Creating & Scaling Meaningful Performance Tests in CI

Ryan Brooks

About me

- Android lead at Emerge Tools

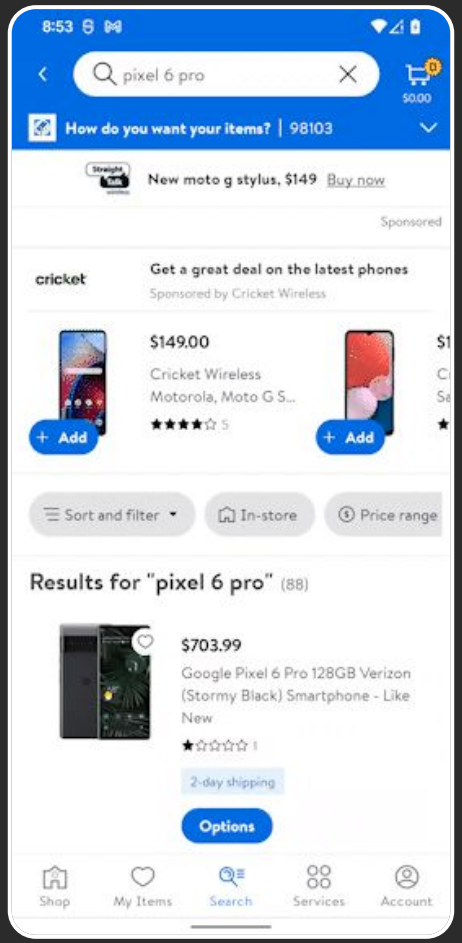
- Formerly Android engineer at Airbnb working on SDUI (Ghost platform)

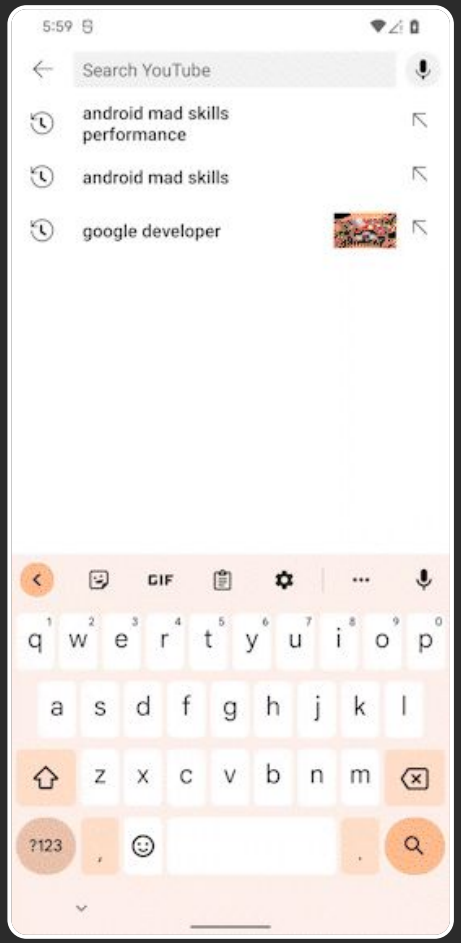
Overview

- What is mobile performance?
- A framework for choosing what to test
- Strategies to measure & test
- Scaling perf tests & perf testing tradeoffs

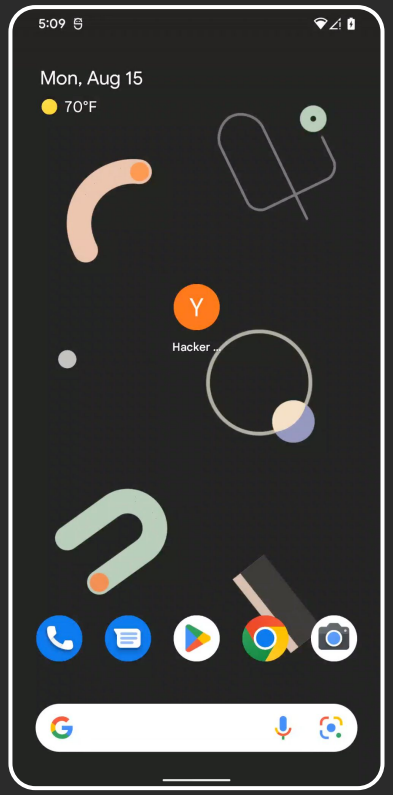
What is mobile performance?


10:14
Wed, Aug 17
87°F
Hacker ...

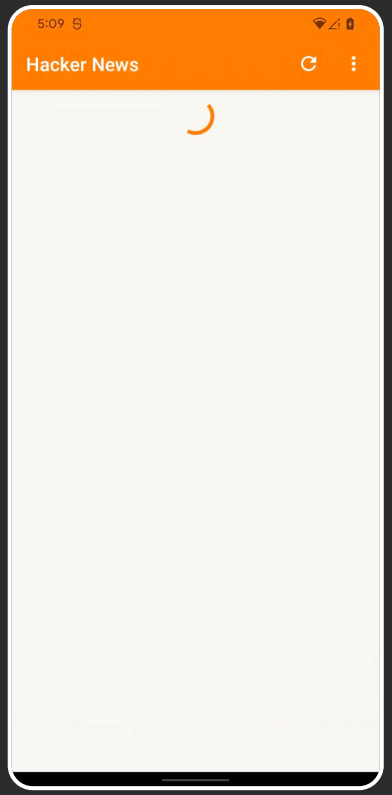

8:53
pixel 6 pro
$0.00
How do you want your items? | 98103
New moto g stylus, $149 Buy now
Sponsored
cricket
Get a great deal on the latest phones
Sponsored by Cricket Wireless
$149.00
Cricket Wireless
Motorola, Moto G S...
Add
Add
Sort and filter
In-store
Price range
Results for "pixel 6 pro" (88)
$703.99
Google Pixel 6 Pro 128GB Verizon (Stormy Black) Smartphone - Like New
2-day shipping
Options
Shop
My Items
Search
Services
Account

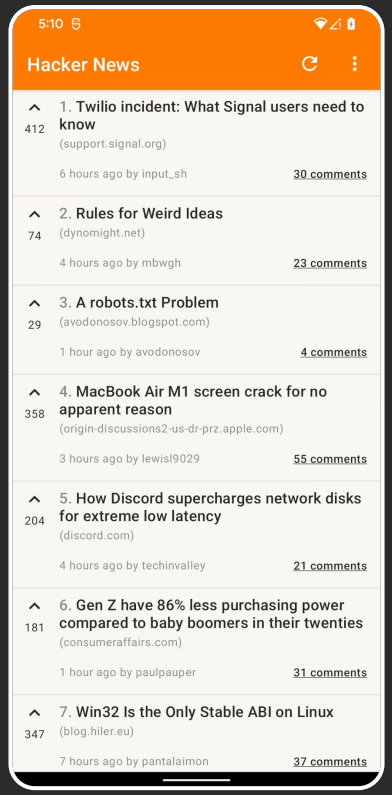

5:59
Search YouTube
android mad skills performance
android mad skills
google developer
GIF

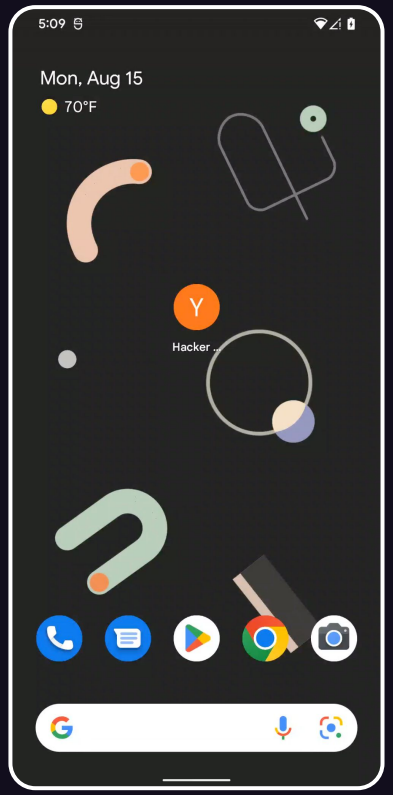

10:14
Wed, Aug 17
87°F
Hacker ...


5:09
Mon, Aug 15
70°F
Hacker ...


5:09

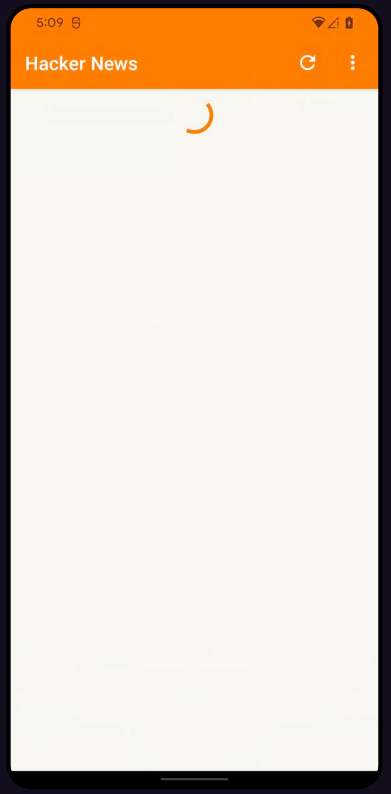

5:09
Hacker News


5:10
Hacker News
412
1. Twilio incident: What Signal users need to know
(support.signal.org)
6 hours ago by input_sh
30 comments
74
2. Rules for Weird Ideas
(dynomight.net)
4 hours ago by mbwgh
23 comments
29
3. A robots.txt Problem
(avodonosov.blogspot.com)
1 hour ago by avodonosov
4 comments
358
4. MacBook Air M1 screen crack for no apparent reason
(origin-discussions2-us-dr-prz.apple.com)
3 hours ago by lewisl9029
55 comments
204
5. How Discord supercharges network disks for extreme low latency
(discord.com)
4 hours ago by techinvalley
21 comments
181
6. Gen Z have 86% less purchasing power compared to baby boomers in their twenties
(consumeraffairs.com)
1 hour ago by paulpauper
31 comments
347
7. Win32 Is the Only Stable ABI on Linux
(blog.hiler.eu)
7 hours ago by pantalaimon
37 comments

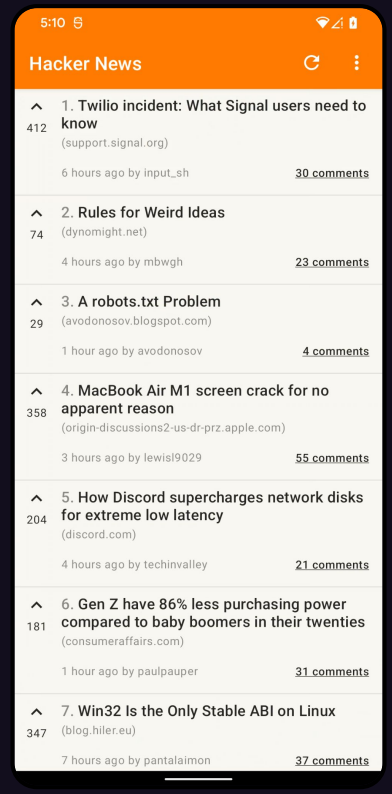


- Process start
- Dex/resources loaded
- JIT compilation
- bindApplication (Application.onCreate)
- Content providers & SDK inits

- MainActivity onCreate
- Dependencies injected/instantiated
- ViewModels/fragments created
- First frame render (time to initial display)



- Fetch data from backend
- Transform data
- Render components
- First contentful paint (time to full display)
  - Indicated by reportFullyDrawn

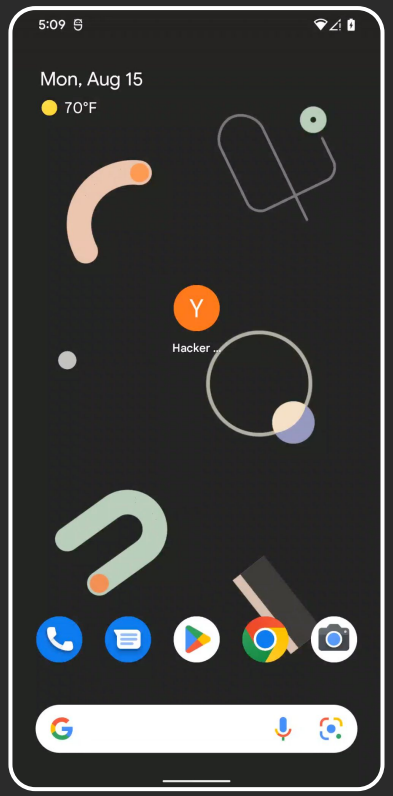


What is mobile performance?

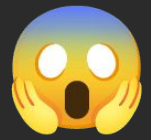
- Time to initial display
  - SDK inits
  - Android component creations (activities, fragments, etc)
  - Dependency instantiation & injection
- Time to full display
  - Network request
  - Rendering UI components

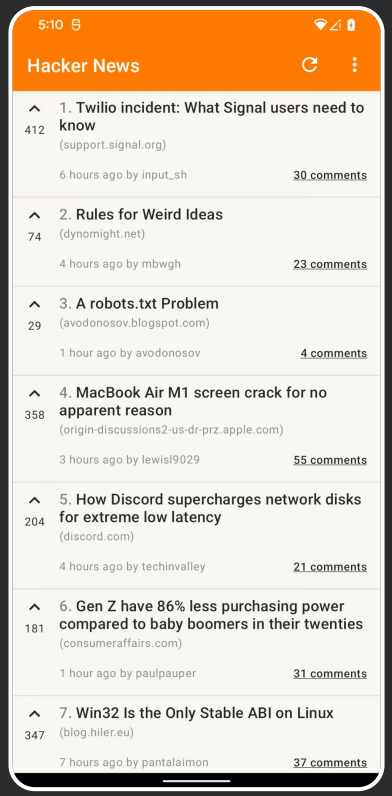
What is **mobile** performance?

- TODO: How this scales to all tests, not just our startup example



- Process start
- Dex/resources loaded
- JIT/AOT compilation
- Content providers & SDK inits
- Application onCreate
- Main Activity onCreate
- Dependencies instantiated
- ViewModels/fragments created
- Fetch data from backend
- Render components



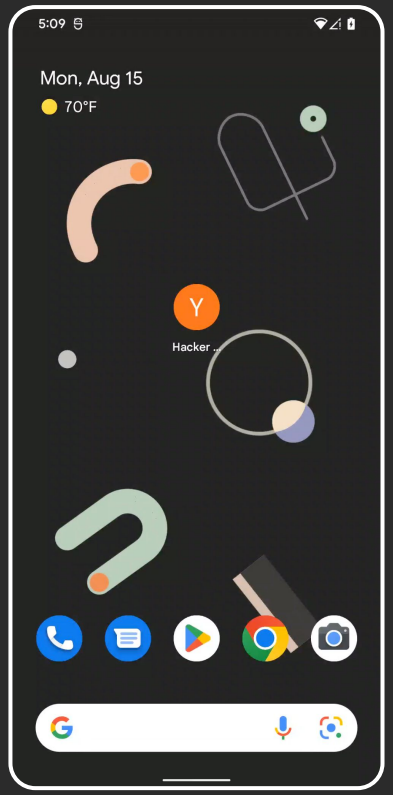


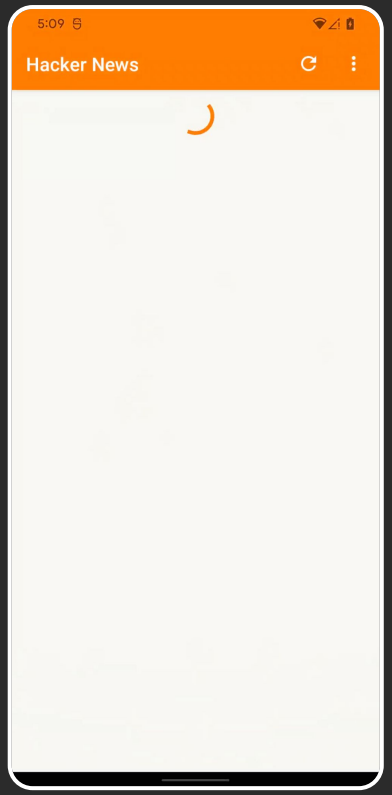
What’s worth testing?

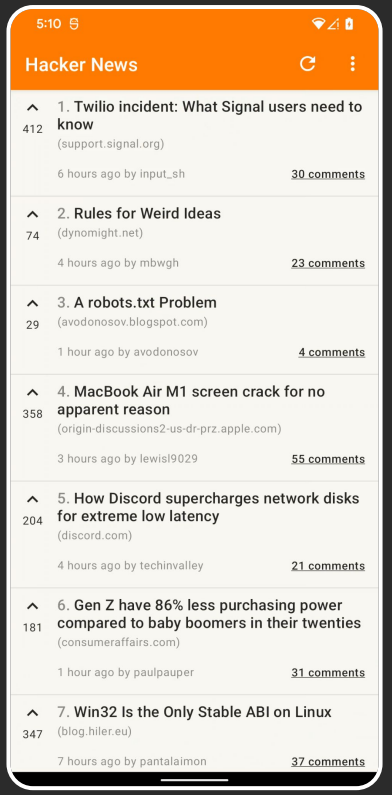
- **In our control**
  - SDK inits
  - Component creations (activities, fragments, etc)
  - Dependency management
  - Rendering UI

- **Out of our control**
  - Network request latency
  - I/O operations
  - Communication with processes outside our app’s control

What’s worth testing?







Testing vs Tracking

- Production ⟹ highest quality data & scale
- Testing should compliment tracking

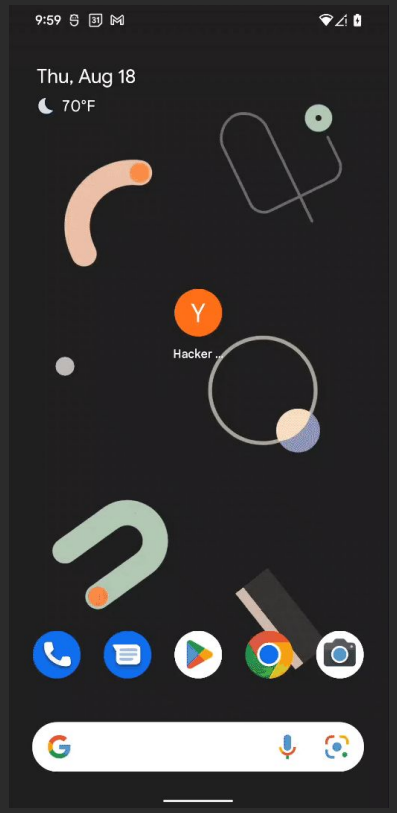
A framework for choosing what to test

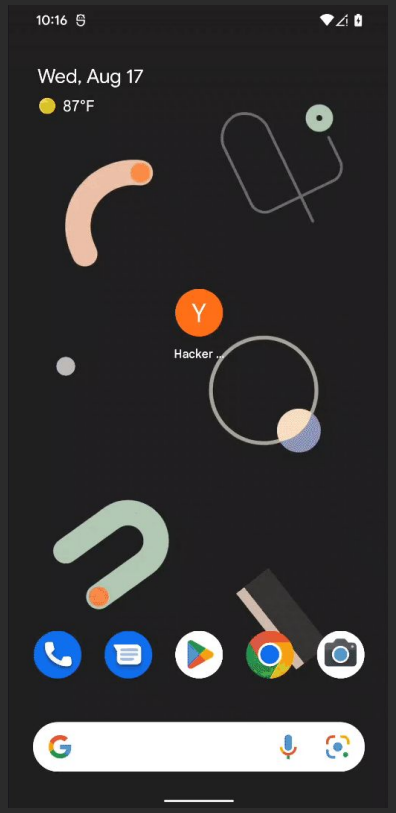
Controllable

Reproducible

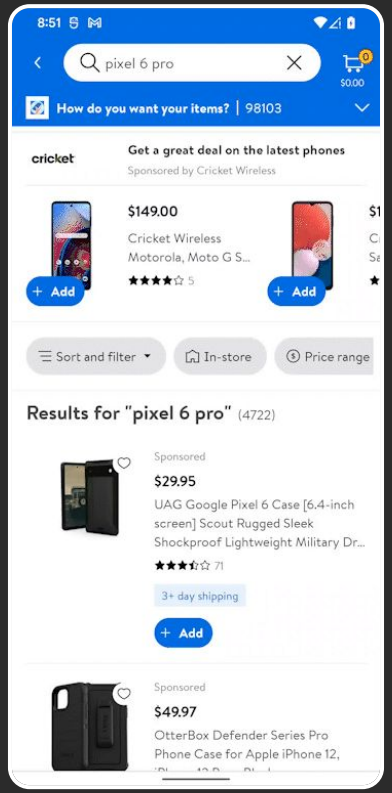
Impactful

Controllable

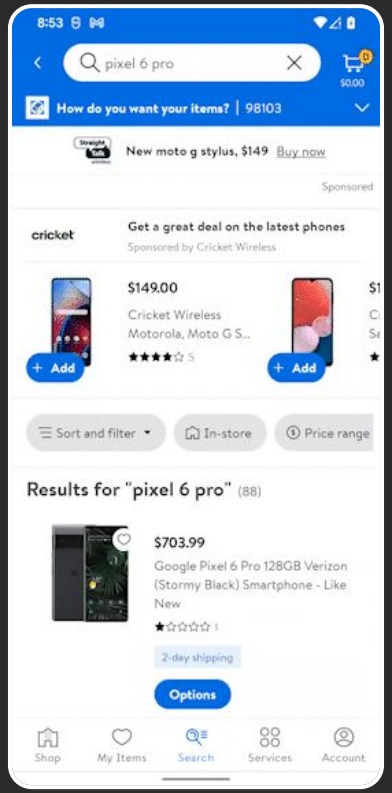




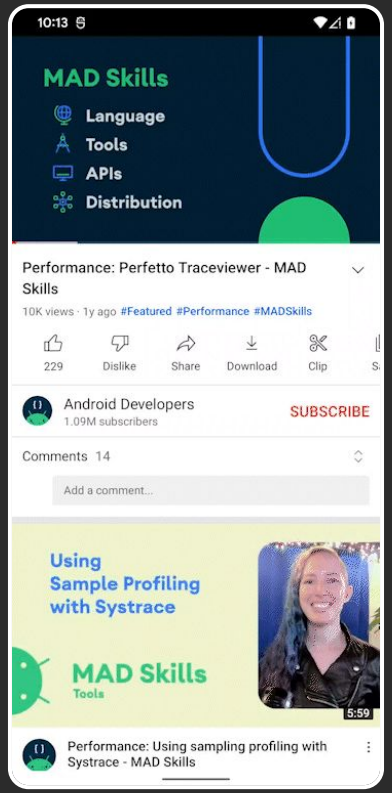
Reproducible 🔁



Reproducible 🔁



Impactful 🔥


10:13
MAD Skills
Language
Tools
APIs
Distribution
Performance: Perfetto Traceviewer - MAD Skills
10K views · 1y ago #Featured #Performance #MADSkills
229
Dislike
Share
Download
Clip
Android Developers
1.09M subscribers
SUBSCRIBE
Comments 14
Add a comment...
Using Sample Profiling with Systrace
MAD Skills
Tools
5:59
Performance: Using sampling profiling with Systrace - MAD Skills

What to avoid measuring in perf testing

- **Non-Controllable**
  - Network operations w/o mocking
  - Inter-process communication (binder)

- **Non-Reproducible**
  - Dynamic content (e.g. news feed)

- **Non-Impactful**
  - Non-primary user flows
  - Interactions that are masked

Testing performance

Measure

- Perfetto
- Simpleperf
- AS Profiler

Test

- Macrobenchmark
- UIAutomator
- Espresso

Measure

- Perfetto
- Simpleperf
- ~~AS Profiler~~

Test

- Macrobenchmark
- UIAutomator
- ~~Espresso~~

Testing performance

1. Measure

Tracing vs Sampling

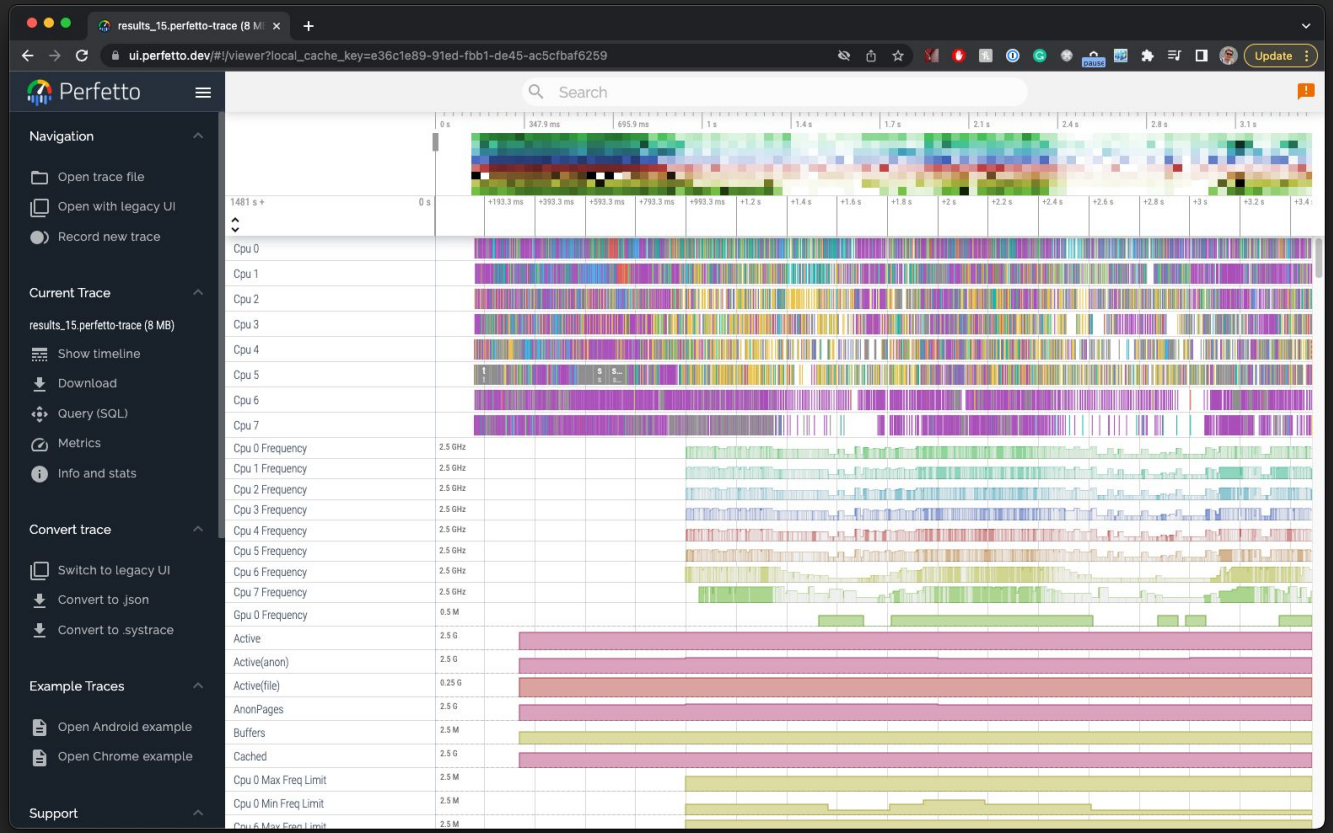
Tracing

- Record a start & stop time for a given “span”
- More traces recorded, performance impact increases

Sampling

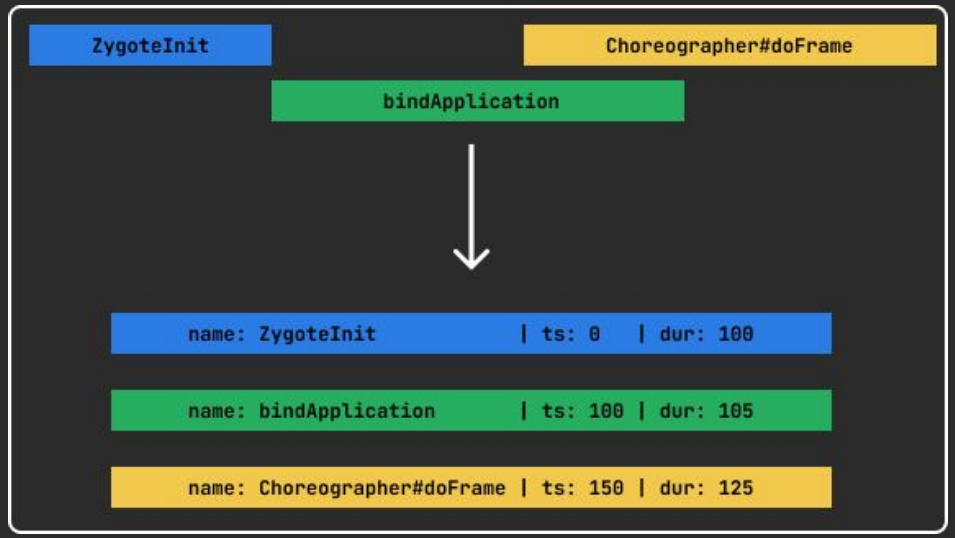
- Take a “snapshot” at a given moment
- Performance + detail impact determined by how often snapshot taken


Perfetto


results_15.perfetto-trace (8 M
ui.perfetto.dev/#!/viewer?local_cache_key=e36c1e89-91ed-fbb1-de45-ac5cfbaf6259
Update
Perfetto
Search
Navigation
Open trace file
Open with legacy UI
Record new trace
Current Trace
results_15.perfetto-trace (8 MB)
Show timeline
Download
Query (SQL)
Metrics
Info and stats
Convert trace
Switch to legacy UI
Convert to .json
Convert to .systrace
Example Traces
Open Android example
Open Chrome example
Support
Cpu 0
Cpu 1
Cpu 2
Cpu 3
Cpu 4
Cpu 5
Cpu 6
Cpu 7
Cpu 0 Frequency
Cpu 1 Frequency
Cpu 2 Frequency
Cpu 3 Frequency
Cpu 4 Frequency
Cpu 5 Frequency
Cpu 6 Frequency
Cpu 7 Frequency
Gpu 0 Frequency
Active
Active(anon)
Active(file)
AnonPages
Buffers
Cached
Cpu 0 Max Freq Limit
Cpu 0 Min Freq Limit

Perfetto

```
SELECT
  slice.name as name,
  slice.ts as startTime,
  slice.dur as duration
FROM slice ...
WHERE (
    process.name LIKE "com.your.application" AND (
      (slice.name LIKE "ZygoteInit") OR
      (slice.name LIKE "bindApplication")
      ...
    )
  )
)
```



Perfetto

Pros 👍

- SQL layer
- Well documented & supported
- Little to no noticeable perf impact

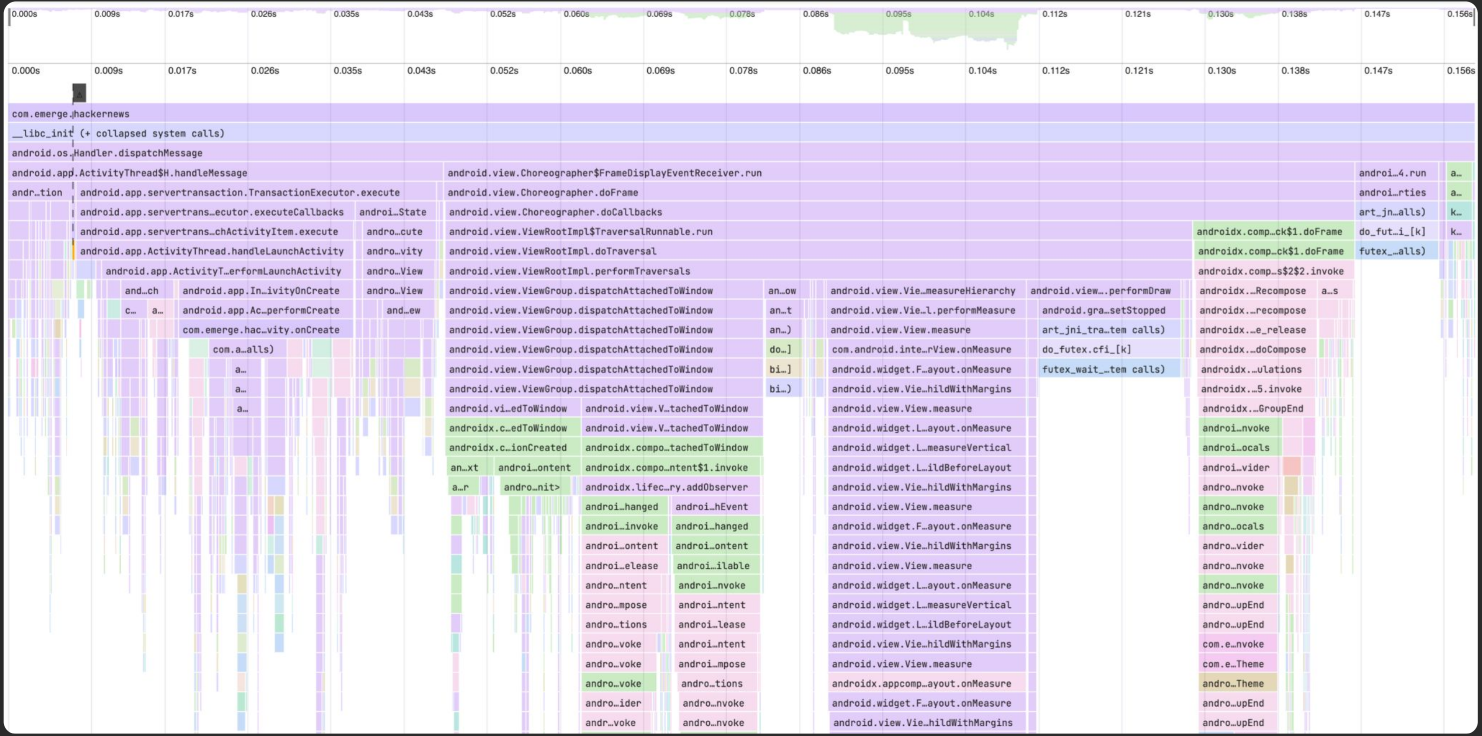
Cons 👎

- Can sample, but missing critical support
- High learning curve for Web UI & other tooling

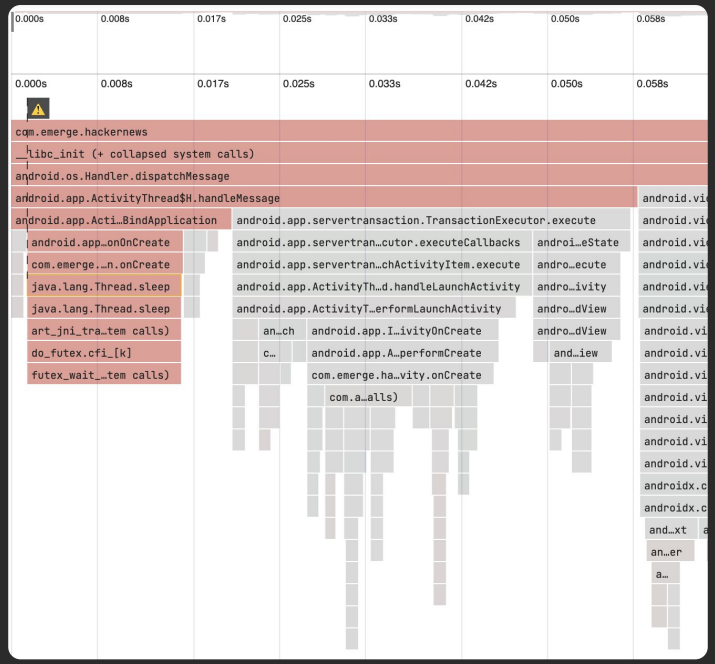
Simpleperf

```
./simpleperf record --app com.your.app \
    -f 200 \
    ...
    -o /path/to/output.data
```

Simpleperf - Flamegraph output



Simpleperf - Differential flamegraph



Brendangregg.com - Resources on diff flamegraphs

Simpleperf

Pros

- Samples all methods during period
- Helper scripts convert to common formats (.folded, Gecko)
- Handles deobfuscation, symbolication & demangling

Cons

- Can affect perf results
- Poorly documented
- Configuration & options are cumbersome

Comparing

- Take measurements of experiment & control
- Use relative comparisons
- Diff tracing/sampling data to pinpoint source of change

Testing performance

1. Measure
2. **Test**

Controlling variance

- CPU speed, page cache state, thermals, etc can all affect perf significantly
- Run multiple iterations to reduce outliers
- Control as much on device as possible

UI testing frameworks & variance

```
// waitForIdle
while (true) {
  long currentTimeMs = SystemClock.uptimeMillis();

  long elapsedGlobalTimeMs = currentTimeMs - startTimeMs;
  long remainingGlobalTimeMs = globalTimeoutMs -
elapsedGlobalTimeMs;
  if (remainingGlobalTimeMs <= 0) {
    throw Exception(...)
  }
  long elapsedIdleTimeMs = currentTimeMs - mLastEventTimeMs;
  long remainingIdleTimeMs = idleTimeoutMs - elapsedIdleTimeMs;
  if (remainingIdleTimeMs <= 0) {
    return;
  }
  mLock.wait(remainingIdleTimeMs);
}
```

UI testing frameworks & variance

```
// UIDevice.waitForIdle
while (true) {
  long currentTimeMs = SystemClock.uptimeMillis();

  long elapsedGlobalTimeMs = currentTimeMs - startTimeMs;
  long remainingGlobalTimeMs = globalTimeoutMs -
elapsedGlobalTimeMs;
  if (remainingGlobalTimeMs <= 0) {
    throw Exception(...)
  }
  long elapsedIdleTimeMs = currentTimeMs - mLastEventTimeMs;
  long remainingIdleTimeMs = idleTimeoutMs - elapsedIdleTimeMs;
  if (remainingIdleTimeMs <= 0) {
    return;
  }
  mLock.wait(remainingIdleTimeMs);
}
```

Quick note - Spans

No matter the test

Macrobenchmark

```
@get:Rule
val benchmarkRule = MacrobenchmarkRule()

@Test
fun startup() = benchmarkRule.measureRepeated(
    packageName = "com.emerge.hackernews",
    metrics = listOf(StartupTimingMetric()),
    iterations = 40,
    startupMode = StartupMode.COLD,
    compilationMode = CompilationMode.None(),
) {  this: MacrobenchmarkScope
    val intent = Intent()
    startActivityAndWait(intent)
}
```

Macrobenchmark

- Leverages Perfetto SQL under the hood
- Controls variance - locks clocks, clears page cache, etc
- Runs in separate APK to mitigate perf impact on target app

Macrobenchmark - Pros & Cons

Pros 👍

- Easiest implementation, lots of freebies
- Complicated aspects abstracted away
- Also generates Baseline profiles

Cons 👎

- Inflexible to larger demands
- Controls variance, to a point
- Doesn’t sample

Microbenchmark

Talk about how it leverages perfetto

Talk about how it’s configurable

Iteration count,

UIAutomator

- Leveraged by Macrobenchmark for UI testing
- Runs separate from target app’s process, mitigating perf impact
- For UI testing, but perf metrics need to be manually measured

UIAutomator/Custom impl - Pros & Cons

Pros

- Full control of device, iterations, measurements
- Run outside of target app process
- Can still leverage Macrobenchmark & Benchmark libs

Cons

- Manual configuration & orchestration of tests
- Higher barrier of entry & maintenance
- Have to manually measure results

Pitfalls

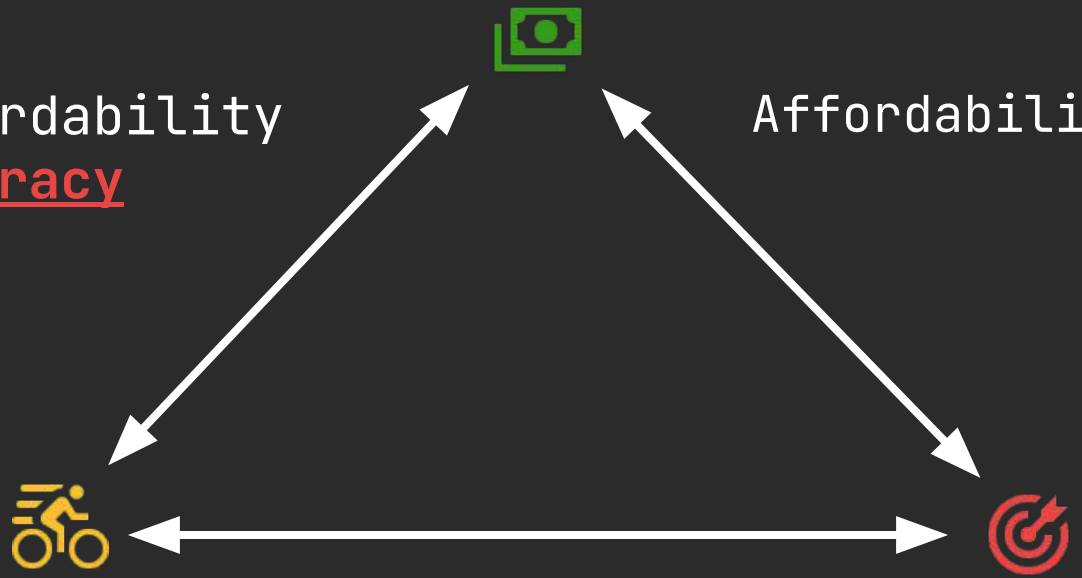
- **Variance**
- Real world performance
- **Emulators** vs Physical devices
- **Espresso** vs UIAutomator
- **Overhead**

Scaling

Performance testing tradeoffs

Speed & affordability impacts accuracy

Affordability & accuracy impacts speed



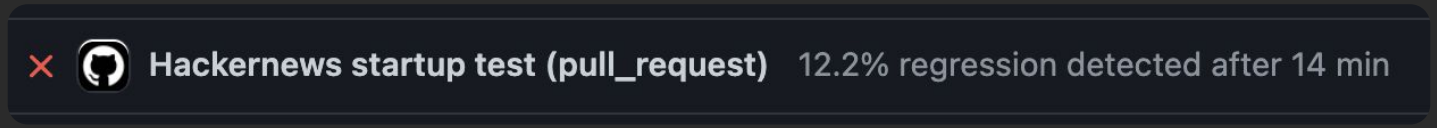
Speed & accuracy impacts affordability

Perf testing tradeoffs

- Type of devices — Cost 💵 & Accuracy 🎯
- Cadence of perf tests — Cost 💵 & Speed 🚴
- Num iterations/test — Speed 🚴 & Accuracy 🎯

Delivering actionable results

- Pinpointing source of change to code (per PR)
- Flamegraphs
- Checks & enforcement



Bringing it all together

Bringing it all together

1. What will we measure & why?

Bringing it all together

1. What will we measure & why?
2. **How will we measure & test?**

Bringing it all together

1. What will we measure & why?
2. How will we measure & test?
3. **How will we scale?**


emerge
tools

Q/A

 ryan@emergetools.com

 @rbro112 / @emergetools