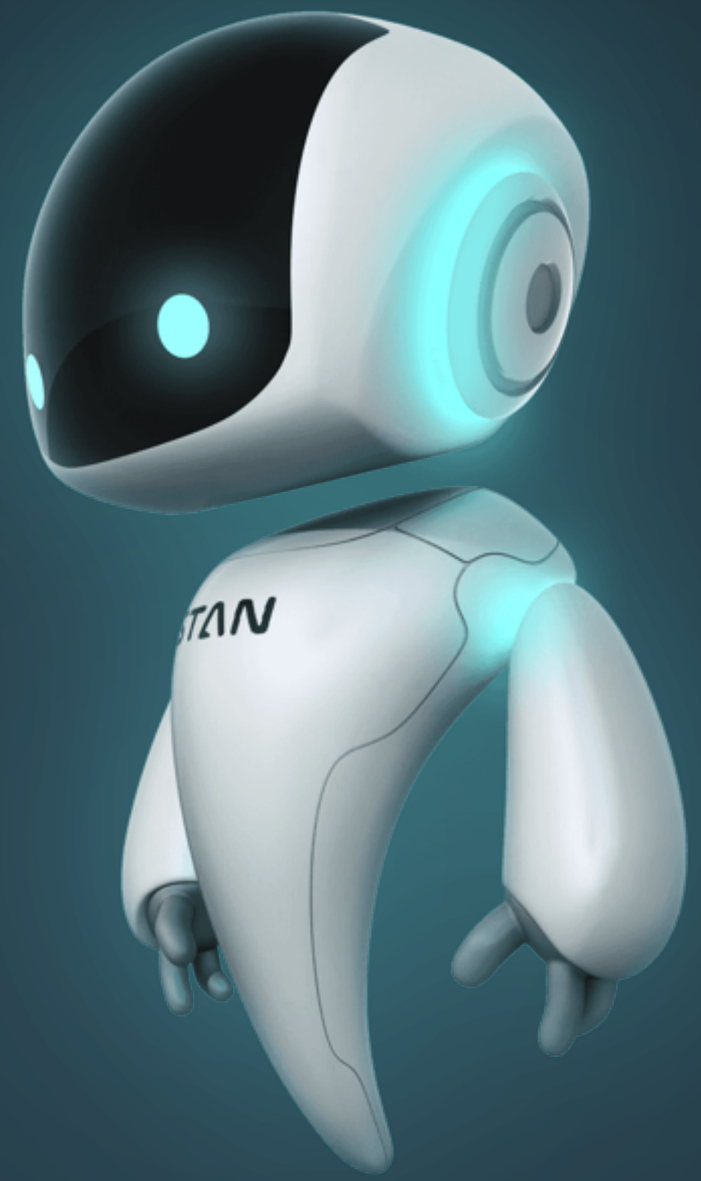
Does My Profiler Tell The Truth?

Fabian Lange



INSTANA®

Profilers are Measurement Tools

INSTANA®

„A fool with a tool is still a fool.“

–Grady Booch

Fabian Lange @CodingFabian · 7h

having "quoted" him, I wonder if @Grady_Booch actually did use the "fool with a tool" phrase :-)

Grady Booch @Grady_Booch · 45m

@CodingFabian i have indeed used that phrase many times in my misspent youth

7:29 PM - 10 Sep 2015 · Details

Available Tools

Mission Control

YourKit

Solaris Studio

VisualVM

AppDynamics

NetBeans Profiler

New Relic

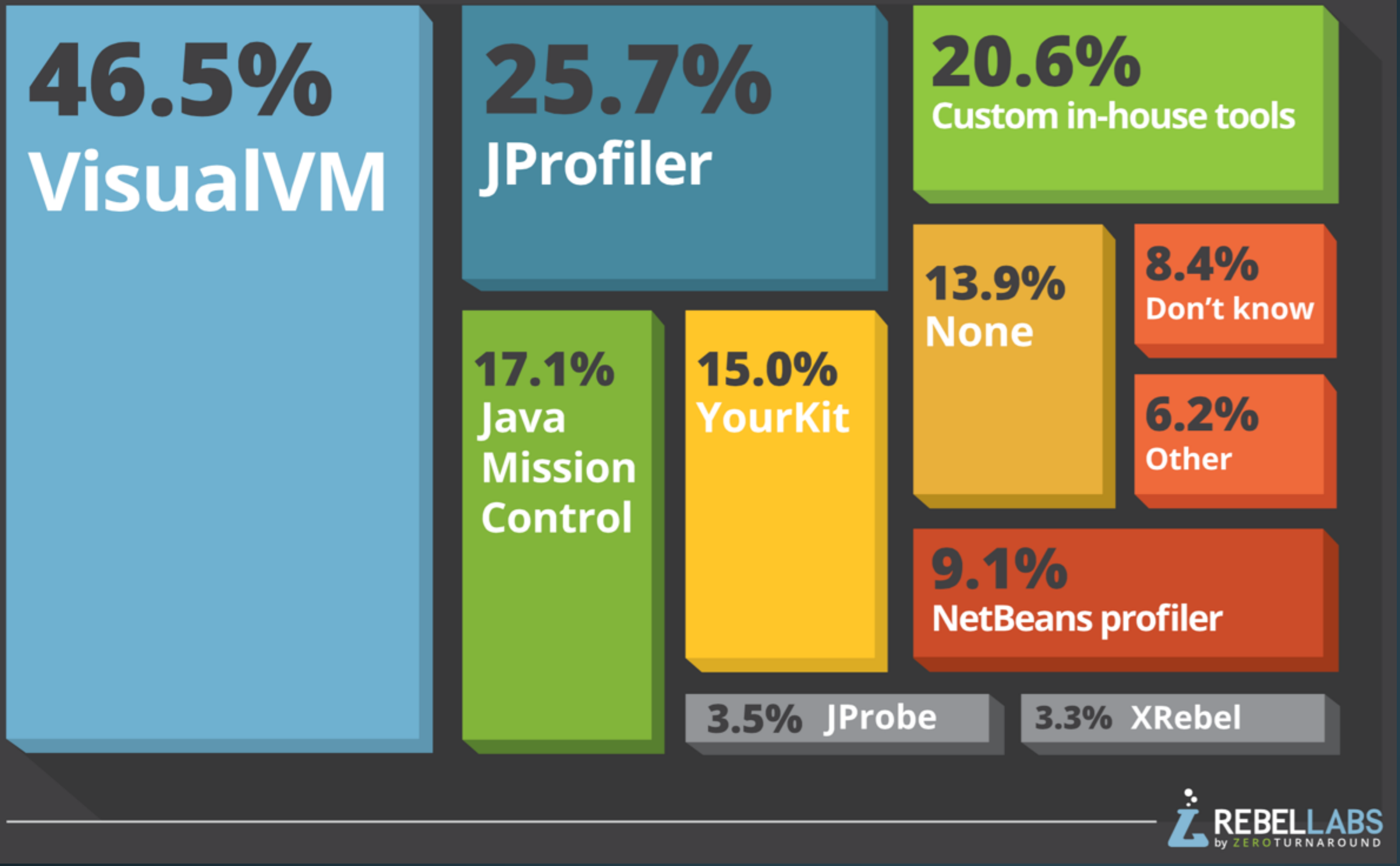
JProfiler

Honest Profiler

JProbe

Dynatrace

Satoris



Source: RebelLabs Developer Productivity Report, Page 22: 0t.ee/javaperformance15

Java Virtual Machine

Tool Interface

JSR-163

Java Platform Profiling Architecture defines APIs:

java.lang.instrument

ClassFileTransformer

java.lang.management

MemoryMXBean

ThreadMXBean

How To Measure Code?

Native Agent

written in C

inserted into the JVM using -agentpath

Java Agent

using java.lang.instrument package

loaded using -javaagent

Function Index

- **Memory Management**
  - Allocate
  - Deallocate
- **Thread**
  - Get Thread State
  - Get Current Thread
  - Get All Threads
  - Suspend Thread
  - Suspend Thread List
  - Resume Thread
  - Resume Thread List
  - Stop Thread
  - Interrupt Thread
  - Get Thread Info
  - Get Owned Monitor Info
  - Get Owned Monitor Stack Depth Info
  - Get Current Contended Monitor
  - Run Agent Thread
  - Set Thread Local Storage
  - Get Thread Local Storage
- **Thread Group**
  - Get Top Thread Groups
  - Get Thread Group Info
  - Get Thread Group Children
- **Stack Frame**
  - Get Stack Trace
  - Get All Stack Traces
  - Get Thread List Stack Traces
  - Get Frame Count
  - Pop Frame
  - Get Frame Location
  - Notify Frame Pop
- **Force Early Return**
  - Force Early Return - Object
  - Force Early Return - Int
  - Force Early Return - Long
  - Force Early Return - Float
  - Force Early Return - Double
  - Force Early Return - Void
- **Heap**
  - Follow References
  - Iterate Through Heap
  - Get Tag
  - Set Tag
  - Get Objects With Tags
  - Force Garbage Collection
- **Heap (1.0)**
  - Iterate Over Objects Reachable From Object
  - Iterate Over Reachable Objects
  - Iterate Over Heap
  - Iterate Over Instances Of Class
- **Local Variable**
  - Get Local Variable - Object
  - Get Local Instance
  - Get Local Variable - Int
  - Get Local Variable - Long
  - Get Local Variable - Float
  - Get Local Variable - Double
  - Set Local Variable - Object
  - Set Local Variable - Int
  - Set Local Variable - Long
  - Set Local Variable - Float
  - Set Local Variable - Double
- **Breakpoint**
  - Set Breakpoint
  - Clear Breakpoint
- **Watched Field**
  - Set Field Access Watch
  - Clear Field Access Watch
  - Set Field Modification Watch
  - Clear Field Modification Watch
- **Class**
  - Get Loaded Classes
  - Get Classloader Classes
  - Get Class Signature
  - Get Class Status
  - Get Source File Name
  - Get Class Modifiers
  - Get Class Methods
  - Get Class Fields
  - Get Implemented Interfaces
  - Get Class Version Numbers
  - Get Constant Pool
  - Is Interface
  - Is Array Class
  - Is Modifiable Class
  - Get Class Loader
  - Get Source Debug Extension
  - Retransform Classes
  - Redefine Classes
- **Object**
  - Get Object Size
  - Get Object Hash Code
  - Get Object Monitor Usage
- **Field**
  - Get Field Name (and Signature)
  - Get Field Declaring Class
  - Get Field Modifiers
  - Is Field Synthetic
- **Method**
  - Get Method Name (and Signature)
  - Get Method Declaring Class
  - Get Method Modifiers
  - Get Max Locals
  - Get Arguments Size
  - Get Line Number Table
  - Get Method Location
  - Get Local Variable Table
  - Get Bytecodes
  - Is Method Native
  - Is Method Synthetic
  - Is Method Obsolete
  - Set Native Method Prefix
  - Set Native Method Prefixes
- **Raw Monitor**
  - Create Raw Monitor
  - Destroy Raw Monitor
  - Raw Monitor Enter
  - Raw Monitor Exit
  - Raw Monitor Wait
  - Raw Monitor Notify
  - Raw Monitor Notify All
- **JNI Function Interception**
  - Set JNI Function Table
  - Get JNI Function Table
- **Event Management**
  - Set Event Callbacks
  - Set Event Notification Mode
  - Generate Events
- **Extension Mechanism**
  - Get Extension Functions
  - Get Extension Events
  - Set Extension Event Callback
- **Capability**
  - Get Potential Capabilities
  - Add Capabilities
  - Relinquish Capabilities
  - Get Capabilities
- **Timers**
  - Get Current Thread CPU Timer Information
  - Get Current Thread CPU Time
  - Get Thread CPU Timer Information
  - Get Thread CPU Time
  - Get Timer Information
  - Get Time
  - Get Available Processors
- **Class Loader Search**
  - Add To Bootstrap Class Loader Search
  - Add To System Class Loader Search
- **System Properties**
  - Get System Properties
  - Get System Property
  - Set System Property
- **General**
  - Get Phase
  - Dispose Environment
  - Set Environment Local Storage
  - Get Environment Local Storage
  - Get Version Number
  - Get Error Name
  - Set Verbose Flag
  - Get JLocation Format

„One cannot measure Java code without interfering with the JVM“

–Werner Heisenberg, 1927

Error Sources

INSTANA®

Overhead

INSTANA

Overhead

Runtime Delay

CPU Consumption

Thread Scheduling

Memory Consumption

Network Saturation

Diskspace Usage

Accuracy

INSTANA®

Accuracy

```
System.currentTimeMillis()
/**
  * Returns the current time in milliseconds. Note that while the unit of time of the return value is a millisecond,
  * the granularity of the value depends on the underlying operating system and may be larger. For example, many
  * operating systems measure time in units of tens of milliseconds.
  *
  * See the description of the class <code>Date</code> for a discussion of slight discrepancies that may arise between
  * "computer time" and coordinated universal time (UTC).
  */

System.nanoTime()
/**
  * Returns the current value of the running Java Virtual Machine's high-resolution time source, in nanoseconds.
  *
  * This method can only be used to measure elapsed time and is not related to any other notion of system or wall-clock
  * time. The value returned represents nanoseconds since some fixed but arbitrary <i>origin</i> time (perhaps in the
  * future, so values may be negative). The same origin is used by all invocations of this method in an instance of a
  * Java virtual machine; other virtual machine instances are likely to use a different origin.
  *
  * This method provides nanosecond precision, but not necessarily nanosecond resolution (that is, how frequently the
  * value changes) - no guarantees are made except that the resolution is at least as good as that of
  * {@link #currentTimeMillis()}.
  *
  * The values returned by this method become meaningful only when the difference between two such values, obtained
  * within the same instance of a Java virtual machine, is computed.
  */
```

Time

INSTANA®

Time

Wall-Clock Time

“Real” time which has passed since start.

Measurable with a clock on the wall.

CPU Time

Time the CPU was busy.

Measurable but questionable.

Lots of Data

INSTANA®

Data Collection

INSTANA

Data Collection

(Stack) Sampling

Checking JVM activity in regular intervals.

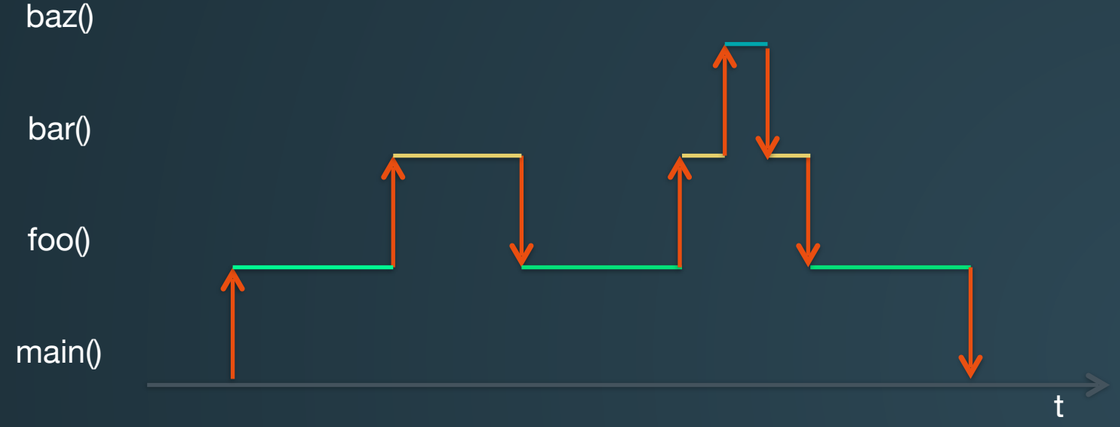
Instrumentation

Injection of measurement code.

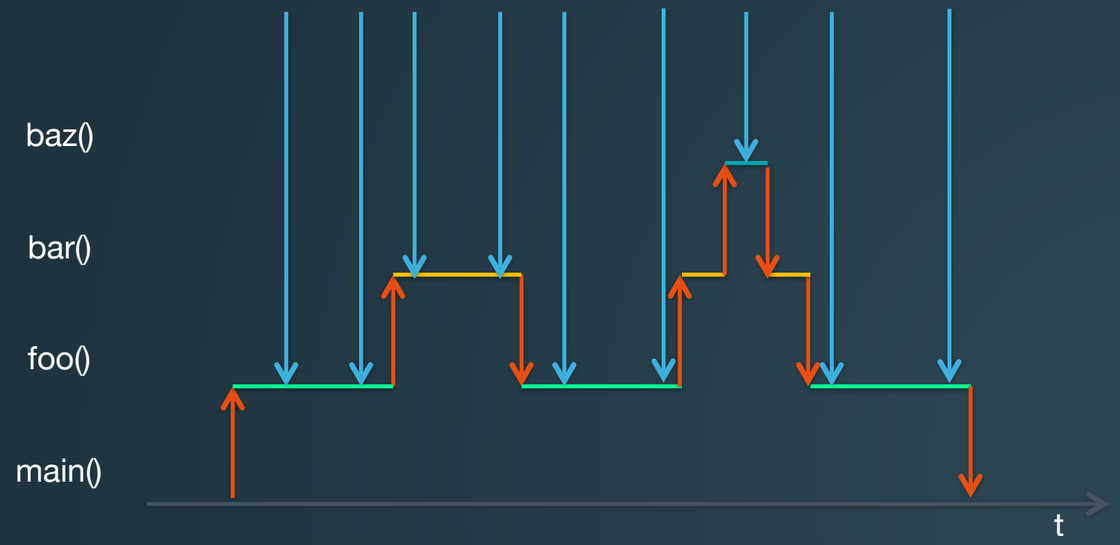
Sampling II

Reducing data by omission.

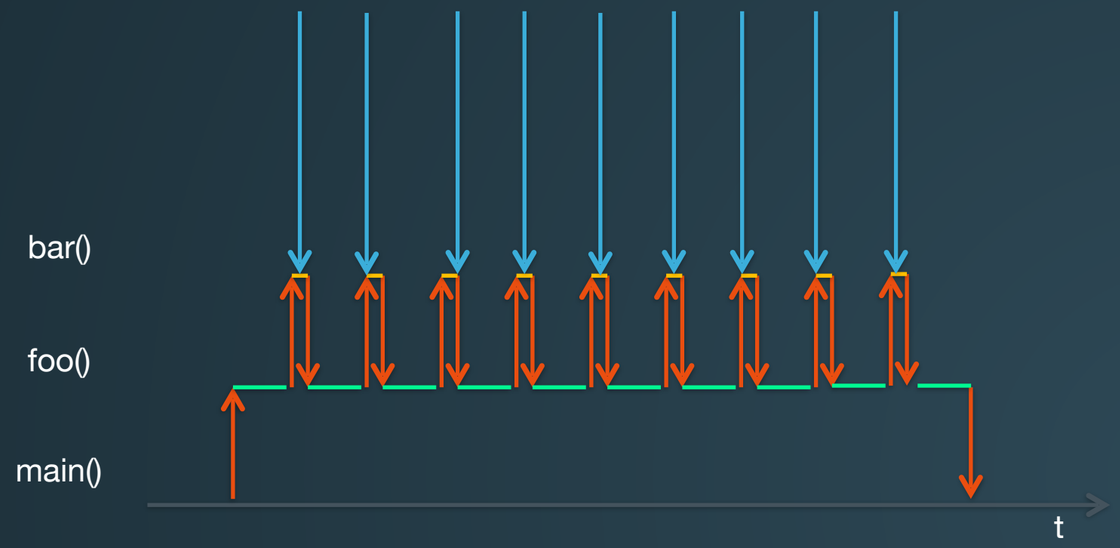
Sampling



Sampling - Great



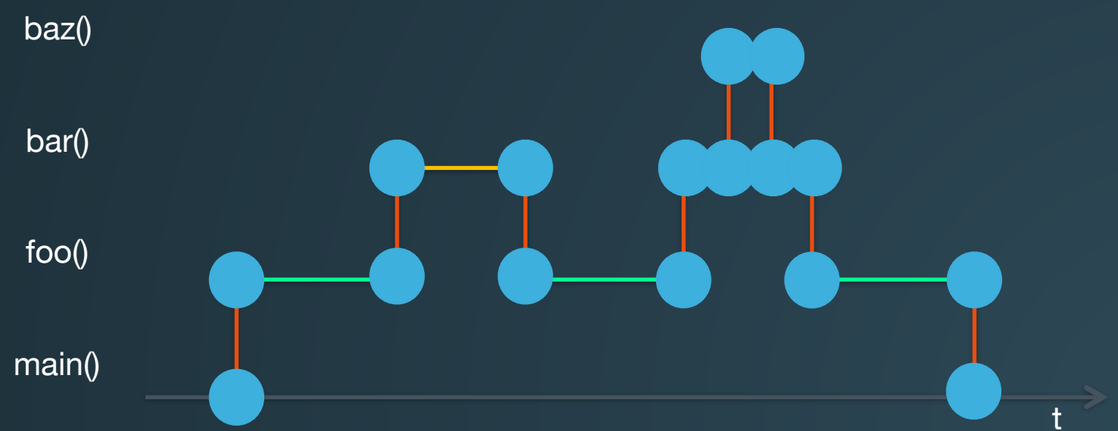
Sampling - Not So Great



Instrumentation

INSTANA®

Instrumentation



Sampling
vs
Instrumentation

DEMO

Code on Github

https://github.com/CodingFabian/SamplingVsInstrumentation

INSTANA®

Using JProfiler

DEMO

INSTANA®

Using HProf

DEMO

INSTANA®

Using Honest Profiler

DEMO

INSTANA®

Safepoints

INSTANA®

Safepoints

Sampling thread has to wait for steady state to interrogate other threads.

Safepoints are in-between code, so conceptually sampling never sees running code.

Honest Profiler uses JVMTI AsyncGetCallTrace which does not wait for safepoints.

github.com/RichardWarburton/honest-profiler

jeremymanson.blogspot.co.uk/2013/07/lightweight-asynchronous-sampling.html

From My Daily Work

INSTANA®

Performance Tuning Guide

Start off with Sampling

Do not take results to serious

Look for bottlenecks

< 10 ms is most of the time irrelevant when profiling

Get better results from benchmarking

Check code, bytecode, assembly

How Much Influence has Instrumentation Code?

DEMO

INSTANA®

Does My Profiler Tell The Truth?

NO

INSTANA®

Use JMH for Benchmarks

INSTANA®

Further Reading

Dapper, a Large-Scale Distributed Systems Tracing Infrastructure

static.googleusercontent.com/media/research.google.com/de//pubs/archive/36356.pdf

Evaluating the Accuracy of Java Profilers

www-plan.cs.colorado.edu/klipto/mytkowicz-pldi10.pdf

How to Measure Java Performance

blog.codecentric.de/en/2011/10/measure-java-performance-sampling-or-instrumentation/

Java Microbenchmark Harness

openjdk.java.net/projects/code-tools/jmh/

Richard Warbutons Honest Profiler

github.com/RichardWarburton/honest-profiler

Want to know more?

@CodingFabian

fabian.lange@instana.com

speakerdeck.com/CodingFabian

github.com/CodingFabian

INSTANA