JVM TI: как сделать плагин для виртуальной машины

Андрей Паньгин, Одноклассники

Обо мне

@AndreiPangin

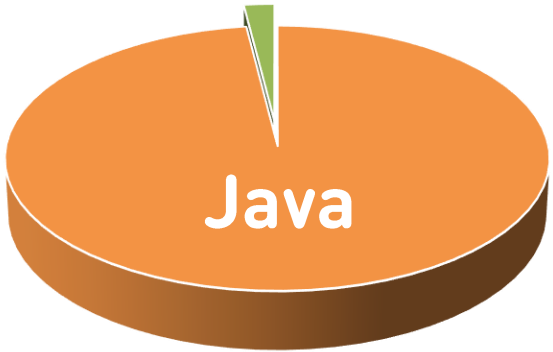
Ведущий разработчик

Специалист по HotSpot JVM

Автор one-nio и async-profiler

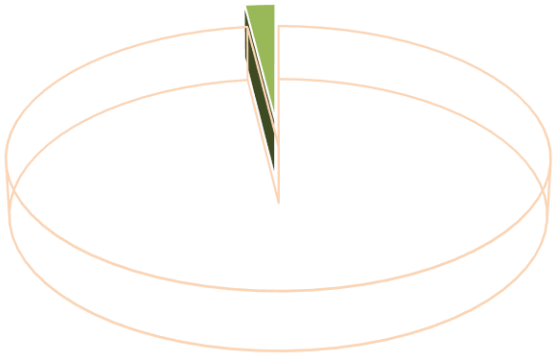
В Одноклассниках

- 71 M MAU
- 6000 серверов в 4 ЦОД
- 2.5 Тбит/с
- 200 (микро)сервисов
- до 100 K QPS на сервер



В Одноклассниках

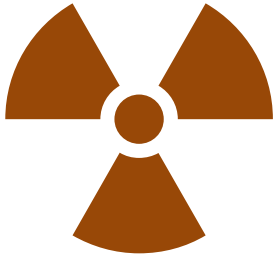
- 71 M MAU
- 6000 серверов в 4 ЦОД
- 2.5 Тбит/с
- 200 (микро)сервисов
- до 100 K QPS на сервер



Achtung! Achtung!

Код на C

Байткод



JVM

Java ошибка №1

Java ошибка №1

```
java.lang.NullPointerException
 at one.app.databus.DatabusHandler.addDetails(DatabusHandler.java:462)
 at one.app.databus.DatabusHandler.handleActivity(DatabusHandler.java:223)
 at one.app.activity.ActivityService.reportActivity(ActivityService.java:139)
 at one.app.activity.ActivityService$RunnableActivity.run(ActivityService.java:176)
 at java.util.concurrent.FutureTask.run(FutureTask.java:266)
 at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
 at java.lang.Thread.run(Thread.java:745)
```

Где NullPointerException?

```
public class DatabusHandler {

    private void addDetails(Activity activity, EventBuilder event) {
        ...
        case CHAT_ADD_USERS:
            for (long userID : activity.getData().getUsers()) {
                event.addUser(userID);
            }
```

Где NullPointerException?

```
public class DatabusHandler {

    private void addDetails(Activity activity, EventBuilder event) {
        ...
        case CHAT_ADD_USERS:
            for (long userID : activity.getData().getUsers()) {
                event.addUser(userID);
            }
```

null?

Где NullPointerException?

```
public class DatabusHandler {

    private void addDetails(Activity activity, EventBuilder event) {
        ...
        case CHAT_ADD_USERS:
            for (long userID : activity.getData().getUsers()) {
                event.addUser(userID);
            }
```

null?

Где NullPointerException?

```
public class DatabusHandler {

    private void addDetails(Activity activity, EventBuilder event) {
        ...
        case CHAT_ADD_USERS:
            for (long userID : activity.getData().getUsers()) {
                event.addUser(userID);
            }
```

List<Long> null?

Подробный NPE message

```
java.lang.NullPointerException: Called 'getUsers()' method on null object
 at one.app.databus.DatabusHandler.addDetails(DatabusHandler.java:462)
 at one.app.databus.DatabusHandler.handleActivity(DatabusHandler.java:223)
 at one.app.activity.ActivityService.reportActivity(ActivityService.java:139)
 at one.app.activity.ActivityService$RunnableActivity.run(ActivityService.java:176)
 at java.util.concurrent.FutureTask.run(FutureTask.java:266)
 at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor.java:1142)
 at java.lang.Thread.run(Thread.java:745)
```

Подробный NPE message

JDK / JDK-4834738

RFE: NullPointerException: Better info

Details

Type:	Enhancement	Status:	CLOSED
Priority:	P4	Resolution:	Won't Fix
Affects Version/s:	1.4.1, 1.4.2, 5.0	Fix Version/s:	8-pool

Подробный NPE message

JDK / JDK-4834738

RFE: NullPointerException: Better info

JDK / JDK-6717558

NullPointerException's from JVM lack any detail message

Details

Type:	Enhancement	Status:	CLOSED
Priority:	P4	Resolution:	Duplicate
Affects Version/s:	5.0	Fix Version/s:	None

Подробный NPE message

JDK / JDK-4834738

RFE: NullPointerException: Better info

JDK / JDK-6717558

NullPointerException's from JVM lack any detail message

JDK / JDK-8057897

NullPointerException message should name the null variable.

Details

Type:	Enhancement	Status:	CLOSED
Priority:	P4	Resolution:	Won't Fix
Affects Version/s:	8u20	Fix Version/s:	None

Тем временем в других JVM

- Volker Simonis — SAP JVM Internals

  https://jug.ru/2015/10/volker-simonis-sap-jvm-internals/

Тем временем в других JVM

- Volker Simonis — SAP JVM Internals

  https://jug.ru/2015/10/volker-simonis-sap-jvm-internals/

JDK / JDK-8218628

Add detailed message to NullPointerException describing what is null.

Details

Type:	Enhancement	Status:	IN PROGRESS
Priority:	P4	Resolution:	Unresolved
Affects Version/s:	13	Fix Version/s:	13

Альтернативы

- Своя сборка OpenJDK
  - поддержка

Альтернативы

- Своя сборка OpenJDK
  - поддержка
- Плагин к JVM
  - а что, так можно было? 😲

JVM Tool Interface

- API для профайлеров, отладчиков и т.п.
  - JDK 5+
- Стандартный
  - https://docs.oracle.com/en/java/javase/11/docs/specs/jvmti.html

JVM Tool Interface

- C/C++ API
  - jdk/include/jvmti.h

JVM Tool Interface

- C/C++ API
  - jdk/include/jvmti.h
- Нативный агент
  - .dll / .so / .dylib

```
java -agentpath:/path/to/mytool.so
```

JVM Tool Interface

- C/C++ API
  - jdk/include/jvmti.h
- Нативный агент
  - .dll / .so / .dylib

```
java -agentpath:/path/to/mytool.so
```

- Не путать

  с Java агентом!

```
java -javaagent:myagent.jar
```

Demo

https://github.com/odnoklassniki/jvmti-tools/richNPE

Simplest way to auto-restart a JVM on StackOverflowError

0

It does not seem that there is a -XX option to restart a JVM on **StackOverflowError**. What is the simplest way to auto-restart a JVM when it gets a **StackOverflowError**?

error-handling jvm stack-overflow Edit tags

share edit close flag

stackoverflow

Simplest way to auto-restart a JVM on StackOverflowError

0

Generating a Java thread dump on an event/exception

0

I worked mainly with the IBM SDK so there is a specific JVM argument you can use in order to enable dumps (heap, thread, system core) on specific events or exceptions (java.lang.OutOfMemoryError, SIGTERM, etc...)

I want to be able to do the same thing using the Oracle JDK. I only see the argument: **-XX:+HeapDumpOnOutOfMemoryError** which will only generate a heap dump for the specific exception java.lang.OutOfMemoryError.

1

Basically I do not have access to the code, so I want to be able to have the JVM generate both a heap dump and a Java thread dump for analysis (java.lang.OutOfMemoryError is one of many other events).

stackoverflow

Simplest way to auto-restart a JVM on StackOverflowError

0

★

Generating a Java thread dump on an event/exception

0

★

1

Generate Java Heap Dump on uncaught Exception

3

★

2

I try to generate a Heap Dump when a uncaught exception is fired. I tried using jmap, but because the process is finished when the exception happens this is not possible.

Using a UncaughtExceptionHandler is no option either, because I only have the binaries of the programs that is executed.

Can anyone help me?

EDIT: It is important that the technique is available through a command line or similar, because I need to automated this. Using a GUI is no option

Обход хипа и стеков в JVM TI

- GetAllStackTraces
- GetThreadListStackTraces

Thread dump

Обход хипа и стеков в JVM TI

- GetAllStackTraces
- GetThreadListStackTraces

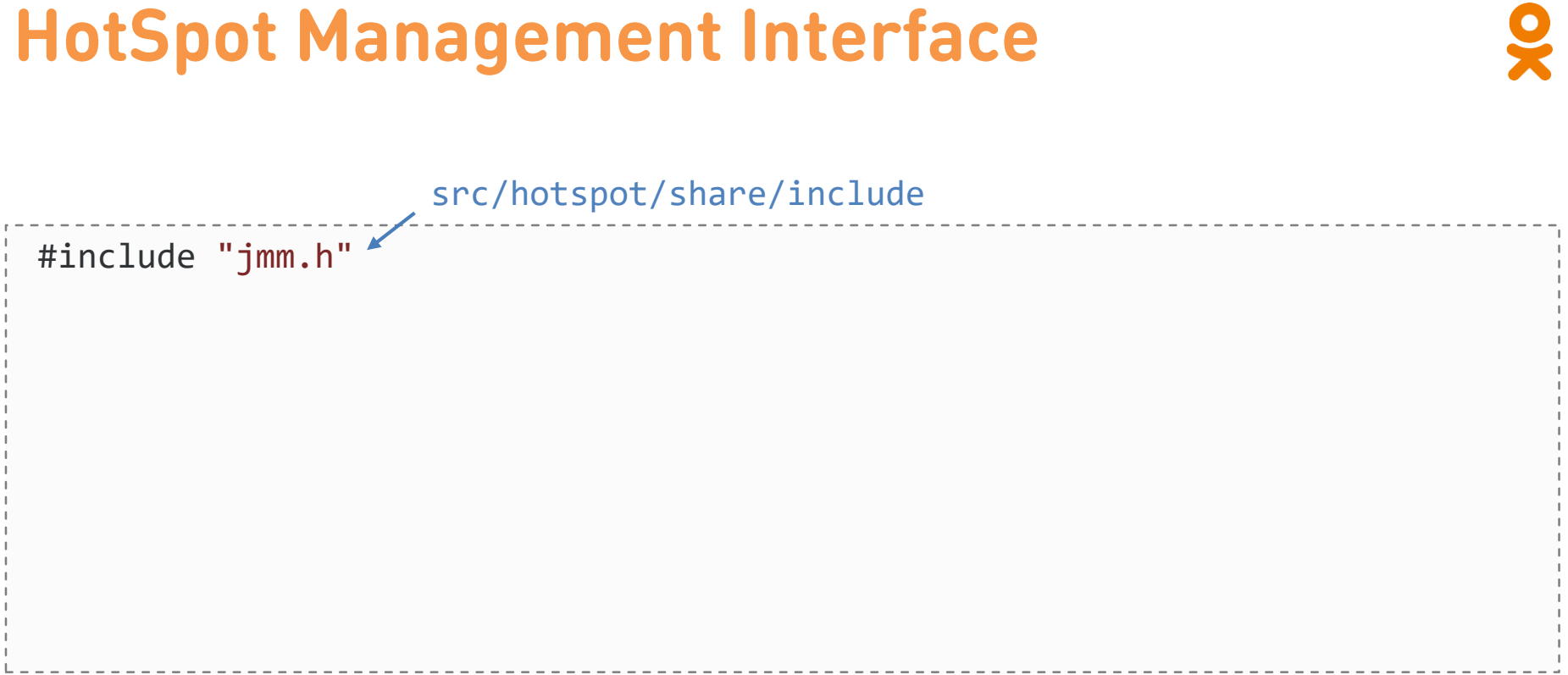
Thread dump

- IterateThroughHeap
- FollowReferences

Heap dump

HotSpot Management Interface

HotSpot Management Interface



HotSpot Management Interface

```
#include "jmm.h"

JNIEXPORT void* JNICALL JVM_GetManagement(jint version);

void JNICALL ExceptionCallback(jvmtiEnv* jvmti, JNIEnv* env, ...) {

    JmmInterface* jmm = (JmmInterface*) JVM_GetManagement(JMM_VERSION_1_0);
```

HotSpot Management Interface

```
#include "jmm.h"

JNIEXPORT void* JNICALL JVM_GetManagement(jint version);

void JNICALL ExceptionCallback(jvmtiEnv* jvmti, JNIEnv* env, ...) {

    JmmInterface* jmm = (JmmInterface*) JVM_GetManagement(JMM_VERSION_1_0);

    jmm->DumpHeap0(env, env->NewStringUTF("dump.hprof"), JNI_FALSE);
```

https://stackoverflow.com/questions/23632653/generate-java-heap-dump-on-uncaught-exception

Другие типы событий

Thread Start / End
Class Load / Prepare

GC Start / Finish
Compiled Method Load

Method Entry / Exit
Breakpoint / Single Step

Monitor Wait
Field Access / Modification

Demo

https://github.com/odnoklassniki/jvmti-tools/vmtrace

Фичи JVM TI

обязательные

capabilities

Фичи JVM TI

обязательные

- method / field info
- thread info
- get stack trace
- force gc
- get object size
- raw monitors
- add to classpath

capabilities

Фичи JVM TI

обязательные

- method / field info
- thread info
- get stack trace
- force gc
- get object size
- raw monitors
- add to classpath

capabilities

в любой момент

- get bytecodes
- get constant pool
- tag objects
- monitor events
- gc events
- redefine classes

Фичи JVM TI

обязательные

- method / field info
- thread info
- get stack trace
- force gc
- get object size
- raw monitors
- add to classpath

capabilities

в любой момент

- get bytecodes
- get constant pool
- tag objects
- monitor events
- gc events
- redefine classes

OnLoad

- access local variables
- exception events
- method entry/exit events
- field access events
- breakpoints

Фичи JVM TI

обязательные

- method / field info
- thread info
- get stack trace
- force gc
- get object size
- raw monitors
- add to classpath

capabilities

в любой момент

- get bytecodes
- get constant pool
- tag objects
- monitor events
- gc events
- redefine classes

OnLoad

- access local variables
- exception events
- method entry/exit events
- field access events
- breakpoints

libjdwp.so – тоже JVM TI агент

Накладные расходы

пассивные
подключение capability

активные
использование фичи

Накладные расходы

пассивные
подключение capability

активные
использование фичи

- can_generate_exception_events
  все исключения в slow path с деоптимизацией

- can_access_local_variables
  выключение Escape Analysis

Накладные расходы

пассивные
подключение capability

активные
использование фичи

- can_generate_exception_events
  все исключения в slow path с деоптимизацией

- can_access_local_variables
  выключение Escape Analysis

 jdwp агент => оверхед

Накладные расходы

пассивные
подключение capability

активные
использование фичи

- SetBreakpoint
  деоптимизация метода

- MethodEntry/Exit, FieldAccess
  Interpreted-only режим

Несколько агентов

```
java -agentpath:/lib/first.so  -agentpath:/lib/second.so
```

Несколько агентов

```
java -agentpath:/lib/first.so  -agentpath:/lib/second.so
```

- У каждого свой jvmtiEnv
  - свой набор capabilities
  - независимые* events

Несколько агентов

```
java -agentpath:/lib/first.so  -agentpath:/lib/second.so
```

- У каждого свой jvmtiEnv
  - свой набор capabilities
  - независимые* events
- Эксклюзивные capabilities
  - suspend
  - breakpoint_events
  - field_access_events

Несколько агентов: проблемы

```
jvmti->GenerateEvents(JVMTI_EVENT_COMPILED_METHOD_LOAD);
```

- Сгенерировал один => получили все

Несколько агентов: проблемы

```
jvmti->GenerateEvents(JVMTI_EVENT_COMPILED_METHOD_LOAD);
```

- Сгенерировал один => получили все
- Баг JDK-8222072
- Быть готовым к повторным событиям

Demo: не только агенты

Получить место вызова

```
String getLocation() {
    StackTraceElement caller = Thread.currentThread().getStackTrace()[3];
    return caller.getFileName() + ':' + caller.getLineNumber();
}
```

"AsyncTask.java:14"

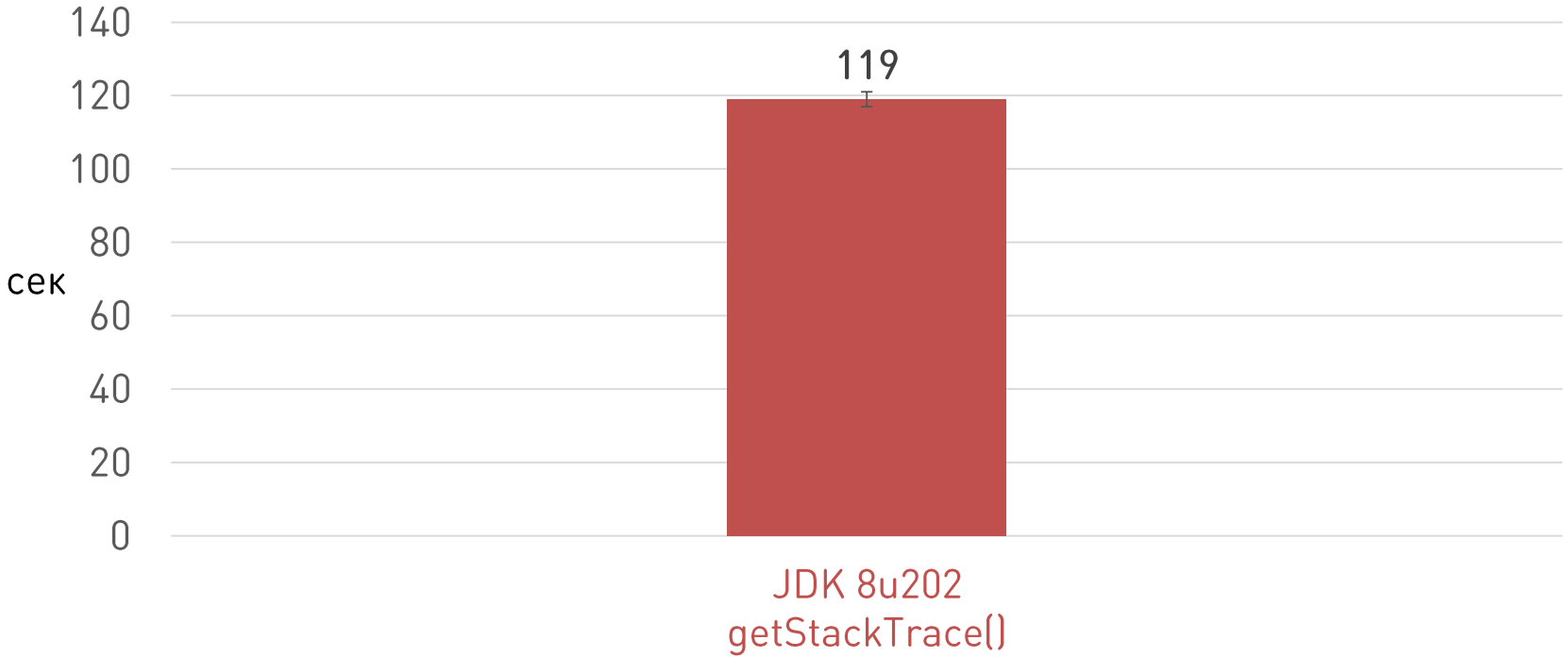
Получить место вызова


```
String getLocation() {
    StackTraceElement caller = Thread.currentThread().getStackTrace()[3];
    return caller.getFileName() + ':' + caller.getLineNumber();
}
```

"AsyncTask.java:14"

https://jug.ru/talks/meetups/everything-you-wanted-to-know-about-stack-traces-and-heap-dumps/

Бенчмарк: 10 млн вызовов



Получить место вызова: Java 9+

```
String getLocation() {
    return StackWalker.getInstance().walk(s -> {
        StackWalker.StackFrame frame = s.skip(3).findFirst().get();
        return frame.getFileName() + ':' + frame.getLineNumber();
    });
}
```

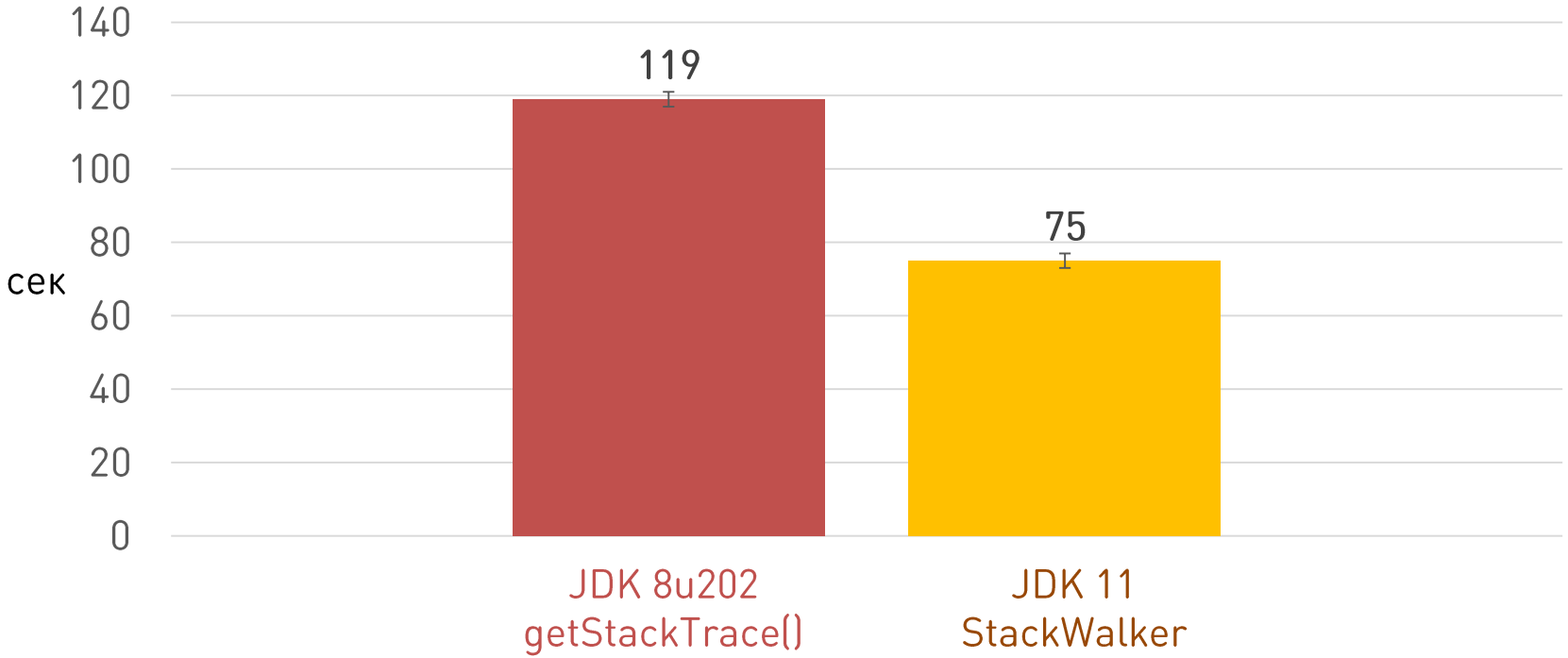
"AsyncTask.java:14"

Получить место вызова: Java 9+

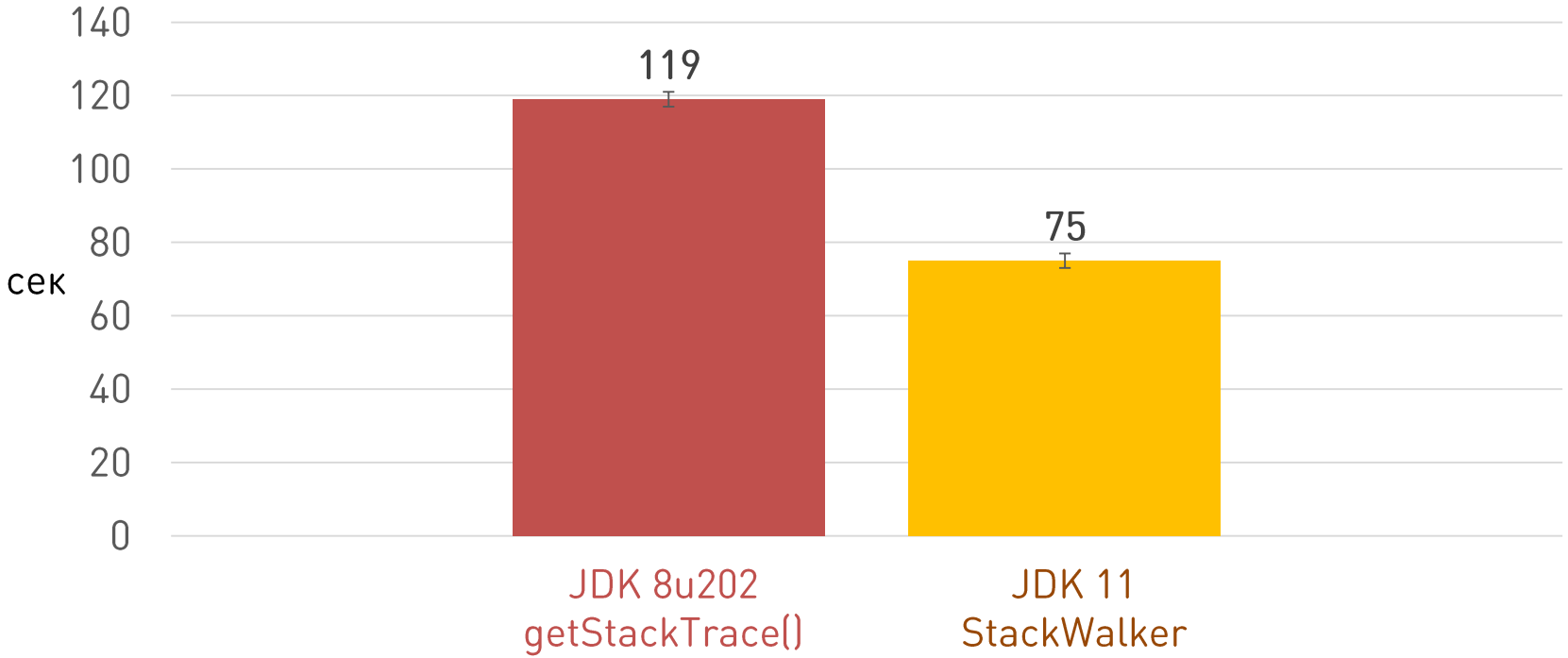
```
String getLocation() {
    return StackWalker.getInstance().walk(s -> {
        StackWalker.StackFrame frame = s.skip(3).findFirst().get();
        return frame.getFileName() + ':' + frame.getLineNumber();
    });
}
```

"AsyncTask.java:14"

Бенчмарк: 10 млн вызовов



Бенчмарк: 10 млн вызовов



Медленный StackWalker
https://bugs.openjdk.java.net/browse/JDK-8151751

JVM TI в помощь

```
GetStackTrace(jthread thread,
              jint start_depth,        = 3
              jint max_frame_count,    = 1
              jvmtiFrameInfo* frame_buffer,
              jint* count_ptr)
```

JVM TI в помощь

```
GetStackTrace(jthread thread,
              jint start_depth,       = 3
              jint max_frame_count,   = 1
              jvmtiFrameInfo* frame_buffer,
              jint* count_ptr)
```

JNI библиотека + JVM TI функции = 

```
public class StackFrame {

    public static native String getLocation(int depth);

    static {
        System.loadLibrary("stackframe");
    }
}
```

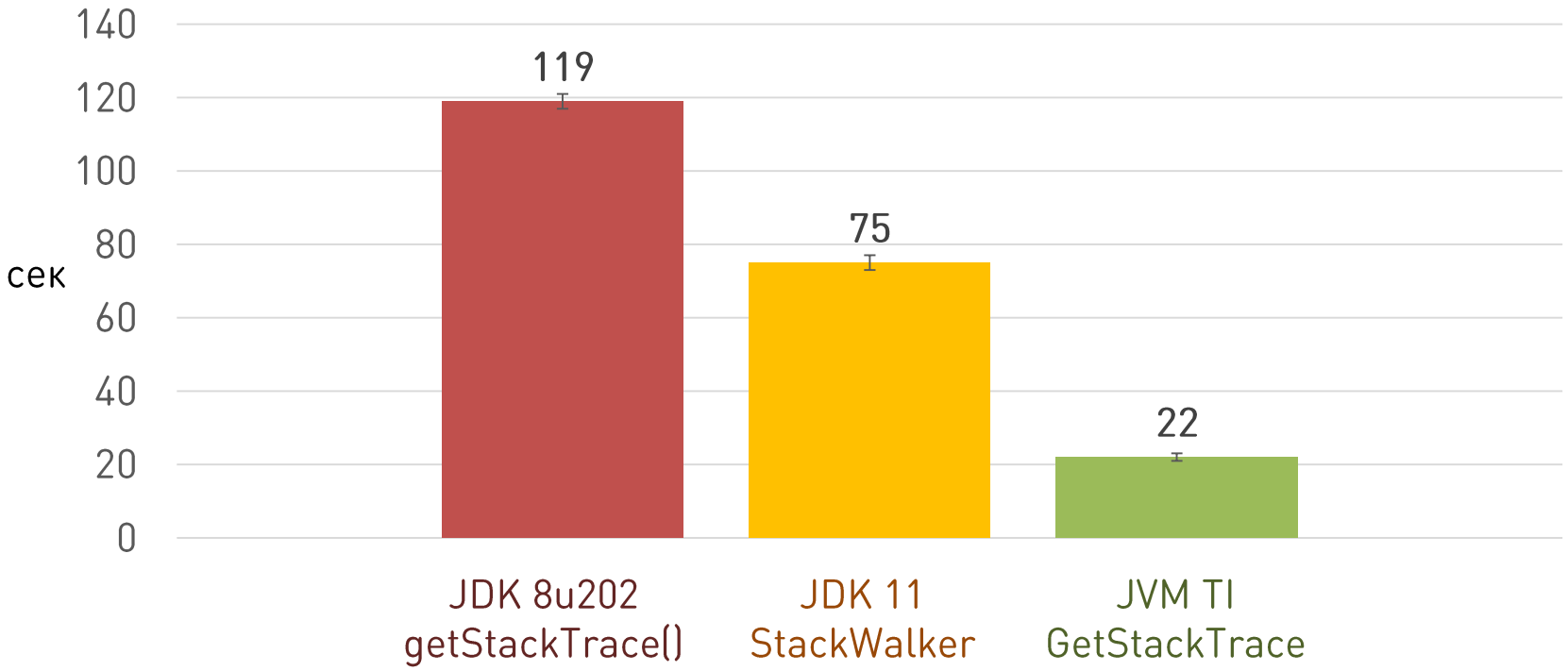
JNI библиотека + JVM TI функции = 🔥

```
jint JNI_OnLoad(JavaVM* vm, void* reserved) {
    vm->GetEnv((void**) &jvmti, JVMTI_VERSION_1_0);
    ...
}
```

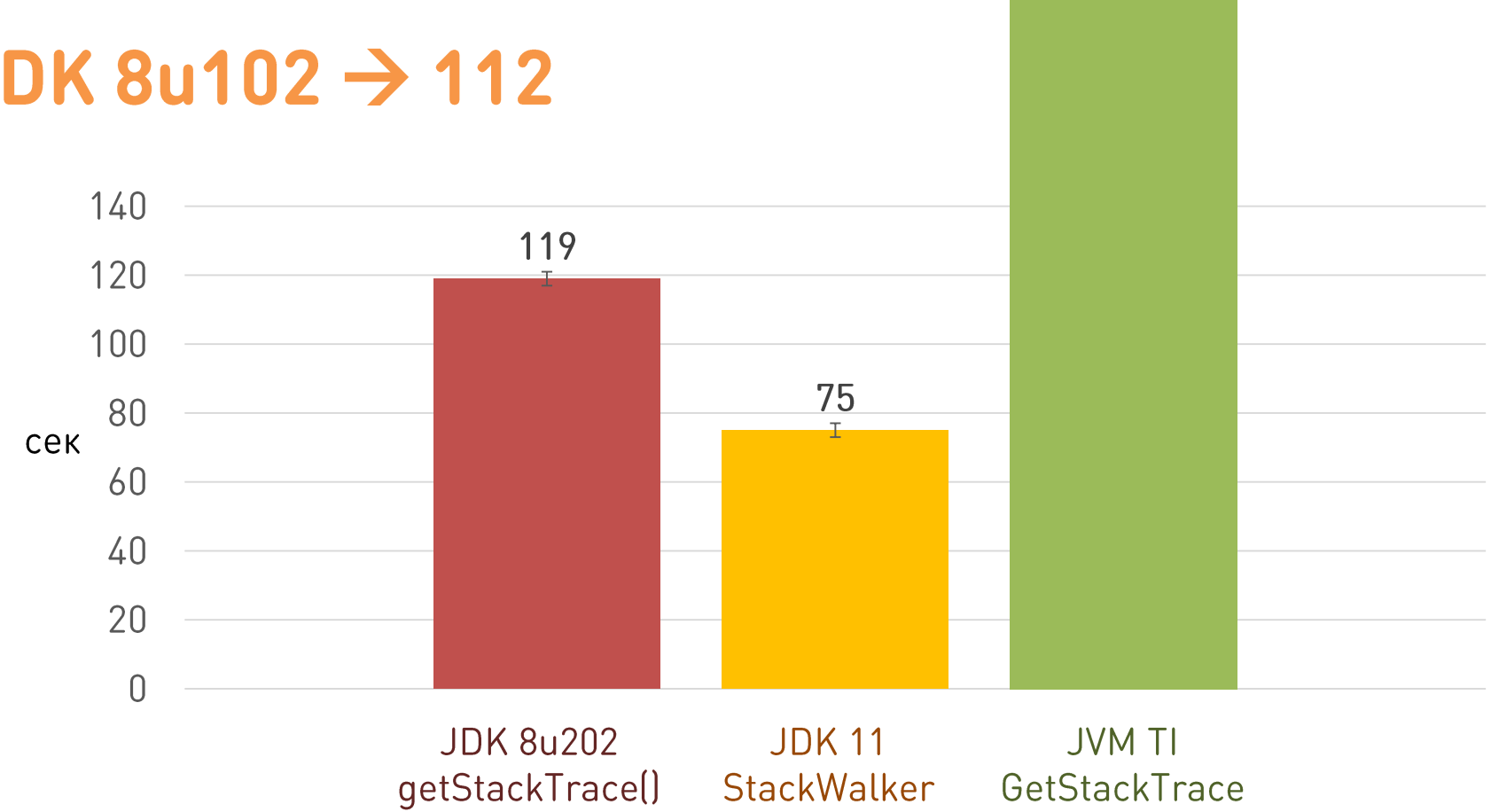
JNI библиотека + JVM TI функции = 

```
jint JNI_OnLoad(JavaVM* vm, void* reserved) {
    vm->GetEnv((void**) &jvmti, JVMTI_VERSION_1_0);
    ...
}

JNIEXPORT jstring JNICALL
Java_StackFrame_getLocation(JNIEnv* env, jclass unused, jint depth) {
    jvmtiFrameInfo frame;
    jint count;
    jvmti->GetStackTrace(NULL, depth, 1, &frame, &count);
```

Бенчмарк: 10 млн вызовов



JDK 8u102 → 112



Ме-е-едленный JVM TI

- Баг JDK-8185348
  - GetMethodName
  - GetMethodDeclaringClass
  - …

Ме-е-едленный JVM TI

- Баг JDK-8185348
  - GetMethodName
  - GetMethodDeclaringClass
  - …

- jvmtiEnter.xsl

Ме-е-едленный JVM TI

- Баг JDK-8185348
  - GetMethodName
  - GetMethodDeclaringClass
  - …
- jvmtiEnter.xsl
- Решение – кешировать результаты
  - map<jmethodID, …>

Способы подключения JVM TI

- Из Java кода
  - System.loadLibrary

Способы подключения JVM TI

- Из Java кода
  - System.loadLibrary
- При запуске JVM
  - agentpath

Способы подключения JVM TI

- Из Java кода
  - System.loadLibrary
- При запуске JVM
  - agentpath
- Dynamic Attach

Dynamic Attach

- Загрузка агента в рантайме

Dynamic Attach

- Загрузка агента в рантайме
- jcmd JDK 9+

```
jcmd <pid> JVMTI.agent_load /path/to/agent.so [arguments]
```

Dynamic Attach

- Загрузка агента в рантайме
- jcmd JDK 9+

```
jcmd <pid> JVMTI.agent_load /path/to/agent.so [arguments]
```

- jattach

  github.com/apangin/jattach

  Пример: async-profiler

Поддержка Dynamic Attach

```
JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM* vm, char* options, void* reserved)


JNIEXPORT jint JNICALL
Agent_OnAttach(JavaVM* vm, char* options, void* reserved)
```

Поддержка Dynamic Attach

```
JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM* vm, char* options, void* reserved)


JNIEXPORT jint JNICALL
Agent_OnAttach(JavaVM* vm, char* options, void* reserved)
```

без OnLoad capabilities

может вызываться повторно

Demo: attach

Что нового в Java 9

Что нового в Java 9

Java Module System

Приватный API

```
ByteBuffer buf = ByteBuffer.allocateDirect(1024);

((sun.nio.ch.DirectBuffer) buf).cleaner().clean();
```

Приватный API

```
ByteBuffer buf = ByteBuffer.allocateDirect(1024);

((sun.nio.ch.DirectBuffer) buf).cleaner().clean();
```

```
Exception in thread "main" java.lang.IllegalAccessError:
class agent.demo6.PrivateApi (in unnamed module @0x3ac3fd8b)
cannot access class jdk.internal.ref.Cleaner (in module java.base)
```

Приватный API

```
Field f = FileDescriptor.class.getDeclaredField("fd");

f.setAccessible(true);
```

Приватный API

```
Field f = FileDescriptor.class.getDeclaredField("fd");

f.setAccessible(true);
```

```
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by agent.demo6.Reflection
         to field java.lang.String.value
WARNING: Please consider reporting this to the maintainers of
         agent.demo6.Reflection
```

Cassandra VS модули

```
--add-exports java.base/jdk.internal.misc=ALL-UNNAMED
--add-opens java.base/jdk.internal.module=ALL-UNNAMED
--add-exports java.base/jdk.internal.ref=ALL-UNNAMED
--add-exports java.base/sun.nio.ch=ALL-UNNAMED
--add-exports java.management.rmi/com.sun.jmx.remote.internal.rmi=ALL-UNNAMED
--add-exports java.rmi/sun.rmi.registry=ALL-UNNAMED
--add-exports java.rmi/sun.rmi.server=ALL-UNNAMED
--add-opens jdk.management/com.sun.management.internal=ALL-UNNAMED
```

https://github.com/apache/cassandra/blob/trunk/conf/jvm11-server.options

Новое в JVM TI 9

- GetAllModules

- AddModuleReads
- AddModuleExports
- AddModuleOpens

Новое в JVM TI 9

- GetAllModules
- AddModuleReads
- AddModuleExports
- AddModuleOpens

1. Регистрируем VMInit
2. Получаем все модули
3. Берём список packages
4. Открываем всё
5. PROFIT!

Demo

https://github.com/odnoklassniki/jvmti-tools/antimodule

Полезности в Java 11

Полезности в Java 11

JEP 331: Легковесное профилирование аллокаций

```
SampledObjectAlloc(jvmtiEnv* jvmti,
                   JNIEnv* env,
                   jthread thread,
                   jobject object,
                   jclass object_klass,
                   jlong size)
```

Полезности в Java 11

JEP 331: Легковесное профилирование аллокаций

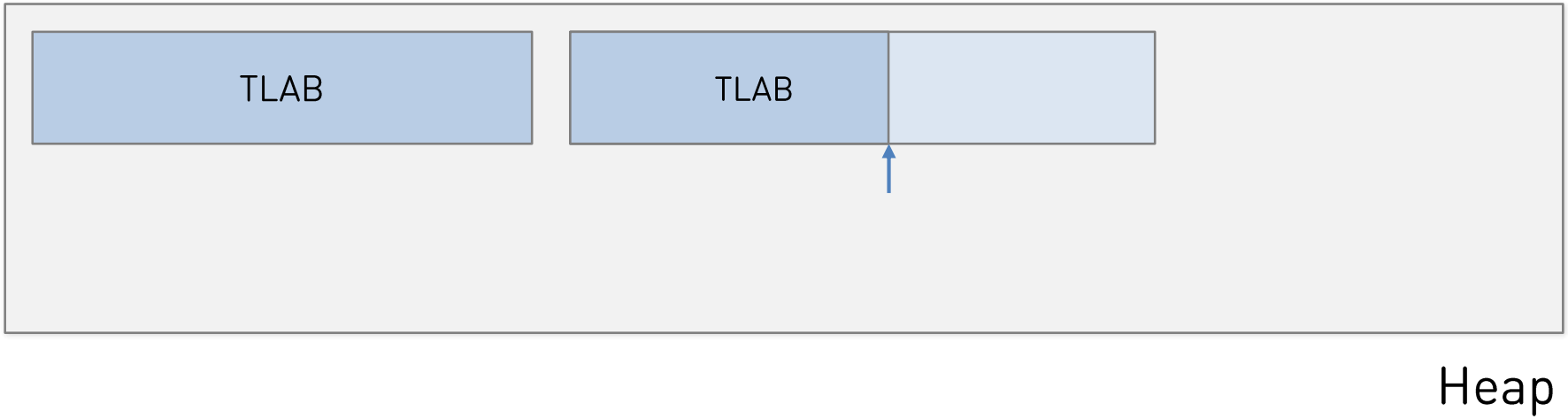
```
SampledObjectAlloc(jvmtiEnv* jvmti,
                   JNIEnv* env,
                   jthread thread,
                   jobject object,
                   jclass object_klass,
                   jlong size)

jvmti->SetHeapSamplingInterval(jint sampling_interval)
```

Профилирование аллокаций

- Инструментирование – медленно

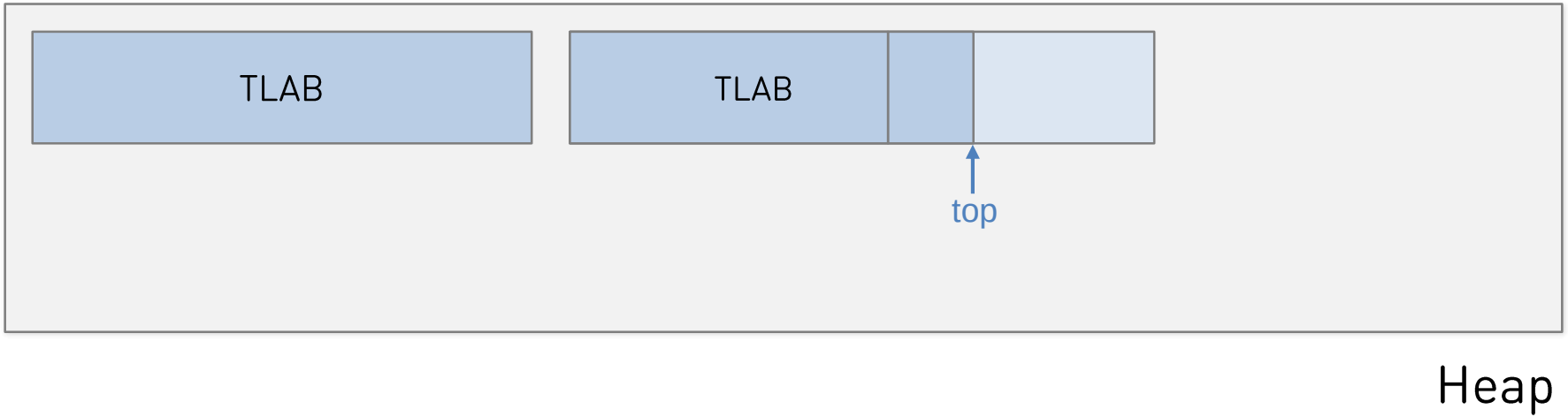
Профилирование аллокаций

- Инструментирование – медленно
- Сэмплирование



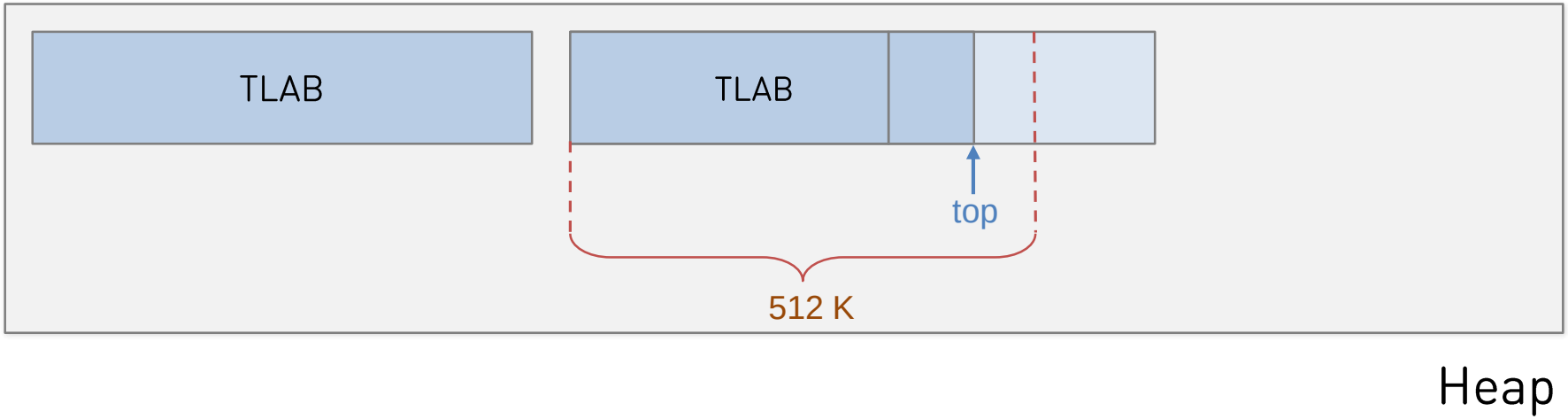
Профилирование аллокаций

- Инструментирование – медленно
- Сэмплирование



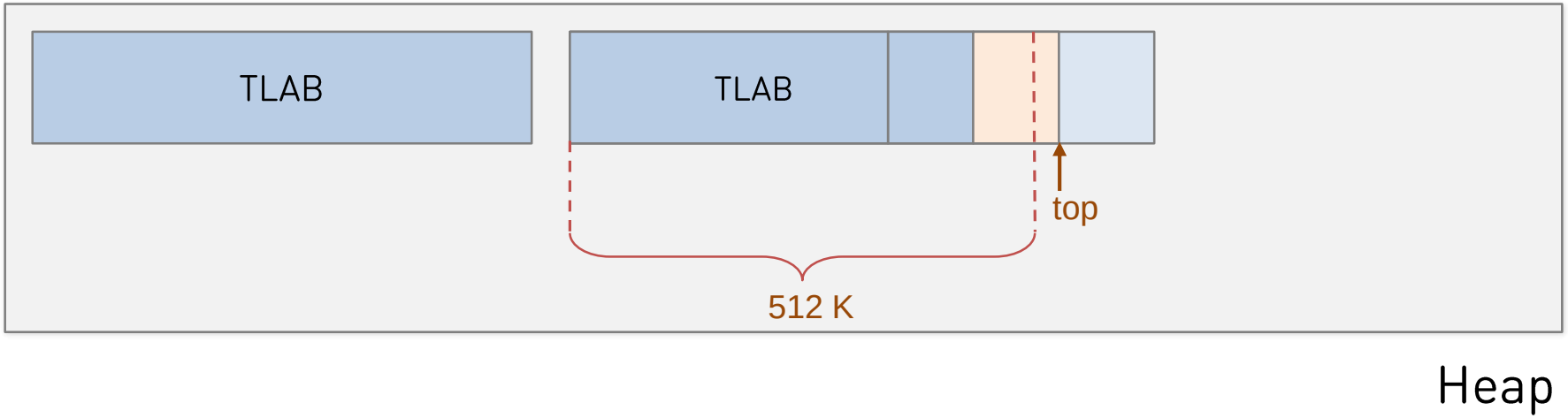
Профилирование аллокаций

- Инструментирование – медленно
- Сэмплирование



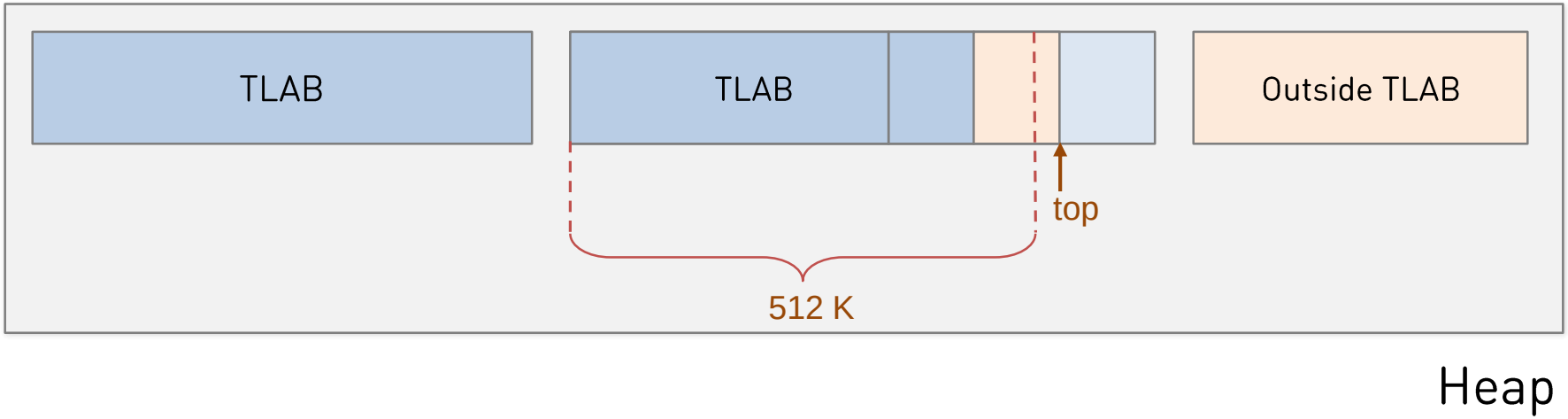
Профилирование аллокаций

- Инструментирование – медленно
- Сэмплирование



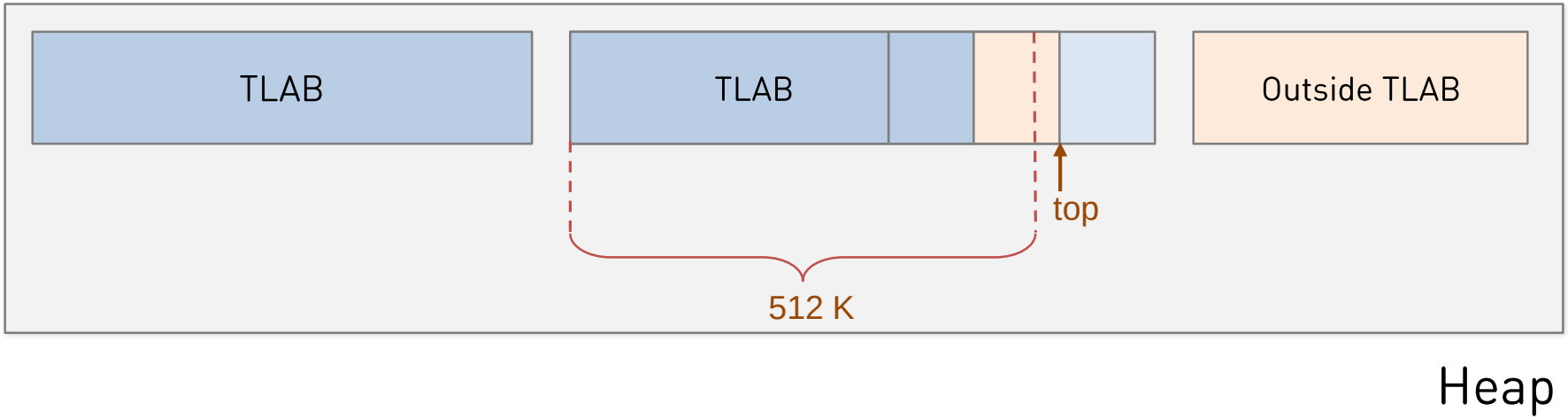
Профилирование аллокаций

- Инструментирование – медленно
- Сэмплирование



Профилирование аллокаций

- Инструментирование – медленно
- Сэмплирование < 5% overhead



Профилирование аллокаций

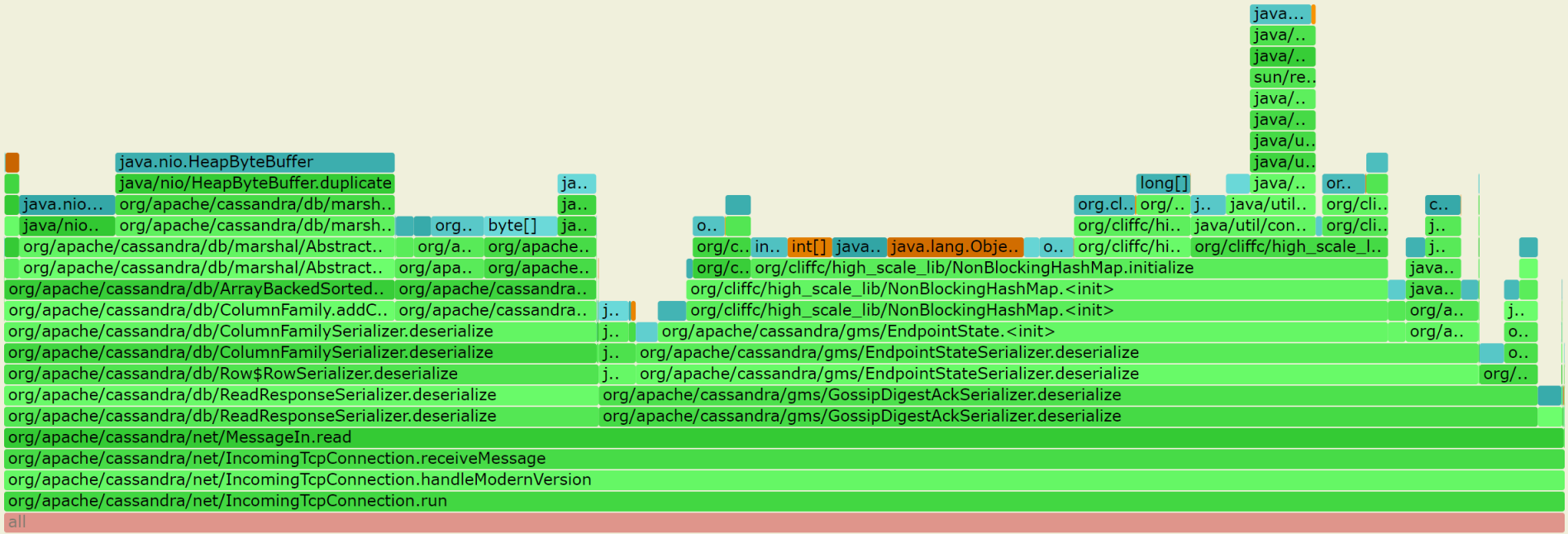
- Flight Recorder в JDK 7+
  github.com/jvm-profiling-tools/async-profiler

Профилирование аллокаций

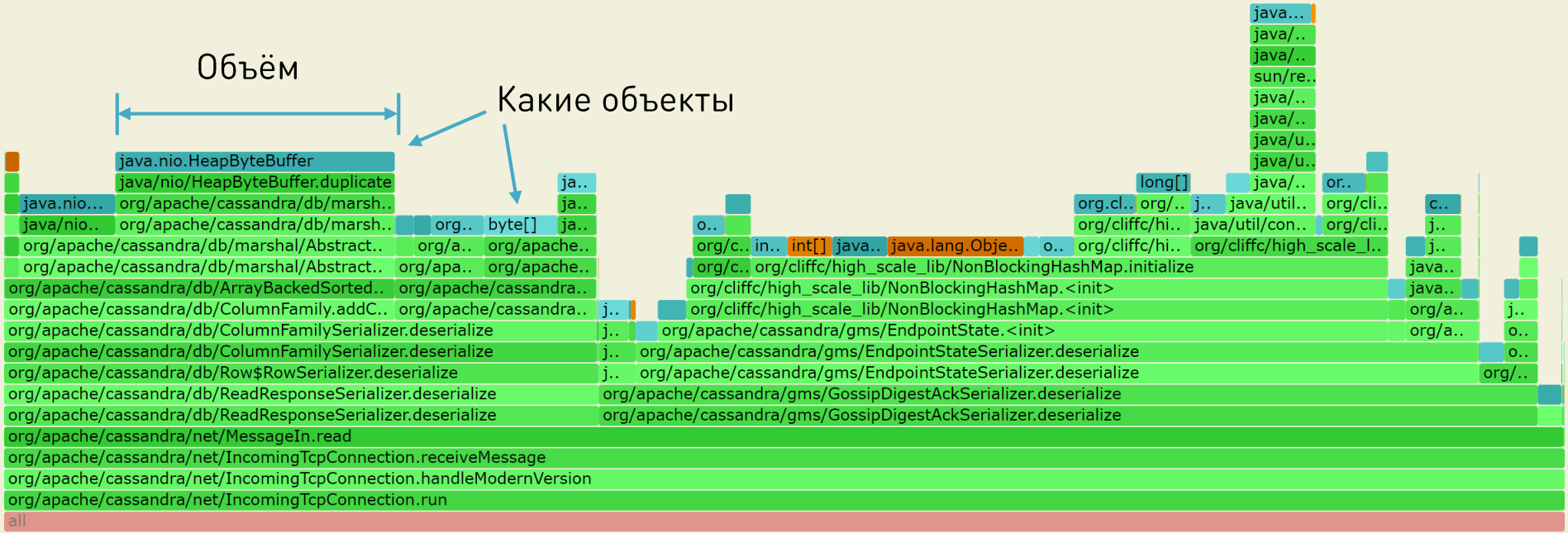
- Flight Recorder в JDK 7+
  github.com/jvm-profiling-tools/async-profiler

- Публичный API в JDK 11
  github.com/odnoklassniki/jvmti-tools/heapsampler

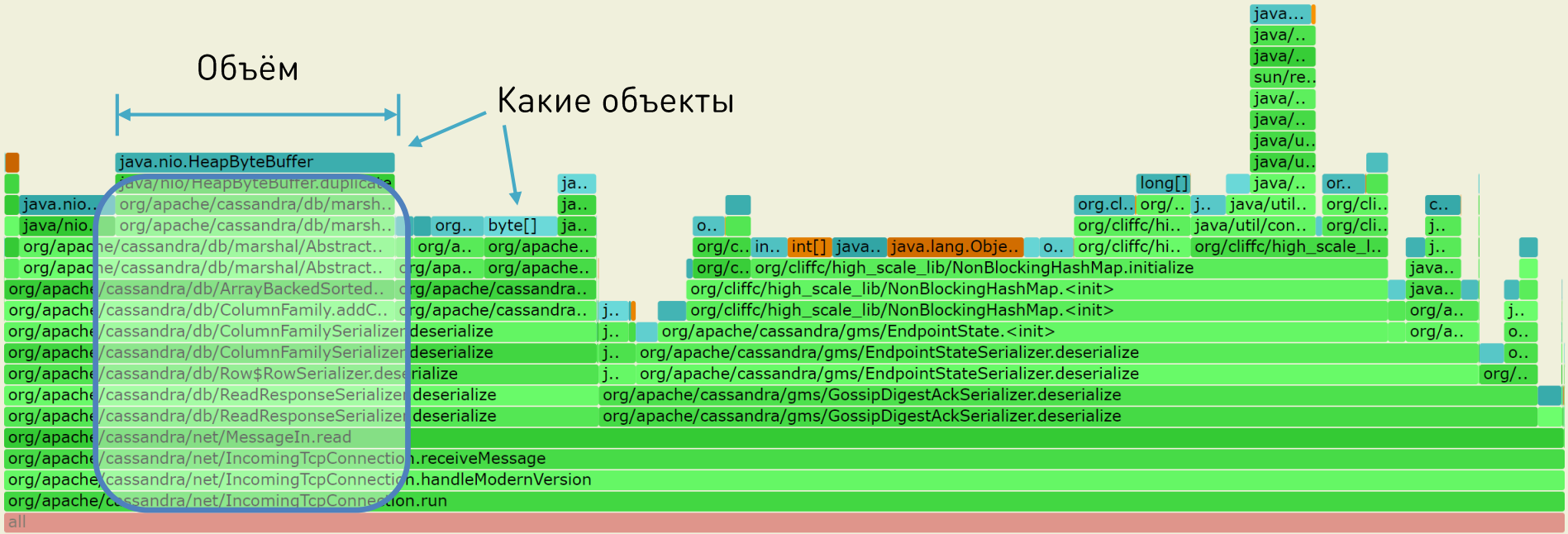
Allocation Flame Graph



Allocation Flame Graph



Allocation Flame Graph



Выводы

- JVM TI – стандартный API для взаимодействия с JVM

Выводы

- JVM TI – стандартный API для взаимодействия с JVM
  - из Java приложения
  - в качестве агента
  - подключается динамически

Выводы

- JVM TI – стандартный API для взаимодействия с JVM
  - из Java приложения
  - в качестве агента
  - подключается динамически
- То, что нужно для плагина!

Примеры плагинов

github.com/odnoklassniki/jvmti-tools

- richNPE
- vmtrace
- stackframe
- antimodule
- heapsampler

andrey.pangin@corp.mail.ru

https://v.ok.ru/vacancies.html