Android Performance Bundle
Performance on Android

Mobile game revenue to pass console, PC for first time

Arjun Kharpal | @ArjunKharpal
Friday, 22 Apr 2016 | 5:54 AM ET

Smartphones Overtake Computers as Top E-Commerce Traffic Source

Smartphone And Tablet Revenue Is Bigger Than The Entire Consumer Electronics Market
Measuring is crucial!

1. Automated UI Testing
Measuring is crucial!

1. Automated UI Testing

2. Real world benchmarking of phones
Measuring is crucial!

1. Automated UI Testing
2. Real world benchmarking of phones
3. Benchmarking in the app store
Performance on Android

Energy leak

GUI Lagging

Memory Bloating
Performance on Android

Energy leak

GUI Lagging

Memory Bloating
What does GUI lagging mean?

• „App not responding“
What does GUI lagging mean?

- „Janky“ frames
  - 60 fps Framerate
  - = 16-ms-per-frame
What does GUI lagging mean?

- „Janky“ frames
  - 60 fps Framerate
  - = 16-ms-per-frame
How to measure fluency?
How to measure fluency?

- FPS
- Rendering time
- Issued frames
“Analyzing GUI Running Fluency for Android Apps”

State Key Lab of Computer Science, Institute of Software, Chinese Academy of Sciences
Beijing, China
Existing tools

Android Systrace

Profile GPU Rendering
Existing tools

**FPS Meter**

**Instrument the source code**
ARFluency - Implementation

<<Process>>
SurfaceFlinger

notify

AppContentLayer

<<Layer>>
Content Layer

Hardware Composer

draw

Android App
ARFluency - Implementation

Code injection: libsurfaceflinger.so

<<Process>>
SurfaceFlinger

Hardware Composer

notify
draw

Dumpsys

AppContentLayer

StatusBar Layer

Content Layer

Android App
Experiments

<table>
<thead>
<tr>
<th>Application</th>
<th>Issued frames (%)</th>
<th>Frame rate (fps)</th>
<th>Rendering time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td>1</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>Shadowsocks</td>
<td>1</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>Google Translate</td>
<td>1</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>Google Photos</td>
<td>4</td>
<td>24</td>
<td>14</td>
</tr>
<tr>
<td>Antutu Benchmark</td>
<td>16</td>
<td>26</td>
<td>13</td>
</tr>
</tbody>
</table>

23.05.2017
Android Performance Bundle - Noah Hollmann
Critique
Different perspectives
Many device configurations
Many device configurations
Many device configurations
Android is unpredictable
Android is unpredictable
“Mining Test Repositories for Automatic Detection of UI Performance Regressions in Android Apps”

13th International Conference on Mining Software Repositories (MSR’16), May 2016, Austin, Texas, United States. IEEE, 2016, Proceedings of the 13th International Conference on Mining Software Repositories.
“Mining Test Repositories for Automatic Detection of UI Performance Regressions in Android Apps”

13th International Conference on Mining Software Repositories (MSR’16), May 2016, Austin, Texas, United States. IEEE, 2016, Proceedings of the 13th International Conference on Mining Software Repositories.
Automated Testing

Test Suite (UI events) → Test execution With Robotium and Genymotion → Dumpsys Logcat → Test Repository

Simulated device profile
The diagram shows the time (ms) per frame for different stages of an application's execution. The chart includes metrics for executing, processing, and drawing operations.

- **Speed Limit**: 16 mspf
- **BUCKET1**: #Janky=3
- **BUCKET2**: #Janky=0
- **BUCKET3**: #Janky=1
- **BUCKET4**: #Janky=3
- **EXIT**: #Janky=3

The chart highlights areas where the application performance is considered janky, with E1, E2, and E3 indicating critical points in the execution process.
[v1, 4.1, samsung, S3, x86, wifi]
Automated Testing

Test Suite (UI events) → Test execution With Robotium and Genymotion → Compare to test repository

Simulated device profile

Dumpsys Logcat
Finding similar configurations

[v1; 4:1; samsung; S3; x86; wifi]
Finding similar configurations

[v1; 4:1; samsung; S3; x86; wifi]
[v1:2; 4:1; samsung; S3; x86; wifi]
Finding similar configurations

[v1; 4:1; samsung; S3; x86; wifi]
[v1:2; 4:1; samsung; S3; x86; wifi]
Finding similar configurations

[v1; 4:1; samsung; S3; x86; wifi]
[v1:2; 4:1; samsung; S3; x86; wifi]
[v1:1; 4:1; samsung; S3; x86; wifi]
[v1; 4:2; samsung; S3; x86; wifi]
[v1:2; 4:1; samsung; S3; x86; 4G]
Finding outliers

Most Similar test runs → Outliers → Similar outliers (Outliers+, Outliers-)
Association Rule Mining

V5.0.0 => Outlier-
Wifi => Outlier+
Evaluation

Exo-Player

Space Blaster

K-9 Mail client
Evaluation

0 Questions
13 Tasks
6 Bugs

Current Status:
- New: 42.1%
- In progress: 31.6%
- Resolved: 26.3%

Raised vs Resolved:
- Resolved
- Raised

23.05.2017
Evaluation

Device Lab

SDKs: 4.*

Upgraded Device Lab

SDKs: 5.*, 6.0.0
Experiment - Performance Degradation

![Graphs showing performance comparison between two different API levels](image_url)
Experiment - Performance Degradation

10/10
Experiment – Context isolation
Experiment – UI Event filtering

20 Events / 4 Janks

Precision: 53%
Recall: 83%
Critique
What you could do...
WHAT REALLY HAPPENS IN TEST CONFERENCES

REPEAT AFTER ME:
TESTERS ARE VALUABLE
WHAT WE DO IS INTERESTING
AND IMPORTANT

PRESENTED
BY
TEST GURU

TESTERS ARE VALUABLE
WHAT WE DO IS INTERESTING
AND IMPORTANT

AG.
“How Developers Detect and Fix Performance Bottlenecks in Android Apps”

Mario Linares-Vásquez, Christopher Vendome, Qi Luo, Denys Poshivanyk
The College of William and Mary, Williamsburg, VA, USA

485 participants  Sept 2015  > 1 year experience
Performance on Android

- Energy leak
- GUI Lagging
- Memory Bloating
Datacollection

• 381,161 Java projects
• 16,331 potential repositories
• 24,340 contributors
• 628 survey responses
• 485 valid responses

Valid response rate: 2% (out of 24,340 contributors)
Preferred practices for detecting bottlenecks

- R + M
- M
- R + M + T
- R
- M + T + O
- T

Review
Manual
Tools
Other
There are many tools available!
Preferred practices for detecting bottlenecks

- R + M: 115
- M: 111
- R + M + T: 71
- R: 56
- M + T + O: 21
- T: 17

Legend:
- Review
- Manual
- Tools
- Other
GUI Lagging: Solution Practices

<table>
<thead>
<tr>
<th>Practice</th>
<th>Number of People</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threads</td>
<td>74</td>
</tr>
<tr>
<td>GUI Optimization</td>
<td>47</td>
</tr>
<tr>
<td>Caching</td>
<td>40</td>
</tr>
<tr>
<td>Memory Management</td>
<td>32</td>
</tr>
<tr>
<td>Network Resources (Cloud)</td>
<td>17</td>
</tr>
<tr>
<td>Delegation</td>
<td>4</td>
</tr>
</tbody>
</table>

Number of people
Memory bloat: Solution Practices

- GUI Optimization: 45 people
- Memory Mgmt.: 30 people
- Network Resources: 3.5 people
- On-Demand Resources: 2 people

Number of people
Energy leak: Solution Practices

- DELEGATION (CLOUD)
- REDUCE GPS CALLS
- REDUCE WAKELOCKS

Number of people
Critique
Papers & Sources

• **Mining test repositories for automatic detection of UI performance regressions in Android apps**
  María Gómez, Romain Rouvoy, Bram Adams and Lionel Seinturier. MSR 2016.

• **How developers detect and fix performance bottlenecks in Android apps**
  Mario Linares-Vásquez, Christopher Vendome, Qi Luo and Denys Pshyvanyk. ICSME 2015.

• **Analyzing GUI running fluency for Android Apps**

• **Android developers**