The guide to Hardware acceleration

"G(PU) force"- Royi Benyossef

SAMSUNG NE



Introduction

Who? What?







Introduction Royi Benyossef Android developer since 2009



Android developer since 2009 **Tech community activist, speaker and founder**



Android developer since 2009 Tech community activist, speaker and founder **Mentor at * accelerator**





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



Introduction Rovi Renvosed

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











Innovation arm focused on software products and services



Innovation arm focused on software products and services Investments (seed to B rounds)



Innovation arm focused on software products and services Investments (seed to B rounds) Independent product creation



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?



X

What do we want?

To understand how the GPU works



What do we want?

To understand how the GPU works?

What do we want?

To understand how the GPU works **Why?**

What do we want?

To understand how the GPU works **Why?**

• Smoother animations



What do we want?

To understand how the GPU works **Why?**

- Smoother animations
- Sleek transitions

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?



Hardware acceleration

Using the specific hardware to do stuff faster

Hardware acceleration

Using the specific hardware to do stuff faster **In our case**

Hardware acceleration

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

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

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

Hardware acceleration

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

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**

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

Special hardware

- Designed for rapid matrix mathematical operations robustly

Special hardware

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

Special hardware

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

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 <u>immediately</u>



Definition Android drawing models Software drawing model - Invalidate() performed immediately

- "Dirty region" calculated



Definition Android drawing models Software drawing model - Invalidate() performed <u>immediately</u> - "Dirty region" contains: - Calling (changed) view



Android drawing models

- 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)



Android drawing models

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



Android drawing models

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



Android drawing models

- 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



Android drawing models

- 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



Android drawing models

- 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

Android drawing models

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

Android drawing models

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

Android drawing models

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

Android drawing models

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



Android drawing models

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



Android drawing models

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

Android drawing models

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



Android drawing models

- 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)

Android drawing models

Android drawing models

Software drawing model (immediate, inefficient) Hardware drawing model

- Invalidate() flags a view as "dirty"

Android drawing models

- Invalidate() flags a view as "dirty"
- Changes saved to "display lists"

Android drawing models

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

Android drawing models

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

Android drawing models

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

Android drawing models

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



Android drawing models

- 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)



Android drawing models

- 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



Android drawing models

- 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)



Android drawing models

- 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

Android drawing models

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

Android drawing models

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

Android drawing models

- 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



Android drawing models

- 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

Android drawing models

- 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)

Understand when to use which model

Understand when to use which model



Understand when to use which model **Provide predictability to the HW drawing model**

Understand when to use which model Provide predictability to the HW drawing model **Debug & troubleshoot**

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



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

Can I get some?

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

Can I get some?

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

Can I get some?

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

Can I get some?

- Complex OpenGL/ES implementation
- Support came slow

Can I get some?

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

Can I get some?

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

Can I get some?

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



Predictability; Invisible UI elements

Can I get some?

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





Predictability; Invisible UI elements

X

Can I get some?

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



- 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!

When using standard views only - OK! When API supports calls in your custom views

When using standard views only - OK! When API supports calls in your custom views (On standard Android, tested devices)

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


- Displays:
 - A graph for each visible app
 - Vertical bars = frames

-					⁴⁶ /1 😼 1
3	T-1000) mi	metic p	oly-allo.	Q
BACKLO	DG ICEB	ОХ	EPICS	DONE	MY WOR
	Glen identifi kill_humans		hostile		
1	point				STAR
🔶 I	earn to lov.	e as ł	iumans do		
	Jnit can liqu dll_humans		elf		
100	NAKE UP G	IST			
6	Ø)				
∯ ¹	Jnit go bacł	k in tii	ne		
PTS: 0			1		FEB
PTS: 0	الد اله		L I		EB
21587	لرجيع		11.	1.1.4	114
PTS: 0					MAR
	4				

- Displays:
 - A graph for each visible app
 - Vertical bars = frames
 - Height of vertical bars = time in Ms

BACKLOG ICEBOX EPICS DONE MY WOR Image: Select in identified as hostile kill_humans Image: Select in iteration is a state in iteratina state in iteration is a state in iterationa	Glen identified as hostile kill_humans 1 point STAF Learn to love as humans do ESTIMA Unit can liquify self	Glen identified as hostile kill_humans 1 point STAI Earn to love as humans do ESTIMAT White can liquify self kill_humans ESTIMAT WAKE UP GIST	🤁 T-1000 m	imetic poly-allo	• ۲ ۲ م
kil_humans 1 point STAL Image: state	kil_humans 1 point STAI Learn to love as humans do ESTIMA Unit can liquify self kil_humans ESTIMA WAKE UP GIST	kill_humans 1 point STAT Image: state of the st	BACKLOG ICEBOX	EPICS DONE	MY WOR
ESTIMA Contican liquify self kill_humans	ESTIMA Unit can liquify self kill_humans ESTIMA WAKE UP GIST	ESTIMA ESTIMA Whit can liquify self kill_humans ESTIMA WAKE UP GIST WAKE UP GIST Unit go back in time	kill_humans	is hostile	STAI
kill_humans	kill_humans	kill_humans ESTIMAT Kill_humans ESTIMAT Kill_humans Unit go back in time	+ Learn to love as	humans do	ESTIMA
	WAKE UP GIST	WAKE UP GIST		self	

- Displays:
 - A graph for each visible app
 - Vertical bars = frames
 - Height of vertical bars = time in Ms
 - Color of part = stage in pipeline

R	T-1000	mimetic p	oly-allo.	"≱ ⊒ 1 ୍୍
BACK	LOG ICEBC)X EPICS	DONE	MY WOR
1	Glen identifie	ed as hostile		
	1 point			STAF
	Learn to love	e as humans do	D	
				ESTIMAT
1	Unit can liqu kill_humans	ify self		
‡	WAKE UP GI	ST		
	4			
₽	Unit go back	in time		
PTS:	0			FEE
PTS:		بالروي		
PIS	a a a a	u de la		1.4
				MAE

- 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)

(≝ 1
*	T-1000 m	imetic p	oly-allo	. Q
BACKLO	G ICEBOX	EPICS	DONE	MY WOR
	Glen identified a ill_humans	s hostile		
1	point			STAR
📌 L	earn to love as.	humans do		
	Jnit can liquify : ill_humans	self		
444	WAKE UP GIST			
sile i	Jnit go back in t			
ф, ч	nin go back in i	.ine		
PTS: 0				FEB
PTS:	di sta			EB
PTS: 0	Link		14	
PTS:0				MAE

- 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

*	T-1000 m	nimetic p	oly-allo	୍ଧ 🖬 1
BACKL	OG ICEBOX	EPICS	DONE	MY WOR
	Glen identified a	as hostile		
	1 point			STAF
*	Learn to love as	s humans do		
	Unit can liquify kill_humans	self		
100	WAKE UP GIST			
	2			
₽	Unit go back in	time		
		-		
PTS: 0		ta.		FEB
2151			iu. d	
PTS: 0			11	

- 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)

" ~					⁴⁶ /1 😼 1
	T-	1000 m	imetic p	oly-allo.	. Q
BACKI	_0G	ICEBOX	EPICS	DONE	MY WOR
()		identified a umans	is hostile		
	1 poi	nt			STAR
*	Learr	n to love as	humans do	0	
()		can liquify umans	self		
₽		E UP GIST			
	4				
₽	Unit	go back in t	time		
PTS: (2		-		FEB
PTS.			11	_	EB
		and to	di.	1.1.4	J 14

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

Stages and their meaning

Input - handles input events If too long

- Check the input-handler event callbacks

Stages and their meaning

Input - handles input events If too long

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

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

X

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

X

Stages and their meaning

Animation - time it takes for animations to run

- Check animation callbacks:
 - ObjectAnimator

Stages and their meaning

- Check animation callbacks:
 - ObjectAnimator
 - ViewPropertyAnimator

Stages and their meaning

- Check animation callbacks:
 - ObjectAnimator
 - ViewPropertyAnimator
 - Transitions

Stages and their meaning

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

Stages and their meaning

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

Rendering Stages and their meaning





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

Stages and their meaning

Measure - calc. views + hierarchy up to root **If too long**

- Check code added to OnMeasure/OnLayout

Stages and their meaning

Measure - calc. views + hierarchy up to root **If too long**

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

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

Stages and their meaning

Draw - handles the actual rendering operations **If too long**

Stages and their meaning

Draw - handles the actual rendering operations **If too long**

- Check code added to OnDraw/DispatchDraw

Stages and their meaning

Draw - handles the actual rendering operations **If too long**

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

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

Stages and their meaning Upload - time to load CPU bitmaps to GPU If too long

- Make sure the sizes match (avoid scaling)

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()

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

Stages and their meaning

Issue - the time it takes to draw the display lists

Swap - the time it takes to swap buffers

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

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

//Vs

canvas.drawPoints(mThousandPointArray);

X

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

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

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



How many times is too much

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



How many times is too much

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

How many times is too much

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

How many times is too much

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

G	
Chatum Latinum	
Chatum Latinum	
Ĥ	alex 15 min. ago Lorem ipsum dolor sit amet, orci nullam cra
	joanna 11 min. ago Omnis aptent magnis suspendisse ipsum, semper egestas magna auctor maecenas
Ĥ	shailen 9 min. ago eu nibh, rhoncus wisi posuere lacus, ad erat egestas quam, magna ante ultricies sem
	alex 8 min. ago rhoncus wisi posuere lacus, ad erat egestas quam, magna ante ultricies sem lacus
	shailen 8 min. ago Enim justo nisl sit proin, quis vestibulum vivamus suscipit penatibus et id, tempus mauris a lacus blandit, aenean praesent arcu scelerisque sociosqu. Nonummy at ut ullamcorper nulla, ligula id, nullam donec

Overdraw How many times is too much Debug GPU Overdraw Diagnosis

How many times is too much

Debug GPU Overdraw **Diagnosis**

- W/ same pixel drawn > 1 in the frame

How many times is too much

Debug GPU Overdraw **Diagnosis**

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

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!



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

Work smarter, not harder

Remove unneeded backgrounds Flatten view hierarchy

- Debug hierarchy!

Work smarter, not harder

Remove unneeded backgrounds Flatten view hierarchy **Reduce transparency**

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!

Debug hierarchy! *Root issue explained* Hierarchy is represented by a logical tree

Hierarchy is represented by a logical tree **Tree iteration during the layout & measure steps**

Hierarchy is represented by a logical tree **Tree iteration during the layout & measure steps**

- Impacted by tree depth (height)

Hierarchy is represented by a logical tree **Tree iteration during the layout & measure steps**

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

Hierarchy is represented by a logical tree Iteration impacted by tree depth + # iterations **Tree depth = amount of nested views**

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

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)

Remove redundant nested layouts

Remove redundant nested layouts (Lint can help)

		102	Window	Help
Inspect Code				
Code Cleanup				
Run Inspection by Name		Ctr	I+Alt+Shift	+1
	<u>C</u> ode Cleanup	<u>C</u> ode Cleanup	<u>C</u> ode Cleanup	<u>C</u> ode Cleanup

- 👳 ViewsSimple (2,700 items)
- Android (2 items)
- Android > Lint > Correctness (169 items)
- Android > Lint > Correctness > Messages (3 items)
- Android > Lint > Internationalization (17 items)
- Android > Lint > Internationalization > Bidirectional Text (32 items)
- Android > Lint > Performance (1,643 items)
 - 🔻 🖶 FrameLayout can be replaced with <merge> tag (1 item)
 - 🔻 🗖 app (1 item)
 - O This '<FrameLayout>' can be replaced with a '<merge>' tag
 - 🕨 🔮 Inefficient layout weight (1 item)
 - Missing baselineAligned attribute (2 items)
 - Unused resources (1,630 items)
 - Useless parent layout (9 items)
- Android > Lint > Security (1 item)
- Android > Lint > Usability (2 items)
- Android > Lint > Usability > Icons (3 items)
- Android > Lint > Usability > Typography (5 items)
- Class structure (3 items)
- Dachestian radundancu (52 itams)

Name

score_chiclet_pvp_details_standard.xml

Location

file [app] - ... \app\src\main\res\layout\score_chiclet_pvp_details_standard.xml

Problem synopsis

This <FrameLayout> can be replaced with a <merge> tag (at line $\frac{2}{2}$)

Suppress

Suppress with @SuppressLint (Java) or tools:ignore (XML)

Troubleshoot Hierarchy! (Lint)

X

Remove redundant nested layouts (Lint can help) Adopt merge/include

Remove redundant nested layouts (Lint can help) Adopt merge/include:

- <Include/> to layout = additional layer

Remove redundant nested layouts (Lint can help) Adopt merge/include:

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

Remove redundant nested layouts (Lint can help) Adopt merge/include Adopt a "cheaper" layout:

- Flatten hierarchy

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

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

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

- 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**

Scope and layers

How to know it's the GPU? (specific by HW/FW) **Set the scope:**

- Application, Activity, Window and layer

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

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

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**

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

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

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

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
Scope and layers

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

Scope and layers

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

Scope and layers

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



Scope and layers

- 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



Scope and layers

- 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

Scope and layers

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



Scope and layers

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



Scope and layers

- 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

Scope and layers

- 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

Scope and layers

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



Scope and layers

- 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!)



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**

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

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

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)

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)

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)

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)

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)

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();

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();

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);

X

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**

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!)

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**

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)**

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**

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)**

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

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)

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)



X

"We should be building **great things. Things that** Don't **yet** exist"



Hope you liked it Thanks for listening!



Royi Benyossef (royi@samsungnext.com)

