

# Dirty Hacks With Java Reflection (includes one or two useful hints)

**Dr Heinz M. Kabutz**

Last updated 2016-09-06



Javaspecialists.eu  
java training

## Short Introduction to Speaker

- **Heinz Kabutz**

- Born in Cape Town, South Africa, now live on Crete
- PhD Computer Science from University of Cape Town
  - University famous for world's first successful heart transplant

- **Created The Java Specialists' Newsletter**

- Monthly advanced newsletter for Java professionals
- <http://www.javaspecialists.eu>

- **One of the first Java Champions**

- <https://java-champions.dev.java.net/>





## Reflection is like Opium

- **A bit too strong for every day use**
  - But can relieve serious pain
- **Please do not become a reflection addict!**

## Modifying/Reading Private/Final Fields

- **We can access private fields by making it accessible**
  - Requires security manager support
- **Note: value field is final and private!**

```
import java.lang.reflect.*;
```

```
public class PrivateFinalFieldTest {  
    public static void main(String... args)  
        throws NoSuchFieldException, IllegalAccessException {  
        Field value = String.class.getDeclaredField("value");  
        value.setAccessible(true);  
        value.set("hello!", "cheers".toCharArray());  
        System.out.println("hello!");  
    }  
}
```

**cheers**

## Optimization methodology

- 1. Load test to identify bottlenecks**
    - Identify the easiest to fix
  - 2. Derive a hypothesis for the cause of the bottleneck**
    - Create a test to isolate the factor identified by the hypothesis
      - This is important, we have often been fooled by profilers!
  - 3. Alter the application or configuration**
  - 4. Test that the change improves the situation**
    - Also make sure the system still works correctly
- Repeat process until targets are met

## Big Gains Quickly

- **Amdahl's law applies**

- **Consider an 4 layered application**

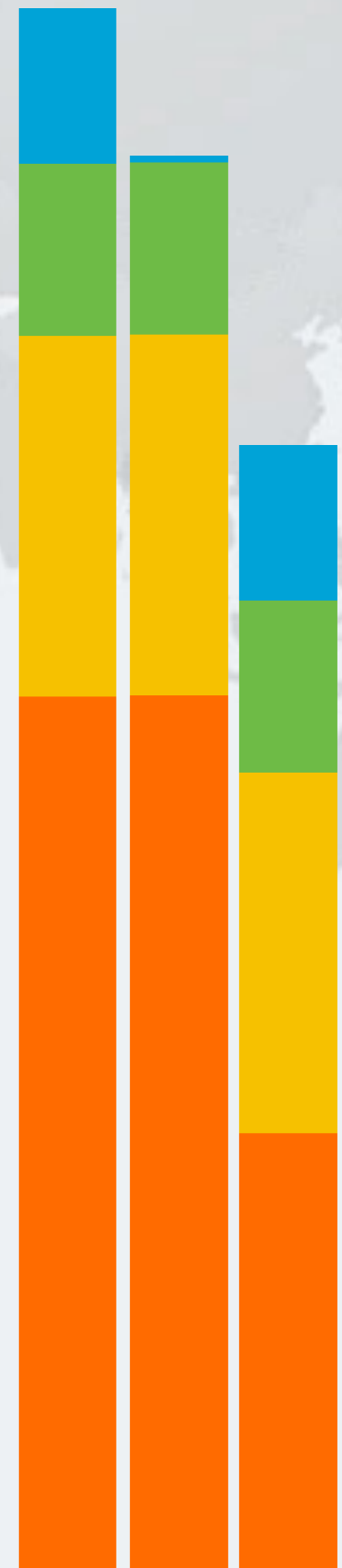
- **Servlet takes 10%**
- **Business component takes 11%**
- **EJB takes 23%**
- **SQL takes 56%**

- **Scenario 1, tuning Servlet gives 20x improvement**

- **"Google" says that servlets are slow**
- **$0.10/20 + 0.11/1 + 0.23/1 + 0.56 /1 = 0.905$**

- **Scenario 2, tuning SQL give 2x improvement**

- **We *measure* and discover SQL is the bottleneck**
- **$0.10/1 + 0.11/1 + 0.23/1 + 0.56/2 = 0.72$**



# System Overview - The Box

**People**

Usage Patterns,  
Rates

**Application**

Lock Contention

**JVM**

Garbage Collector,  
Number of Threads

**Hardware**

CPU, Memory,  
Disk, Network





# Dirty Hacks With Java Reflection (includes one or two useful hints)

**Dr Heinz M. Kabutz**

<http://www.javaspecialists.eu>

Twitter: @heinzkabutz

Email: [heinz@kabutz.net](mailto:heinz@kabutz.net)



Javaspecialists.eu  
java training