Performance Challenges in Facebook Android Development

Frank Qixing Du
Facebook Inc.
Dec 5, 2012
O'Reilly Velocity, Beijing

About Speaker

- Joined Facebook since Facebook Android App 1.7.2
- Worked on Facebook Android Frontend UI, Push Notifications, APIs, Fragments, Faceweb, Architect & Performance etc.
- Release Manager of Facebook Android v1.9
- Worked for a Mobile Development Platform Startup in California, and an NYC Company

About Speaker

- Facebook: https://facebook.com/du.frank
- Weibo: http://weibo.com/frankdu

Outline

- Facebook Overview
- Mobile: Web vs. Native
- Challenges in Mobile Context
- Facebook Android Performance
  - Why Android Stutter More?
  - Measure Improvement
  - Garbage Collection
  - Memory
  - View Optimization
  - Main Thread
- Summary
- Q &A

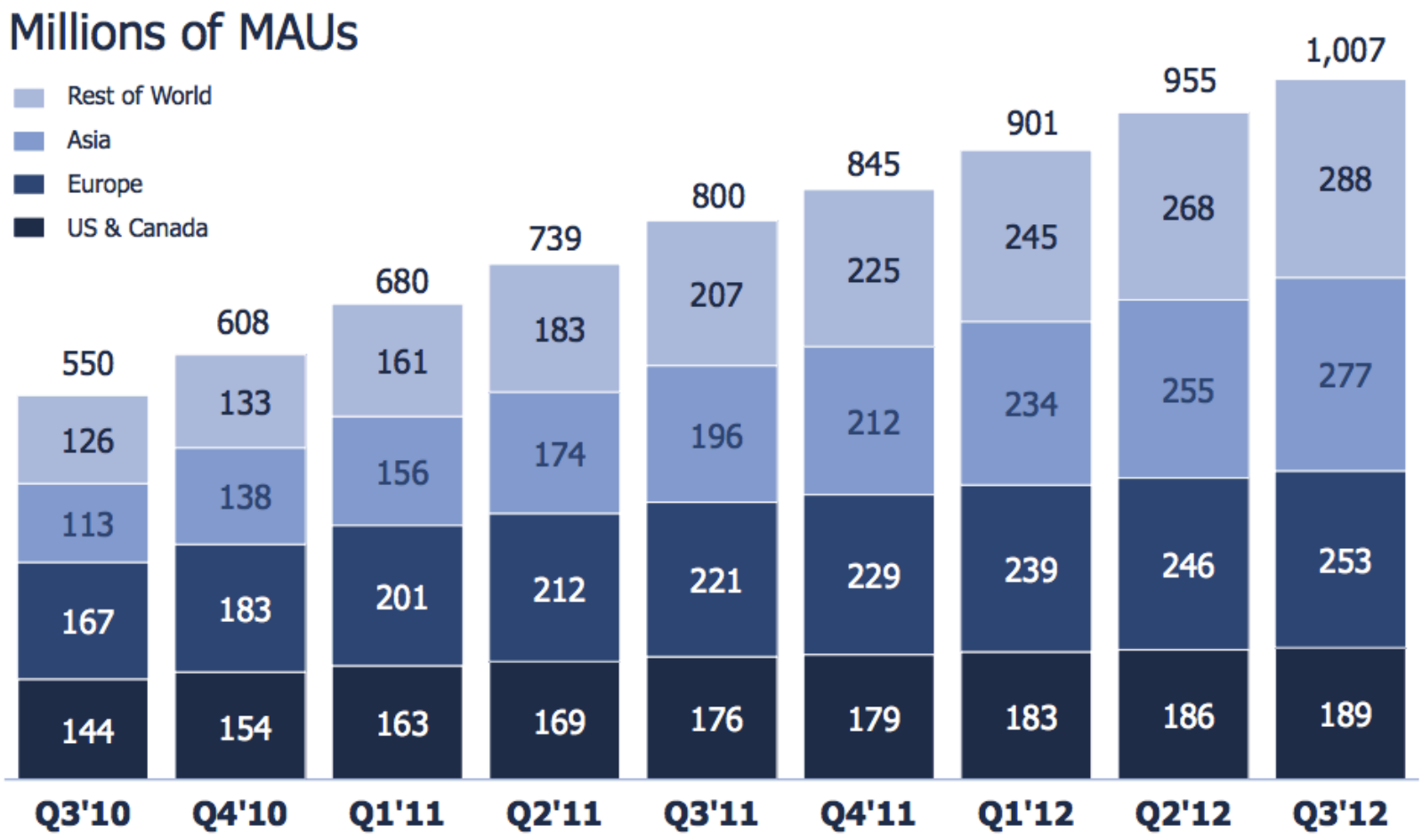
Facebook

- Mission
  - Make the world more open and connected. Help people stay connected with friends and family.
- Serving Over 1 Billion Users from Silicon Valley
- About 80% MAU outside US/Canada
- 584 Million DAU
- 604 Million Mobile MAU
- 4000+ Employees

* The numbers were around Sep 2012. Please check the source for details.

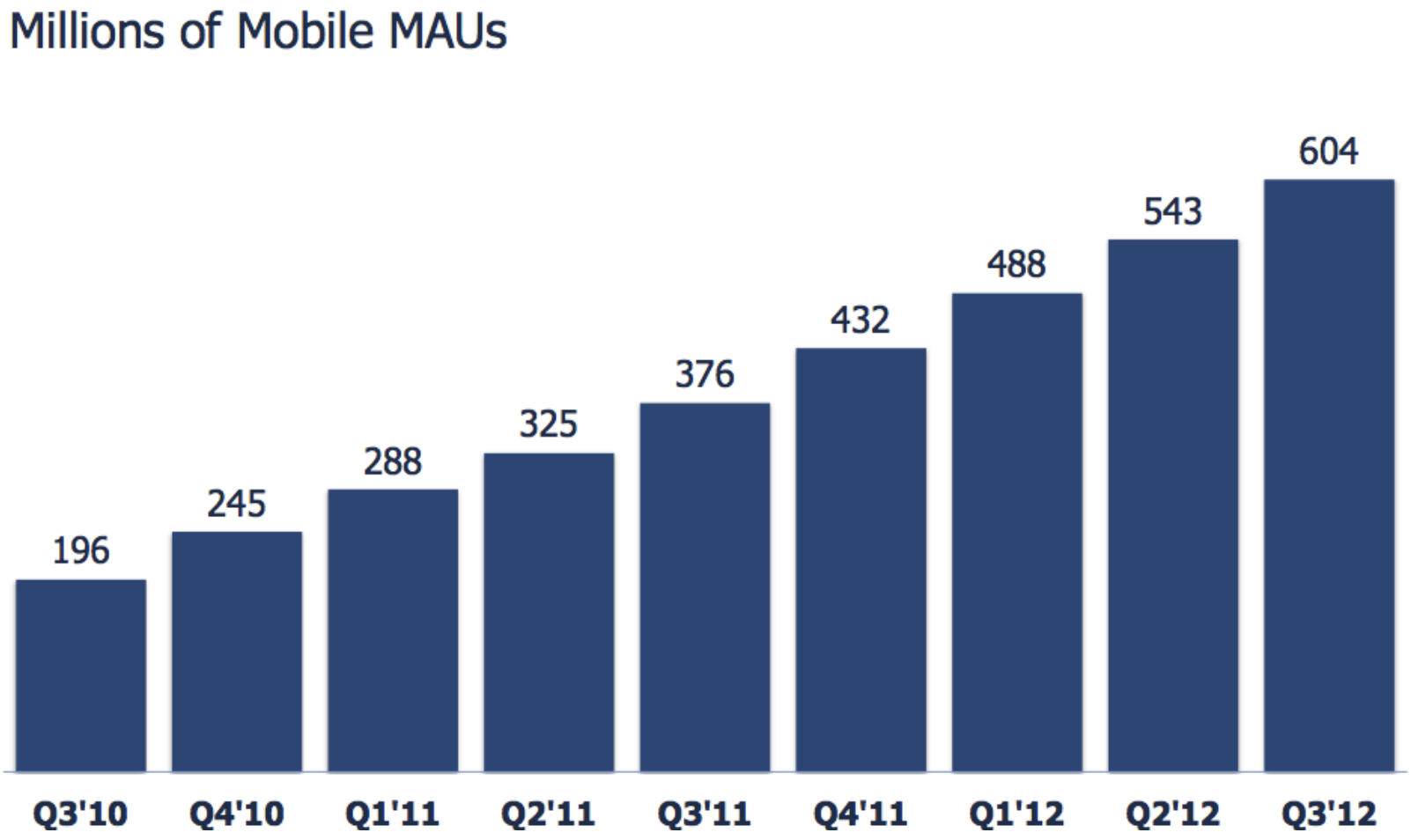
Source: http://newsroom.fb.com/Key-Facts

Facebook Growth



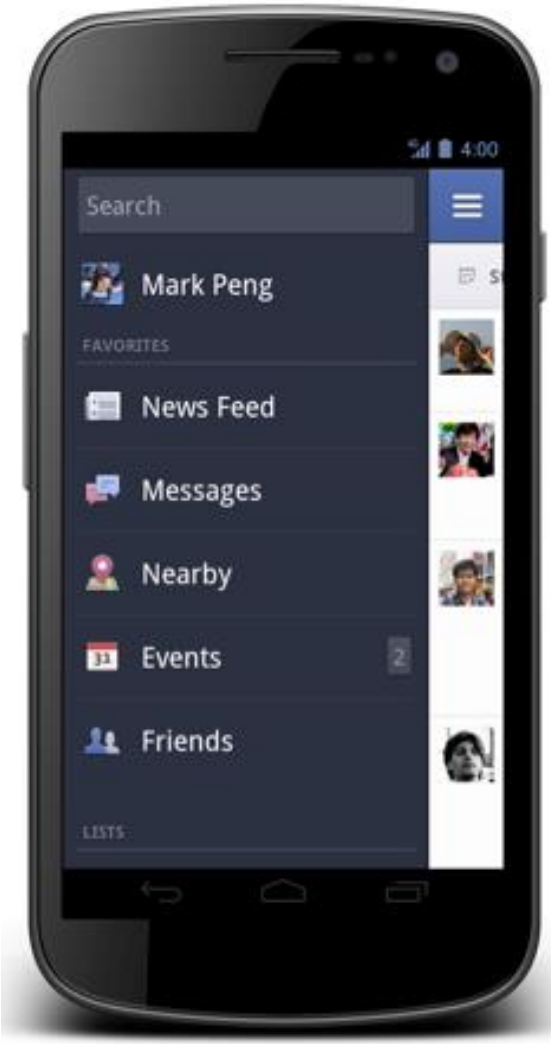
Source: http://investor.fb.com/results.cfm

Facebook Mobile Growth



Source: http://investor.fb.com/results.cfm

iOS, Android and More



Facebook Mobile Everywhere

- Mobile sites for ALL phones
- Feature phones (m-site + app)
- Blackberry
- Windows Phone 7/8
- iOS
- Android

Facebook Mobile Everywhere

Facebook

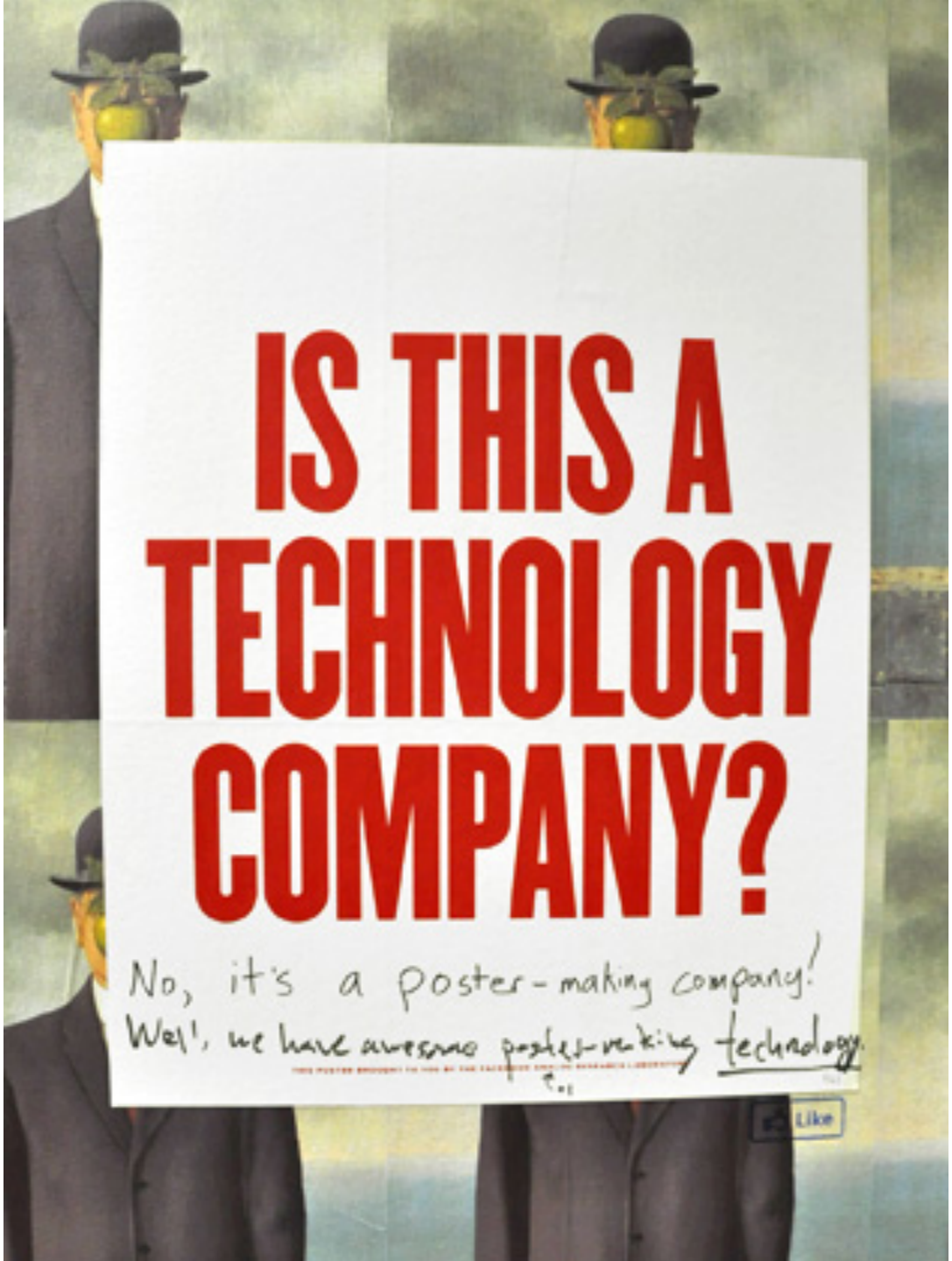
In Posters…


WHAT WOULD YOU DO IF YOU WEREN'T AFRAID?
I would do this.
You Should Ask Eminem
LET MY KID PLAY WITH KNIVES
COMMUTE AT 150mph
DO A BARREL ROLL
PROMOTE SYNERGY (LIKE A BOSS)
THIS POSTER BROUGHT TO YOU BY THE FACEBOOK ANALOG RESEARCH LABORATORY


MOVE
FAST AND
BREAK
THINGS


LC-24
IS THIS A
TECHNOLOGY
COMPANY?


STAY
FOCUSED
& KEEP
SHIPPING


IS THIS A TECHNOLOGY COMPANY?
No, it's a poster-making company!
Well, we have awesome poster-making technology.
Like

Mobile Client Solutions

2 Major Categories:

- Mobile Web
- Mobile Native

Mobile Web

- Advantage
  - Browsers are Standard
  - Reusable codes across platforms
  - Fast Shipping via daily push
- Disadvantage
  - Less control on data pipeline
  - Limited consumption patterns
  - Performance

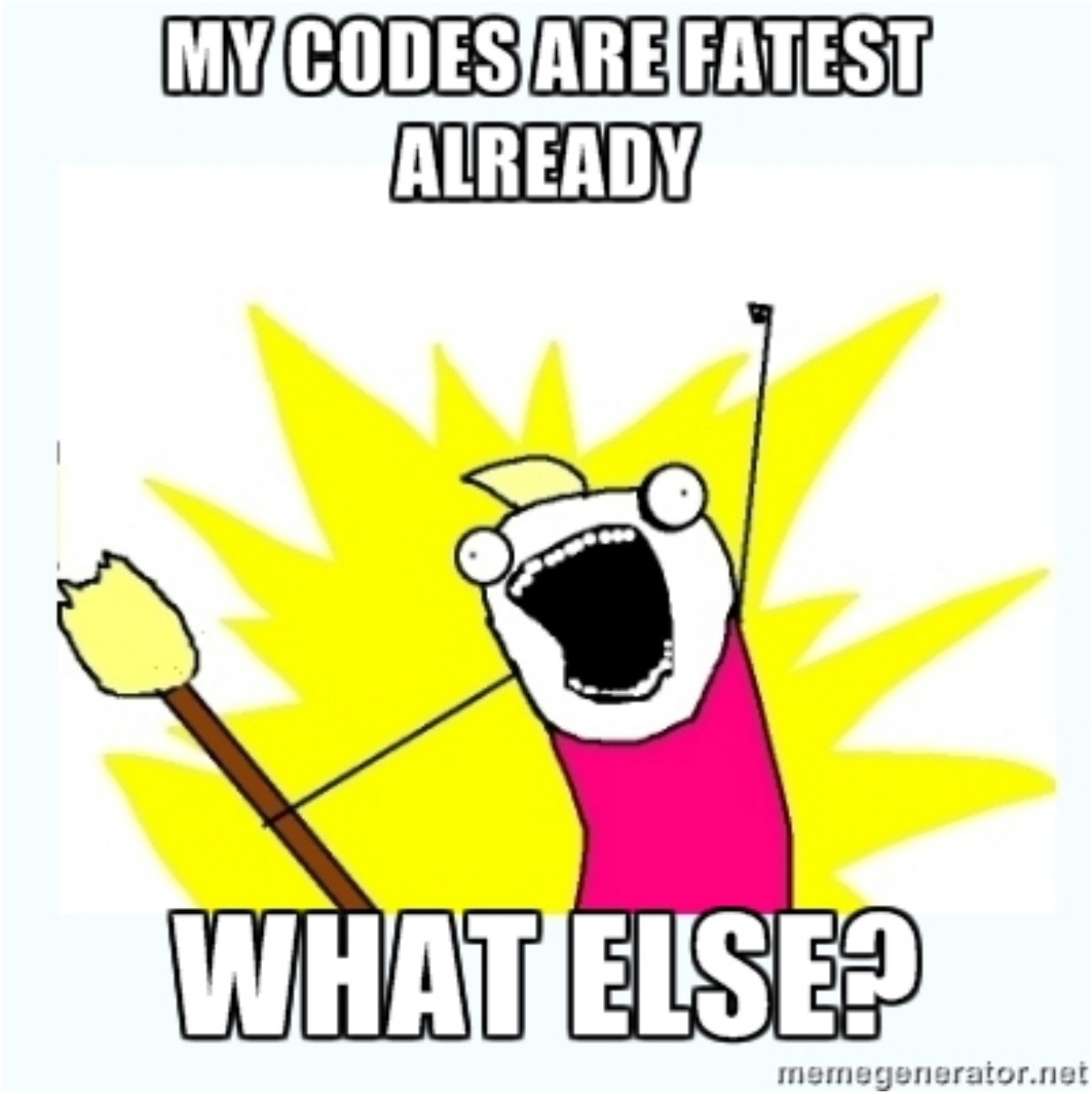
Native App

- Advantages
  - Server send data only
  - Better leverage on hardware (GPS, sensors etc.)
  - User consumption optimization
- Disadvantages
  - Slower shipping
  - Harder programming model than Web
  - Code reuse

Challenges in Mobile Context

- Unreliable Network
- Limited Hardware
- Battery Limit
- User Sentiment
- Worldwide Users
- Device/OS Diversity


USERS DON'T CARE
BUT EVERYTHING FAST
memegenerator.net


MY CODES ARE FATEST ALREADY
WHAT ELSE?
memegenerator.net

Facebook Android Development

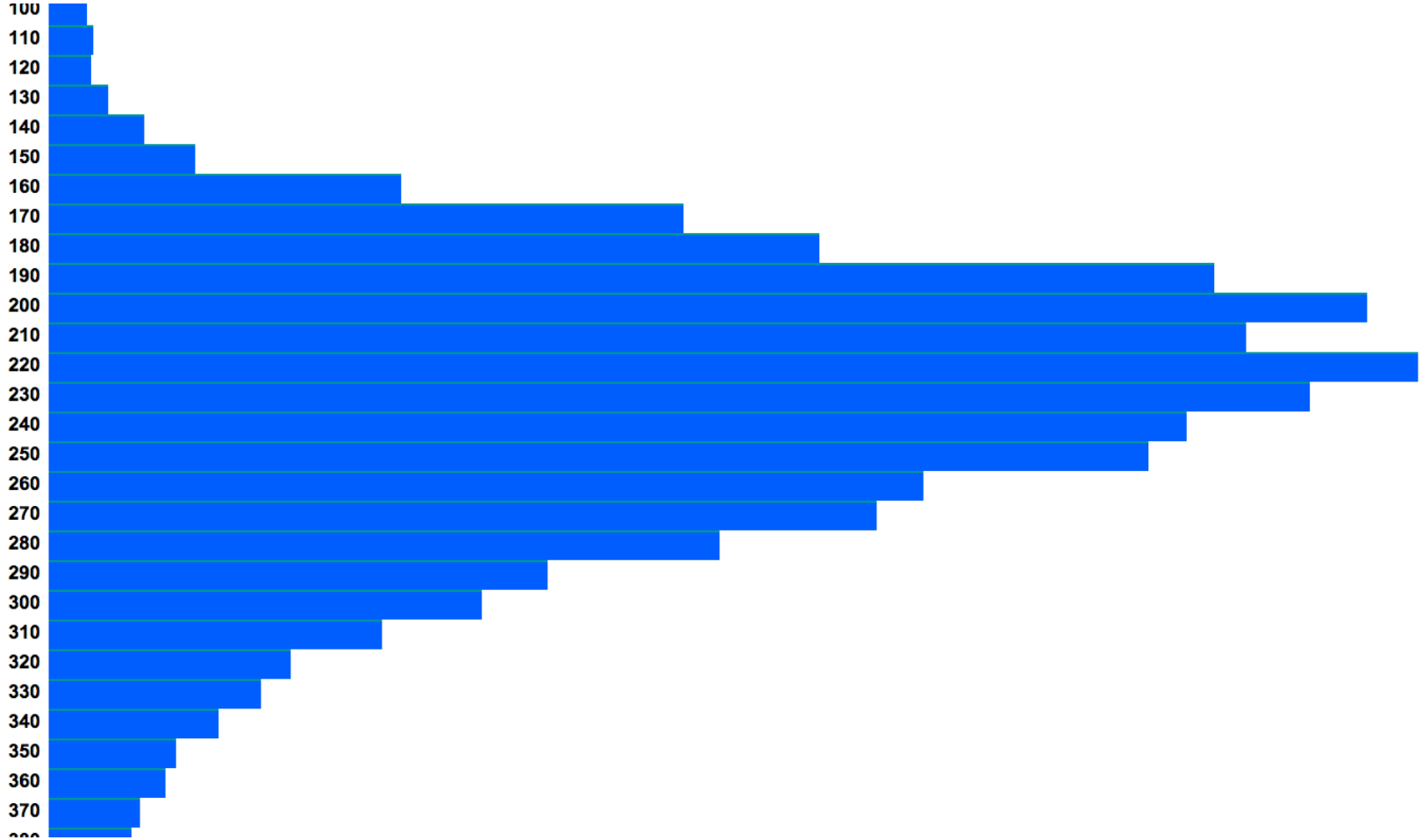
A Scrolling Performance Story

Simple Math

- 60 fps – 16.7ms per frame
- 30 fps – 33.3ms per frame
- As a Reference:
  - 60 fps: Console/desktop games target for
  - 30 fps: Most mobile games target for
- 100 ~ 200ms
  - Threshold when users perceive slowness *

* http://developer.android.com/training/articles/perf-anr.html

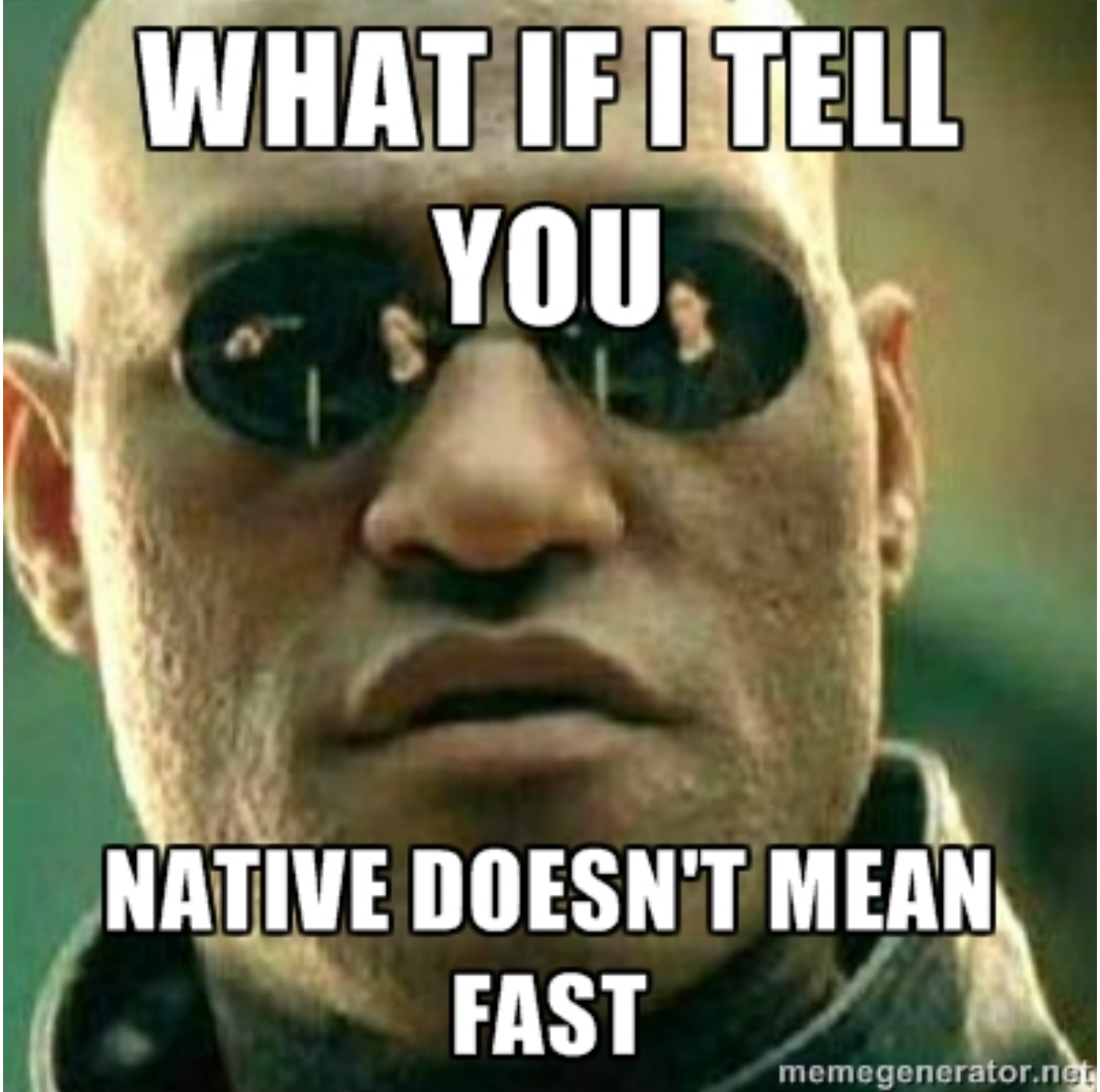
Human Reaction Time Distribution



Source: http://www.humanbenchmark.com/tests/reactiontime/stats.php

Butter Smooth Experience

- Super Fast Codes
  - Most efficient algorithms
  - Native programming model
- User Perception of Being Fast
  - Always responsive to users input
  - Most effective interaction flow
  - Keep users informed


WHAT IF I TELL YOU
NATIVE DOESN'T MEAN FAST
memegenerator.net

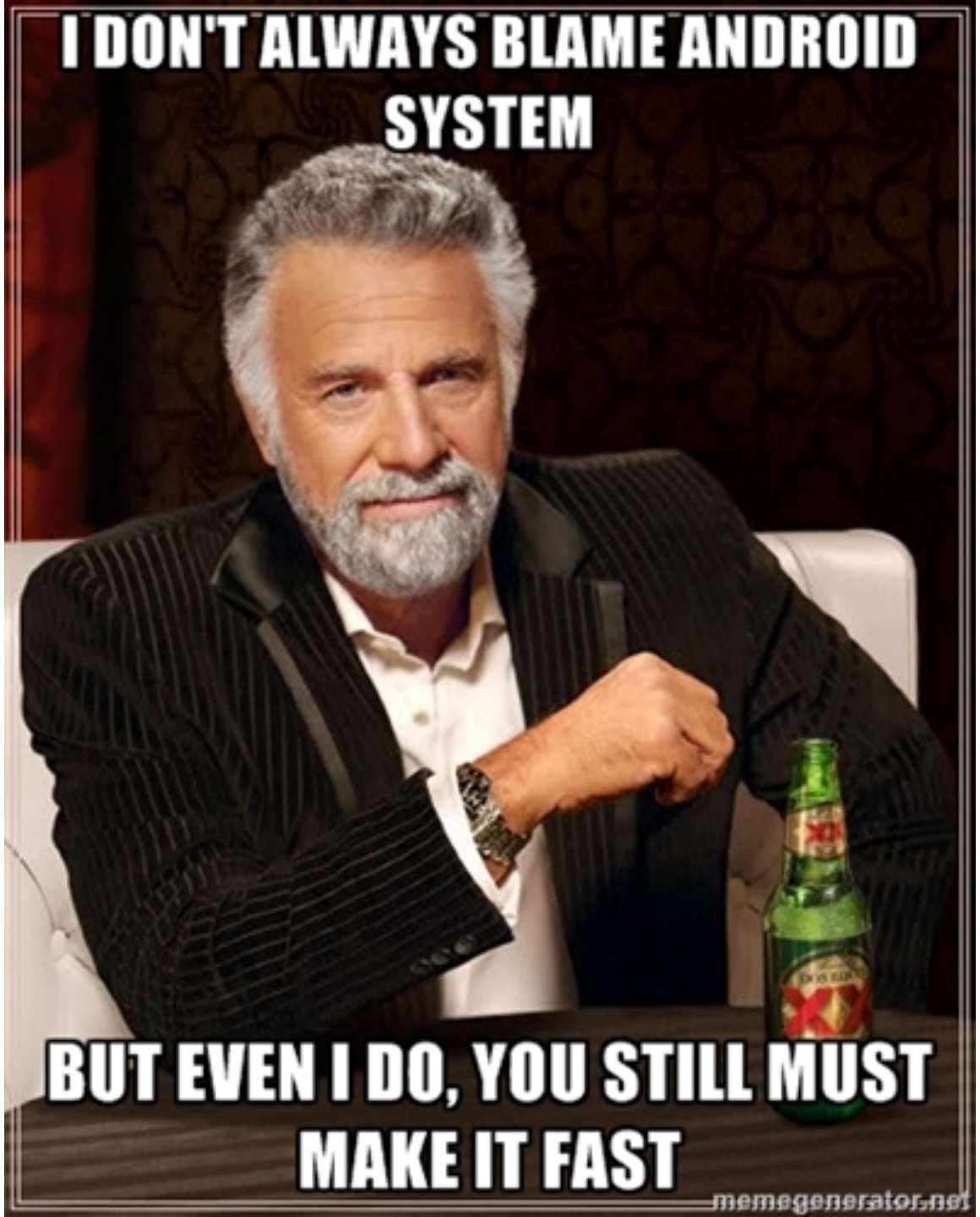
Our Android Performance Challenges

- Why Android stutter more?
- Measure Improvement
- Garbage Collection
- Memory
- View Optimization
- Main Thread
- User Perception

Android Scrolling Stutter More?

Different Graphics Strategies

- iOS CoreAnimation - retained mode
  - CoreAnimation runs on bg thread
  - Continuously move UI objects even UI thread is blocked
- Android – Immediate mode
  - Compute immediate view locations
  - Post to MessageQueue of UI thread with timers
  - Other UI ops (draw/measure etc.) can delay animation


I DON'T ALWAYS BLAME ANDROID SYSTEM
BUT EVEN I DO, YOU STILL MUST MAKE IT FAST
memegenerator.net

Why So Important?

- iOS Animation Smoothness
- Android Animation
  - Android 2.3 or earlier
  - ICS
  - Jelly Bean
- Facebook
  - User engagement
  - Supper complicated UI composition
  - Push system limits

Measure Improvement

- FPS
- Tracer
- Performance Marker
- Tracking Graphs
- PM Sentiment

GC

```
D/dalvikvm(18824): JIT code cache reset in 2 ms (1048444 bytes 2/0)
D/dalvikvm(18824): GC_CONCURRENT freed 4622K, 28% free 12497K/17187K, paused 5ms+26ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 1404K, 29% free 12325K/17187K, paused 113ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 124K, 27% free 13646K/18659K, paused 86ms
D/dalvikvm(18824): GC_CONCURRENT freed 1191K, 14% free 16063K/18659K, paused 5ms+24ms
D/dalvikvm(  659): GC_CONCURRENT freed 1729K, 82% free 5048K/26915K, paused 7ms+4ms
D/dalvikvm(18824): GC_CONCURRENT freed 5863K, 29% free 14648K/20579K, paused 3ms+16ms
D/dalvikvm(  870): GC_EXPLICIT freed 167K, 33% free 27046K/39971K, paused 11ms+3ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 204K, 30% free 14446K/20579K, paused 91ms
D/dalvikvm(18824): GC_FOR_ALLOC freed <1K, 28% free 15909K/22051K, paused 66ms
D/dalvikvm(  432): GC_CONCURRENT freed 5798K, 40% free 26232K/43171K, paused 3ms+10ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 2841K, 19% free 19175K/23523K, paused 138ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 481K, 16% free 25297K/29923K, paused 120ms
D/dalvikvm(  432): GC_EXPLICIT freed 2967K, 40% free 26150K/43171K, paused 5ms+13ms
D/dalvikvm(19140): GC_CONCURRENT freed 1790K, 40% free 3103K/5155K, paused 1ms+2ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 3531K, 21% free 27401K/34467K, paused 129ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 2K, 19% free 30715K/37795K, paused 161ms
D/dalvikvm(18824): WAIT_FOR_CONCURRENT_GC blocked 80ms
D/dalvikvm(18824): WAIT_FOR_CONCURRENT_GC blocked 89ms
```

GC

```
D/dalvikvm(18824): JIT code cache reset in 2 ms (1048444 bytes 2/0)
D/dalvikvm(18824): GC_CONCURRENT freed 4622K, 28% free 12497K/17187K, paused 5ms+26ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 1404K, 29% free 12325K/17187K, paused 113ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 124K, 27% free 13646K/18659K, paused 86ms
D/dalvikvm(18824): GC_CONCURRENT freed 1191K, 14% free 16063K/18659K, paused 5ms+24ms
D/dalvikvm(  659): GC_CONCURRENT freed 1729K, 82% free 5048K/26915K, paused 7ms+4ms
D/dalvikvm(18824): GC_CONCURRENT freed 5863K, 29% free 14648K/20579K, paused 3ms+16ms
D/dalvikvm(  870): GC_EXPLICIT freed 167K, 33% free 27046K/39971K, paused 11ms+3ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 204K, 30% free 14446K/20579K, paused 91ms
D/dalvikvm(18824): GC_FOR_ALLOC freed <1K, 28% free 15909K/22051K, paused 66ms
D/dalvikvm(  432): GC_CONCURRENT freed 5798K, 40% free 26232K/43171K, paused 3ms+10ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 2841K, 19% free 19175K/23523K, paused 138ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 481K, 16% free 25297K/29923K, paused 120ms
D/dalvikvm(  432): GC_EXPLICIT freed 2967K, 40% free 26150K/43171K, paused 5ms+13ms
D/dalvikvm(19140): GC_CONCURRENT freed 1790K, 40% free 3103K/5155K, paused 1ms+2ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 3531K, 21% free 27401K/34467K, paused 129ms
D/dalvikvm(18824): GC_FOR_ALLOC freed 2K, 19% free 30715K/37795K, paused 161ms
D/dalvikvm(18824): WAIT_FOR_CONCURRENT_GC blocked 80ms
D/dalvikvm(18824): WAIT_FOR_CONCURRENT_GC blocked 89ms
```


WAT? NO GC?
ISN'T IT CALLED JAVA?
memegenerator.net

Garbage Collection

- No GC
- Reduce GC
- GC Comparison:
  - GC_FOR_ALLOC
  - GC_EXPLICIT
  - GC_CONCURRENT
- Compare
  - Instagram vs Pinterest vs Google+

Memory

- Memory Analysis
  - Image
    - Purgeable bitmaps
  - Remove unnecessary allocation
    - Cached values
    - Boxing/unboxing
  - Reduce allocation
    - Object Pooling
    - Shared resources
  - Memory leakage
    - Activity Context vs Application Context
    - Allow android system to release resources
    - Manage activities using resources heavily

View Optimization

- View Recycling
- Deflate View Hierarchy

View Optimization cont.

- Why View Recycling?
  - View inflation is expensive (e.g. ~70ms)
  - Render content is expensive (e.g. 15~30ms)
  - The above both happens on the UI thread
  - Scrolling stutter is noticeable with long pause

View Optimization cont.

Custom View Recycler:

- Take over Android ListView recycling
  - What's the problem with Android impl?
    - Best performance for views of same height
    - Inefficient: Drop views if uneven height
  - Take Over View Recycling

- Custom view recycling
  - Recycle small parts of large views

View Optimization cont.

Deflate the view hierarchy:

- 3 passes: measure/layout/draw
- System limit
- Kill unnecessary nesting levels
- <merge />
- <ViewStub />

Main Thread

- Ideally, only update view state on main thread
- Move heavy operations off main thread
  - Examples: Network/DB/File/Preferences operations
  - Methods
    - StrictMode to rescue
    - Service & Worker Threads
    - AsyncTask
  - Discuss: Pre-computing vs lazy computing
- Must run on main thread?
  - Defer heavy ops to non-critical time
  - Monitor main thread idling
  - Tiny task units (< 17ms or < 100ms?)

User Perception

- Clean UI design
- Transition animation
- Fasten content loading
  - Disk/memory cache
  - Image prefetching and warming
- Visual Hints
  - Progress bar
  - Loading indicator

import now, __future__

- Project Butter

- Romain Guy:
  “...a lot of the work we have to do today is because of certain choices made years ago... ...having the UI thread handle animations is the biggest problem. We are working on other solutions to try to improve this (schedule drawing on vsync instead of block on vsync after drawing, possible use a separate rendering thread, etc.) An easy solution would of course to create a new UI toolkit but there are many downsides to this also.”

Summary

- At Facebook, performance is a serious feature
- Improve memory efficiency
- Reduce GC as much as possible
- Move heavy ops off UI thread
- Defer heavy ops at critical time
- User perception of smoothness
- Measure the improvement quantitatively

More readings

- Roots of Android Lagginess? http://fineoils.blogspot.com/2012/01/roots-of-android-lagginess.html
- Android Graphics, Animations and Tips & Tricks http://www.curious-creature.org/2010/12/02/android-graphics-animations-and-tips-tricks/
- Best Practice for Performance http://developer.android.com/training/best-performance.html
- Allocation tracker http://www.curious-creature.org/2009/02/07/track-memory-allocations-on-android/
- Systrace http://developer.android.com/tools/help/systrace.html
- Traceview http://developer.android.com/tools/help/traceview.html

Q & A

Thank you!