The guide to Hardware acceleration

“G(PU) force”- Royi Benyossef

Introduction

Who? What?

Introduction

Royi Benyossef

Introduction

Royi Benyossef

Android developer since 2009

Introduction

Royi Benyossef

Android developer since 2009

Tech community activist, speaker and founder

Introduction

Royi Benyossef

Android developer since 2009

Tech community activist, speaker and founder

Mentor at * accelerator

Introduction

Royi Benyossef

Android developer since 2009
Tech community activist, speaker and founder
Mentor at * accelerator
Google expert since Dec 2012

Introduction

Royi Benyossef

Android developer since 2009
Tech community activist, speaker and founder
Mentor at * accelerator
Google expert since Dec 2012
Ecosystem relations manager at Samsung NEXT Tel-Aviv

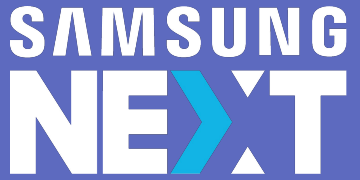
Introduction

Royi Benyossef

Android developer since 2009
Tech community activist, speaker and founder
Mentor at * accelerator
Google expert since Dec 2012
Ecosystem relations manager at Samsung NEXT Tel-Aviv
Investor at Samsung NEXT Tel-Aviv

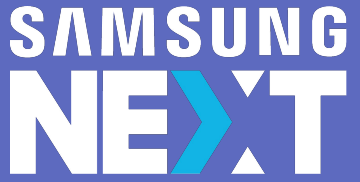
Introduction

Samsung NEXT



Introduction

Samsung NEXT



Innovation arm focused on software products and services

Introduction

Samsung NEXT

Innovation arm focused on software products and services

Investments (seed to B rounds)

Introduction

Samsung NEXT

Innovation arm focused on software products and services

Investments (seed to B rounds)

Independent product creation

Introduction

Samsung NEXT

Innovation arm focused on software products and services
Investments (seed to B rounds)
Independent product creation
M&A & partnerships for Samsung & Samsung NEXT

Prolog

Gee brain, what are we going to do tonight?

Motivation

What do we want?

Motivation

What do we want?

To understand how the GPU works

Motivation

What do we want?

To understand how the GPU works?

Motivation

What do we want?

To understand how the GPU works

Why?

Motivation

What do we want?

To understand how the GPU works

Why?

- **Smoother animations**

Motivation

What do we want?

To understand how the GPU works

Why?

- Smoother animations
- **Sleek transitions**

Motivation

What do we want?

To understand how the GPU works

Why?

- Smoother animations
- Sleek transitions
- **Avoiding really strange bugs!**

Definition

What’s are we talking about?

Definition

Hardware acceleration

Using the specific hardware to do stuff faster

Definition

Hardware acceleration

Using the specific hardware to do stuff faster

In our case

Definition

Hardware acceleration

Using the specific hardware to do stuff faster

In our case - using the GPU to render graphics in apps

Definition

Hardware acceleration

Using the specific hardware to do stuff faster

In our case - using the GPU to render graphics in apps;

= Draw Canvas with the GPU

Definition

Hardware acceleration

Using the specific hardware to do stuff faster

In our case - using the GPU to render graphics in apps;

= Draw Canvas with the GPU = use hardware drawing model

Definition

Hardware acceleration

Using the specific hardware to do stuff faster
In our case - using the GPU to render graphics in apps
Instead of?

Definition

Hardware acceleration

Using the specific hardware to do stuff faster
In our case - using the GPU to render graphics in apps
Instead of - using the CPU

Definition

Hardware acceleration

Using the specific hardware to do stuff faster

In our case - using the GPU to render graphics in apps

Instead of - using the CPU = use the software rendering model

Definition

The GPU

Special hardware

Definition

The GPU

Special hardware

- **Designed for rapid matrix mathematical operations robustly**

Definition

The GPU

Special hardware

- Designed for rapid matrix mathematical operations robustly
- **Original intent = increase graphics performance**

Definition

The GPU

Special hardware

- Designed for rapid matrix mathematical operations robustly
- **Original intent = increase graphics performance**
  (animations = bitmap manipulations = math on matrices)

Definition

The GPU

Special hardware

- Designed for rapid matrix mathematical operations robustly
- Original intent = increase graphics performance
- **Also used for NN based AI (= again, operations on matrices)**

Definition

Android drawing models

Software drawing model

Definition

Android drawing models

Software drawing model

- **Invalidate() performed immediately**

Definition

Android drawing models

Software drawing model

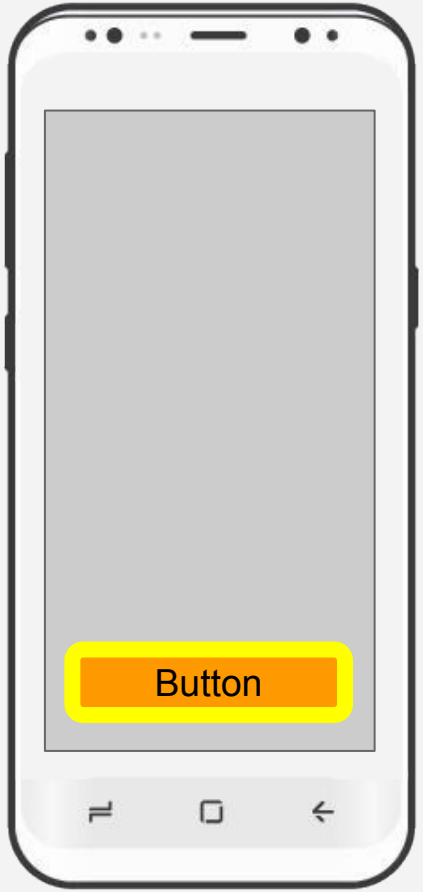
- Invalidate() performed immediately
- **“Dirty region” calculated**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- **“Dirty region” contains:**
  - **Calling (changed) view**

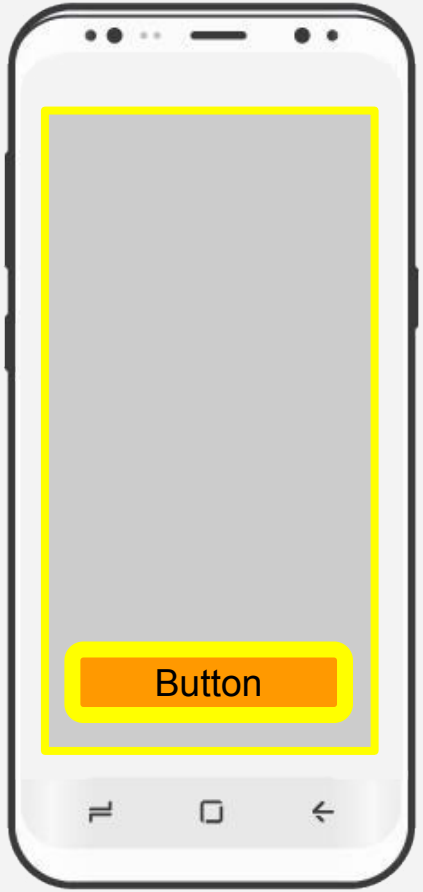


Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- **“Dirty region” contains:**
  - Calling (changed) view
  - **All views that intersect w/ calling view**

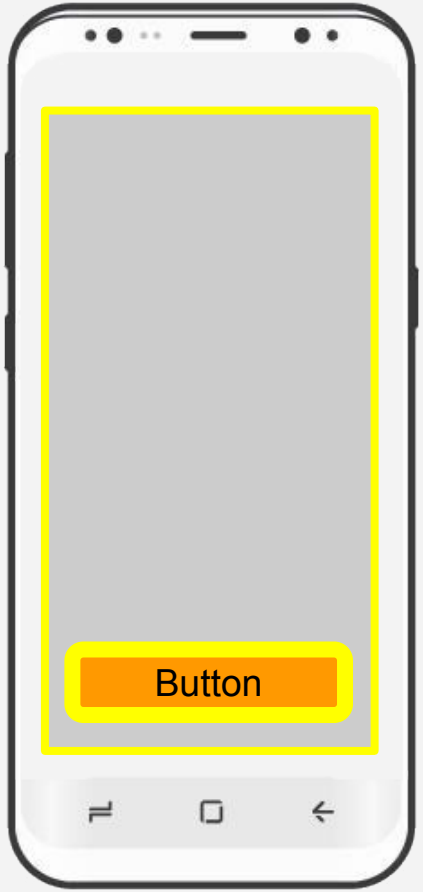


Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- **“Dirty region” contains:**
  - Calling (changed) view
  - **All views that intersect w/ calling view (even if they haven’t changed)**



Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- **“Dirty region” contains:**
  - Calling (changed) view
  - **All views that intersect w/ calling view (even if they haven’t changed)**
    - **Bad**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- **“Dirty region” contains:**
  - Calling (changed) view
  - **All views that intersect w/ calling view (even if they haven’t changed)**
    - **Bad = performance**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- **“Dirty region” contains:**
  - Calling (changed) view
  - **All views that intersect w/ calling view (even if they haven’t changed)**
    - **Bad = performance**
    - **Good**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- **“Dirty region” contains:**
  - Calling (changed) view
  - **All views that intersect w/ calling view (even if they haven’t changed)**
    - **Bad = performance**
    - **Good = hides your bugs**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- **“Dirty region” contains:**
  - Calling (changed) view
  - **All views that intersect w/ calling view (even if they haven’t changed)**
    - **Bad = performance**
    - **Good(-ish) = hides your bugs**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- “Dirty region” calc. (more than you thought)
- **Draws “dirty region”**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- “Dirty region” calc. (more than you thought)
- Draws “dirty region”
- **Uses CPU**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- “Dirty region” calc. (more than you thought)
- Draws “dirty region”
- **Uses CPU**
  - **Good**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- “Dirty region” calc. (more than you thought)
- Draws “dirty region”
- **Uses CPU**
  - **Good:**
    - **Ubiquitously supported**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- “Dirty region” calc. (more than you thought)
- Draws “dirty region”
- **Uses CPU**
  - **Good:**
    - Ubiquitously supported
    - **Predictable functionality**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- “Dirty region” calc. (more than you thought)
- Draws “dirty region”
- **Uses CPU**
  - Good (predictable, supported)
  - **Bad**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- “Dirty region” calc. (more than you thought)
- Draws “dirty region”
- **Uses CPU**
  - Good (predictable, supported)
  - **Bad:**
    - **Predictably & ubiquitously slow**

Definition

Android drawing models

Software drawing model

- Invalidate() performed immediately
- “Dirty region” calc. (more than you thought)
- Draws “dirty region”
- **Uses CPU**
  - Good (predictable, supported)
  - **Bad:**
    - Predictably & ubiquitously slow
    - **In higher demand**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- **Invalidate() flags a view as “dirty”**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- **Changes saved to “display lists”**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- **Upon “V-Sync” diff is drawn**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - **Good**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - **Good:**
    - **Faster**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - **Good:**
    - **Faster (built specifically for that)**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - **Good:**
    - **Faster (built specifically for that)**
    - **Not as used**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - **Good:**
    - Faster (built specifically for that)
    - **Not as used (faster yet)**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - **Good:**
    - Faster (built specifically for that)
    - Not as used (faster yet)
    - **Props (alpha/rotation), don’t invalidate**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
    - **Good (faster X2)**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - Good (faster X2)
  - **Bad**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - Good (faster X2)
  - **Bad:**
    - **Support is specific & not standard**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - Good (faster X2)
  - **Bad:**
    - Support is specific & not standard
    - **Does not hide your bugs**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model

- Invalidate() flags a view as “dirty”
- Changes saved to “display lists”
- Upon “V-Sync” diff is drawn
- **Uses GPU**
  - Good (faster X2)
  - **Bad:**
    - Support is specific & not standard
    - Does not hide your bugs
    - **Consumes more RAM**

Definition

Android drawing models

Software drawing model (immediate, inefficient)

Hardware drawing model (faster, less predictable)

Problem

What's there to fix?

Problem

What's there to fix?

Understand when to use which model

Problem

What's there to fix?

Understand when to use which model

Problem

What's there to fix?

Understand when to use which model

Provide predictability to the HW drawing model

Problem

What's there to fix?

Understand when to use which model

Provide predictability to the HW drawing model

Debug & troubleshoot

Problem

What's there to fix?

Understand when to use which model
Predictability in the HW drawing model
Debug & troubleshoot

Predictability

Can I get some?

Predictability

Can I get some?

API availability

Predictability

Can I get some?

API availability:

- **< 11 - unsupported**

Predictability

Can I get some?

API availability:

- < 11 Unsupported
- **11-14 Supported, default off**

(= “Support”*, RAM req. too high)

Predictability

Can I get some?

API availability:

- < 11 Unsupported
- 11-14 Supported, default off
- **> 14 Supported, default on**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- **OpenGL/ES implementation of Canvas ops**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- **OpenGL/ES implementation of Canvas ops**
  - **A lot of functions to implement**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- **OpenGL/ES implementation of Canvas ops**
  - A lot of functions to implement
  - **Implementation varies based on HW/FW**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- **OpenGL/ES implementation of Canvas ops**
  - A lot of functions to implement
  - Implementation varies based on HW/FW
  - **Complex implementation**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- Complex OpenGL/ES implementation
- **Support came slow**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- Complex OpenGL/ES implementation
- **Support came slow (standard views first)**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- Complex OpenGL/ES implementation
- Support came slow (standard views first)
- **When using unsupported calls**

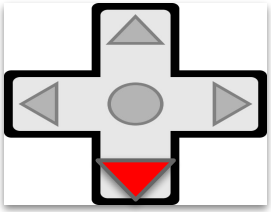
Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- Complex OpenGL/ES implementation
- Support came slow (standard views first)
- **When using unsupported calls:**
  - **Invisible UI elements**



Predictability; Invisible UI elements

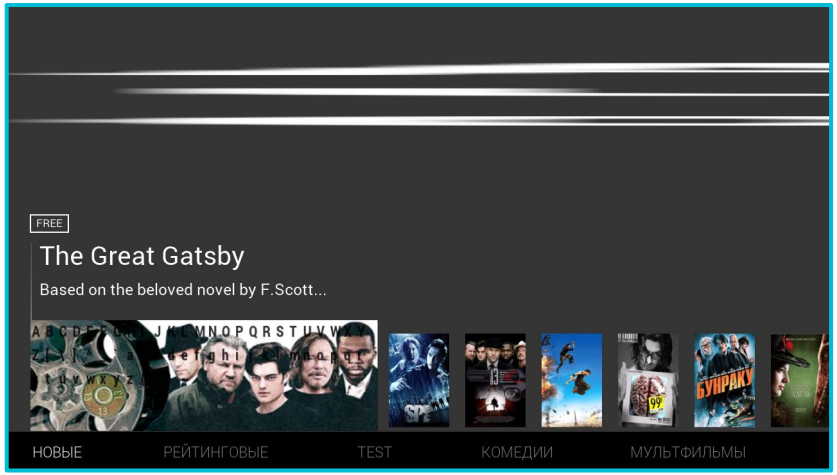
Predictability

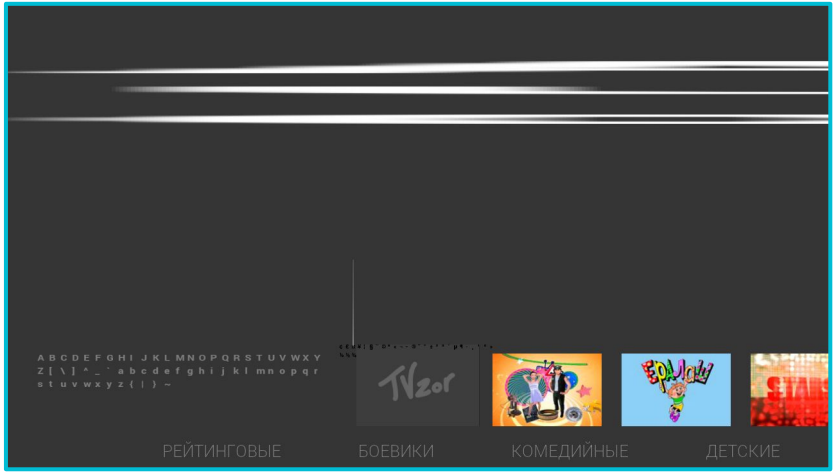
Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- Complex OpenGL/ES implementation
- Support came slow (standard views first)
- **When using unsupported calls:**
  - Invisible UI elements
  - **Badly rendered pixels**





Predictability; Invisible UI elements

Predictability

Can I get some?

API availability: (“11-14”, 14+)

***”Support”**

- Complex OpenGL/ES implementation
- Support came slow (standard views first)
- **When using unsupported calls:**
  - Invisible UI elements
  - Badly rendered pixels
  - **Exceptions**

Predictability

Can I get some?

API availability: (“11-14”, 14+)

*”Support”

- Complex OpenGL/ES implementation
- Support came slow (standard views first)
- When using unsupported calls:
  - Invisible UI elements
  - Badly rendered pixels
  - Exceptions

Predictability

Can I get some?

When using standard views only - OK!

Predictability

Can I get some?

When using standard views only - OK!

When API supports calls in your custom views

Predictability

Can I get some?

When using standard views only - OK!

When API supports calls in your custom views

(On standard Android, tested devices)

Predictability

Can I get some?

When using standard views only - OK!

When API supports calls in your custom views

(/guide/topics/graphics/hardware-accel.html#drawing-support)

Debugging

Tools, usage & prognosis

Rendering

How big is your GPU?

Rendering

How big is your GPU?

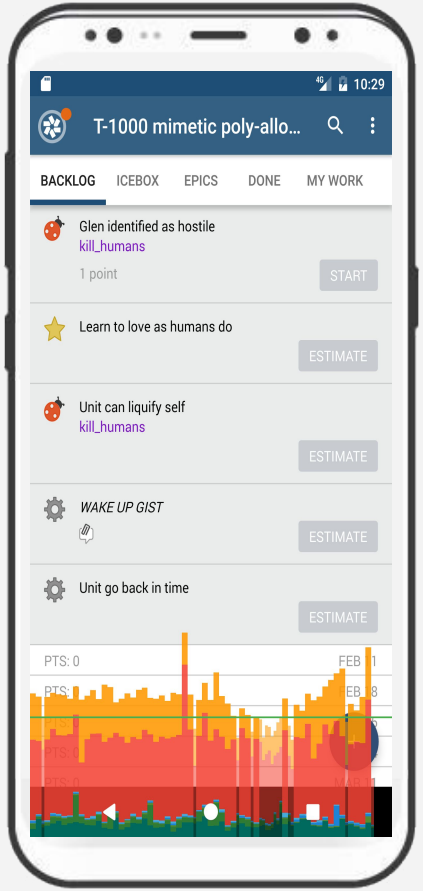
Profile GPU Rendering tool

Rendering

How big is your GPU?

Profile GPU Rendering tool:

- **Enabled via the developer's options menu**

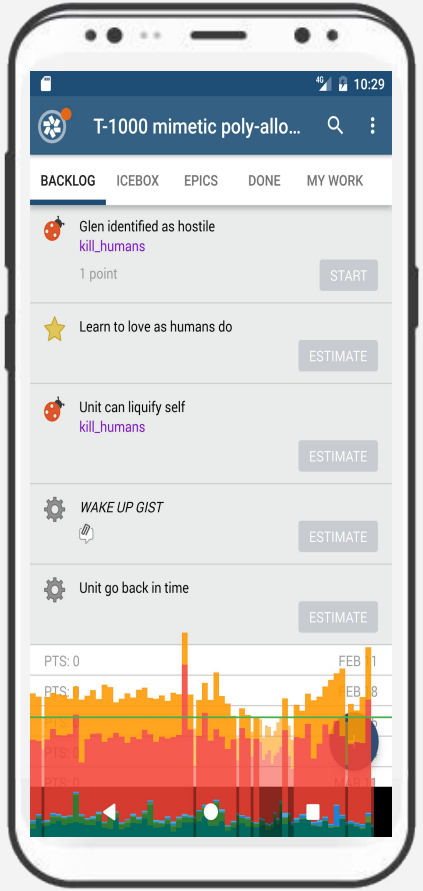


Rendering

How big is your GPU?

Profile GPU Rendering tool:

- **Enabled:**
  - Via the developer's options menu
  - **On devices running API 16+**

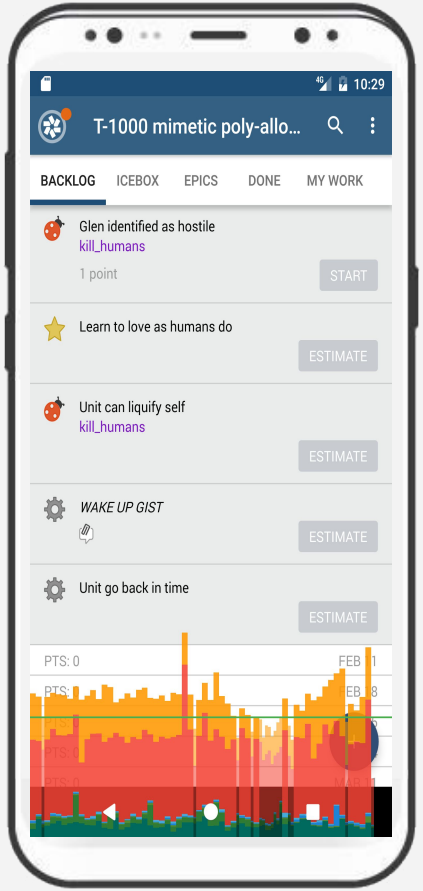


Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- **Displays a graph for each visible app**

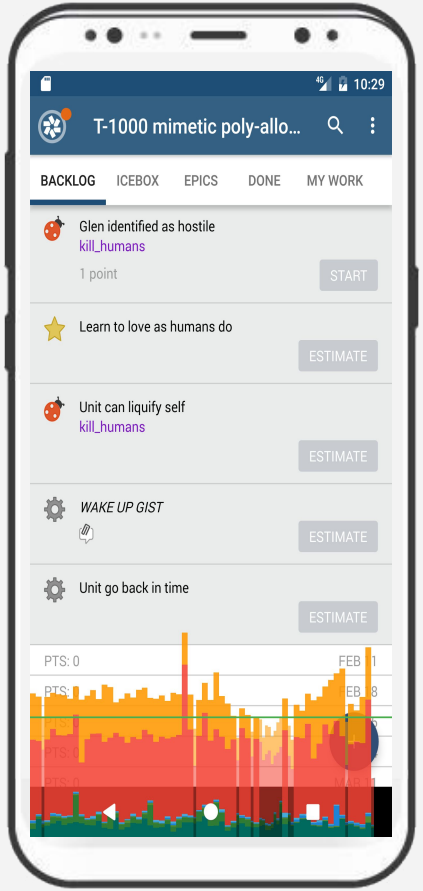


Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- **Displays:**
  - A graph for each visible app
  - **Vertical bars = frames**

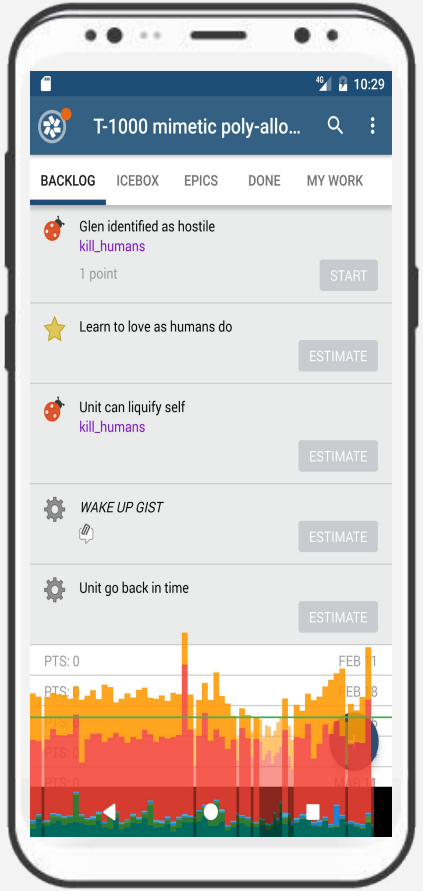


Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- **Displays:**
  - A graph for each visible app
  - Vertical bars = frames
  - **Height of vertical bars = time in Ms**

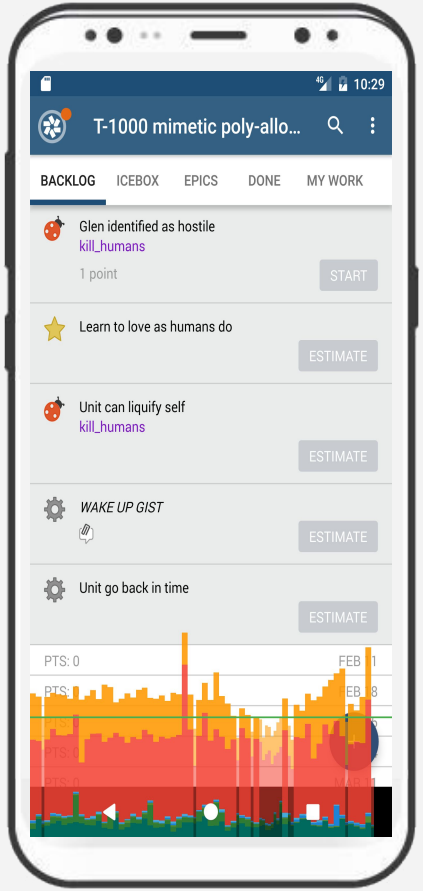


Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- **Displays:**
  - A graph for each visible app
  - Vertical bars = frames
  - Height of vertical bars = time in Ms
  - **Color of part = stage in pipeline**

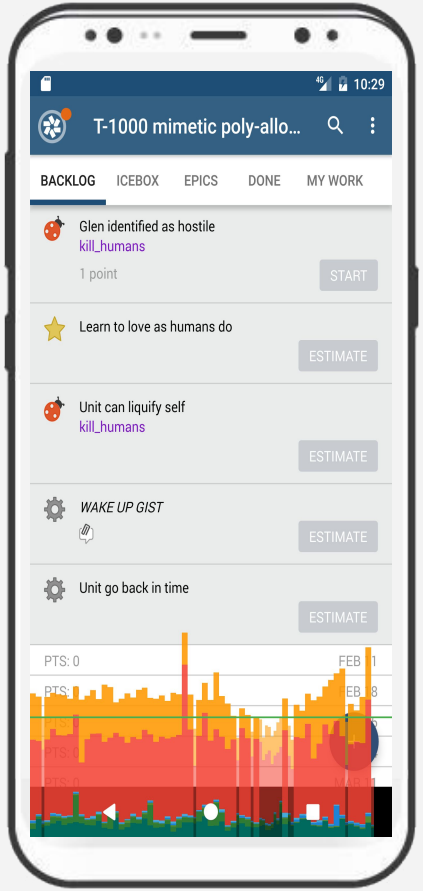


Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- **Displays:**
  - A graph for each visible app
  - Vertical bars = frames
  - Height of vertical bars = time in Ms
  - **Color of part = stage in pipeline (changes by API)**

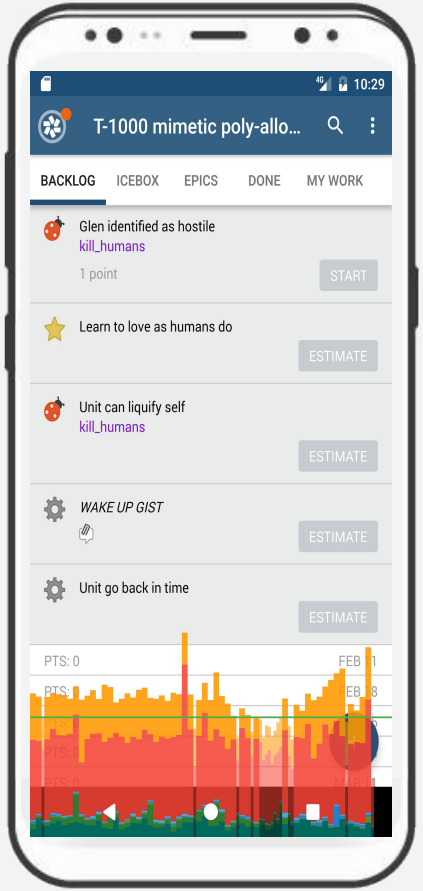


Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- **Displays:**
  - A graph for each visible app
  - Vertical bars = frames
  - Height of vertical bars = time in Ms
  - Color of part = stage in pipeline
  - **Vertical line = 16Ms**

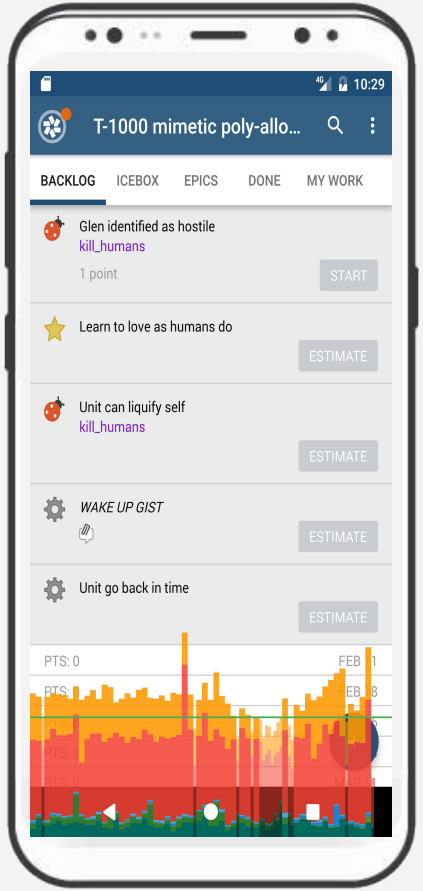


Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- **Displays:**
  - A graph for each visible app
  - Vertical bars = frames
  - Height of vertical bars = time in Ms
  - Color of part = stage in pipeline
  - **Vertical line = 16Ms (for 60 FPS)**



Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- Displays:
  - A graph for each visible app
  - Vertical bars = frames
  - Height of vertical bars = time in Ms
  - Color of part = stage in pipeline
  - Vertical line = 16Ms (for 60 FPS)
- **Diagnosis**

Rendering

How big is your GPU?

Profile GPU Rendering tool:

- Enabled (16+, developer options)
- Displays:
  - A graph for each visible app
  - Vertical bars = frames
  - Height of vertical bars = time in Ms
  - Color of part = stage in pipeline
  - Vertical line = 16Ms (for 60 FPS)
- **Diagnosis graph > line = problem!**

Rederring

Stages and their meaning

Rendering

Stages and their meaning

Input

Rendering

Stages and their meaning

Input - handles input events

Rendering

Stages and their meaning

Input - handles input events

If too long

- **Check the input-handler event callbacks**

Rendering

Stages and their meaning

Input - handles input events

If too long

- Check the input-handler event callbacks
- **Might be too much/too complex work**

Rendering

Stages and their meaning

Input - handles input events

If too long

- Check the input-handler event callbacks
- Might be too much/too complex work
- **Remove tasks/offload to BG thread**

Rendering

Stages and their meaning

Animation

Rendering

Stages and their meaning

Animation - time it takes for animations to run

Rendering

Stages and their meaning

Animation - time it takes for animations to run

If too long

Rendering

Stages and their meaning

Animation - time it takes for animations to run

If too long

- **Check animation callbacks**

Rendering

Stages and their meaning

Animation - time it takes for animations to run

If too long

- **Check animation callbacks:**
  - **ObjectAnimator**

Rendering

Stages and their meaning

Animation - time it takes for animations to run

If too long

- **Check animation callbacks:**
  - ObjectAnimator
  - **ViewPropertyAnimator**

Rendering

Stages and their meaning

Animation - time it takes for animations to run

If too long

- **Check animation callbacks:**
  - ObjectAnimator
  - ViewPropertyAnimator
  - **Transitions**

Rendering

Stages and their meaning

Animation - time it takes for animations to run

If too long

- Check animation callbacks
- **Likely to be due to a prop. change in anim.**

Rendering

Stages and their meaning

Animation - time it takes for animations to run

If too long

- Check animation callbacks
- **Likely to be due to a prop. change in anim. (Example: RecyclerView Fling animation)**

Rendering

Stages and their meaning

Measure

Rendering

Stages and their meaning

Measure - calc. views + hierarchy up to root

Rendering

Stages and their meaning

Measure - calc. views + hierarchy up to root

If too long

Rendering

Stages and their meaning

Measure - calc. views + hierarchy up to root

If too long

- **Check code added to OnMeasure/OnLayout**

Rendering

Stages and their meaning

Measure - calc. views + hierarchy up to root

If too long

- Check code added to OnMeasure/OnLayout
- **Debug your hierarchy!**

Rendering

Stages and their meaning

Measure - calc. views + hierarchy up to root

If too long

- Check code added to OnMeasure/OnLayout
- Debug your hierarchy!
- **Check for overdraw**

Rendering

Stages and their meaning

Draw

Rendering

Stages and their meaning

Draw - handles the actual rendering operations

Rendering

Stages and their meaning

Draw - handles the actual rendering operations

If too long

Rendering

Stages and their meaning

Draw - handles the actual rendering operations

If too long

- **Check code added to OnDraw/DispatchDraw**

Rendering

Stages and their meaning

Draw - handles the actual rendering operations

If too long

- Check code added to OnDraw/DispatchDraw
- **Debug hierarchy!**

Rendering

Stages and their meaning

Draw - handles the actual rendering operations

If too long

- Check code added to OnDraw/DispatchDraw
- Debug hierarchy!
- **Check for overdraw**

Rendering

Stages and their meaning

Upload

Rendering

Stages and their meaning

Upload - time to load CPU bitmaps to GPU

Rendering

Stages and their meaning

Upload - time to load CPU bitmaps to GPU
(Impacted by RAM, GPU and CPU capacity)

Rendering

Stages and their meaning

Upload - time to load CPU bitmaps to GPU

If too long

Rendering

Stages and their meaning

Upload - time to load CPU bitmaps to GPU

If too long

- **Make sure the sizes match (avoid scaling)**

Rendering

Stages and their meaning

Upload - time to load CPU bitmaps to GPU

If too long

- Make sure the sizes match (avoid scaling)
- **Take advantage of prepareToDraw()**

Rendering

Stages and their meaning

Upload - time to load CPU bitmaps to GPU

If too long

- Make sure the sizes match (avoid scaling)
- Take advantage of prepareToDraw()
- **Check for overdraw**

Rendering

Stages and their meaning

Issue - the time it takes to draw the display lists

Rendering

Stages and their meaning

Issue - the time it takes to draw the display lists

Swap - the time it takes to swap buffers

Rendering

Stages and their meaning

Issue - the time it takes to draw the display lists

Swap - the time it takes to swap buffers

If too long

Rendering

Stages and their meaning

Issue - the time it takes to draw the display lists

Swap - the time it takes to swap buffers

If too long

- **Try aggregating commands**

```
for (int i = 0; i < 1000; i++) {
    canvas.drawPoint()
}

//Vs

canvas.drawPoints(mThousandPointArray);
```

Rendering

Stages and their meaning

Misc. - on the main thread, not rendering

Rendering

Stages and their meaning

Misc. - on the main thread, not rendering

If too long

Rendering

Stages and their meaning

Misc. - on the main thread, not rendering

If too long

- **There's work that needs to be on a BG thread**

Rendering

Stages and their meaning

Misc. - on the main thread, not rendering

If too long

- There's work that needs to be on a BG thread
- **Use Systrace and/or Tracer**

Overdraw

How many times is too much?

Overdraw

How many times is too much

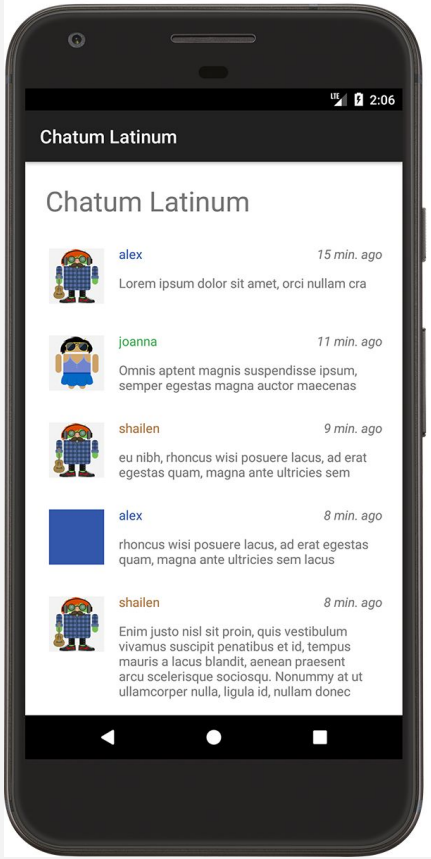
Debug GPU Overdraw

Overdraw

How many times is too much

Debug GPU Overdraw

- **Enabled via the developer's options menu**

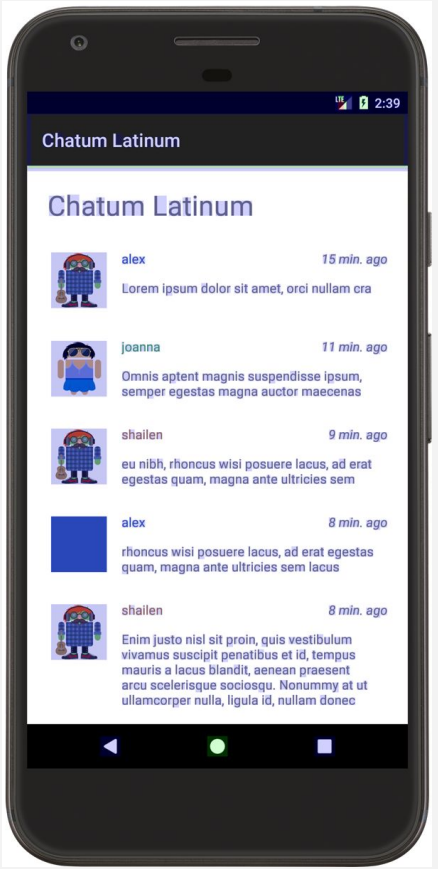


Overdraw

How many times is too much

Debug GPU Overdraw

- Enabled via the developer's options menu
- **Displays:**
  - **Blue = Overdrawn 1 time**



Overdraw

How many times is too much

Debug GPU Overdraw

- Enabled via the developer's options menu
- **Displays:**
  - Blue = Overdrawn 1 time
  - **Green = Overdraw 2 times**

Overdraw

How many times is too much

Debug GPU Overdraw

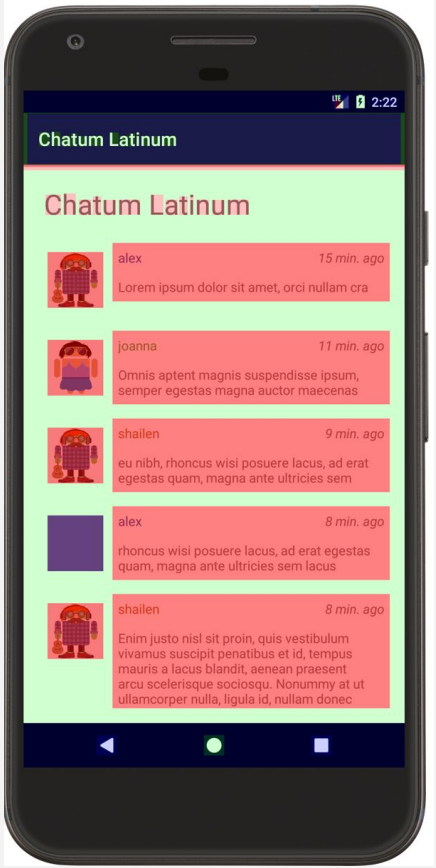
- Enabled via the developer's options menu
- **Displays:**
  - Blue = Overdrawn 1 time
  - Green = Overdraw 2 times
  - **Pink = Overdraw 3 times**

Overdraw

How many times is too much

Debug GPU Overdraw

- Enabled via the developer's options menu
- **Displays:**
  - Blue = Overdrawn 1 time
  - Green = Overdraw 2 times
  - Pink = Overdraw 3 times
  - **Red = Overdraw 4 times**



Overdraw

How many times is too much

Debug GPU Overdraw

Diagnosis

Overdraw

How many times is too much

Debug GPU Overdraw

Diagnosis

- **W/ same pixel drawn > 1 in the frame**

Overdraw

How many times is too much

Debug GPU Overdraw

Diagnosis

- W/ same pixel drawn > 1 in the frame
- **Extra (redundant) GPU effort**

Overdraw

How many times is too much

Debug GPU Overdraw

Diagnosis

- W/ same pixel drawn > 1 in the frame
- Extra (redundant) GPU effort
- **Fix whenever possible**

Overdraw fix

Work smarter, not harder

Overdraw fix

Work smarter, not harder

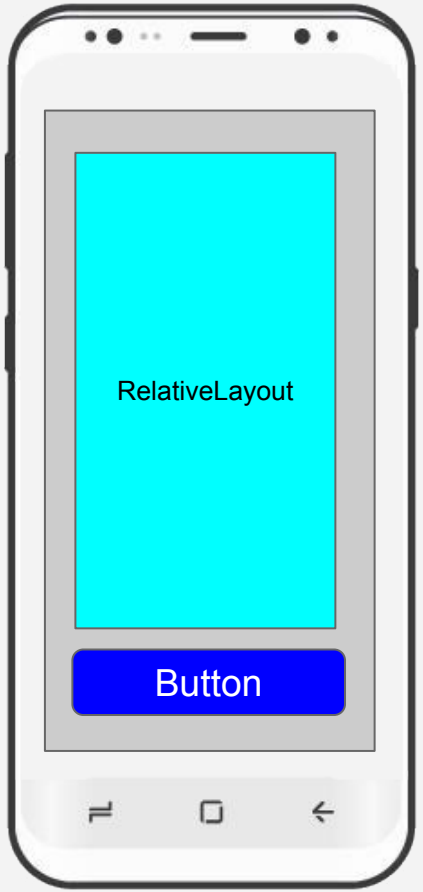
Remove unneeded backgrounds

Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

- **RelativeLayout**

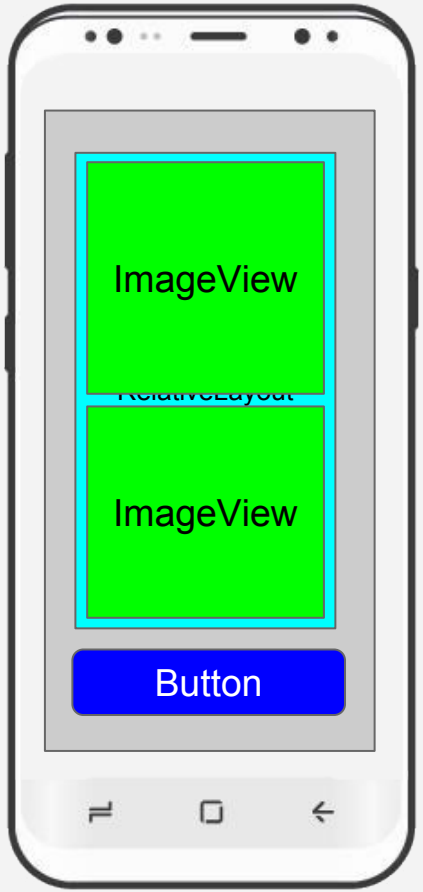


Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

- **RelativeLayout > 2 ImageViews**

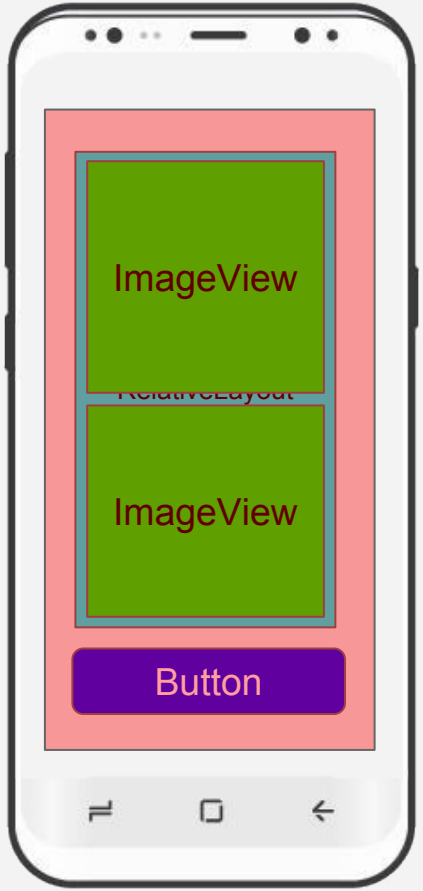


Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

- **RelativeLayout > 2 ImageViews = overdraw!**

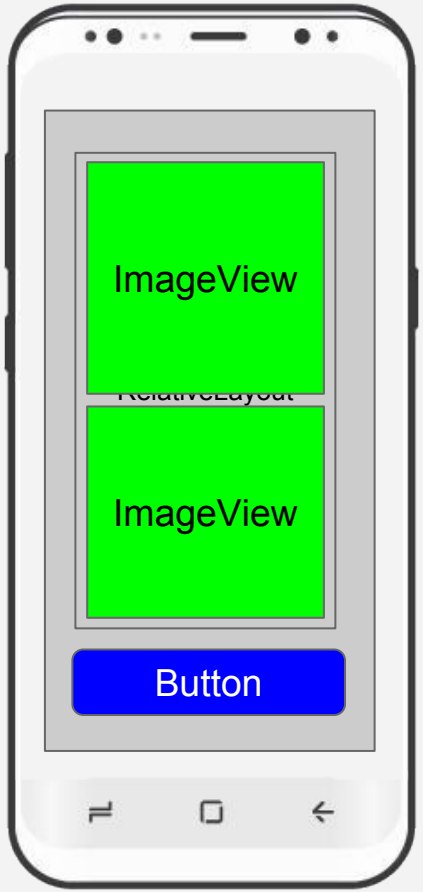


Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

- RelativeLayout > 2 ImageViews = overdraw!
- Remove background from RL = no overdraw
- **Use Layout Inspector for less obvious issues**



Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

Flatten view hierarchy

Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

Flatten view hierarchy

- **Debug hierarchy!**

Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

Flatten view hierarchy

Reduce transparency

Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

Flatten view hierarchy

Reduce transparency

- **Like a background you don’t see**

Overdraw fix

Work smarter, not harder

Remove unneeded backgrounds

Flatten view hierarchy

Reduce transparency

- Like a background you don’t see
- **+ 2X measure**

Debug hierarchy!

The time is now

Debug hierarchy!

Root issue explained

Hierarchy is represented by a logical tree

Debug hierarchy!

Root issue explained

Hierarchy is represented by a logical tree

Tree iteration during the layout & measure steps

Debug hierarchy!

Root issue explained

Hierarchy is represented by a logical tree

Tree iteration during the layout & measure steps

- **Impacted by tree depth (height)**

Debug hierarchy!

Root issue explained

Hierarchy is represented by a logical tree

Tree iteration during the layout & measure steps

- **Impacted**
  - **By tree depth (height)**
  - **By # of iterations needed**

Debug hierarchy!

Root issue explained

Hierarchy is represented by a logical tree

Iteration impacted by tree depth + # iterations

Tree depth = amount of nested views

Debug hierarchy!

Root issue explained

Hierarchy is represented by a logical tree
Iteration impacted by tree depth + # iterations
Tree depth = amount of nested views
 # iterations = increases for relative positioning

Debug hierarchy!

Root issue explained

Hierarchy is represented by a logical tree
Iteration impacted by tree depth + # iterations
Tree depth = amount of nested views
 # iterations = increases for relative positioning
(=Double taxation)

Troubleshoot hierarchy!

What to do?

Troubleshoot Hierarchy!

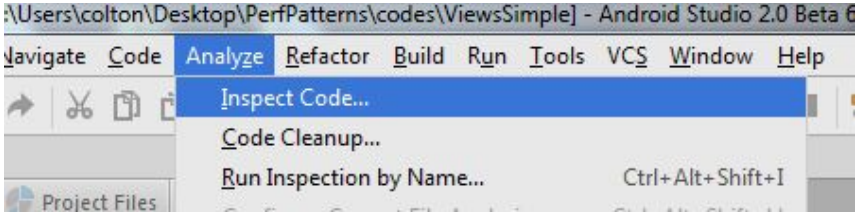
What to do?

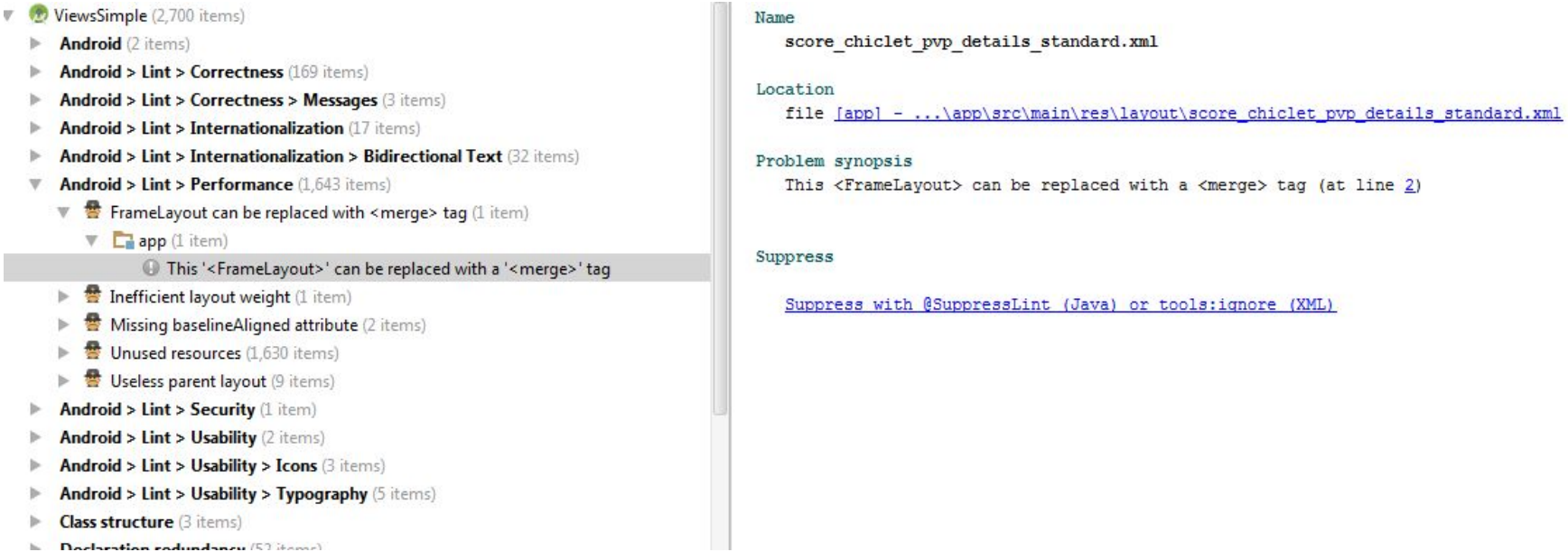
Remove redundant nested layouts

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)





Troubleshoot Hierarchy! (Lint)

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include:

- **<Include/> to layout = additional layer**

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include:

- <Include/> to layout = additional layer
- **<Merge/> + <Include> = direct placement**

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include

Adopt a “cheaper” layout

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include

Adopt a “cheaper” layout:

- **Flatten hierarchy**

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include

Adopt a “cheaper” layout:

- Flatten hierarchy
- **Use non-relative layouts (reduce iterations)**

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include

Adopt a “cheaper” layout:

- Flatten hierarchy
- **Use non-relative layouts (reduce iterations) (ConstraintLayout is king)**

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include

Adopt a “cheaper” layout:

- Flatten hierarchy
- **Use non-relative layouts (reduce iterations) (ConstraintLayout is king - API 24+)**

Troubleshoot Hierarchy!

What to do?

Remove redundant nested layouts (Lint can help)

Adopt merge/include

Adopt a “cheaper” layout:

- Flatten hierarchy
- **Use non-relative layouts (reduce iterations) (ConstraintLayout is king - API 24+) (GridLayout/TableLayout/LinearLayout)**

General debugging

How would you know it’s the GPU?

General debugging

How would you know it’s the GPU?

How would you know it’s the GPU?

General debugging

How would you know it's the GPU?

How would you know it's the GPU?

- **Issue reproducible on specific:**
  - **Devices (regardless of API level)**

General debugging

How would you know it's the GPU?

How would you know it's the GPU?

- **Issue reproducible on specific:**
  - Devices (regardless of API level)
  - **API levels (specifically)**

General debugging

How would you know it's the GPU?

How would you know it's the GPU?

- **Issue reproducible on specific:**
  - Devices (regardless of API level)
  - API levels (specifically)
  - **GPU "families"**

General tips & tricks

Scope

How to know it’s the GPU? (specific by HW/FW)

Set the scope

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope:

- **Application, Activity, Window and layer**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope:

- Application, Activity, Window and layer
- **Decrease granularity until you find the culprit**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope:

- Application, Activity, Window and layer
- Decrease granularity until you find the culprit
- **Debug and/or settle on the right scope**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- **Buffer cache since API 1**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- **LAYER_TYPE_NONE**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- **LAYER_TYPE_NONE**
  - **No buffer**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- **LAYER_TYPE_NONE**
  - No buffer
  - **Rendered by software**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- **LAYER_TYPE_HARDWARE**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- **LAYER_TYPE_HARDWARE:**
  - **Backed by hardware texture buffer**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- **LAYER_TYPE_HARDWARE:**
  - Backed by hardware texture buffer
  - **Rendered in the hardware model**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- **LAYER_TYPE_HARDWARE:**
  - Backed by hardware texture buffer
  - Rendered in the hardware model
  - **Best for performance**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- **LAYER_TYPE_HARDWARE:**
  - Backed by hardware texture buffer
  - Rendered in the hardware model
  - **Best for performance**
  - **Anim. added to texture w/o redrawing**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- LAYER_TYPE_HARDWARE (HW/HW)
- **LAYER_TYPE_SOFTWARE**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- LAYER_TYPE_HARDWARE (HW/HW)
- **LAYER_TYPE_SOFTWARE**
    - **Backed by a Bitmap**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- LAYER_TYPE_HARDWARE (HW/HW)
- **LAYER_TYPE_SOFTWARE**
  - Backed by a Bitmap
  - **Rendered in the software model**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- LAYER_TYPE_HARDWARE (HW/HW)
- **LAYER_TYPE_SOFTWARE**
  - Backed by a Bitmap
  - Rendered in the software model
  - **Best for compatibility**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- LAYER_TYPE_HARDWARE (HW/HW)
- LAYER_TYPE_SOFTWARE
- **Canvas.isHardwareAccelerated()**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)

Set the scope (either for debug or as a solution)

Layer up:

- Buffer cache since API 1
- LAYER_TYPE_NONE (None/SW)
- LAYER_TYPE_HARDWARE (HW/HW)
- LAYER_TYPE_SOFTWARE
- **Canvas.isHardwareAccelerated()**
  (window can still be drawn in the SW model!)

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- **60 FPS animations of a complex view is hard**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- 60 FPS animations of a complex view is hard
- **Rendering the view to a HW texture is easier**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- 60 FPS animations of a complex view is hard
- **Rendering the view to a HW texture is easier (rendering just a texture and not the view)**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- 60 FPS animations of a complex view is hard
- **Rendering the view to a HW texture is easier**
  - **Prop changes that will not invalidate:**
    - **alpha (transparency)**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- 60 FPS animations of a complex view is hard
- **Rendering the view to a HW texture is easier**
  - **Prop changes that will not invalidate:**
    - alpha (transparency)
    - **x,y,translationX,translationY(position)**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- 60 FPS animations of a complex view is hard
- **Rendering the view to a HW texture is easier**
  - **Prop changes that will not invalidate:**
    - alpha (transparency)
    - x,y,translationX,translationY(position)
    - **scaleX, scaleY (size)**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- 60 FPS animations of a complex view is hard
- **Rendering the view to a HW texture is easier**
  - **Prop changes that will not invalidate:**
    - alpha (transparency)
    - x,y,translationX,translationY(position)
    - scaleX, scaleY (size)
    - **rotationX,rotationY (3D orientation)**

General tips & tricks

Scope and layers

How to know it's the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- 60 FPS animations of a complex view is hard
- **Rendering the view to a HW texture is easier**
  - **Prop changes that will not invalidate:**
    - alpha (transparency)
    - x,y,translationX,translationY(position)
    - scaleX, scaleY (size)
    - rotationX,rotationY (3D orientation)
    - **pivotX,pivotY**

```
view.setLayerType(View.LAYER_TYPE_HARDWARE, null);
ObjectAnimator.ofFloat(view, "rotationY", 180).start();
```

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations

- 60 FPS animations of a complex view is hard
- Rendering the view to a HW texture is easier
- **But only enable them while animation!**

```
view.setLayerType(View.LAYER_TYPE_HARDWARE, null);
ObjectAnimator animator = ObjectAnimator.ofFloat(view,
"rotationY", 180);
animator.addListener(new AnimatorListenerAdapter() {
    @Override
    public void onAnimationEnd(Animator animation) {
        view.setLayerType(View.LAYER_TYPE_NONE, null);
    }
});
animator.start();
```

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations
Custom view top tip

```
private class PieView extends View {
public PieView(Context context) {
        super(context);
      if (!isInEditMode()) {
          setLayerType(View.LAYER_TYPE_HARDWARE,
      null);
      }
     }
}
```

General tips & tricks

Scope and layers

- How to know it’s the GPU? (specific by HW/FW)
- Set the scope (either for debug or as a solution)
- Layer up (choose the right one)
- Layers and animations
- Custom view top tip
- **Don't create render objects in draw methods**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations
Custom view top tip
Don't create render objects in draw methods
(=Path, Paint - Garbage collection!)

General tips & tricks

Scope and layers

- How to know it’s the GPU? (specific by HW/FW)
- Set the scope (either for debug or as a solution)
- Layer up (choose the right one)
- Layers and animations
- Custom view top tip
- Don't create render objects in draw methods
- **Don't modify shapes too often**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations
Custom view top tip
Don't create render objects in draw methods
Don't modify shapes too often (new texture maps)

General tips & tricks

Scope and layers

- How to know it’s the GPU? (specific by HW/FW)
- Set the scope (either for debug or as a solution)
- Layer up (choose the right one)
- Layers and animations
- Custom view top tip
- Don't create render objects in draw methods
- Don't modify shapes too often (new texture maps)
- **Don't modify bitmaps too often**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations
Custom view top tip
Don't create render objects in draw methods
Don't modify shapes too often (new texture maps)
Don't modify bitmaps too often (new GPU upload)

General tips & tricks

Scope and layers

- How to know it’s the GPU? (specific by HW/FW)
- Set the scope (either for debug or as a solution)
- Layer up (choose the right one)
- Layers and animations
- Custom view top tip
- Don't create render objects in draw methods
- Don't modify shapes too often (new texture maps)
- Don't modify bitmaps too often (new GPU upload)
- **Use alpha with care**

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations
Custom view top tip
Don't create render objects in draw methods
Don't modify shapes too often (new texture maps)
Don't modify bitmaps too often (new GPU upload)
Use alpha with care (off-screen = 2X fill-rate)

General tips & tricks

Scope and layers

How to know it’s the GPU? (specific by HW/FW)
Set the scope (either for debug or as a solution)
Layer up (choose the right one)
Layers and animations
Custom view top tip
Don't create render objects in draw methods
Don't modify shapes too often (new texture maps)
Don't modify bitmaps too often (new GPU upload)
Use alpha with care (off-screen = 2X fill-rate)
(Use LAYER_TYPE_HARDWARE)

Questions?


“**We** should be building
great things. Things that
Don’t **yet** exist”
Mashable

Hope you liked it
Thanks for listening!

Royi Benyossef
(royi@samsungnext.com)

Google Developers
Experts