PerfCake
A Lightweight Performance Testing Framework

From owen directly to the plate

Martin Večeřa <mvecera@perfcake.org>
Pavel Macík <pmacik@perfcake.org>

November 2015
www.perfcake.org
Let’s see what the requirement reads:

ACME REST Service must respond in less than 6ms in 90% of cases.
Standard Measurement (88 runs)
Standard Measurement (real throughput)
PerfCake comes at hand
How to specify the scenario

scenario "Jetty Spring POST"
  run 1.m with 50.threads
  generator "DefaultMessageGenerator"
  sender "HttpSender"
    target "http://${server.host}:8181/jetty-spring"
    method "POST"
  reporter "IterationsPerSecondReporter"
    destination "ConsoleDestination" every 1.s
  message file: "5kB_message.xml"
end
PerfCake comes at hand
PerfCake comes at hand

![Graph showing response time against throughput for different load levels. The graph illustrates how PerfCake performs under varying load conditions, with colors representing different load levels (1 to 200).]
Maybe my SLA is defined in a better way:

ACME REST Service must respond in less than 6ms in 90% of cases with 50 parallel clients.
Hunt for a sweet spot

<table>
<thead>
<tr>
<th>Intervals</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;0.0; 3.0)</td>
<td>43.80%</td>
</tr>
<tr>
<td>&lt;3.0; 6.0)</td>
<td>49.12%</td>
</tr>
<tr>
<td>&lt;6.0; 9.0)</td>
<td>4.94%</td>
</tr>
<tr>
<td>&lt;9.0; 12.0)</td>
<td>2.14%</td>
</tr>
<tr>
<td>&gt;12.0</td>
<td>0.00%</td>
</tr>
</tbody>
</table>
Cool stuff we work on

PerfRepo recent acquisition

IoT – Raspberry Pi 2 vs. Pi

The report shows comparison of 4 test executions. Show test executions

<table>
<thead>
<tr>
<th>Test</th>
<th>Metric</th>
<th>Raspberry Pi 2</th>
<th>Raspberry Pi</th>
<th>Raspberry Pi 2 vs. Raspberry Pi</th>
</tr>
</thead>
<tbody>
<tr>
<td>IoT REST</td>
<td>throughput</td>
<td>813,22</td>
<td>40,90</td>
<td>94,97 %</td>
</tr>
<tr>
<td>IoT MQTT</td>
<td>throughput</td>
<td>309,58</td>
<td>89,85</td>
<td>70,98 %</td>
</tr>
</tbody>
</table>
Cool stuff we work on

Clustering
Arquillian integration

```java
public class Tests {
    @PerfCake(scenario = "perf_test.dsl")
    private PerfCakeController perfCake;

    public void firstTest() {
        perfCake.start();
        perfCake.getProgress();
        ...
    }
}
```
Let’s do some real work

Demo

RaspberryPi 2, MQTT, REST