {
    long vmThread = VMThread.of(javaThread);

    long osThread = jvm.getAddress(vmThread + off_osthread);


}
```

Получение ID потока

```
off_osthread  = jvm.type("JavaThread").offset("_osthread");
off_thread_id = jvm.type("OSThread").offset("_thread_id");
```

```
public int getThreadId(Thread javaThread) {
    long vmThread = VMThread.of(javaThread);

    long osThread = jvm.getAddress(vmThread + off_osthread);

    return jvm.getInt(osThread + off_thread_id);
}
```

Пример: фоновое I/O

	Throughput	Latency 99%
Одна задача	98 MB/s	11.4 ms
С фоновым потоком	75 MB/s	19.2 ms
ioprio_set*	98 MB/s	12.0 ms

* IOPRIO_CLASS_IDLE для фонового потока

Мы не одиноки

Tagir Valeev @tagir_valeev · 23 февр.
Black magic github.com/JetBrains/inte...

1 4

com.intellij.openapi.util.objectTree.ThrowableInterner

А в чём проблема?

- Группировать одинаковые исключения
- У Throwable нет equals и hashCode
- getStackTrace дорогой

«Всё, что вы хотели знать о стек-трейсах и хип-дампах»
https://youtu.be/0pyZERLBZvQ

Секреты JVM

```
public class Throwable {

    /**
     * Native code saves some indication of the stack backtrace in this slot.
     */
    private transient Object backtrace;
```

Секреты JVM

```
public class Throwable {

    /**
     * Native code saves some indication of the stack backtrace in this slot.
     */
    private transient Object backtrace;
```

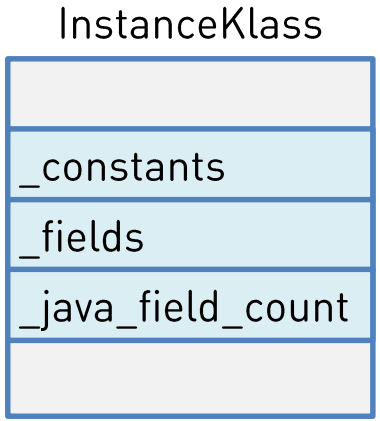
```
// JDK 8
Throwable.class.getDeclaredFields()
```

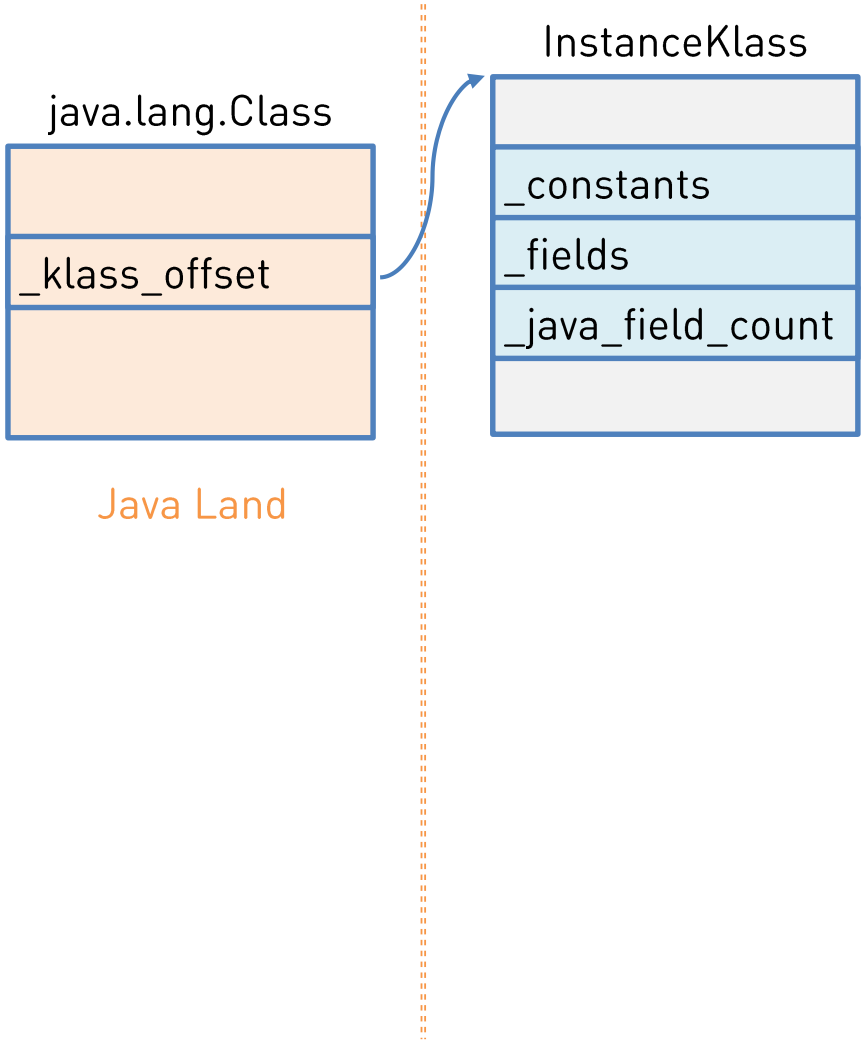
- detailMessage
- cause
- stackTrace
- suppressedExceptions

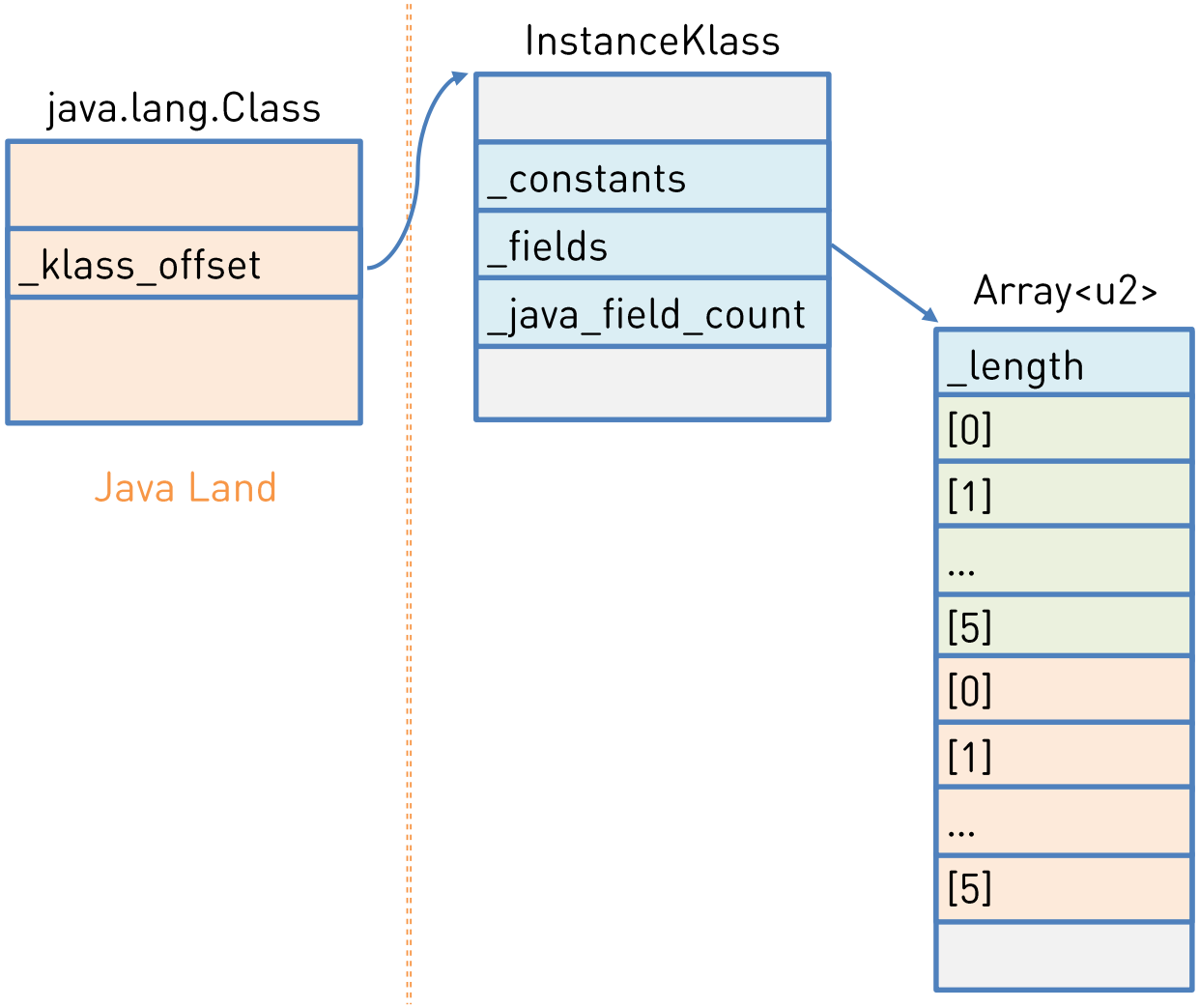
В этом коде прекрасно всё

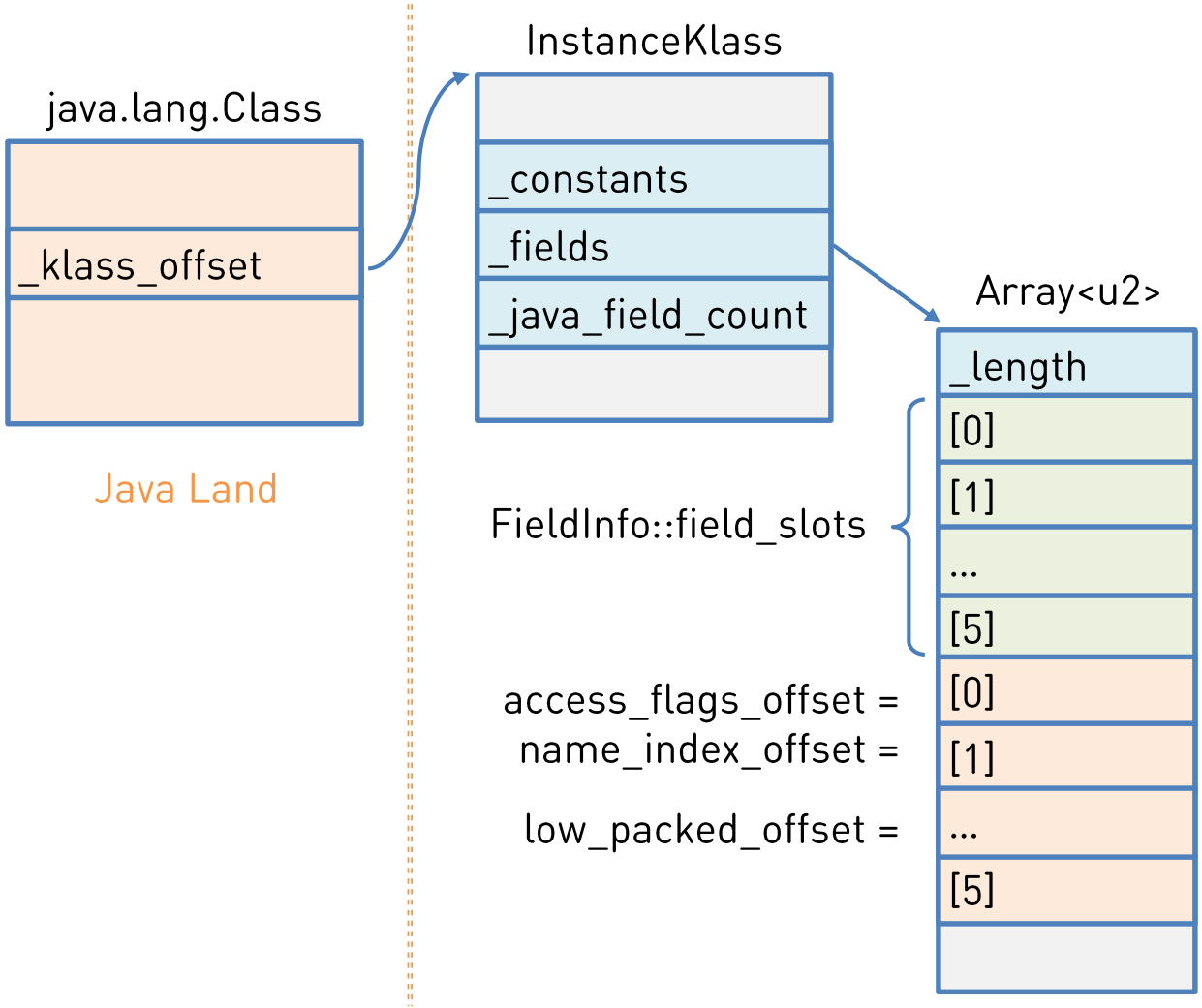
```
// Please don't look, there's nothing interesting here.

if ((SystemInfo.isOracleJvm || SystemInfo.isJetBrainsJvm)
    && SystemInfo.isJavaVersionAtLeast(7, 0, 0)) {

    Field firstField = Throwable.class.getDeclaredFields()[1];
    long firstFieldOffset =
        AtomicFieldUpdater.getUnsafe().objectFieldOffset(firstField);

    BACKTRACE_FIELD_OFFSET = firstFieldOffset == 12 ? 8 :
                             firstFieldOffset == 16 ? 12 :
                             firstFieldOffset == 24 ? 16 : UNKNOWN;
```

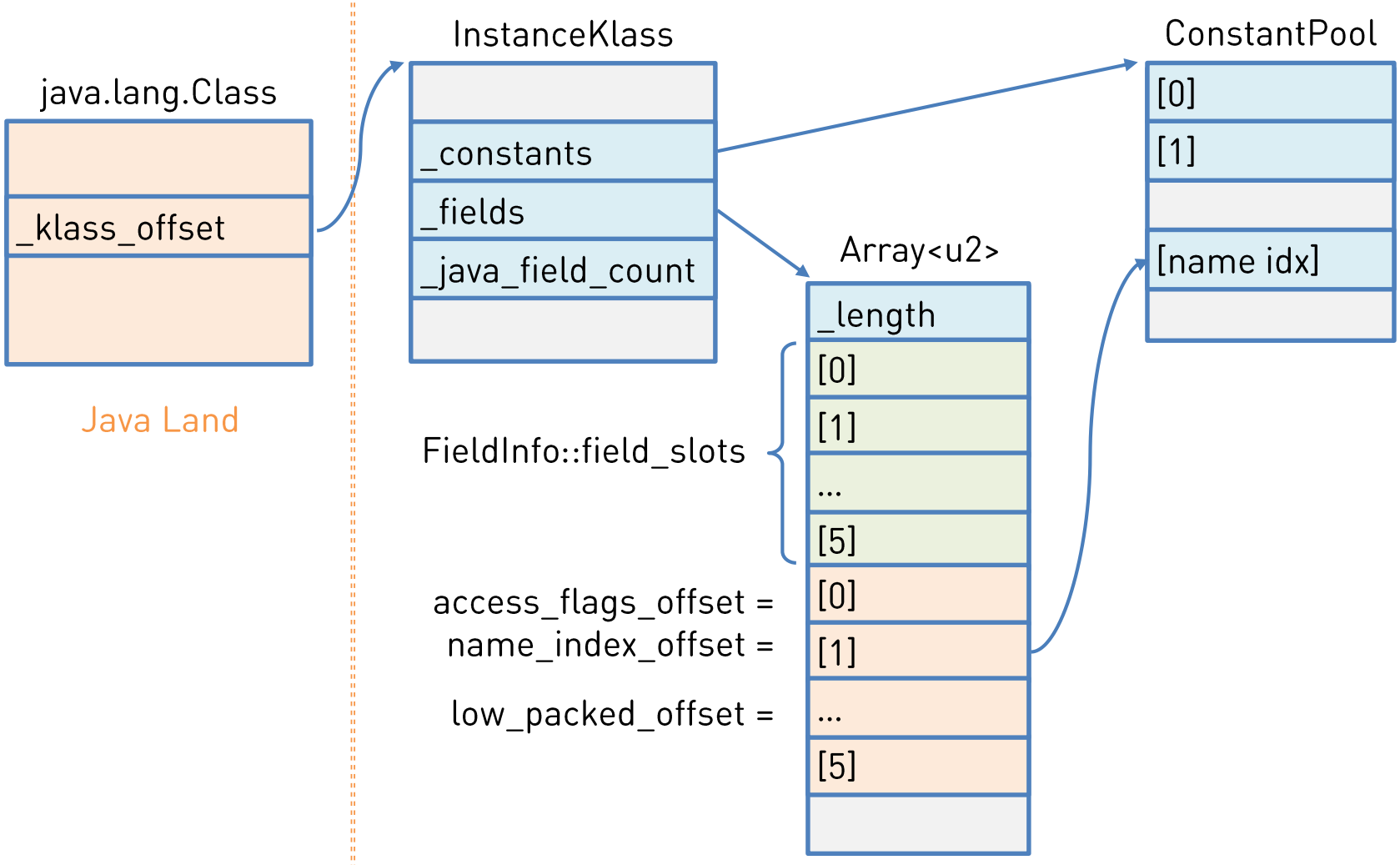
Доступ к полям через VM Structs

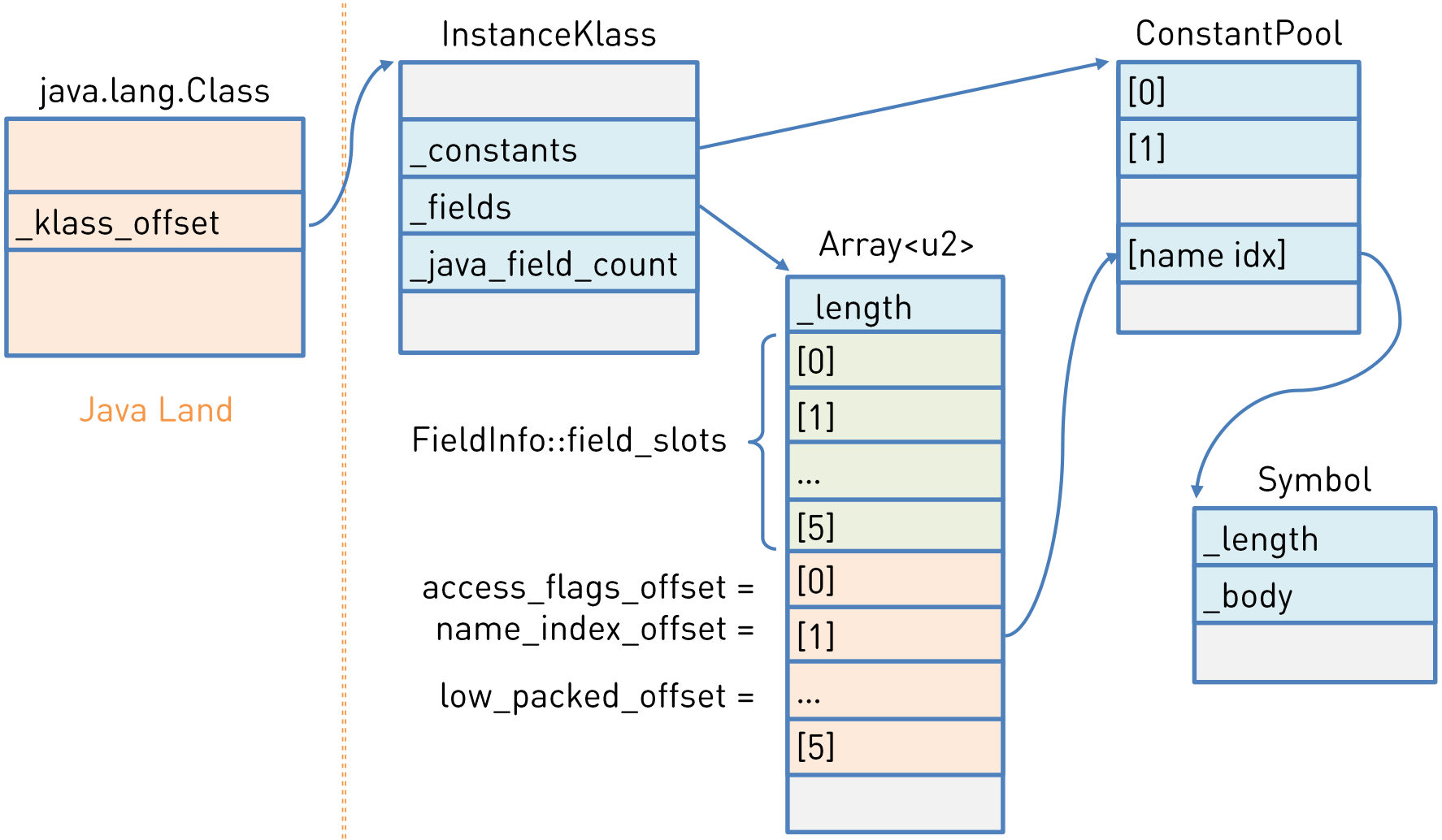

InstanceKlass
_constants
_fields
_java_field_count


InstanceKlass
java.lang.Class
_constants
_klass_offset
_fields
_java_field_count
Java Land


java.lang.Class
_klass_offset
Java Land
InstanceKlass
_constants
_fields
_java_field_count
Array<u2>
_length
[0]
[1]
...
[5]
[0]
[1]
...
[5]


java.lang.Class
_klass_offset
Java Land
InstanceKlass
_constants
_fields
_java_field_count
Array<u2>
_length
[0]
[1]
...
[5]
FieldInfo::field_slots
access_flags_offset =
name_index_offset =
low_packed_offset =
[0]
[1]
...
[5]


java.lang.Class
_klass_offset
Java Land
InstanceKlass
_constants
_fields
_java_field_count
ConstantPool
[0]
[1]
[name idx]
Array<u2>
_length
[0]
[1]
...
[5]
FieldInfo::field_slots
access_flags_offset =
name_index_offset =
low_packed_offset =
[0]
[1]
...
[5]


java.lang.Class
_klass_offset
Java Land
InstanceKlass
_constants
_fields
_java_field_count
ConstantPool
[0]
[1]
[name idx]
Array<u2>
_length
[0]
[1]
...
[5]
FieldInfo::field_slots
access_flags_offset =
name_index_offset =
low_packed_offset =
[0]
[1]
...
[5]
Symbol
_length
_body

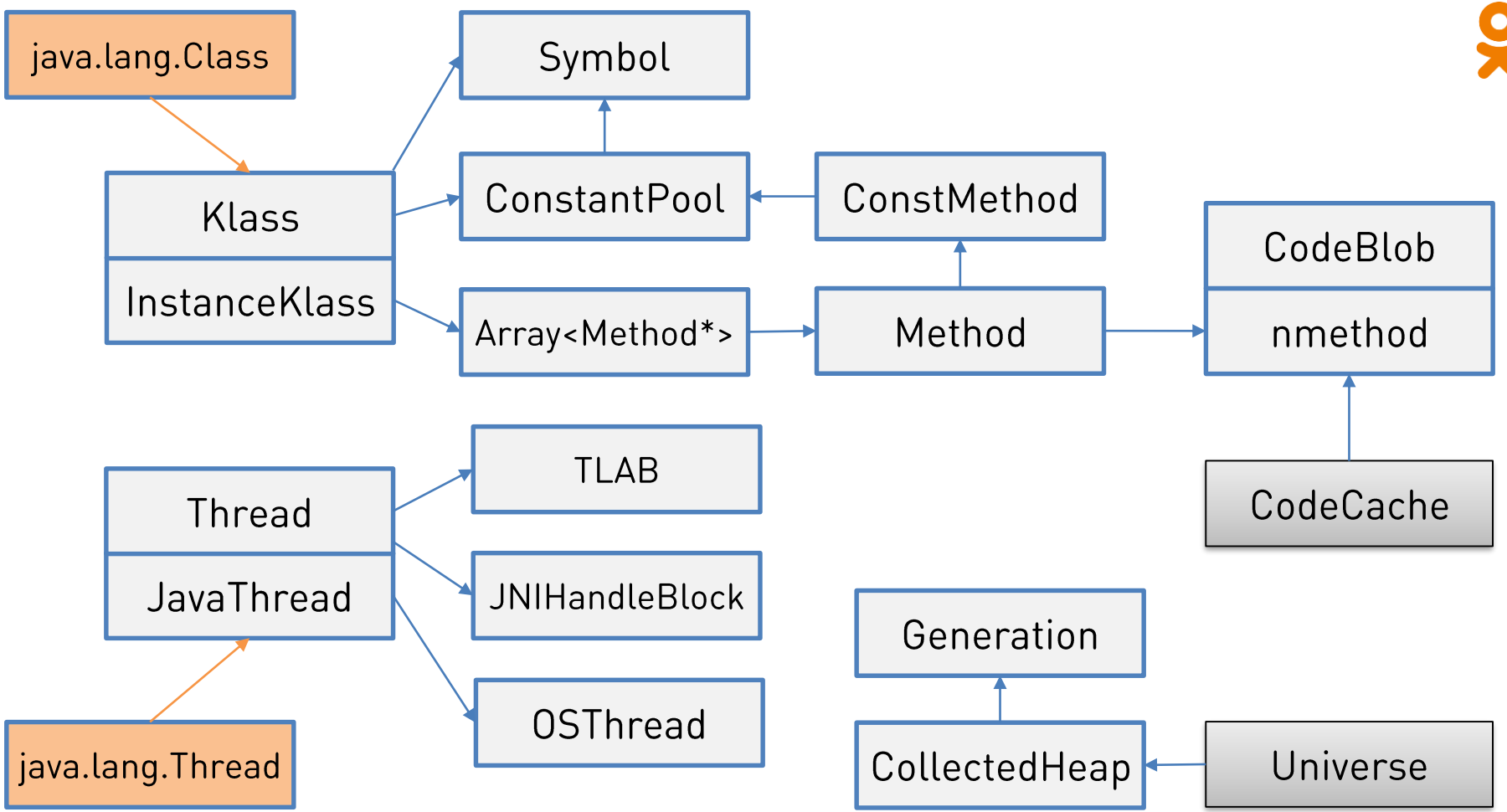
Перечисляем все поля

https://github.com/apangin/helfy/tree/master/test/one/helfy/FieldInfo.java

```
java.lang.Throwable
  12  backtrace
  16  detailMessage
  20  cause
  24  stackTrace
  28  suppressedExceptions
```

Где берёшь ~~такую траву~~ эти структуры?

```
jvm.dump(System.out);

  > 250 констант
  > 700 типов
  > 800 полей
```


java.lang.Class
Symbol
Klass
InstanceKlass
ConstantPool
ConstMethod
CodeBlob
nmethod
Array<Method*>
Method
Thread
JavaThread
TLAB
JNIHandleBlock
OSThread
CodeCache
Generation
CollectedHeap
Universe
java.lang.Thread

VM Structs 80 lvl

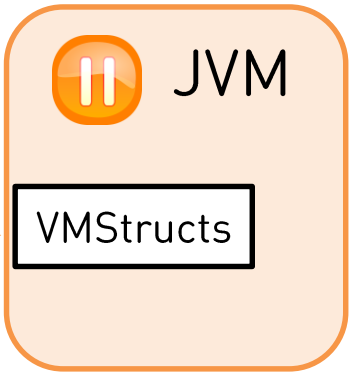
- Достать скомпилированный код
- Подменить entry point метода
- Найти значения локальных переменных

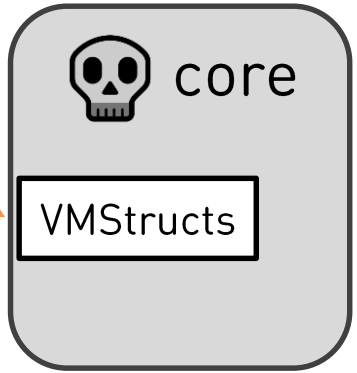
- И многое другое...

https://github.com/apangin/helfy/tree/master/test/

Нетрадиционное применение

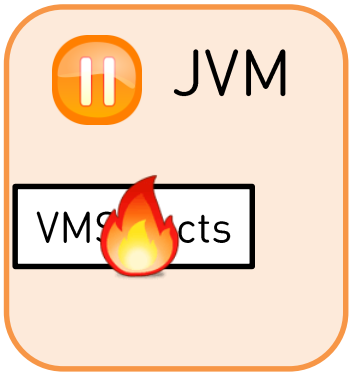
jstack –F
jmap –F

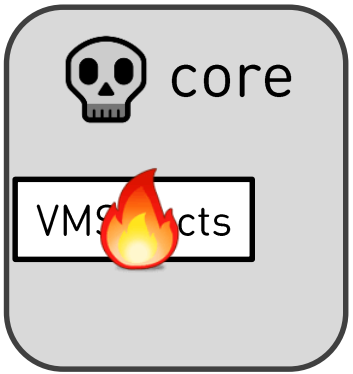




Нетрадиционное применение

jstack –F
jmap –F





Портим VM Structs

```
long vmStructs = jvm.getSymbol("gHotSpotVMStructs");
jvm.putAddress(vmStructs, 0);
```

Портим VM Structs

```
long vmStructs = jvm.getSymbol("gHotSpotVMStructs");
jvm.putAddress(vmStructs, 0);
```

```
$ jmap -F 1234
Attaching to process ID 1234, please wait...
Exception in thread "main" java.lang.reflect.InvocationTargetException
        at ...
Caused by: java.lang.RuntimeException: gHotSpotVMStructs unexpectedly had a NULL type name at index 0
        at sun.jvm.hotspot.HotSpotTypeDataBase.readVMStructs(HotSpotTypeDataBase.java:444)
        at sun.jvm.hotspot.HotSpotTypeDataBase.<init>(HotSpotTypeDataBase.java:91)
        at sun.jvm.hotspot.HotSpotAgent.setupVM(HotSpotAgent.java:391)
        at sun.jvm.hotspot.HotSpotAgent.go(HotSpotAgent.java:305)
        at sun.jvm.hotspot.HotSpotAgent.attach(HotSpotAgent.java:140)
        at sun.jvm.hotspot.tools.Tool.start(Tool.java:185)
        at sun.jvm.hotspot.tools.Tool.execute(Tool.java:118)
        at sun.jvm.hotspot.tools.PMap.main(PMap.java:72)
        ... 6 more
```

- Знание внутренностей JVM позволяет
  - расширить возможности мониторинга и управления
  - преодолеть ограничения Java платформы
- Helfy упрощает работу с VM Structs
- Специфика HotSpot JVM


@AndreiPangin
andrey.pangin@corp.mail.ru
https://v.ok.ru/vacancies.html