Как сделать профилировщик

из палок... и других подручных средств

@antonarhipov

JPoint 2016, Москва

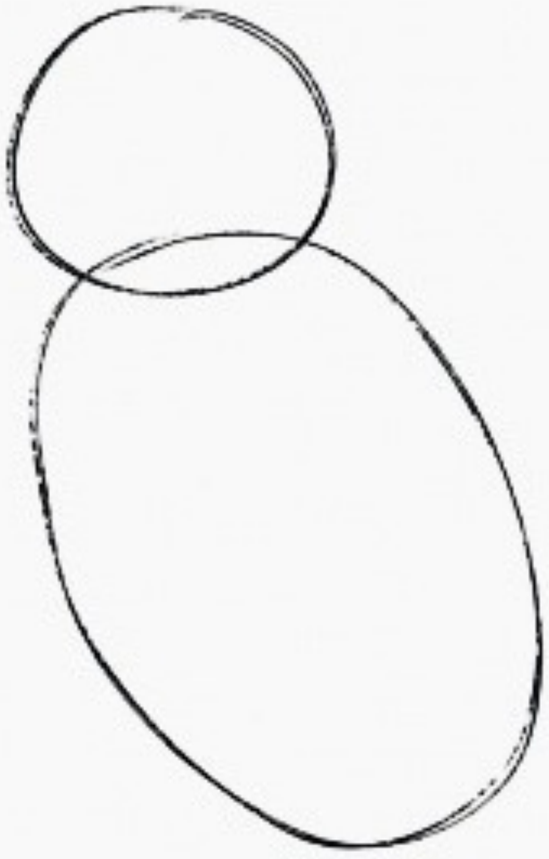
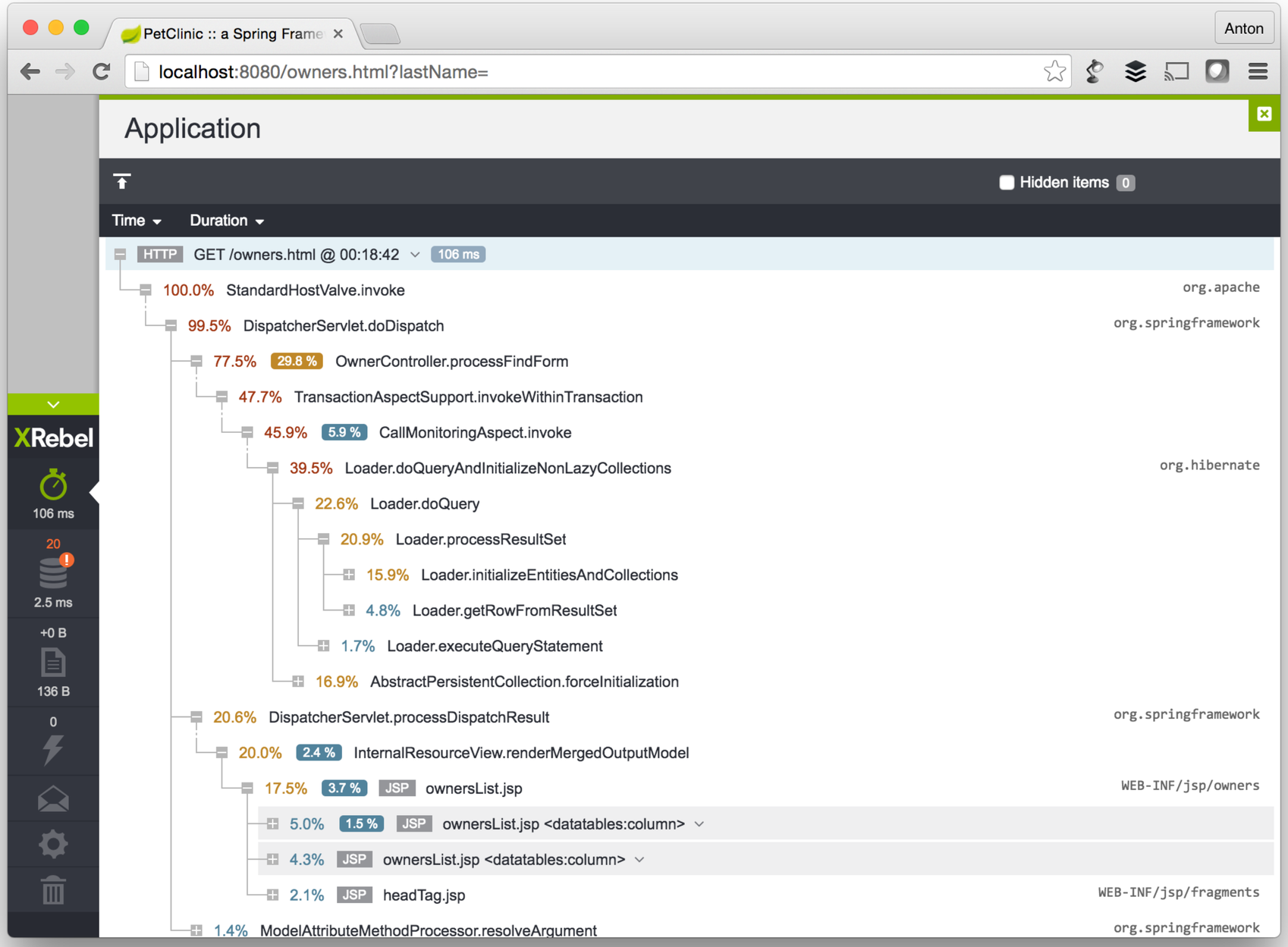


Fig 1. Draw two circles

Fig 2. Draw the rest of the damn Owl


PetClinic :: a Spring Frame
Anton
localhost:8080/owners.html?lastName=
Application
Hidden items 0
Time
Duration
HTTP GET /owners.html @ 00:18:42 106 ms
100.0% StandardHostValve.invoke org.apache
99.5% DispatcherServlet.doDispatch org.springframework
77.5% 29.8 % OwnerController.processFindForm
47.7% TransactionAspectSupport.invokeWithinTransaction
45.9% 5.9 % CallMonitoringAspect.invoke
39.5% Loader.doQueryAndInitializeNonLazyCollections org.hibernate
22.6% Loader.doQuery
20.9% Loader.processResultSet
15.9% Loader.initializeEntitiesAndCollections
4.8% Loader.getRowFromResultSet
1.7% Loader.executeQueryStatement
16.9% AbstractPersistentCollection.forceInitialization
20.6% DispatcherServlet.processDispatchResult org.springframework
20.0% 2.4 % InternalResourceView.renderMergedOutputModel
17.5% 3.7 % JSP ownersList.jsp WEB-INF/jsp/owners
5.0% 1.5 % JSP ownersList.jsp <datatables:column>
4.3% JSP ownersList.jsp <datatables:column>
2.1% JSP headTag.jsp WEB-INF/jsp/fragments
1.4% ModelAttributeMethodProcessor.resolveArgument org.springframework
XRebel
106 ms
20
2.5 ms
+0 B
136 B
0

История одного проекта

2005

Batch / ETL

Java 1.4 HP-UX

Исходные данные : ~10k записей

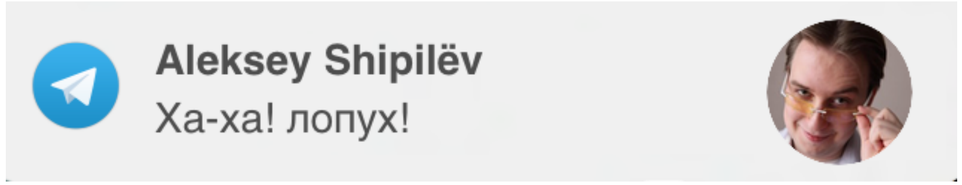
Результат: ~20M записей

j.u.Date / j.u.Calendar

Первый запуск: 25 ч :-(

После оптимизаций: 10 ч

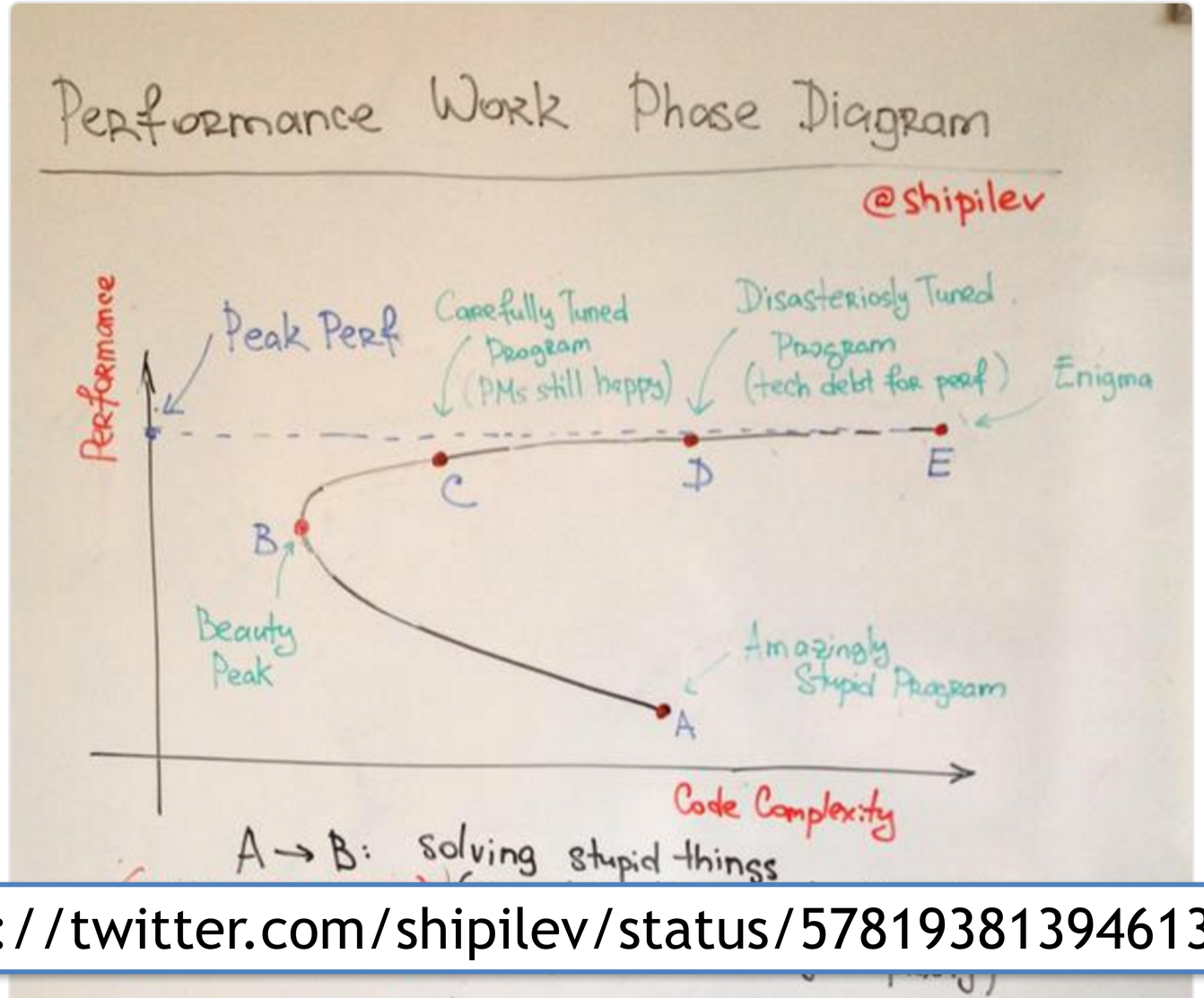
После изменения дизайна: 30 мин :-)



Тот самый случай когда выглядишь умным, но чувствуешь себя полным ДУРАКОМ

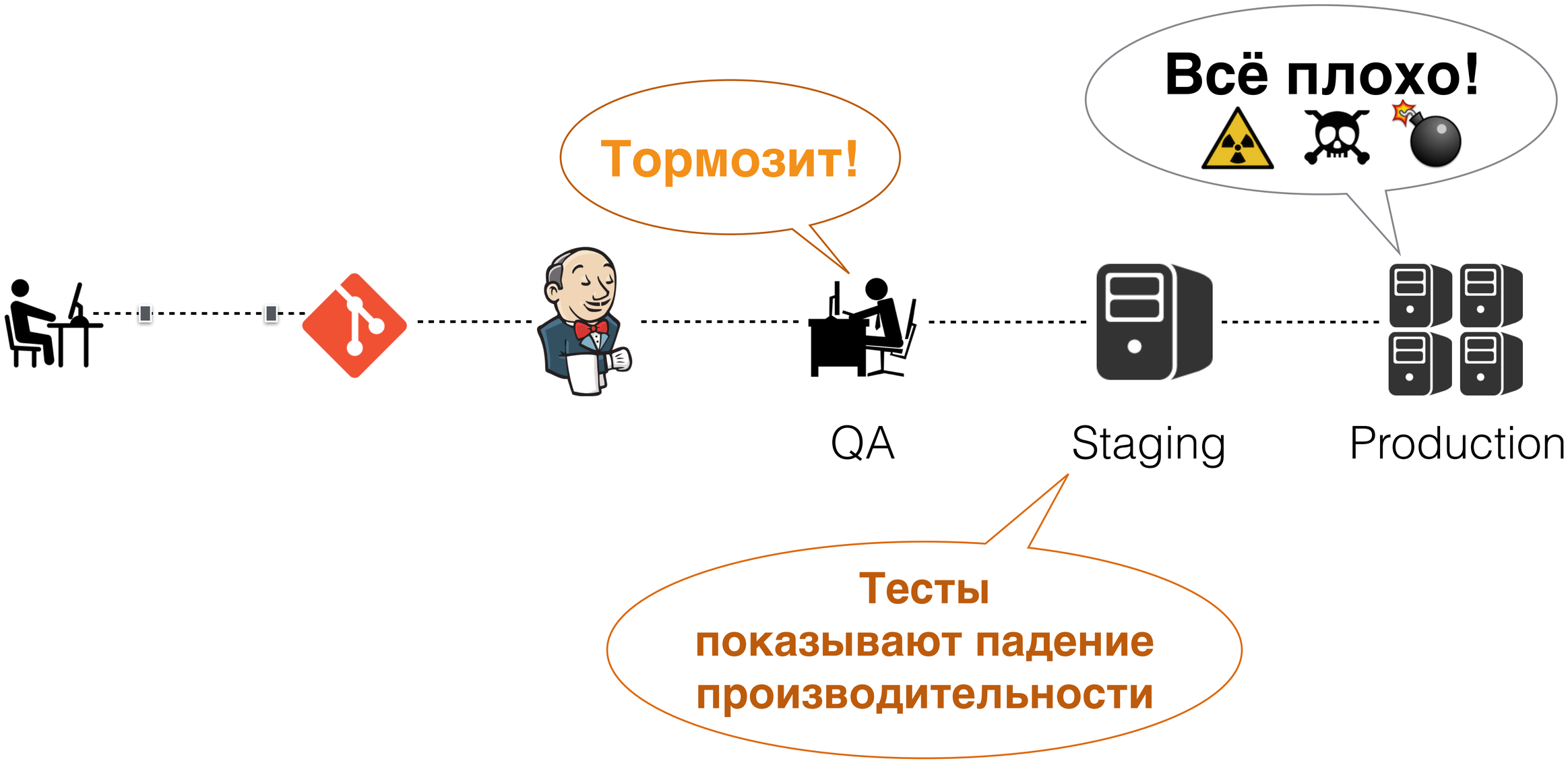
Aleksey Shipilëv
@shipilev

I relax by drawing stuff on my large whiteboard. Here's "Perf Work Phase Diagram" for you.



https://twitter.com/shipilev/status/578193813946134529

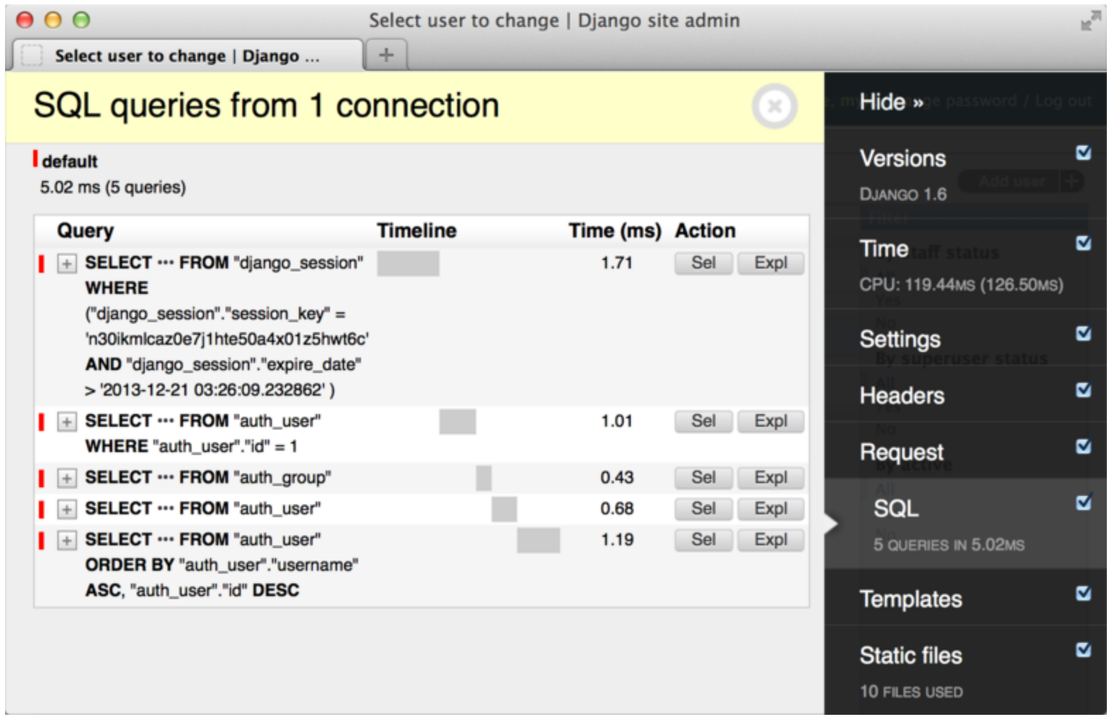
Как и когда вы используете профилировщик?


Всё плохо!
Тормозит!
QA
Staging
Production
Тесты
показывают падение
производительности


REACTIVE
PROACTIVE

“Проактивные” инструменты

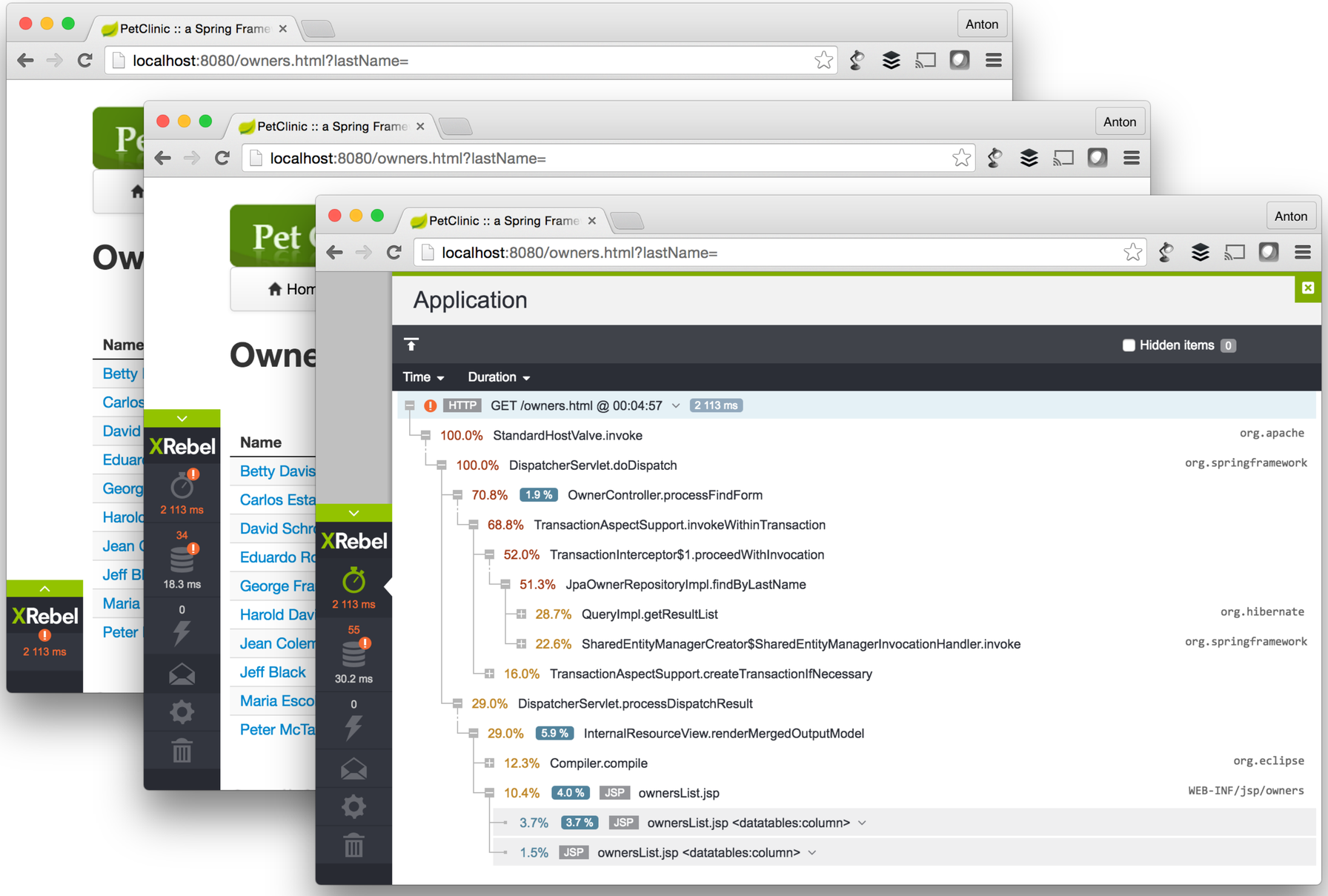
- Chrome DevTools
- FireBug
- YSlow
- …
- **XRebel**



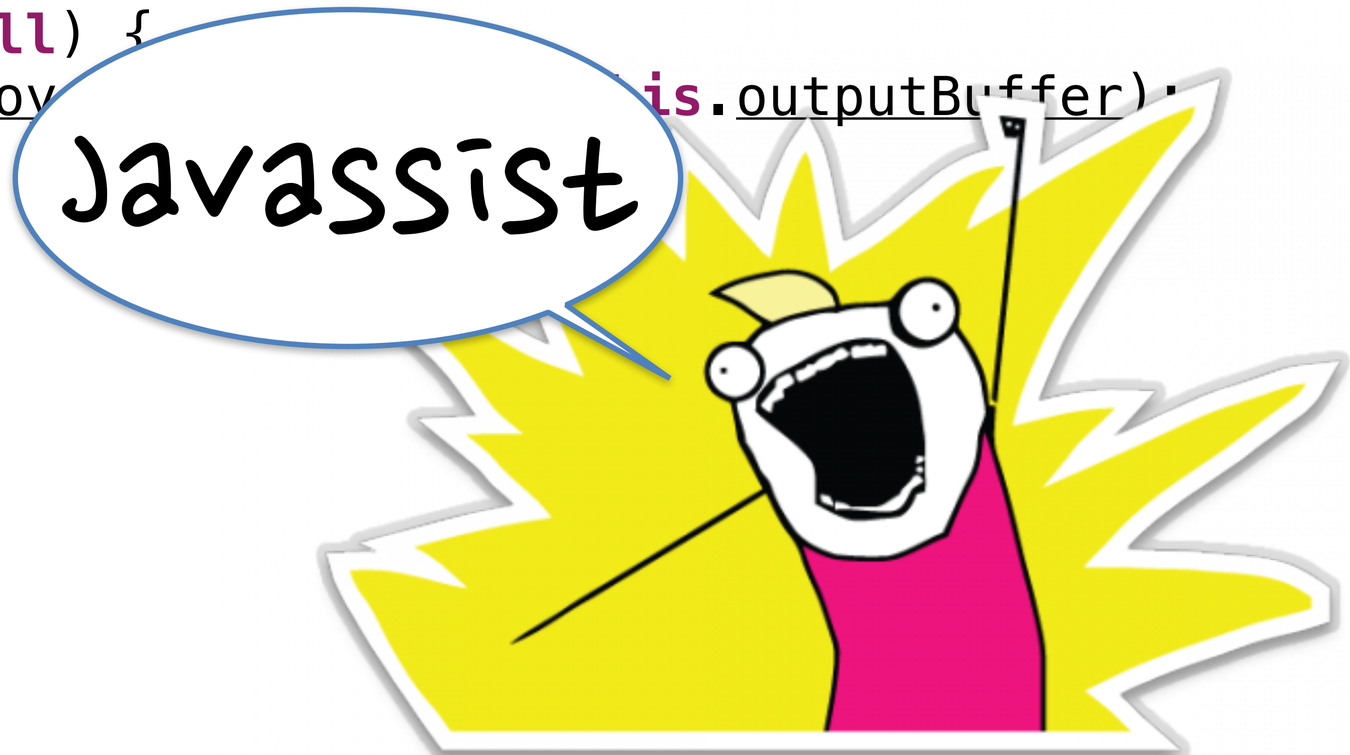
Этюды профайлероводства

- Пользовательский интерфейс
- Профилирование приложения
- Трассировка внешних вызовов
- Трассировка потоков
- Распределённые приложения


UI & UX


PetClinic :: a Spring Frame
Anton
localhost:8080/owners.html?lastName=
Application
Hidden items 0
Time
Duration
HTTP GET /owners.html @ 00:04:57 2 113 ms
100.0% StandardHostValve.invoke org.apache
100.0% DispatcherServlet.doDispatch org.springframework
70.8% 1.9 % OwnerController.processFindForm
68.8% TransactionAspectSupport.invokeWithinTransaction
52.0% TransactionInterceptor$1.proceedWithInvocation
51.3% JpaOwnerRepositoryImpl.findByLastName
28.7% QueryImpl.getResultList org.hibernate
22.6% SharedEntityManagerCreator$SharedEntityManagerInvocationHandler.invoke org.springframework
16.0% TransactionAspectSupport.createTransactionIfNecessary
29.0% DispatcherServlet.processDispatchResult
29.0% 5.9 % InternalResourceView.renderMergedOutputModel
12.3% Compiler.compile org.eclipse
10.4% 4.0 % JSP ownersList.jsp WEB-INF/jsp/owners
3.7% 3.7 % JSP ownersList.jsp <datatables:column>
1.5% JSP ownersList.jsp <datatables:column>
XRebel
2 113 ms
34
18.3 ms
0
55
30.2 ms

```
package org.apache.catalina.connector;

public class Response implements HttpServletResponse {

  public ServletOutputStream getOutputStream() throws IOException {
    if (this.usingWriter) {
      throw new IllegalStateException(…));
    }
    this.usingOutputStream = true;
    if (this.outputStream == null) {
      this.outputStream = new Cov                    is.outputBuffer);
    }
    return this.outputStream;
  }
```

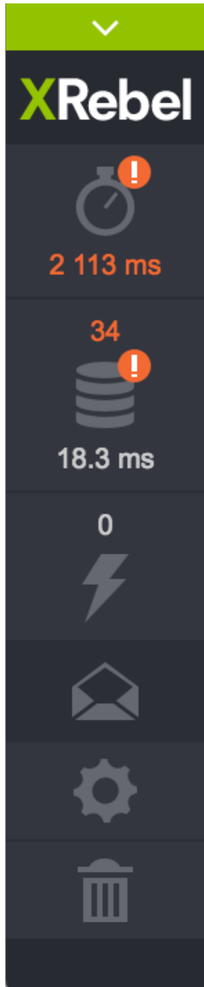


```java
package org.apache.catalina.connector;

public class Response implements HttpServletResponse {

  public ServletOutputStream getOutputStream() throws IOException {
    if (this.usingWriter) {
      throw new IllegalStateException(…));
    }
    this.usingOutputStream = true;
    if (this.outputStream == null) {
      this.outputStream = new CoyoteOutputStream(this.outputBuffer);
    }
    return this.outputStream;
```

```java
ctClass.getDeclaredMethod("getOutputStream").insertAfter(
  "if (" + _interceptor + " != null) {" +
  "  $_ = (ServletOutputStream)" + _interceptor + ".getOutputStream($_);" +
  "}");
```

Javassist

```
package org.apache.catalina.connector;

public class Response implements HttpServletResponse {

  public ServletOutputStream getOutputStream() throws IOException {
    if (this.usingWriter) {
      throw new IllegalStateException(…));
    }
    this.usingOutputStream = true;
    if (this.outputStream == null) {
      this.outputStream = new CoyoteOutputStream(this.outputBuffer);
    }
    if (_interceptor != null) {
      this.outputStream = (ServletOutputStream)_interceptor
              .getOutputStream(this.outputStream);
    }
    return this.outputStream;
  }
```



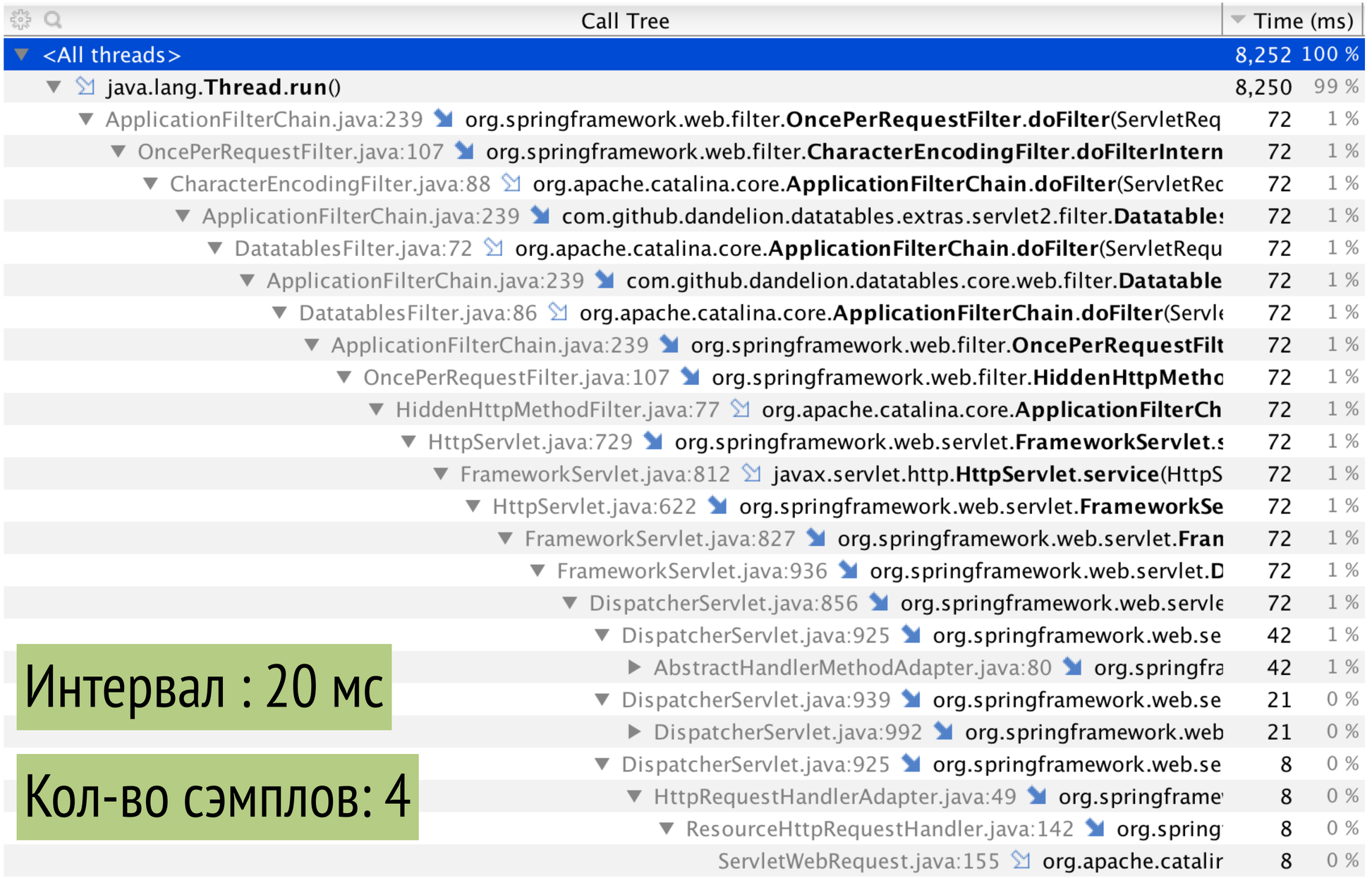
Куда уходит время?

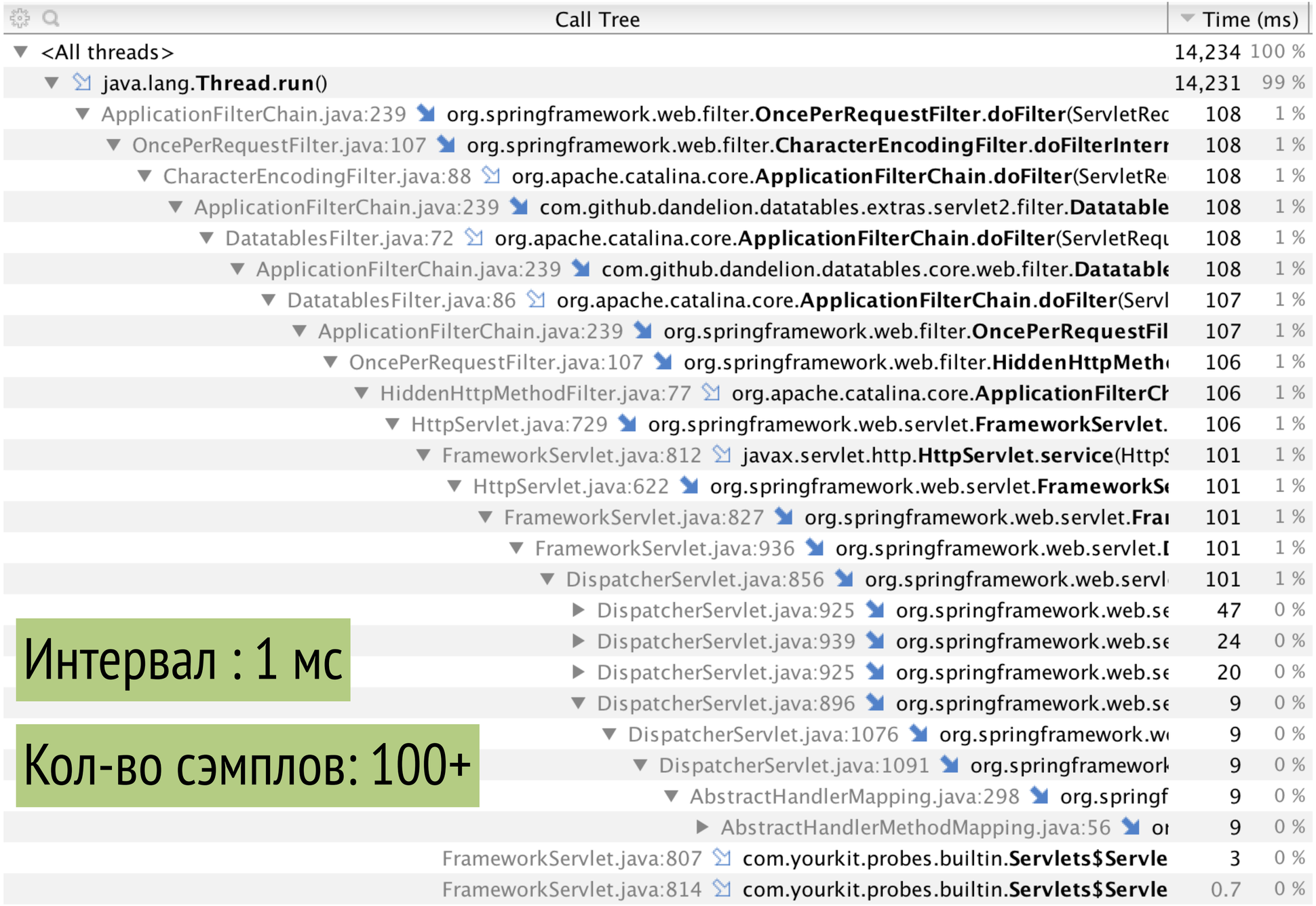
Сэмплирование

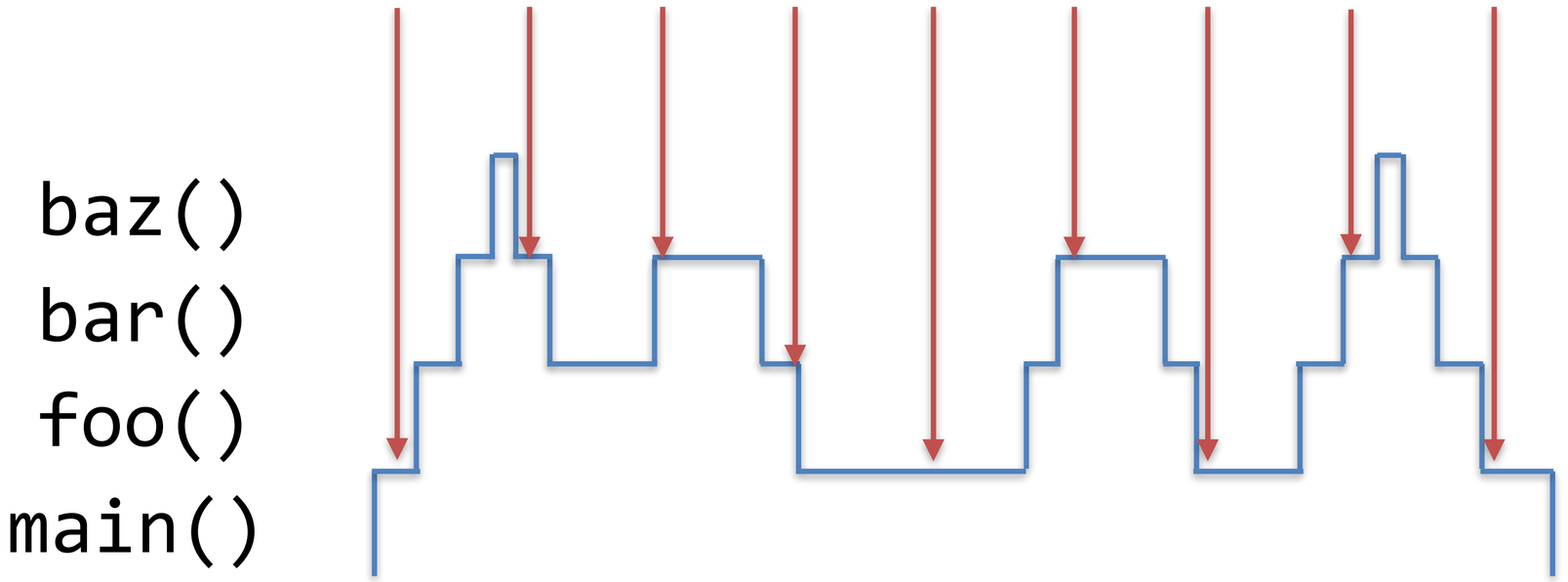
Windows: **Ctrl+Break**

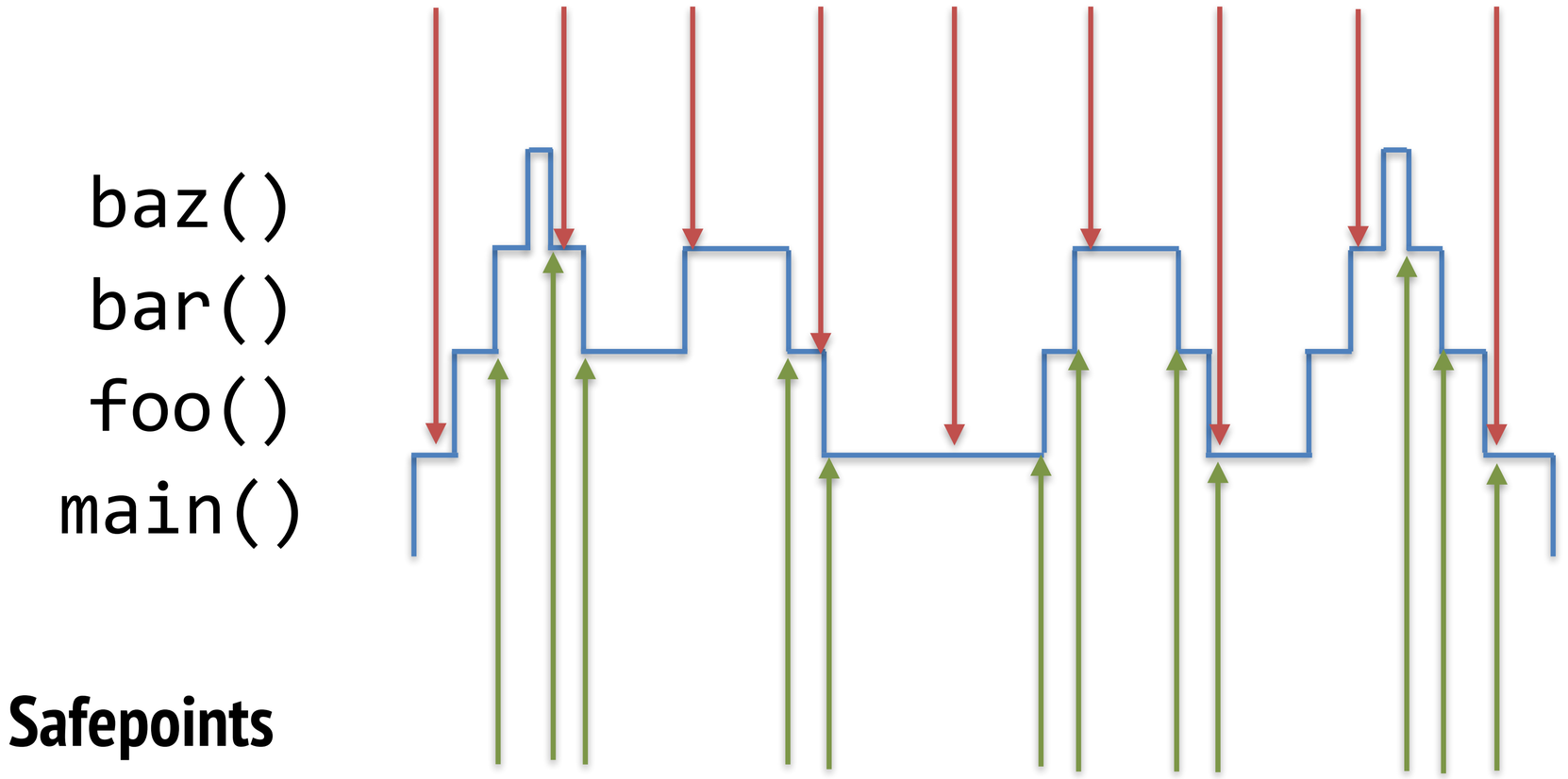
*nix: **kill -3 PID**

```
"http-nio-8080-exec-1" #40 daemon prio=5 os_prio=31 tid=0x00007fd7057ed800 nid=0x7313 runnable
   java.lang.Thread.State: RUNNABLE
       at o.s.s.p.w.OwnerController.processFindForm(OwnerController.java:89)
       at s.r.NativeMethodAccessorImpl.invoke0(Native Method)
       at s.r.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
       at s.r.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
       at j.l.r.Method.invoke(Method.java:497)
```


Call Tree
Time (ms)
<All threads> 8,252 100 %
java.lang.Thread.run() 8,250 99 %
ApplicationFilterChain.java:239 org.springframework.web.filter.OncePerRequestFilter.doFilter(ServletReq 72 1 %
OncePerRequestFilter.java:107 org.springframework.web.filter.CharacterEncodingFilter.doFilterIntern 72 1 %
CharacterEncodingFilter.java:88 org.apache.catalina.core.ApplicationFilterChain.doFilter(ServletRec 72 1 %
ApplicationFilterChain.java:239 com.github.dandelion.datatables.extras.servlet2.filter.Datatable 72 1 %
DatatablesFilter.java:72 org.apache.catalina.core.ApplicationFilterChain.doFilter(ServletRequ 72 1 %
ApplicationFilterChain.java:239 com.github.dandelion.datatables.core.web.filter.Datatable 72 1 %
DatatablesFilter.java:86 org.apache.catalina.core.ApplicationFilterChain.doFilter(Servl 72 1 %
ApplicationFilterChain.java:239 org.springframework.web.filter.OncePerRequestFilt 72 1 %
OncePerRequestFilter.java:107 org.springframework.web.filter.HiddenHttpMetho 72 1 %
HiddenHttpMethodFilter.java:77 org.apache.catalina.core.ApplicationFilterCh 72 1 %
HttpServlet.java:729 org.springframework.web.servlet.FrameworkServlet.s 72 1 %
FrameworkServlet.java:812 javax.servlet.http.HttpServlet.service(HttpS 72 1 %
HttpServlet.java:622 org.springframework.web.servlet.FrameworkSe 72 1 %
FrameworkServlet.java:827 org.springframework.web.servlet.Fran 72 1 %
FrameworkServlet.java:936 org.springframework.web.servlet.D 72 1 %
DispatcherServlet.java:856 org.springframework.web.servle 72 1 %
DispatcherServlet.java:925 org.springframework.web.se 42 1 %
AbstractHandlerMethodAdapter.java:80 org.springfra 42 1 %
DispatcherServlet.java:939 org.springframework.web.se 21 0 %
DispatcherServlet.java:992 org.springframework.web 21 0 %
DispatcherServlet.java:925 org.springframework.web.se 8 0 %
HttpRequestHandlerAdapter.java:49 org.springframe 8 0 %
ResourceHttpRequestHandler.java:142 org.spring 8 0 %
ServletWebRequest.java:155 org.apache.catalir 8 0 %
Интервал : 20 мс
Кол-во сэмплов: 4

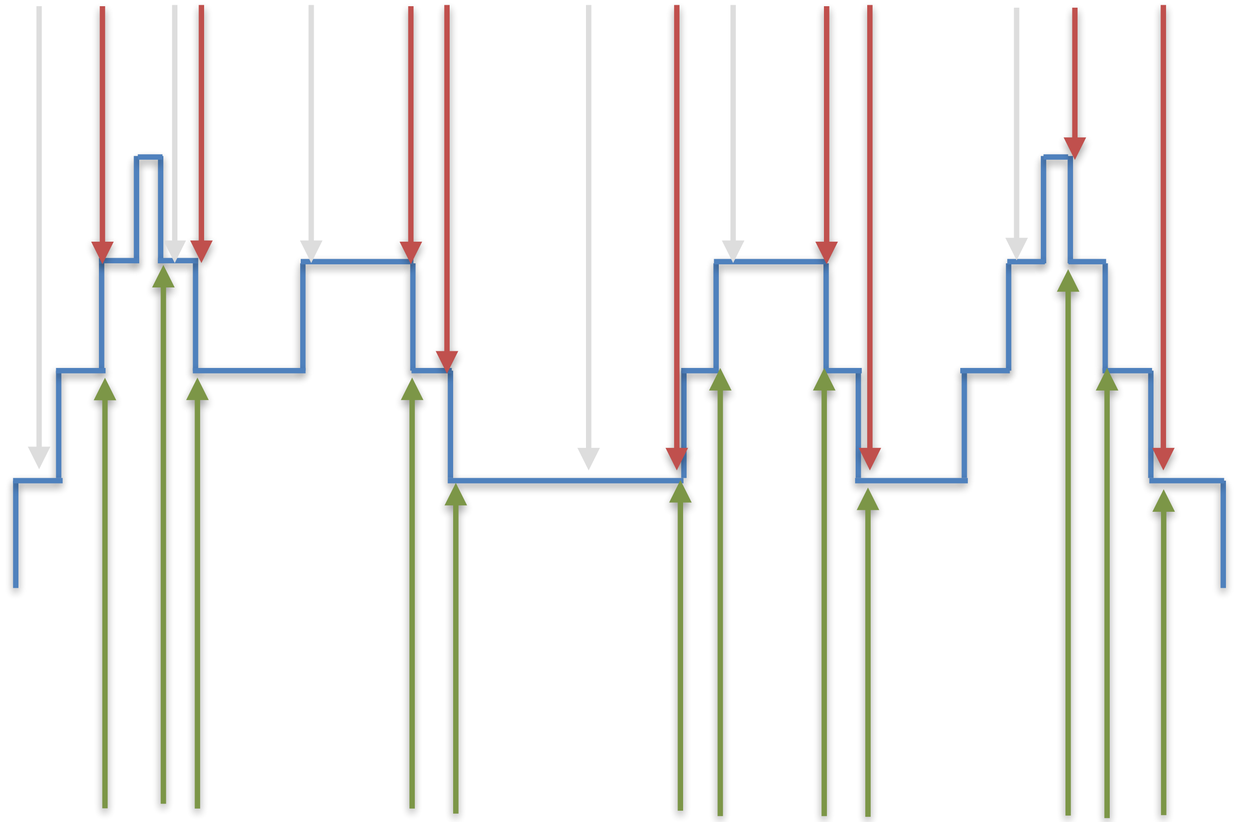

Call Tree
Time (ms)
<All threads> 14,234 100 %
java.lang.Thread.run() 14,231 99 %
ApplicationFilterChain.java:239 org.springframework.web.filter.OncePerRequestFilter.doFilter(ServletRec 108 1 %
OncePerRequestFilter.java:107 org.springframework.web.filter.CharacterEncodingFilter.doFilterInterr 108 1 %
CharacterEncodingFilter.java:88 org.apache.catalina.core.ApplicationFilterChain.doFilter(ServletRe 108 1 %
ApplicationFilterChain.java:239 com.github.dandelion.datatables.extras.servlet2.filter.Datatable 108 1 %
DatatablesFilter.java:72 org.apache.catalina.core.ApplicationFilterChain.doFilter(ServletRequ 108 1 %
ApplicationFilterChain.java:239 com.github.dandelion.datatables.core.web.filter.Datatable 108 1 %
DatatablesFilter.java:86 org.apache.catalina.core.ApplicationFilterChain.doFilter(Servl 107 1 %
ApplicationFilterChain.java:239 org.springframework.web.filter.OncePerRequestFil 107 1 %
OncePerRequestFilter.java:107 org.springframework.web.filter.HiddenHttpMeth 106 1 %
HiddenHttpMethodFilter.java:77 org.apache.catalina.core.ApplicationFilterCh 106 1 %
HttpServlet.java:729 org.springframework.web.servlet.FrameworkServlet. 106 1 %
FrameworkServlet.java:812 javax.servlet.http.HttpServlet.service(HttpS 101 1 %
HttpServlet.java:622 org.springframework.web.servlet.FrameworkSe 101 1 %
FrameworkServlet.java:827 org.springframework.web.servlet.Fra 101 1 %
FrameworkServlet.java:936 org.springframework.web.servlet.I 101 1 %
DispatcherServlet.java:856 org.springframework.web.servl 101 1 %
DispatcherServlet.java:925 org.springframework.web.se 47 0 %
DispatcherServlet.java:939 org.springframework.web.se 24 0 %
DispatcherServlet.java:925 org.springframework.web.se 20 0 %
DispatcherServlet.java:896 org.springframework.web.se 9 0 %
DispatcherServlet.java:1076 org.springframework.w 9 0 %
DispatcherServlet.java:1091 org.springframework 9 0 %
AbstractHandlerMapping.java:298 org.springf 9 0 %
AbstractHandlerMethodMapping.java:56 or 9 0 %
FrameworkServlet.java:807 com.yourkit.probes.builtin.Servlets$Servle 3 0 %
FrameworkServlet.java:814 com.yourkit.probes.builtin.Servlets$Servle 0.7 0 %
Интервал : 1 мс
Кол-во сэмплов: 100+


baz()
bar()
foo()
main()



Safepoints

```
baz()
bar()
foo()
main()
```

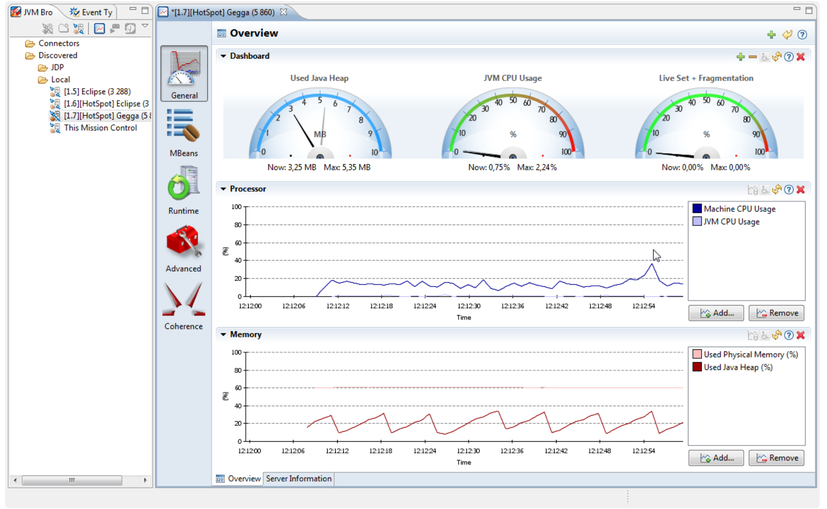


Safepoints

Как обуздать safepoint bias?

Java Mission Control

Java 7u40



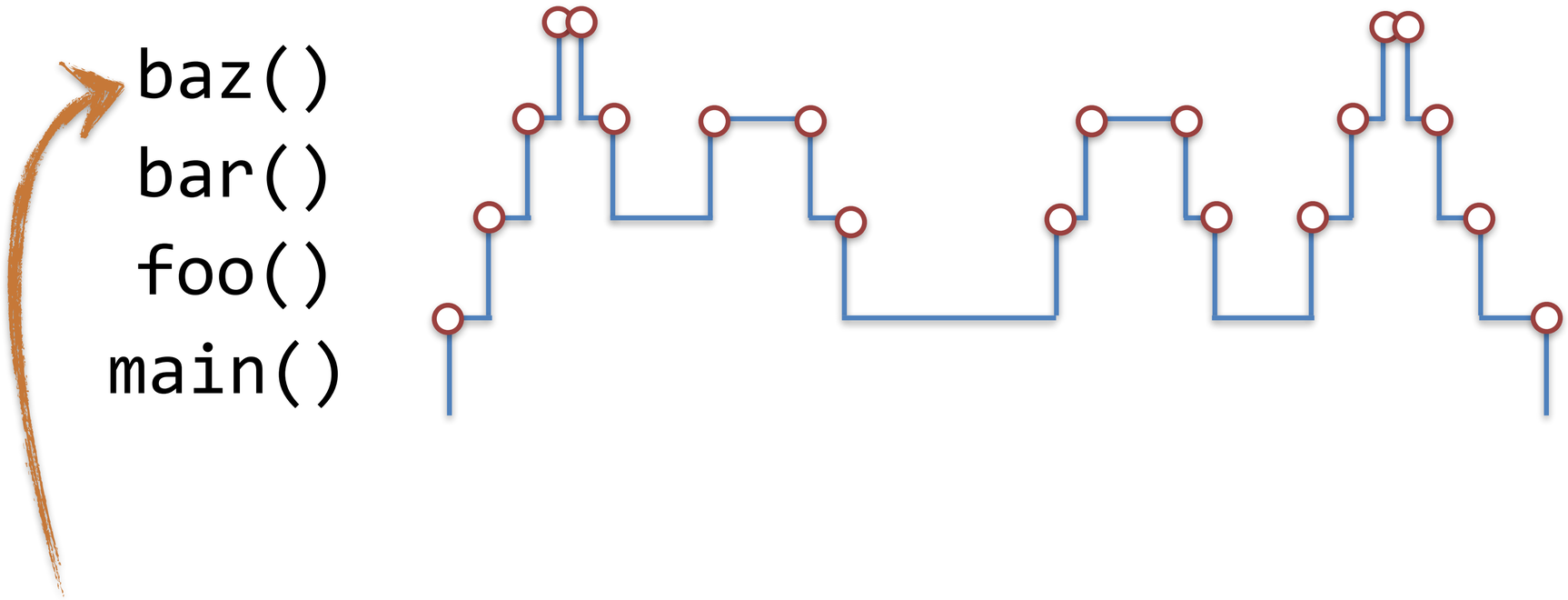
Honest Profiler

JVMTI

AsyncGetCallTrace

https://github.com/RichardWarburton/honest-profiler

Инструментирующий профилировщик



Высокие накладные расходы для быстрых методов

```
public void businessMethod() {
  long start = System.currentTimeMillis();
  work();
  Profiler.log(System.currentTimeMillis()-start);
}
```

```
public void businessMethod() {
  long start = System.nanoTime();
  work();
  Profiler.log(System.nanoTime()-start);
}
```

```
public void businessMethod() {
  Profiler.start("businessMethod");
  try {
    work();
  } finally {
    Profiler.log("businessMethod");
  }
}
```

System.nanoTime

Параллельный поток для замеров времени

- sleep-wakup-update
- yield-wakup-update
- busy-loop-update

```
class Profiler {

  Loop loop;

  public static void start(String method) {
    long now = loop.getTime();
    …
  }
```

Busy Loop

```
public class Loop implements Runnable {
    private volatile long time;
    public void run() {
        while (running) {
            time = System.nanoTime();
            sleep();
        }
    }

    public final long getTime() {
        return time;
    }
```

Наносекунды в приближении

Чтение памяти имеет цену

Занимает циклы процессора = ns

Каждый слой ПО добавляет к цене.

JVM, JNI, OS, HW

Nanotrusting the NanoTime

http://shipilev.net/blog/2014/nanotrusting-nanotime/

Наносекунды в приближении

granularity_nanotime:	26.300 +- 0.205 ns	Linux
latency_nanotime:	25.542 +- 0.024 ns	
granularity_nanotime:	29.322 +- 1.293 ns	Solaris
latency_nanotime:	29.910 +- 1.626 ns	
granularity_nanotime:	371,419 +- 1,541 ns	Windows
latency_nanotime:	14,415 +- 0,389 ns	

Nanotrusting the NanoTime

http://shipilev.net/blog/2014/nanotrusting-nanotime/

Thread.sleep

- Win8: 1.7 мс (JNI + socket poll: 1.0 ms)
- Linux: 0.1 мс
- OS X: 0.05 мс
- Всё таки загружает ядро процессора на 25-50%

Thread.yield

- Планировшик в Windows может проигнорировать наш поток в случае высокой загрузки процессора

Busy Loop

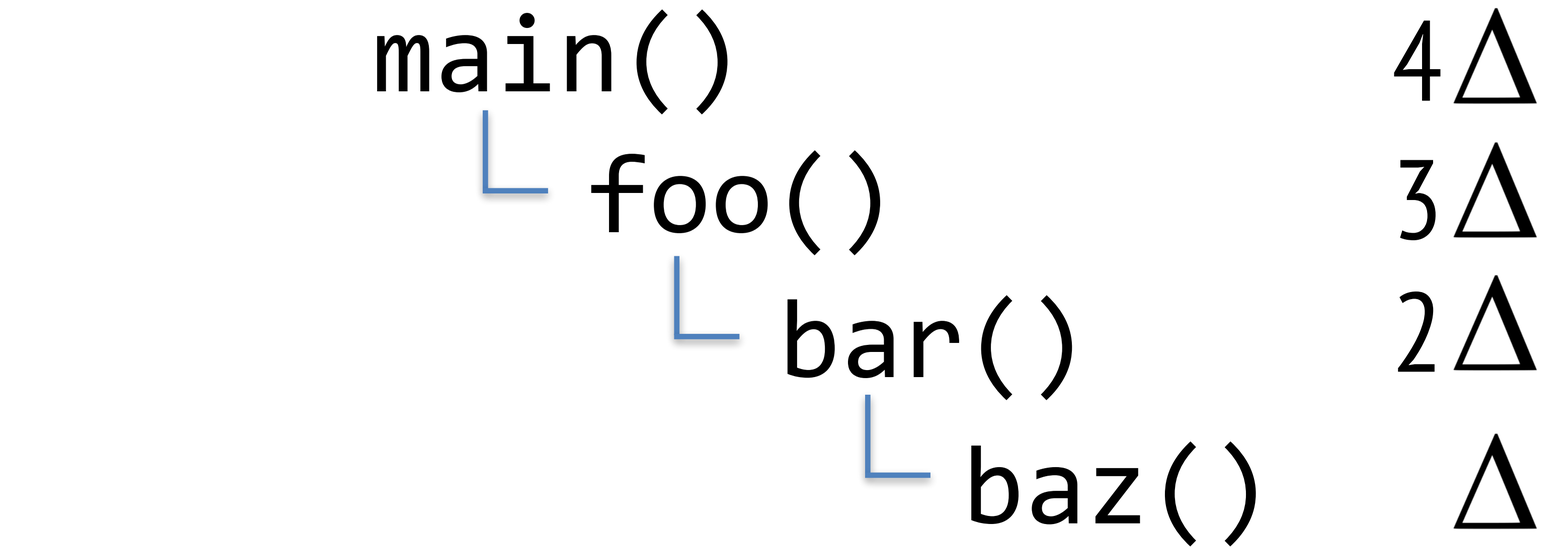
```
public class Loop implements Runnable {
    private volatile long time;
    public void run() {
        while (running) {
            time = System.nanoTime();
            sleep();
        }
    }
    private void sleep() {
      if (!MiscUtil.isWindows()) {
         Thread.yield();
      }
    }
```

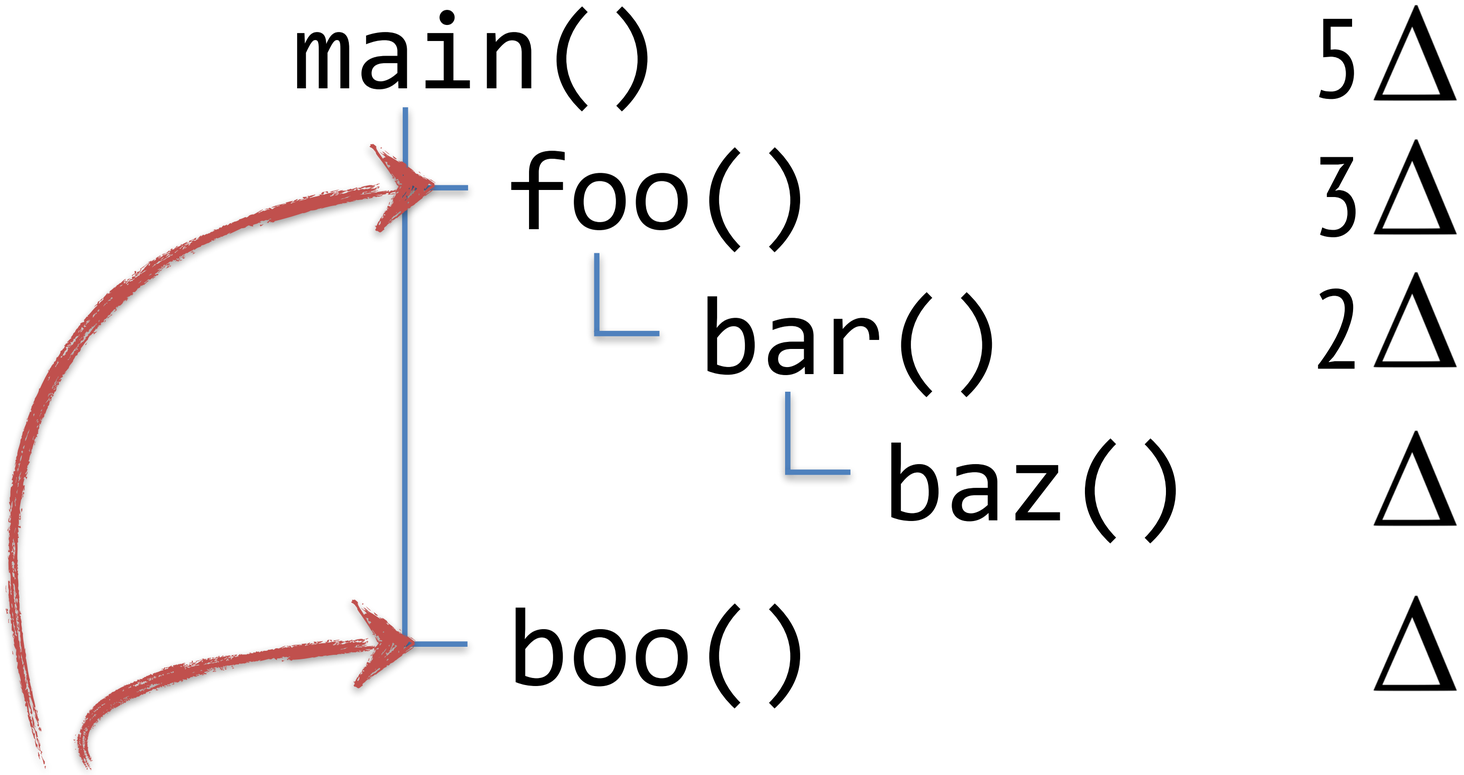
Busy Loop

```
public class Loop implements Runnable {
    private volatile long time;
    public void run() {
        while (running) {
            time = System.nanoTime();
            sleep();
        }
    }
    private void sleep() {
      if (!MiscUtil.isWindows()) {
         Thread.yield();
      }
    }
```

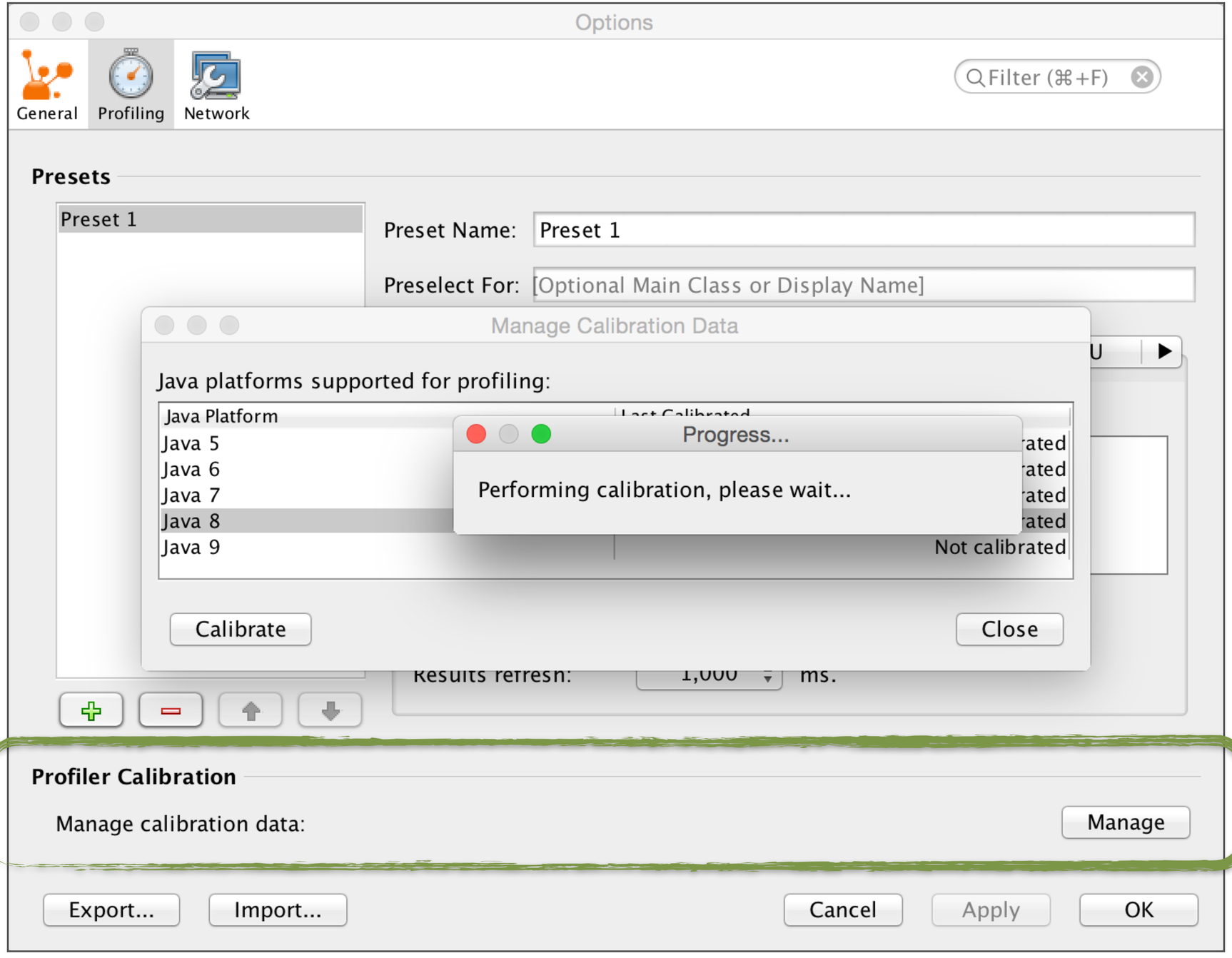
Busy Loop

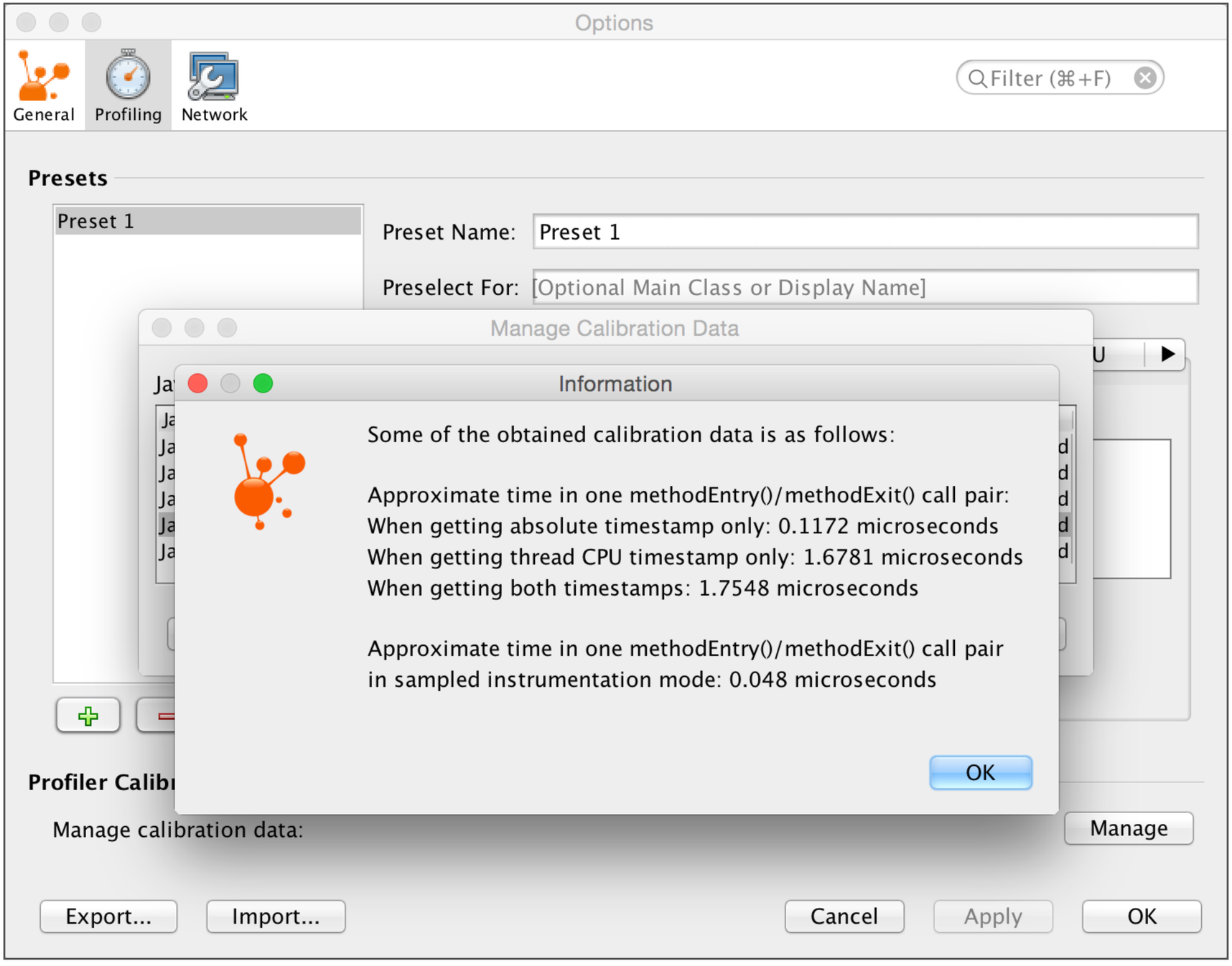
```
public class Loop implements Runnable {
    private volatile long time;
    public void run() {
        while (running) {
            time = System.nanoTime();
            sleep();
        }
    }
    private void sleep() {
      if (!MiscUtil.isWindows()) {
         Thread.yield();
      }
    }
```


main()
foo()
bar()
baz()
4Δ
3Δ
2Δ
Δ



Дисбаланс ошибки для веток с разной глубиной


Options
General
Profiling
Network
Filter (⌘+F)
Presets
Preset 1
Preset Name:
Preset 1
Preselect For:
[Optional Main Class or Display Name]
Manage Calibration Data
Java platforms supported for profiling:
Java Platform
Java 5
Java 6
Java 7
Java 8
Java 9
Not calibrated
Progress...
Performing calibration, please wait...
Calibrate
Close
Profiler Calibration
Manage calibration data:
Manage
Export...
Import...
Cancel
Apply
OK


Options
General
Profiling
Network
Filter (⌘+F)
Presets
Preset 1
Preset Name:
Preset 1
Preselect For:
[Optional Main Class or Display Name]
Manage Calibration Data
Information
Some of the obtained calibration data is as follows:
Approximate time in one methodEntry()/methodExit() call pair:
When getting absolute timestamp only: 0.1172 microseconds
When getting thread CPU timestamp only: 1.6781 microseconds
When getting both timestamps: 1.7548 microseconds
Approximate time in one methodEntry()/methodExit() call pair
in sampled instrumentation mode: 0.048 microseconds
OK
Profiler Calibr
Manage calibration data:
Manage
Export...
Import...
Cancel
Apply
OK

Оптимизации

```
LeafMethodAnalyser(AbstractMethodBodyAnalyser mv) {
   super(mv, not(eq(VISIT_METHOD_INSN)),
             not(eq(VISIT_INVOKE_DYNAMIC_INSN)),
             not(eq(VISIT_JUMP_INSN)));
}
```

Оптимизации

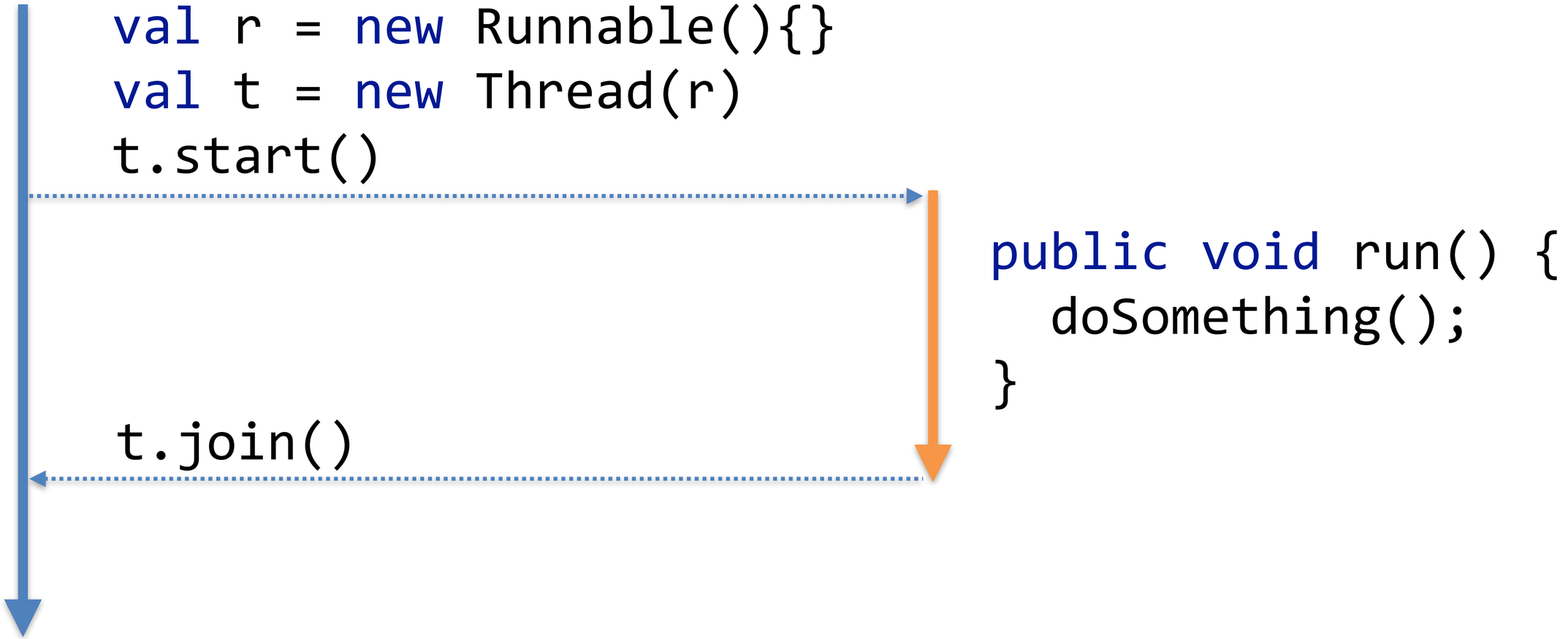
```
public int calculateMargin(Order order){
  return order.grossPrice() / order.salesPrice() * 100;
}

public int grossPrice() {
  return gross;
}

public int grossPrice();
  Code:
    0: aload_0
    1: getfield #2   // Field gross:I
    4: ireturn
```

Не вызывает других методов

Нет циклов

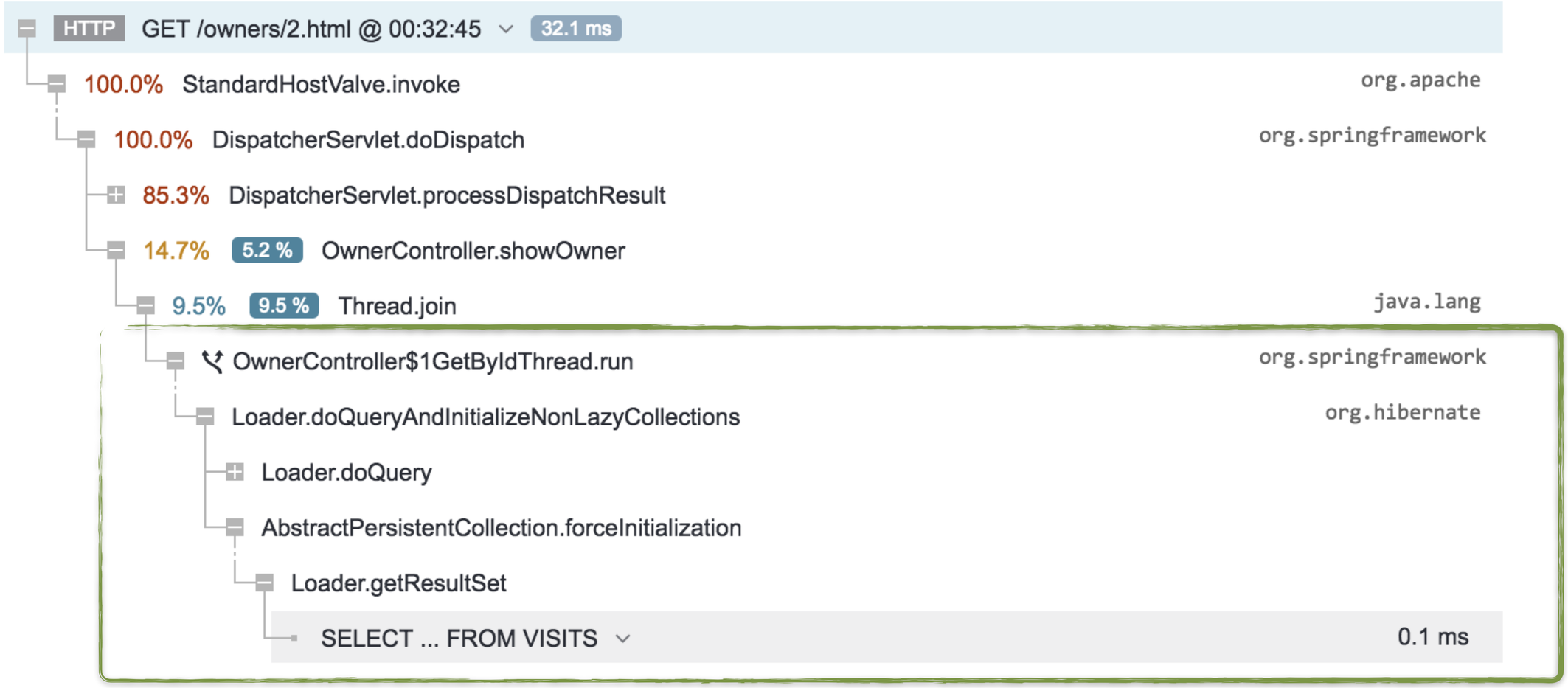
Можно игнорировать

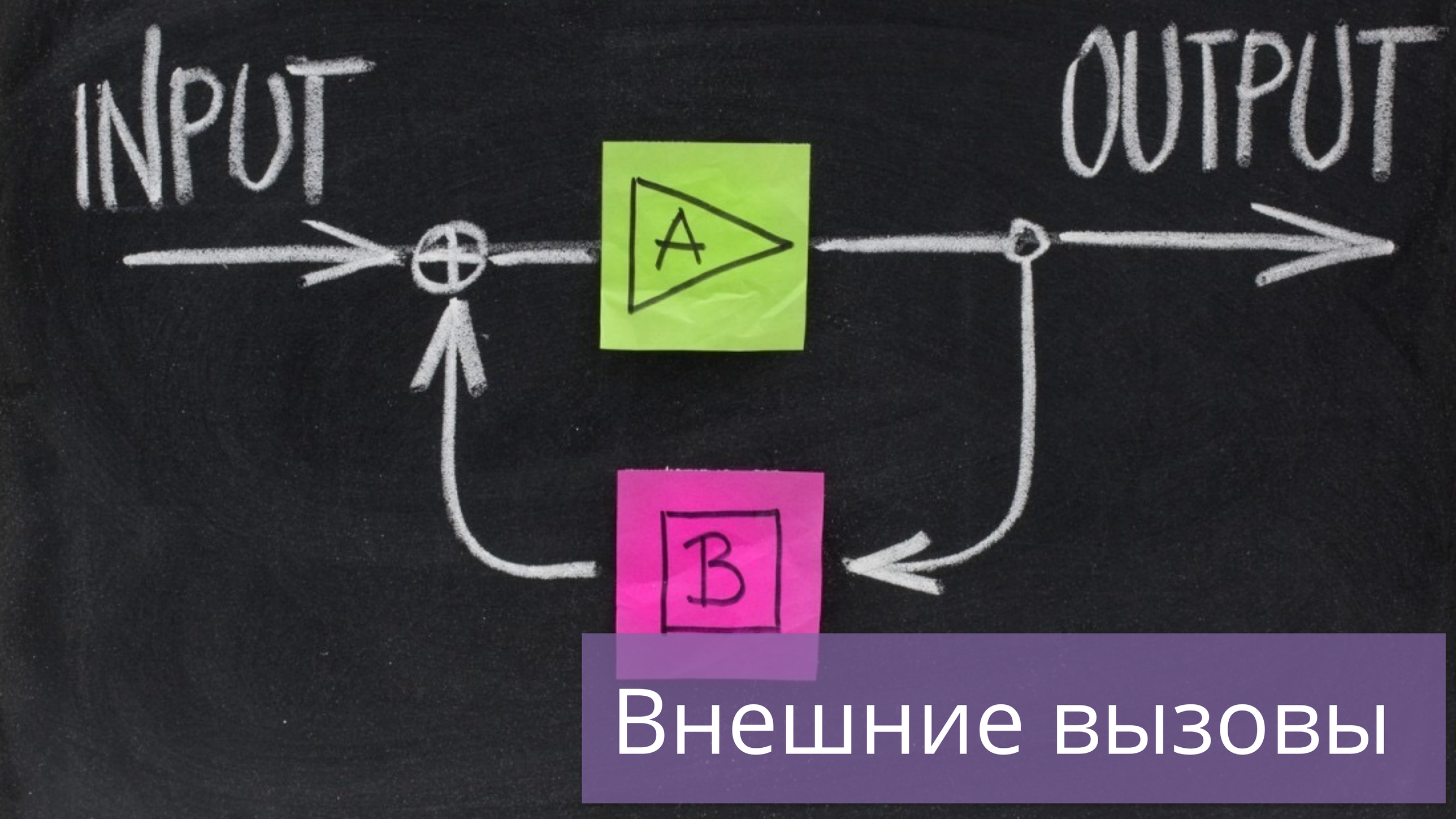
Потоки


val r = new Runnable(){}
val t = new Thread(r)
t.start()
t.join()
public void run() {
doSomething();
}

```
Runnable r = new Runnable(){
   public void run() {
      doSomething();
   }
};
```

```
Runnable r = new Runnable(){
   private WeakReference<Context> ctx;

   public void run() {
      Context sink = ctx.getChildContext();
      sink = Sinks.startAsync(sink);

      try {
         doSomething();
      } finally {
         Sinks.stopAsync(sink);
      }
   }
};
```


HTTP GET /owners/2.html @ 00:32:45 32.1 ms
100.0% StandardHostValve.invoke
org.apache
100.0% DispatcherServlet.doDispatch
org.springframework
85.3% DispatcherServlet.processDispatchResult
14.7% 5.2 % OwnerController.showOwner
9.5% 9.5 % Thread.join
java.lang
OwnerController$1GetByIdThread.run
org.springframework
Loader.doQueryAndInitializeNonLazyCollections
org.hibernate
Loader.doQuery
AbstractPersistentCollection.forceInitialization
Loader.getResultSet
SELECT ... FROM VISITS
0.1 ms



Внешние вызовы

JDBC

> **Вариант 1.** Proxy

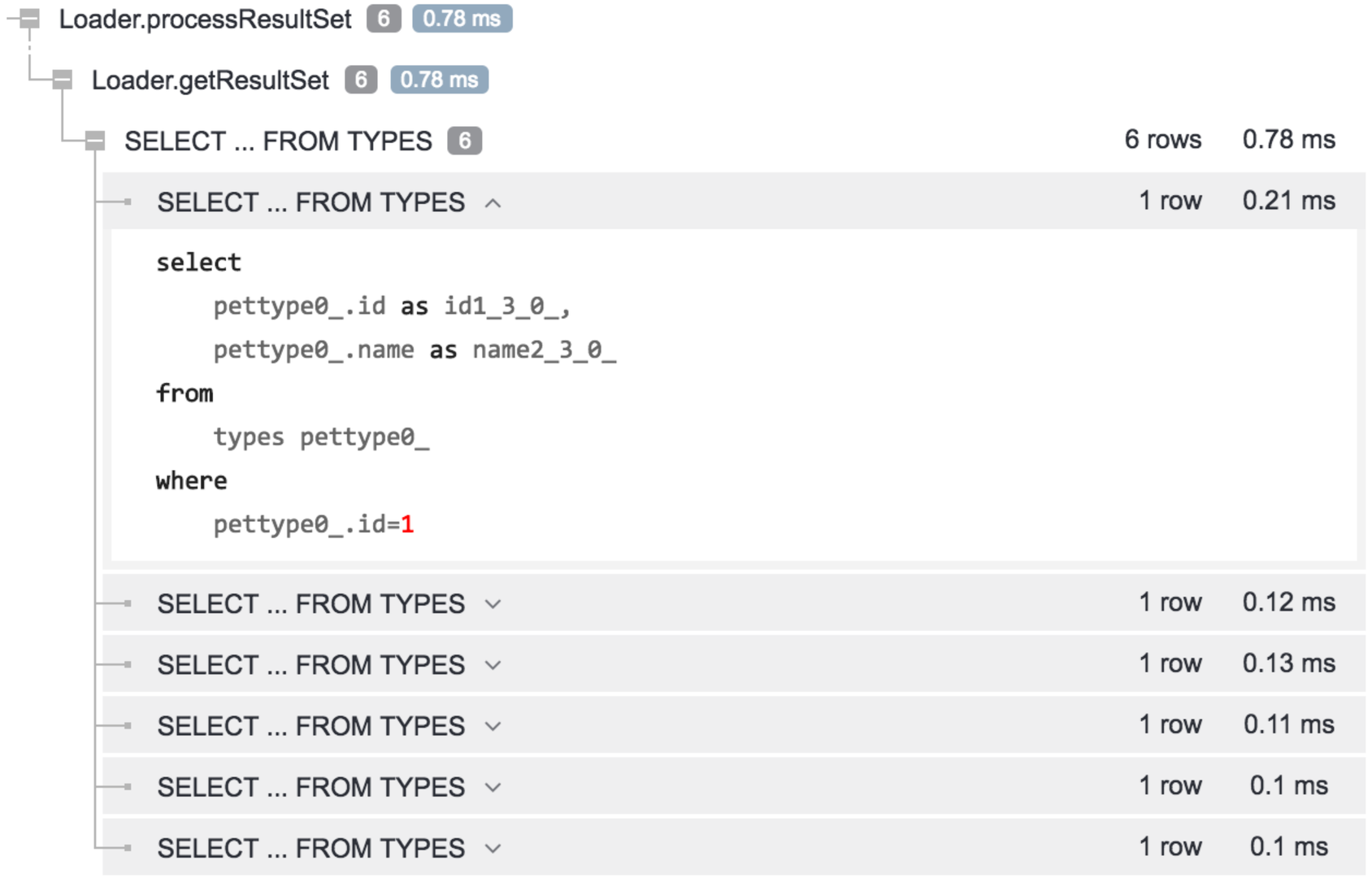
 Привет, ClassCastException

> **Вариант 2.** Список известных нам классов. Что знаем, то и инструментируем.

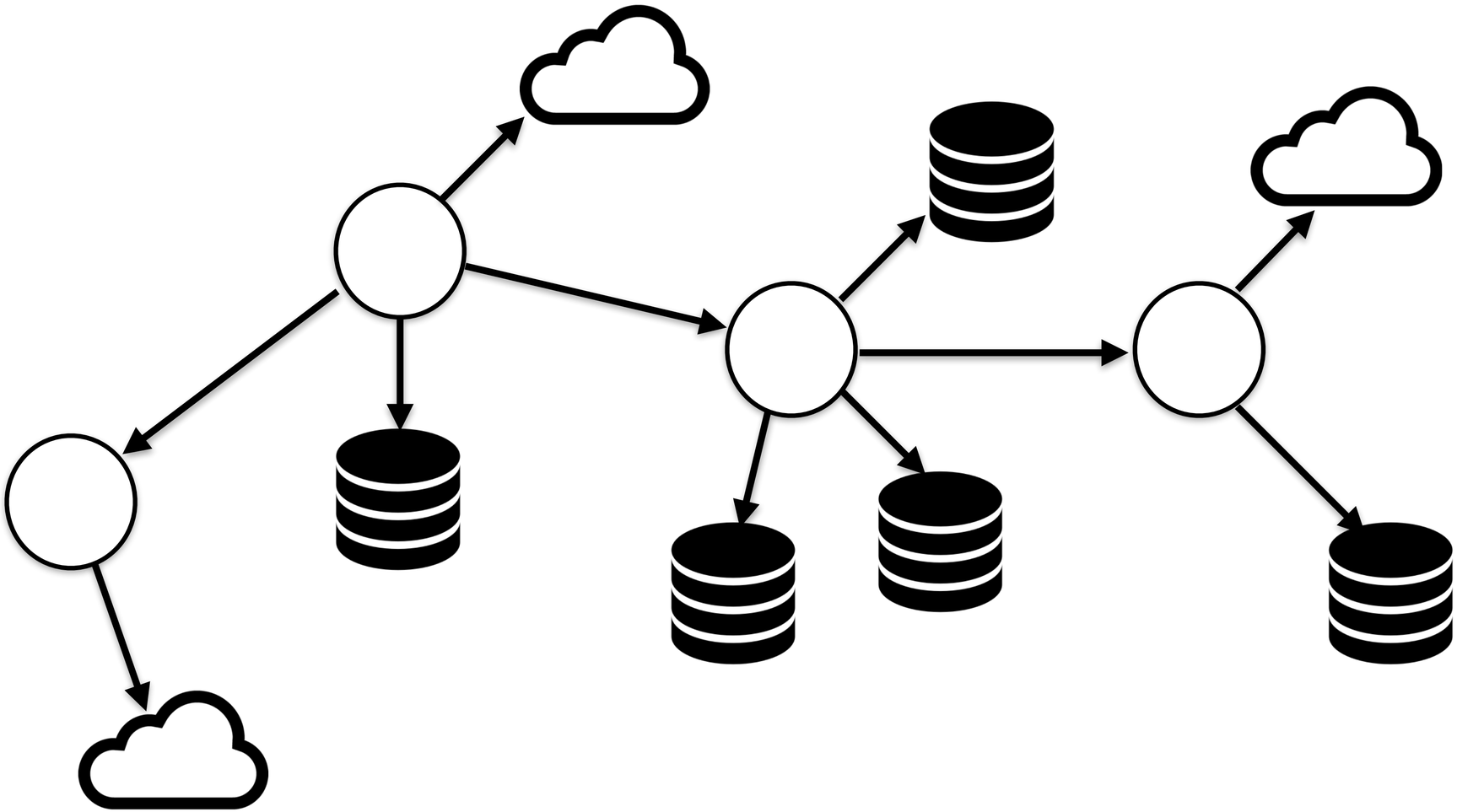
 Привет, DB2

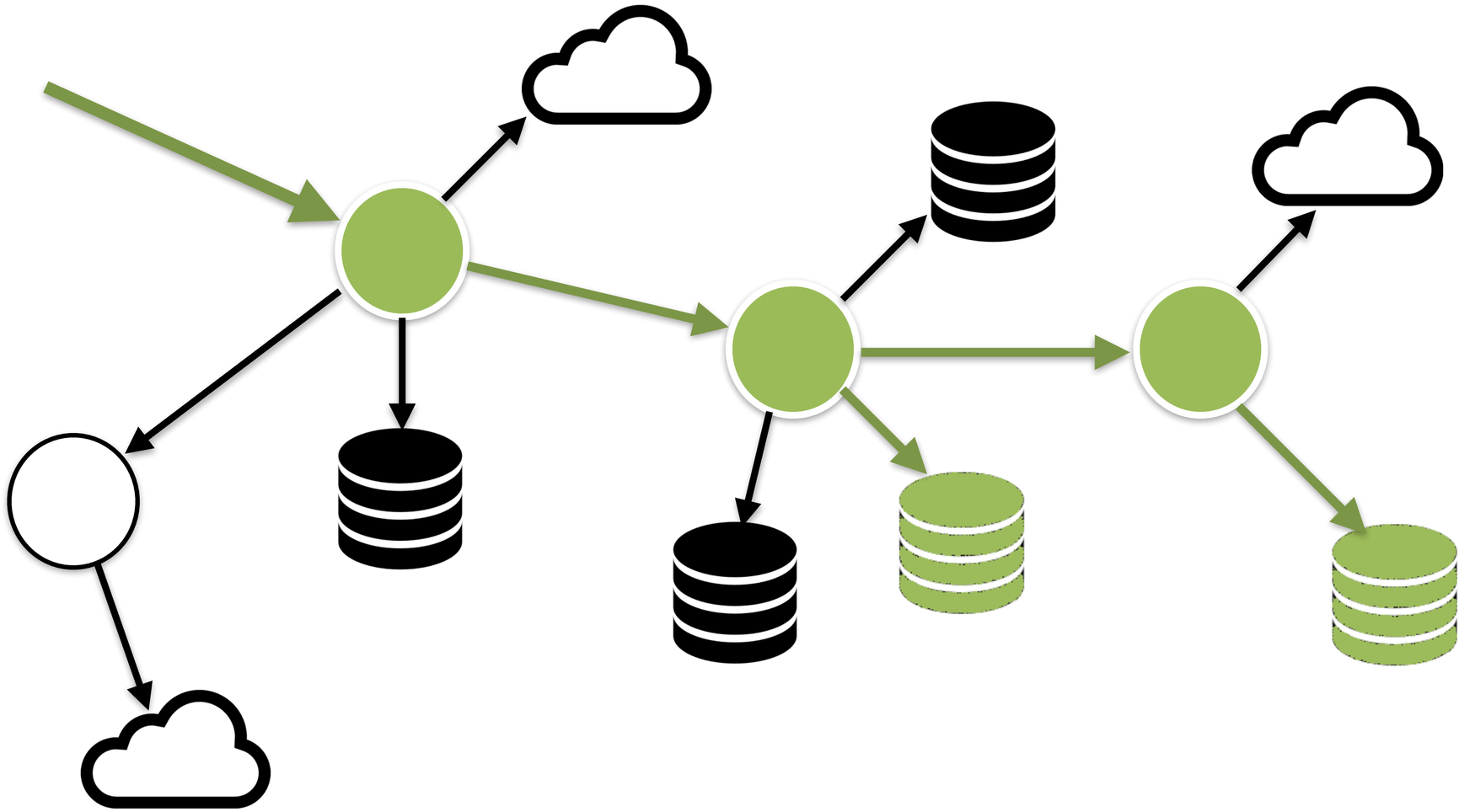
> **Вариант 3.** Есть известные интерфейсы - сканируем иерархию классов

 Замедление старта приложения


Loader.processResultSet 6 0.78 ms
Loader.getResultSet 6 0.78 ms
SELECT ... FROM TYPES 6 6 rows 0.78 ms
SELECT ... FROM TYPES 1 row 0.21 ms
select
pettype0_.id as id1_3_0_,
pettype0_.name as name2_3_0_
from
types pettype0_
where
pettype0_.id=1
SELECT ... FROM TYPES 1 row 0.12 ms
SELECT ... FROM TYPES 1 row 0.13 ms
SELECT ... FROM TYPES 1 row 0.11 ms
SELECT ... FROM TYPES 1 row 0.1 ms
SELECT ... FROM TYPES 1 row 0.1 ms

Распределённые приложения

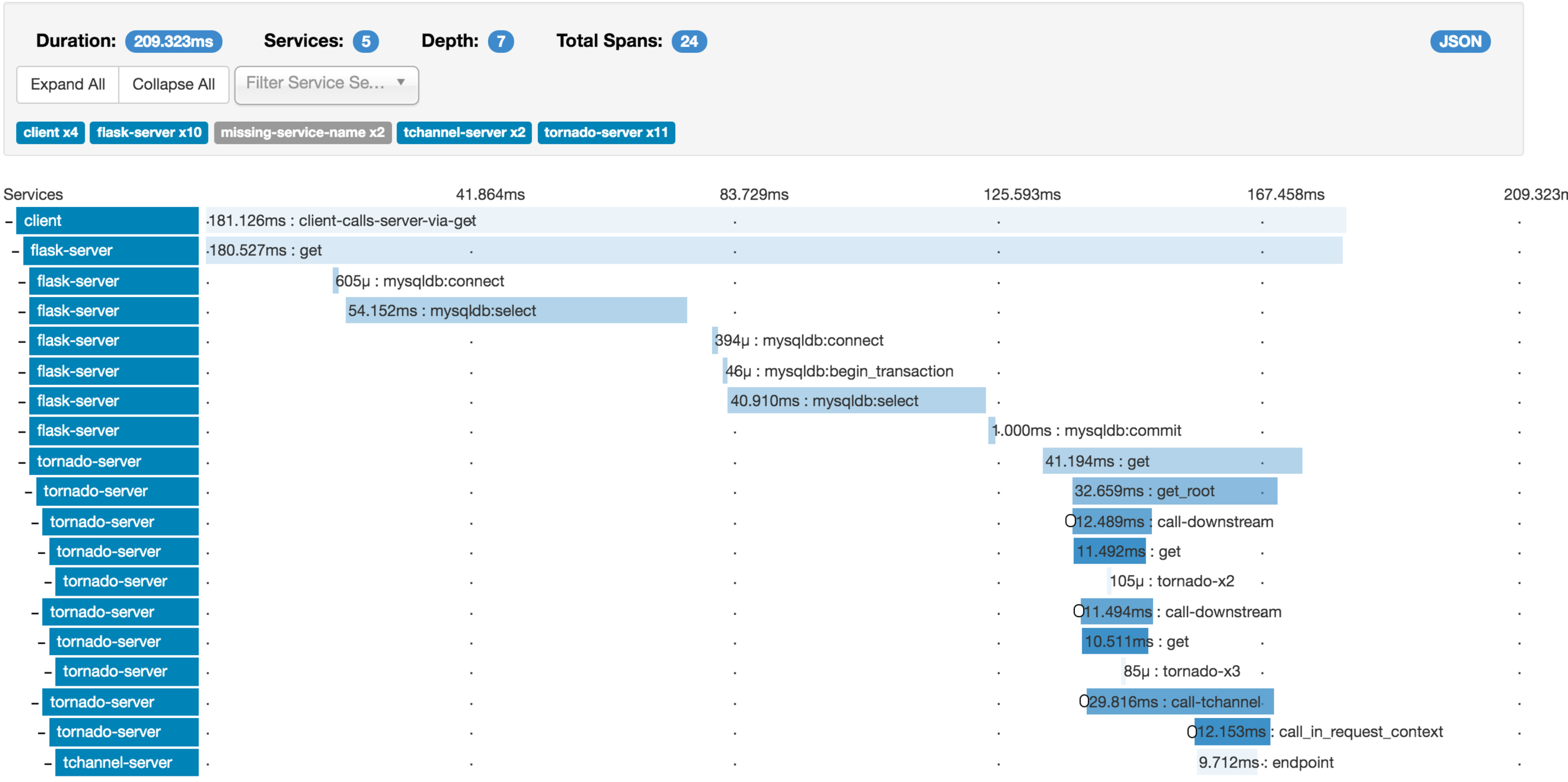




Что уже есть?

- Разные $$$ APM-продукты: Dynatrace, AppDynamics
- Dapper http://research.google.com/pubs/pub36356.html
- ZipKin http://zipkin.io

NB! Требуется серверная часть

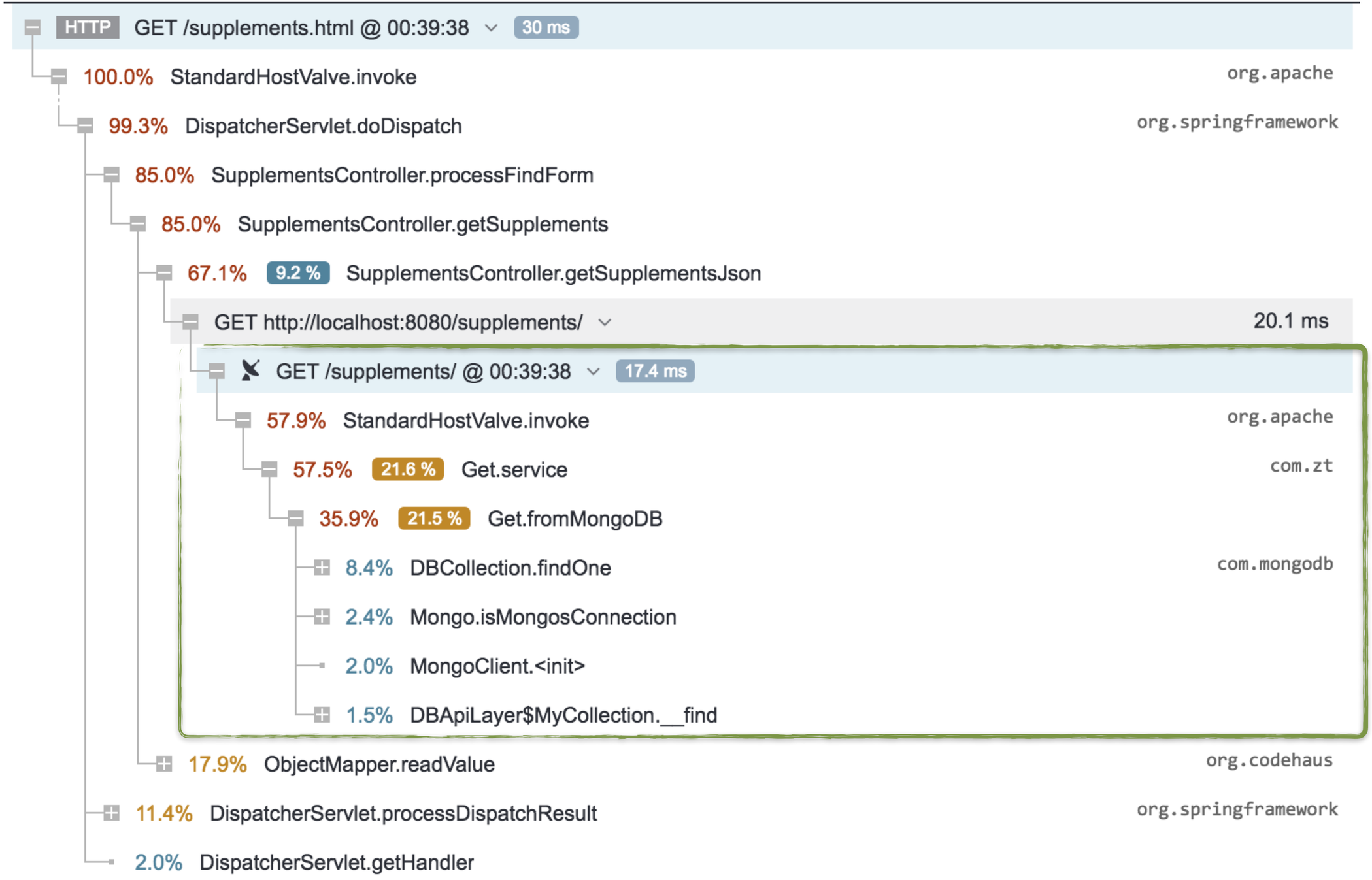

Duration: 209.323ms
Services: 5
Depth: 7
Total Spans: 24
JSON
Expand All
Collapse All
Filter Service Se...
client x4
flask-server x10
missing-service-name x2
tchannel-server x2
tornado-server x11
Services
41.864ms
83.729ms
125.593ms
167.458ms
209.323m
client
flask-server
flask-server
flask-server
flask-server
flask-server
flask-server
flask-server
tornado-server
tornado-server
tornado-server
tornado-server
tornado-server
tornado-server
tornado-server
tornado-server
tornado-server
tchannel-server
181.126ms : client-calls-server-via-get
180.527ms : get
605µ : mysqldb:connect
54.152ms : mysqldb:select
394µ : mysqldb:connect
46µ : mysqldb:begin_transaction
40.910ms : mysqldb:select
1.000ms : mysqldb:commit
41.194ms : get
32.659ms : get_root
12.489ms : call-downstream
11.492ms : get
105µ : tornado-x2
11.494ms : call-downstream
10.511ms : get
85µ : tornado-x3
29.816ms : call-tchannel
12.153ms : call_in_request_context
9.712ms : endpoint

Request headers

GET /supplements/ HTTP/1.1
X-XRebel-Event-Id: 6a61eb15-23f1-40ea-a30d-2f62d616a196
X-XRebel-Version: 3.0.2
Cache-Control: no-cache
Pragma: no-cache
User-Agent: Java/1.8.0_51
Host: localhost:8080

Response headers

HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
X-XRebel-Event-Id: c5603281-a745-416e-8f27-92ba47c76b26
X-XRebel-Version: 3.0.2
X-xr-bookmark: 3197589b-bc9f-44ab-a2aa-12557a39b090
Content-Type: application/json;charset=ISO-8859-1


HTTP GET /supplements.html @ 00:39:38 30 ms
100.0% StandardHostValve.invoke org.apache
99.3% DispatcherServlet.doDispatch org.springframework
85.0% SupplementsController.processFindForm
85.0% SupplementsController.getSupplements
67.1% 9.2 % SupplementsController.getSupplementsJson
GET http://localhost:8080/supplements/ 20.1 ms
GET /supplements/ @ 00:39:38 17.4 ms
57.9% StandardHostValve.invoke org.apache
57.5% 21.6 % Get.service com.zt
35.9% 21.5 % Get.fromMongoDB
8.4% DBCollection.findOne com.mongodb
2.4% Mongo.isMongosConnection
2.0% MongoClient.<init>
1.5% DBApiLayer$MyCollection.__find
17.9% ObjectMapper.readValue org.codehaus
11.4% DispatcherServlet.processDispatchResult org.springframework
2.0% DispatcherServlet.getHandler

This slide is intentionally left blank

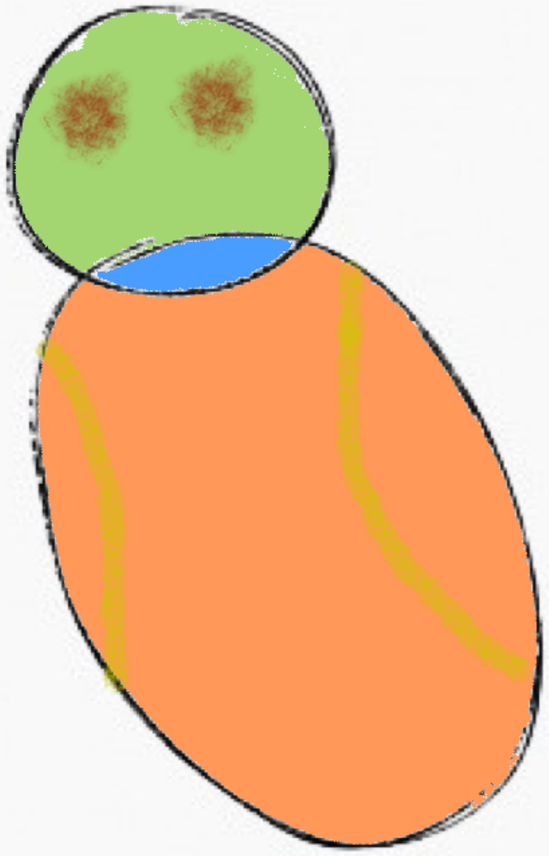


Fig 1. Draw two circles

Fig 2. Draw the rest of the damn Owl

@antonarhipov

anton@zeroturnaround.com

https://speakerdeck.com/antonarhipov

https://www.slideshare.net/arhan