Analyzing Memory Errors in Production

Markus Weninger

Johannes Kepler University Linz, Austria
Institute for System Software
WHO ARE WE?
WHO ARE WE?
WHO ARE WE?
WHO ARE WE?
WHO ARE WE?
WHO ARE WE?
WHO ARE WE?

For what?
To find and fix memory anomalies!
MOTIVATION

Fixing Memory Problems In Production
MOTIVATION

Jake
MOTIVATION

Jake

July
MOTIVATION

Jake

July
MOTIVATION

Fixing Memory Problems In Production

Jake

July

OutOfMemoryError
MOTIVATION

OutOfMemoryError

Jake

July
MOTIVATION

Jake

July

OutOfMemoryError

Slow downs
MOTIVATION

OutOfMemoryError

Jake

July

Slow downs
MOTIVATION

Fixing Memory Problems in Production

Jake

July

OutOfMemoryError

Slow down
* not actually Jake
Fixing Memory Problems In Production

Why is my program crashing?

* not actually Jake
Let’s try a memory analysis tool!

* not actually Jake
Problems
PROBLEM

Fixing Memory Problems In Production
Fixing Memory Problems In Production
I am new to all of this!
PROBLEM

Fixing Memory Problems In Production

I am new to all of this!

What should I look for?

I am new to all of this!
Fixing Memory Problems in Production

I am new to all of this!

What should I look for?

When to use which feature?

I am new to all of this!
PROBLEM

I am new to all of this!

What should I look for?

<another question about memory analysis> ...

When to use which feature?

I am new to all of this!
POLL TIME
Poll Time:

- Used a memory analysis tool before: Dummy Series 1
- Manually took a heap dump: Dummy Series 2
- Knows what trace-based memory analysis is: Dummy Series 3
- Could describe what a dominator tree is: Dummy Series 3
USERS NEED GUIDANCE
MEMORY ANALYSIS 101
MEMORY ANALYSIS 101

Top-down analysis
MEMORY ANALYSIS 101

Top-down analysis

GC root
MEMORY ANALYSIS 101

Top-down analysis

GC root

Fixing Memory Problems In Production
MEMORY ANALYSIS 101

Top-down analysis

GC root
MEMORY ANALYSIS 101

Top-down analysis

GC root
MEMORY ANALYSIS 101

Top-down analysis

GC root
MEMORY ANALYSIS 101

Top-down analysis

GC root

Which single object / GC root keeps the most other objects alive?
MEMORY ANALYSIS 101

Top-down analysis

Bottom-up analysis

Which single object / GC root keeps the most other objects alive?
MEMORY ANALYSIS 101

Top-down analysis

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis
MEMORY ANALYSIS 101

Top-down analysis

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis
MEMORY ANALYSIS 101

Top-down analysis

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis

...
MEMORY ANALYSIS 101

Top-down analysis

GC root

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis

...
MEMORY ANALYSIS 101

Top-down analysis

GC root

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis

...
MEMORY ANALYSIS 101

Top-down analysis

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis

Which single object / GC root keeps the most other objects alive?
Fixing Memory Problems In Production

MEMORY ANALYSIS 101

Top-down analysis

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis

Which single object / GC root keeps the most other objects alive?
MEMORY ANALYSIS 101

Top-down analysis

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis

Fixing Memory Problems In Production
MEMORY ANALYSIS 101

Top-down analysis

Which single object / GC root keeps the most other objects alive?

Bottom-up analysis

Analyze why a certain group of objects accumulates over time

GC root

1 Mio.

10,000

...
DOMINATOR TREE

Object Reference Graph

GC Roots

L1

N1 → D1 → D6 → N6
N2 → D2 → D7 → N7
N3 → D3 → D8 → N8
N4 → D4 → N9
N5 → D5 → N10

L2

DOMINATOR TREE

Object Reference Graph

GC Roots

Dom. Tree Root

Dominator Tree

Fixing Memory Problems In Production
### DOMINATOR TREE

#### Object Reference Graph

- **GC Roots**
  - L1
  - N1
  - N2
  - N3
  - N4
  - N5
  - D1
  - D2
  - D3
  - D4

#### Dominator Tree

- **Dom. Tree Root**
  - D1
  - D2
  - D3
  - D4
  - D5
  - L1
  - N1
  - N2
  - N3
  - N4
  - N5
  - N6
  - N7
  - N8
  - N9
  - N10

---

Fixing Memory Problems In Production
DOMINATOR TREE

Object Reference Graph

GC Roots

L1
N1 D1
N2 D2
N3 D3
N4 D4
N5 D5

D6 N6

Dominator Tree

Dom. Tree Root

N1
L1

N6
D6

N7
D7

N8
D8

N9
D9

N10

What happens if we could free L1?
DOMINATOR TREE

Object Reference Graph

<table>
<thead>
<tr>
<th>GC Roots</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
</tr>
<tr>
<td>N1 -&gt; D1</td>
</tr>
<tr>
<td>N2 -&gt; D2</td>
</tr>
<tr>
<td>N3 -&gt; D3</td>
</tr>
<tr>
<td>N4 -&gt; D4</td>
</tr>
<tr>
<td>N5 -&gt; D5</td>
</tr>
</tbody>
</table>

| L2       |
| D6 -> N6 |
| D7 -> N7 |
| D8 -> N8 |
| D9 -> N9 |

| D1       |
| D2       |
| D3       |
| D4       |
| D5       |

Dominator Tree

<table>
<thead>
<tr>
<th>Dom. Tree Root</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1</td>
</tr>
<tr>
<td>N1</td>
</tr>
<tr>
<td>N2</td>
</tr>
<tr>
<td>D6</td>
</tr>
<tr>
<td>D7</td>
</tr>
<tr>
<td>D8</td>
</tr>
<tr>
<td>N3</td>
</tr>
<tr>
<td>N4</td>
</tr>
<tr>
<td>D9</td>
</tr>
<tr>
<td>D10</td>
</tr>
</tbody>
</table>

| L2             |
| N6             |
| N7             |
| N8             |
| N9             |
| N10            |

These can be collected too
Thanks Markus, but that seems like a lot to learn...
Thanks Markus, but that seems like a lot to learn...
Thanks Markus, but that seems like a lot to learn...

Okay, let us try to improve that in AntTracks.
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase

GC phase

Mut. phase
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase

new X()

GC phase

Mut. phase

Fixing Memory Problems In Production
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase

new X()

GC phase

Mut. phase
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

new X()

Mut. phase

GC phase

Mutator phase

Trace file
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase

new X()
new Y()

GC phase

Mut. phase

Trace file
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase

new X()
new Y()
new Z()

GC phase

Mut. phase

Trace file

Fixing Memory Problems In Production
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

new X()
new Y()
new Z()
new Y()

Mutator phase

GC phase

Mut. phase

Trace file

Fixing Memory Problems In Production

13
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
- new X()
- new Y()
- new Z()
- new Y()

GC phase
- move

Trace file
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
new X()
new Y()
new Z()
new Y()

GC phase
move
move

Mut. phase

Trace file

Fixing Memory Problems In Production
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
- new X()
- new Y()
- new Z()
- new Y()

Mut. phase

GC phase
- move
- move
- move

Trace file
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
new X()
new Y()
new Z()
new Y()

GC phase
move
move
move

Mut. phase
new Y()

Trace file

Fixing Memory Problems In Production
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
- new X()
- new Y()
- new Z()
- new Y()

Mut. phase

GC phase
- move
- move
- move

Mut. phase
- new Y()
- new Z()

Trace file

Fixing Memory Problems In Production
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
- new X()
- new Y()
- new Z()
- new Y()

Mut. phase

GC phase
- move
- move
- move

AntTracks Analyzer
Memory anomaly detection
Memory leak analysis

Trace file
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
- new X()
- new Y()
- new Z()
- new Y()

Mut. phase
- move
- move
- move

GC phase
- new Y()
- new Z()

Trace file

AntTracks Analyzer
Memory anomaly detection
Memory leak analysis

Heap 1
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase

new X()
new Y()
new Z()
new Y()
move
move
move

GC phase

move
move
move

Mut. phase

new Y()
new Z()

Trace file

AntTracks Analyzer

Memory anomaly detection

Memory leak analysis

Heap 1
ANTTRACKS SYSTEM OVERVIEW

**AntTracks VM**

- **Application**
  - Mutator phase: new X(), new Y(), new Z(), new Y()
  - GC phase: move, move, move
  - Mut. phase: new Y(), new Z()

**AntTracks Analyzer**

- Memory anomaly detection
- Memory leak analysis

- Trace file

- Heap 1
- Heap 2

Fixing Memory Problems In Production
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
- new X()
- new Y()
- new Z()

Mut phase
- new Y()

GC phase
- move
- move
- move

AntTracks Analyzer

Memory anomaly detection
Memory leak analysis

Heap 1
Heap 2
... Heap n

Trace file
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

New X()
New Y()
New Z()
New Y()

Mutator phase

GC phase

Move
Move
Move

Mut. phase

new X()
new Y()
new Z()

Trace file

AntTracks Analyzer

Memory anomaly detection

Memory leak analysis

Heap 1
Heap 2
... Heap n
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
- new X(
- new Y(
- new Z(
- new Y(

Mut. phase
- move
- move
- move

GC phase
- new Y(
- new Z(

Trace file

AntTracks Analyzer
Memory anomaly detection
Memory leak analysis

Heap 1
Heap 2
Heap n
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
new X()
new Y()
new Z()
new Y()

Mut. phase

GC phase
move
move
move

Trace file

AntTracks Analyzer
Memory anomaly detection
Memory leak analysis

Heap 1
Heap 2
... Heap n

Fixing Memory Problems In Production
ANTTRACKS SYSTEM OVERVIEW

AntTracks VM

Application

Mutator phase
- new X()
- new Y()
- new Z()
- new Y()

GC phase
- move
- move
- move

Mut. phase
- new Y()
- new Z()

Trace file

AntTracks Analyzer
Memory anomaly detection
Memory leak analysis

Heap 1
Heap 2
Heap n
GUIDED EXPLORATION

<table>
<thead>
<tr>
<th>Application</th>
<th>Monitoring Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring Data</td>
<td></td>
</tr>
</tbody>
</table>

Fixing Memory Problems In Production
**GUIDED EXPLORATION**

<table>
<thead>
<tr>
<th>Application</th>
<th>Monitoring Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring Data</td>
<td>![Detection Diagram]</td>
</tr>
</tbody>
</table>

**Detection** describes the task of automatically detecting a potential problem, i.e., a suspicious pattern.
## GUIDED EXPLORATION

<table>
<thead>
<tr>
<th>Application</th>
<th>Monitoring Tool</th>
</tr>
</thead>
</table>
| Monitoring Data | ![Diagram](image)

### Detection

*Detection* describes the task of automatically detecting a potential problem, i.e., a suspicious pattern.

### Highlighting

*Highlighting* the relevant region on the UI helps users to understand where the automatically gained insight can be found if the view was inspected manually.
## GUIDED EXPLORATION

<table>
<thead>
<tr>
<th>Application</th>
<th>Monitoring Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring Data</td>
<td>![Diagram of Monitoring Tool]</td>
</tr>
</tbody>
</table>

1. **Detection** describes the task of automatically detecting a potential problem, i.e., a suspicious pattern.

2. **Highlighting** the relevant region on the UI helps users to understand where the automatically gained insight can be found if the view was inspected manually.

3. **Explanation** of the highlighted visualization helps users to interpret it and explains concepts that are needed for this interpretation.
GUIDED EXPLORATION

<table>
<thead>
<tr>
<th>Application</th>
<th>Monitoring Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring Data</td>
<td>⬛ ⬛ ⬛ ⬛</td>
</tr>
</tbody>
</table>

**Detection** describes the task of automatically detecting a potential problem, i.e., a suspicious pattern.

**Highlighting** the relevant region on the UI helps users to understand where the automatically gained insight can be found if the view was inspected manually.

**Explanation** of the highlighted visualization helps users to interpret it and explains concepts that are needed for this interpretation.

**Suggestions** on which steps could / should be taken next make it easier for the user to understand what operations are possible and why they might be useful.
GUIDED EXPLORATION

Detected describes the task of automatically detecting a potential problem, i.e., a suspicious pattern.

Highlighting the relevant region on the UI helps users to understand where the automatically gained insight can be found if the view was inspected manually.

Explanation of the highlighted visualization helps users to interpret it and explains concepts that are needed for this interpretation.

Suggestions on which steps could / should be taken next make it easier for the user to understand what operations are possible and why they might be useful.
Overview

Search and book a journey on the easyTravel web frontend with a realistic multi-step booking process.

Start all the tiers of easyTravel and enable architecture and performance flaws by selecting a pre-configured scenario.

Preconfigured dashboards and business transactions show the full capabilities and value that dynaTrace delivers.

easyTravel provides a web portal which allows users to log in, search for journeys to various destinations, select promotional journeys directly that are offered and to book a journey using credit card details. Additionally, a Business-to-Business (B2B) web portal for travel agencies is provided where travel agencies can manage the journeys that they offer and can review reports about made bookings.

easyTravel is a multi-tier application implemented in .Net and Java. The starting of the various tiers and the enabling/disabling of different problem pattern plugins is done via a separate easyTravel Launcher. The Launcher allows the user to conveniently switch between different demo scenarios. Each scenario can define load scripts and certain problem pattern plugins that are enabled. The scenarios can be modified or extended by changing an XML file. This is useful when giving demos and allows you to focus on problem areas that are particularly relevant for a specific demo.

Download (Installer + License)

- easyTravel Demo License
- Latest easyTravel version
- easyTravel for AppMon 7.2
- easyTravel for AppMon 7.1
DEMO: OVERVIEW

Fixing Memory Problems In Production
DEMO: OVERVIEW
(1) DETECTION + (2) HIGHLIGHTING

Memory

GC Overhead

Fixing Memory Problems In Production
DEMO: OVERVIEW
(3) EXPLANATION + (4) SUGGESTION

**Potential memory leak!**

AntTracks has detected a timeframe over which the reachable memory is continuously growing. This is an indicator for a memory leak.

Memory leaks are often caused by **indefinitely growing data structures** - AntTracks can help you find these data structures by calculating their growth over time.

Also, if a memory leak exists, typically objects of a few common types accumulate over time. AntTracks can help you to identify these objects by visualizing the **evolution of the heap composition over time**.

**ANALYZE OBJECT GROUP TRENDS AND DATA STRUCTURE GROWTH**
DEMO: EVOLUTION

Fixing Memory Problems In Production
DEMO: EVOLUTION

Fixing Memory Problems In Production
Date objects allocated in the constructor of Base are the major suspects for a memory leak since about 30% of the overall heap growth is accounted to them.

Inspect who keeps them alive.
DEMO: GRAPH VIEW

Static field(1)

1 root pointer references 1 obj.

Date (137, 043)
DEMO: GRAPH VIEW
Fixing Memory Problems In Production
Too many `ArrayList<Location>` are added here.
ANOTHER DEMO: FINAGLE-HTTP
ANOTHER DEMO: FINAGLE-HTTP

Fixing Memory Problems In Production
ANOTHER DEMO: FINAGLE-HTTP

Objects

Time [ms] vs. Objects [Million]

- Eden
- Survivor
- Old
### ANOTHER DEMO: FINAGLE-HTTP

<table>
<thead>
<tr>
<th>Name</th>
<th>Collected objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>16,677,450</td>
</tr>
<tr>
<td>0 GCs survived</td>
<td>16,673,869</td>
</tr>
<tr>
<td>- Promise$WaitQueue$$anon$4</td>
<td>4,152,026</td>
</tr>
<tr>
<td>- Promise$Monitored</td>
<td>4,151,660</td>
</tr>
<tr>
<td>- Future$$anonfun$onSuccess$1</td>
<td>4,151,598</td>
</tr>
<tr>
<td>- FinagleHttp$$anonfun$runIteration$$anon$$anonfun$run$1$$anonfun$run1::ap</td>
<td>4,151,596</td>
</tr>
<tr>
<td>- FinagleHttp$$anonfun$runIteration$$anon$$anonfun$run$1$$anonfun$run1::ap</td>
<td>4,151,596</td>
</tr>
</tbody>
</table>
ANOTHER DEMO: FINAGLE-HTTP

<table>
<thead>
<tr>
<th>Name</th>
<th>Collected objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>16,677,450</td>
</tr>
<tr>
<td>0 GCs survived</td>
<td>16,673,869</td>
</tr>
<tr>
<td>Promise$WaitQueue$$anon$4</td>
<td>4,152,026</td>
</tr>
<tr>
<td>Promise$Monitored</td>
<td>4,151,660</td>
</tr>
<tr>
<td>Future$$anonfun$onSuccess$1</td>
<td>4,151,598</td>
</tr>
<tr>
<td>FinagleHttp$$anonfun$runIteration$1$$anon$2$$anonfun$run$1$$anon$2$$anonfun$run$1::api</td>
<td>4,151,596</td>
</tr>
<tr>
<td>FinagleHttp$$anonfun$runIteration$1$$anon$2$$anonfun$run$1::api</td>
<td>4,151,596</td>
</tr>
</tbody>
</table>
### ANOTHER DEMO: FINAGLE-HTTP

<table>
<thead>
<tr>
<th>Name</th>
<th>Collected objects</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Overall</strong></td>
<td>16,677,450</td>
</tr>
<tr>
<td><strong>0 GCs survived</strong></td>
<td>16,673,869</td>
</tr>
<tr>
<td>▶ Promise$WaitQueue$anon$4</td>
<td>4,152,026</td>
</tr>
<tr>
<td>▶ Promise$Monitored</td>
<td>4,151,660</td>
</tr>
<tr>
<td>▶ Future$anonfun$onSuccess$1</td>
<td>4,151,598</td>
</tr>
<tr>
<td>▶ FinagleHttp$anonfun$runIteration$1$anon$2$anonfun$run$1$anon</td>
<td>4,151,596</td>
</tr>
<tr>
<td>▶ FinagleHttp$anonfun$runIteration$1$anon$2$anonfun$run$1::ap</td>
<td>4,151,596</td>
</tr>
</tbody>
</table>
### ANOTHER DEMO: FINAGLE-HTTP

<table>
<thead>
<tr>
<th>Name</th>
<th>Collected objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>16,677,450</td>
</tr>
<tr>
<td>0 GCs survived</td>
<td>16,673,869</td>
</tr>
<tr>
<td>Promise$WaitQueue$$anon$4</td>
<td>4,152,026</td>
</tr>
<tr>
<td>Promise$Monitored</td>
<td>4,151,660</td>
</tr>
<tr>
<td>Future$$anonfun$onSuccess$1</td>
<td>4,151,598</td>
</tr>
<tr>
<td>FinagleHttp$$anonfun$runIteration$1$$anon$2$$anonfun$run$1$$anon$2$$anonfun$run$1::ap</td>
<td>4,151,596</td>
</tr>
<tr>
<td>FinagleHttp$$anonfun$runIteration$1$$anon$2$$anonfun$run$1::ap</td>
<td>4,151,596</td>
</tr>
</tbody>
</table>
ANOTHER DEMO: FINAGLE-HTTP

<table>
<thead>
<tr>
<th>Name</th>
<th>Collected objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>16,677,450</td>
</tr>
<tr>
<td>0 GCs survived</td>
<td>16,673,869</td>
</tr>
<tr>
<td>Promise$WaitQueue$$anon$4</td>
<td>4,152,026</td>
</tr>
<tr>
<td>Promise$Monitored</td>
<td>4,151,660</td>
</tr>
<tr>
<td>Future$$anonfun$onSuccess$1</td>
<td>4,151,598</td>
</tr>
<tr>
<td>FinagleHttp$$anonfun$runIteration$1$$anon$2$$anonfun$run$1$$anon$2$$anonfun$run$1::ap</td>
<td>4,151,596</td>
</tr>
<tr>
<td>FinagleHttp$$anonfun$runIteration$1$$anon$2$$anonfun$run$1::ap</td>
<td>4,151,596</td>
</tr>
</tbody>
</table>
DEMO: FINAGLE-HTTP


for (i <- 0 until NUM_REQUESTS) {
    Await.result(response.onSuccess { rep: http.Response =>
        totalLength += rep.content.length
    })
}

DEMO: FINAGLE-HTTP
DEMO: FINAGLE-HTTP

```scala

for (i <- 0 until NUM_REQUESTS) {
  Await.result(response.onSuccess { rep: http.Response =>
    totalLength += rep.content.length
  })
}
```

for (i <- 0 until NUM_REQUESTS) {
    Await.result(response.onSuccess(rep: http.Response =>
        totalLength += rep.content.length))
}

val h = { rep: http.Response =>
    totalLength += rep.content.length
}

for (i <- 0 until NUM_REQUESTS {
    Await.result(response.onSuccess(h))
}

for (i <- 0 until NUM_REQUESTS) {
    Await.result(response.onSuccess {
        rep: http.Response =>
        totalLength += rep.content.length
    })
}

val h = {
    rep: http.Response =>
    totalLength += rep.content.length
}

for (i <- 0 until NUM_REQUESTS) {
    Await.result(response.onSuccess(h))
}

for (i <- 0 until NUM_REQUESTS) {
    Await.result(response.onSuccess { rep: http.Response =>
        totalLength += rep.content.length
    })
}

val h = { rep: http.Response =>
    totalLength += rep.content.length
}

for (i <- 0 until NUM_REQUESTS {
    Await.result(response.onSuccess(h))
}
DEMO: FINAGLE-HTTP

```scala
for (i <- 0 until NUM_REQUESTS) {
    Await.result(response.onSuccess { rep: http.Response =>
        totalLength += rep.content.length
    })
}
val h = { rep: http.Response =>
    totalLength += rep.content.length
}
for (i <- 0 until NUM_REQUESTS) {
    Await.result(response.onSuccess(h))
}
```
DEMO: FINAGLE-HTTP

```scala
for (i <- 0 until NUM_REQUESTS) {
  Await.result(response.onSuccess { rep: http.Response =>
    totalLength += rep.content.length
  })
}
val h = { rep: http.Response =>
  totalLength += rep.content.length
}
for (i <- 0 until NUM_REQUESTS) {
  Await.result(response.onSuccess(h))
}
```
TAKE-AWAYS
TAKE-AWAYS

Memory Analysis

Top-down
Bottom-up

Heap dumps

Trace-based

... and 100 other things
Memory Analysis

- Top-down
- Bottom-up
- Heap dumps
- Trace-based
- ... and 100 other things

Problem

New users need guidance to get started faster

It is not enough to „throw tools at them“
TAKE-AWAYS

Memory Analysis

- Top-down
- Bottom-up
- Heap dumps
- Trace-based

... and 100 other things

Problem

- New users need guidance to get started faster
- It is not enough to „throw tools at them“

Guided Exploration

- Detection
- Highlighting
- Explanation
- Suggestion
TAKE-AWAYS

Memory Analysis
Top-down
Bottom-up
Heap dumps
Trace-based
... and 100 other things

Problem
New users need guidance to get started faster
It is not enough to „throw tools at them“

Guided Exploration
Detection
Highlighting
Explanation
Suggestion

Markus Weninger
Johannes Kepler University
Linz, Austria
markus.weninger@jku.at
http://mevss.jku.at/AntTracks
Additional Notes

- Some icons made by Freepik & Smashicons from https://www.flaticon.com
- Some photos made by bruce mars and Isaque Pereira from Pexels