#perfMatters

About me

Twitter:	@geekanamika
LinkedIn:	https://www.linkedin.com/in/geekanamika/
Medium:	https://medium.com/@anamikatripathi1601/
Github:	https://github.com/geekanamika

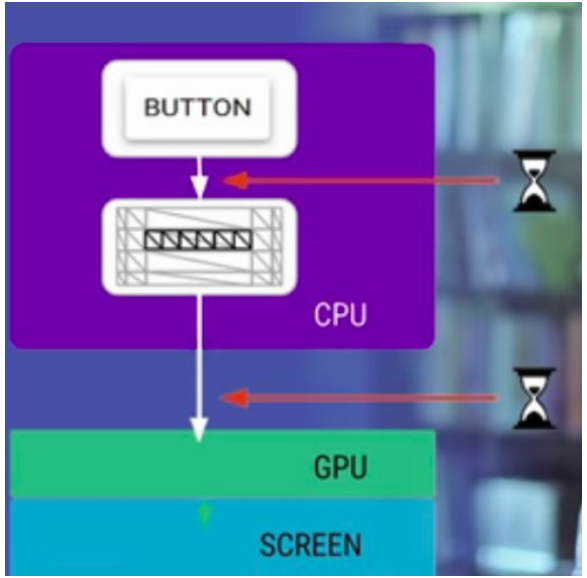
Overdraw

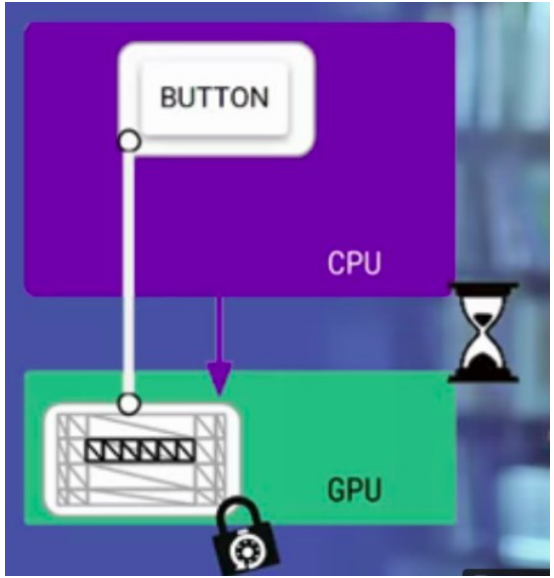
- An app may draw the same pixel more than once within a single frame, an event called *overdraw*.
- For example, if we have a bunch of stacked UI cards, each card hides a portion of the one below it.
- However, the system still needs to draw even the hidden portions of the cards in the stack. This is because stacked cards are rendered according to the painter's algorithm: that is, in back-to-front order.

Android colors UI elements to identify the amount of overdraw as follows:

- **True color:** No overdraw
- **Blue:** Overdrawn 1 time
- **Green:** Overdrawn 2 times
- **Pink:** Overdrawn 3 times
- **Red:** Overdrawn 4 or more times

Reason

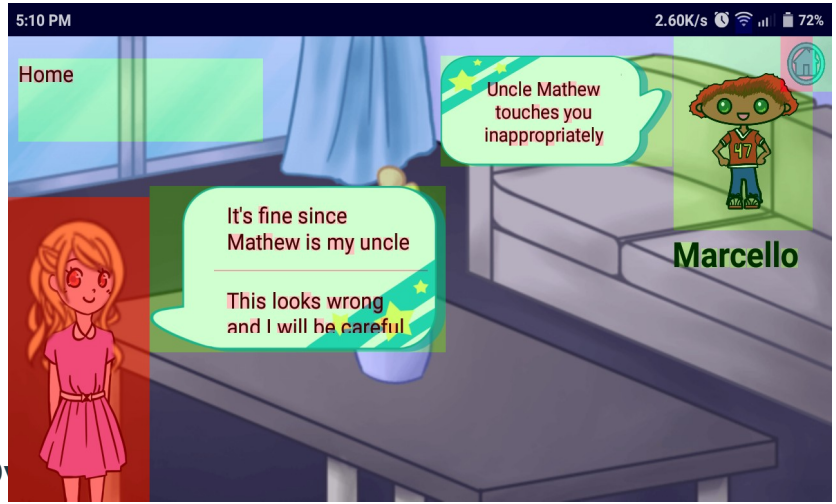

BUTTON
CPU
GPU
SCREEN

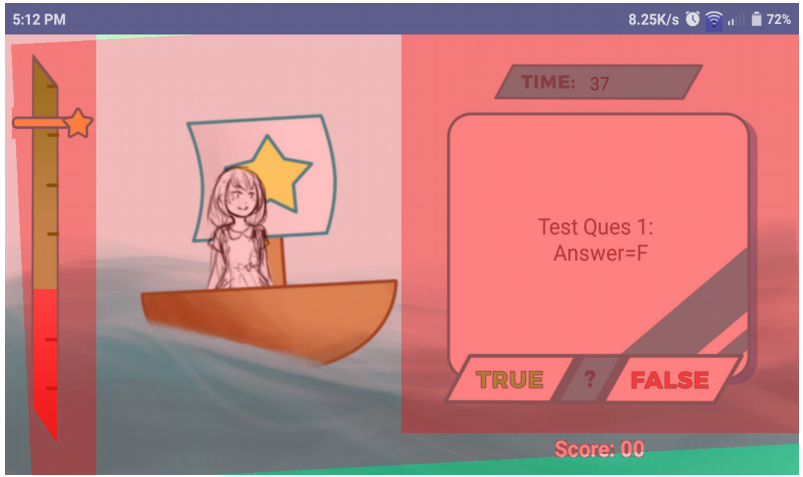

BUTTON
CPU
GPU


Overdraw
13 minutes to Sheraton
22 minutes to Norman
Y. Mineta San Jose...
Visible card
Hides views underneath
Sierra Madre
70°


STORE

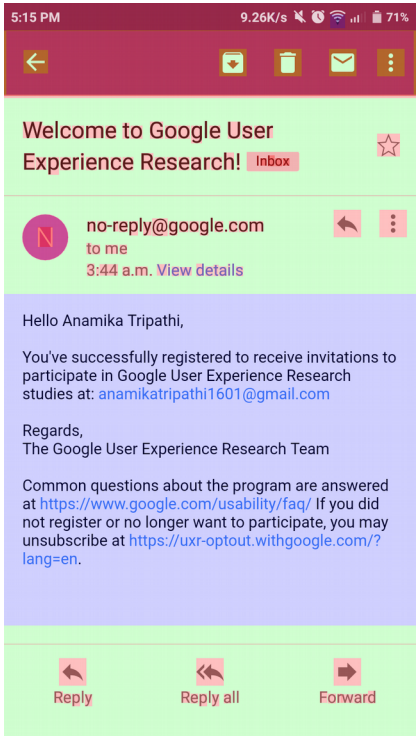

5:11 PM
5.53K/s
72%
Score: 4
METHOD:
FAILURE...
CONDOM
CONTINUE

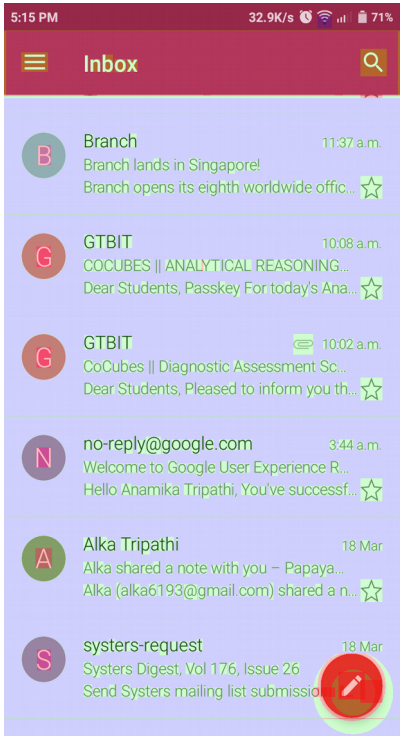

5:10 PM
2.60K/s
72%
Home
Uncle Mathew touches you inappropriately
It's fine since Mathew is my uncle
This looks wrong and I will be careful
Marcello


5:12 PM
8.25K/s
72%
TIME: 37
Test Ques 1:
Answer=F
TRUE
?
FALSE
Score: 00


5:13 PM
12.7K/s
71%
40
HAIR
CLOTHES
ACCESSORIES
5
BUY
5
BUY
5
BUY
5
BUY
5
SELECT
5
BUY
gif


STORE
LIBRARY
gif


5:15 PM
9.26K/s
71%
Welcome to Google User Experience Research!
Inbox
no-reply@google.com
to me
3:44 a.m. View details
Hello Anamika Tripathi,
You've successfully registered to receive invitations to participate in Google User Experience Research studies at: anamikatripathi1601@gmail.com
Regards,
The Google User Experience Research Team
Common questions about the program are answered at https://www.google.com/usability/faq/ If you did not register or no longer want to participate, you may unsubscribe at https://uxr-optout.withgoogle.com/?lang=en.
Reply
Reply all
Forward

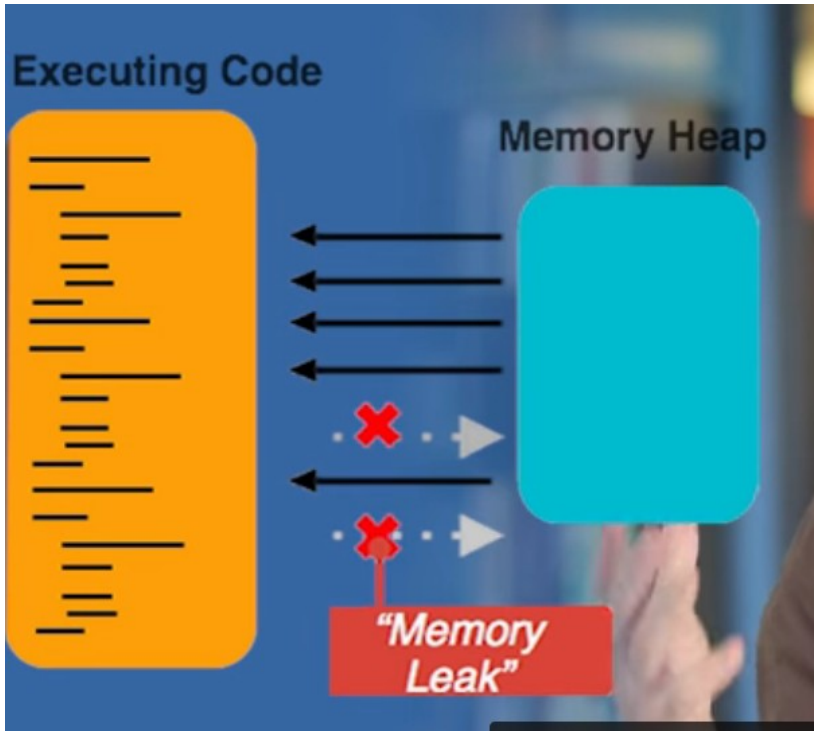

5:15 PM
32.9K/s
71%
Inbox
Branch
11:37 a.m.
Branch lands in Singapore!
Branch opens its eighth worldwide offic...
GTBIT
10:08 a.m.
COCUBES || ANALYTICAL REASONING...
Dear Students, Passkey For today's Ana...
GTBIT
10:02 a.m.
CoCubes || Diagnostic Assessment Sc...
Dear Students, Pleased to inform you th...
no-reply@google.com
3:44 a.m.
Welcome to Google User Experience R...
Hello Anamika Tripathi, You've successf...
Alka Tripathi
18 Mar
Alka shared a note with you – Papaya...
Alka (alka6193@gmail.com) shared a n...
systers-request
18 Mar
Systers Digest, Vol 176, Issue 26
Send Systers mailing list submissio...

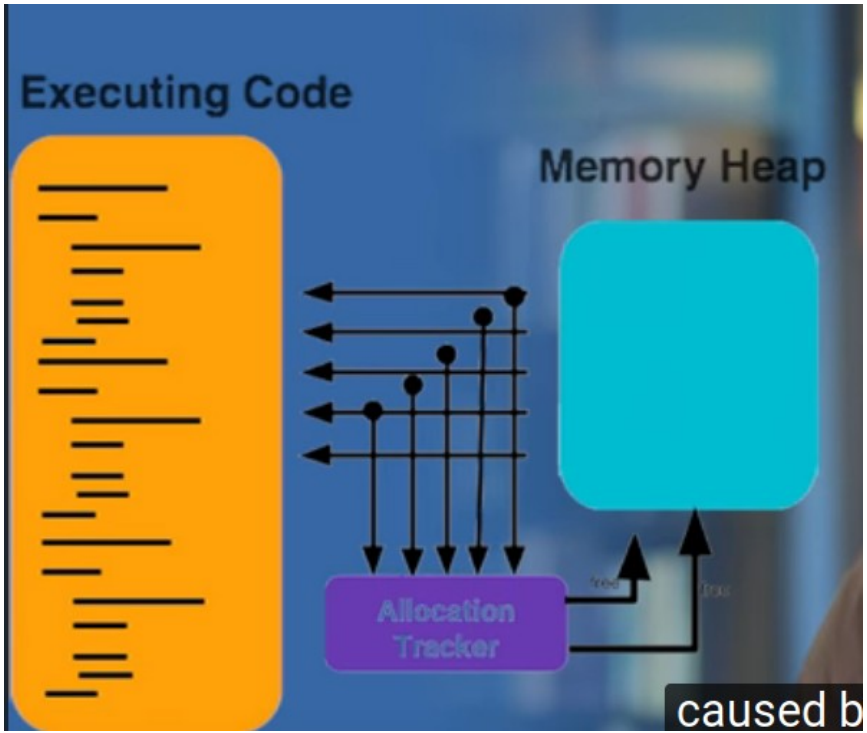
Fixing Overdraw

There are several strategies you can pursue to reduce or eliminate overdraw:

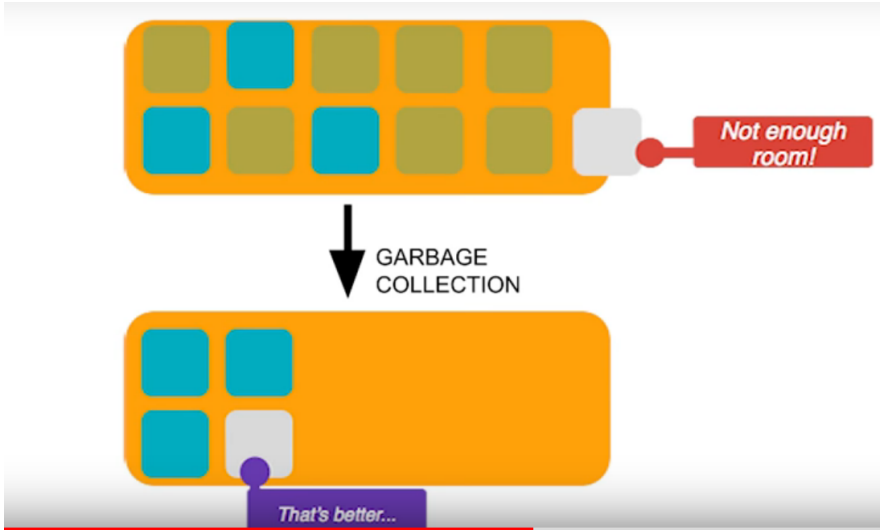
- Removing unneeded backgrounds in layouts.
- Flattening the view hierarchy.
- Reducing transparency.

Memory performance



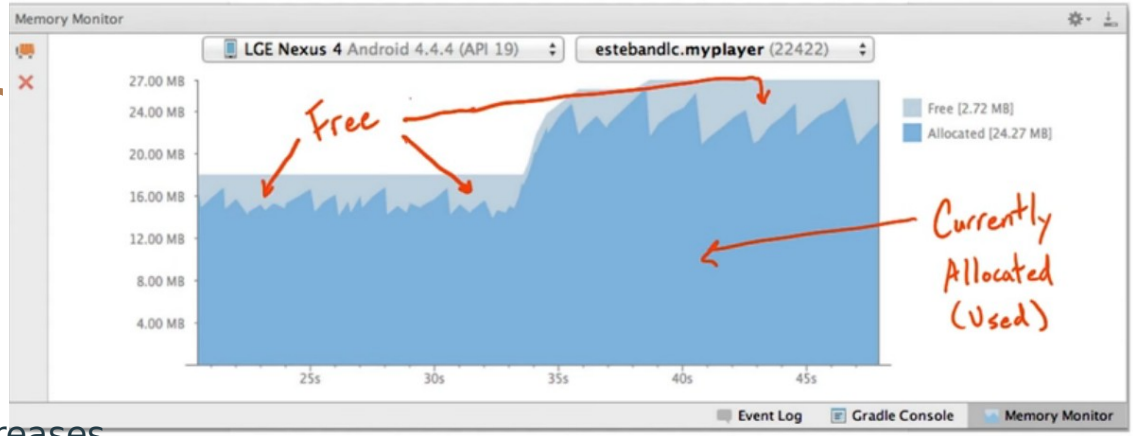


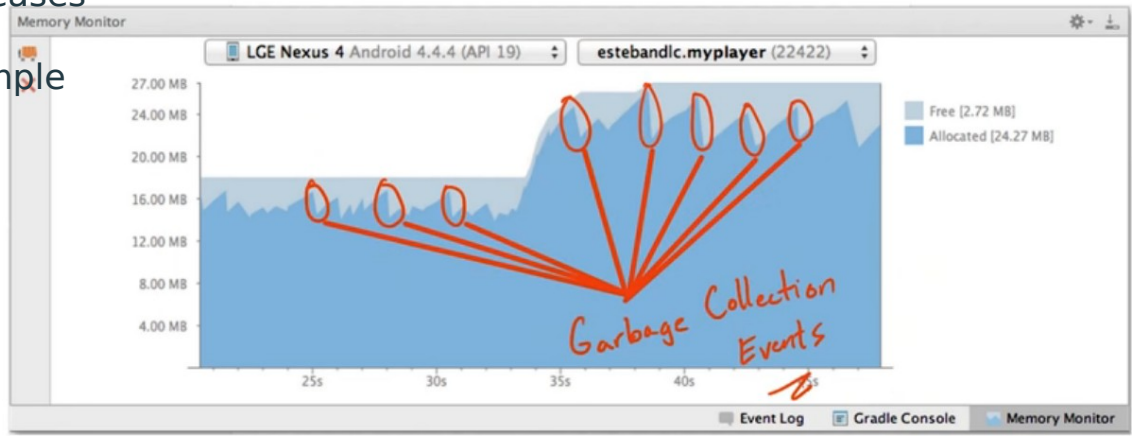
Memory unavailable



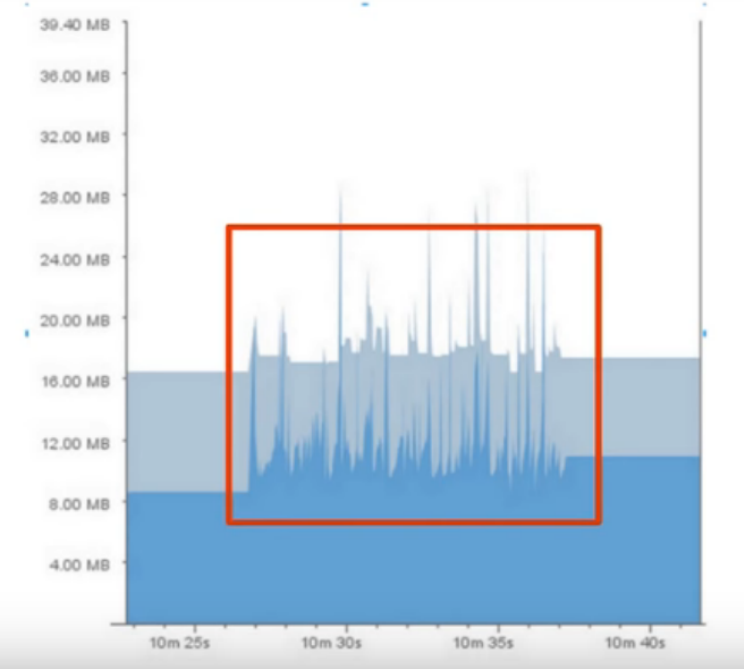
Memory monitor

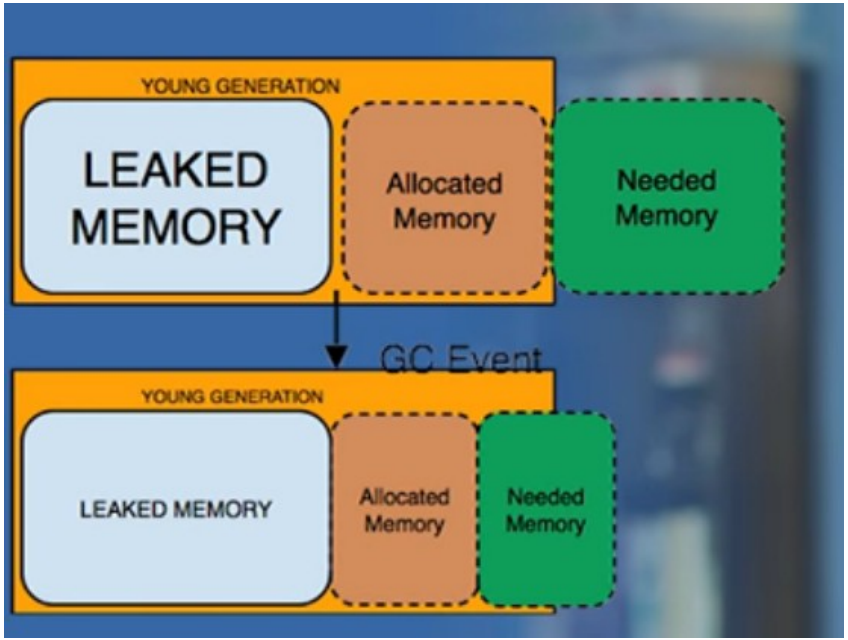
- Memory monitor view
- Allocated and free memory increases as per demand
- Healthy garbage collection example



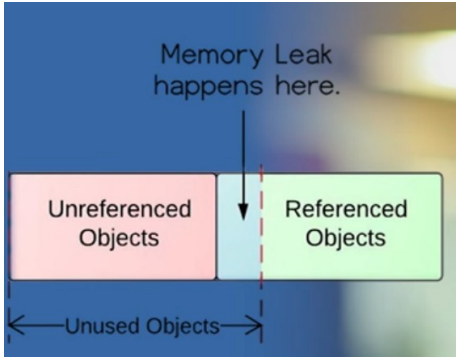


Bad memory allocation ex





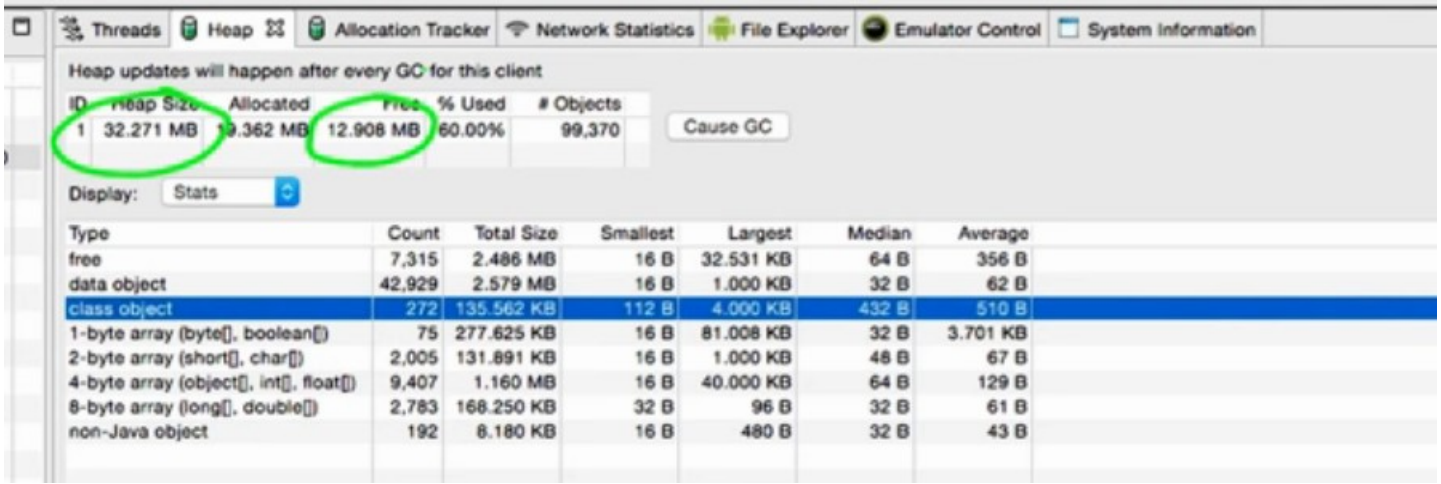
Memory leaks



- Memory leaks are objects which application is no longer using but GC fails to recognize them.
- Result is permanent in heap which never gets freed & take the valuable space.
- https://developer.android.com/studio/profile/memory-profiler.html
- Tip : **Track the object life & clean up the reference when no longer required.**
- If we allocate a high number of objects in a small period of time like inner for look or in onDraw() functions , We're asking for alot of sudden memory which eventually will trigger GC & process becomes slower. **Solution : Try to find these instances & keep them out of loop if they're going to be need many times.**

Memory profiler

Heapviewer ss



Avoid memory leaks tips

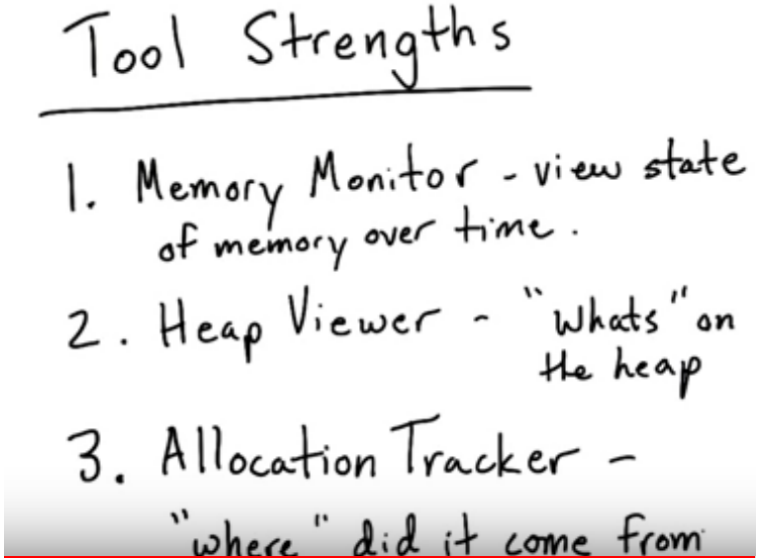
```
collector.setListener(this, mListener);
```

This problem is compounded when the activity is destroyed and a new one is created. In this example, when a new activity is created due to the device orientation changing, an associated Listener is created by the view, but when the activity is destroyed, that listener is never released. This means that no listeners can ever be reclaimed by Java's garbage collector, which creates a memory leak.

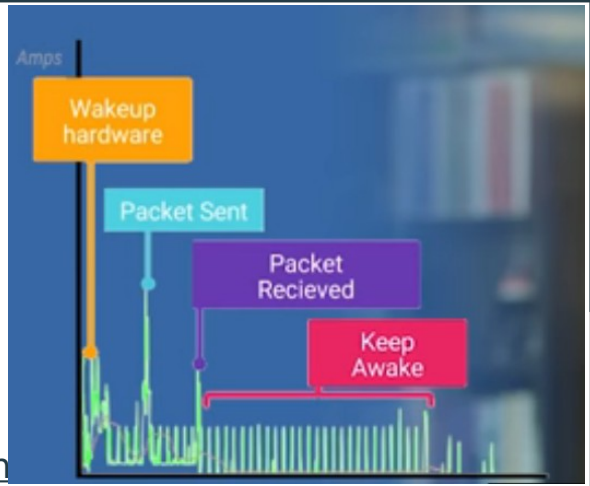
When the device is rotated and the current activity's onStop() method is invoked, make sure to clean up any unnecessary references to view listeners.

Memory tools

- Memory monitor
- Heap viewer
- Allocation tracker



Battery performance



Links

- https://github.com/google/battery-historian
- https://developer.android.com/studio/profile/battery-historian.h
- Download the zip file of battery historian , extract it & check whether device is attached
- **Adb kill-server** for killing the old data
- Command to reset battery data gathering from the terminal: **adb shell dumpsys batterystats --reset**

```
Terminal
TinnyTim-2:Desktop Udacity$ adb devices
List of devices attached
0784f0040074a065        device

TinnyTim-2:Desktop Udacity$ adb shell dumpsys batterystats > batterystats.txt
TinnyTim-2:Desktop Udacity$ adb shell dumpsys batterystats > com.example.android.sunshine.app > sunshine.txt
TinnyTim-2:Desktop Udacity$ python historian.py sunshine.txt > sunshine_battery_stats.html
TinnyTim-2:Desktop Udacity$
```

Battery analyze using Battery manager

Activity:

Analyze

Let's assume you've identified some power heavy functions in your app. Let's look at one quick win we can achieve by using Android's BatteryManager.

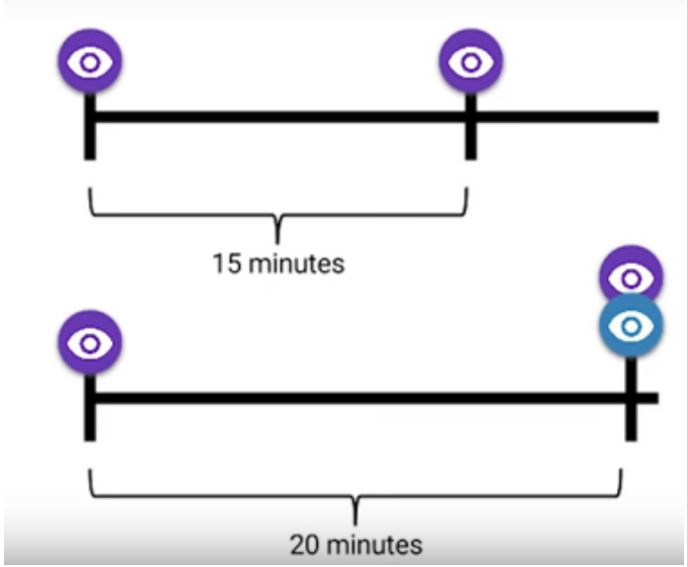
BatteryManager is a simple class that allows you to grab information about the battery state. This is helpful because it allows us to apply a quick optimization. If a non-essential task is started when the device is unplugged, you can opt to hold-off on this task and save some battery. Let's practice implementing this.

Battery manager:
https://classroom.udacity.com/courses/ud825/lessons/3758168642/concepts/37808385480923

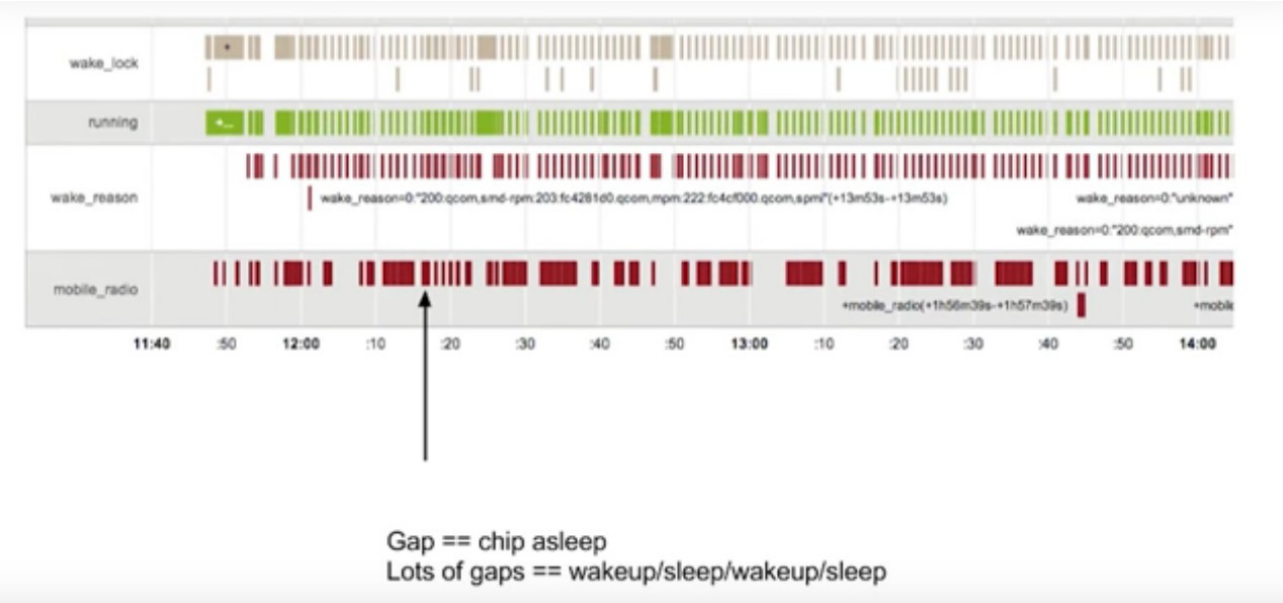
Battery life - Waylock & drain

https://www.youtube.com/watch?v=KzSKIpJepUw&feature=youtu.be

- Wake up phone **PowerManager.wake lock API -** This can go horribly wrong.What if app takes 60min instead of 10sec or keep awake forever.
- Set the inexact time limt
- Use **JobSchedur API** to make work better
- Acquire the wakelock->do work -> Remember to release it.



Network connection & Battery Management



- Smaller the gaps, more wakeup & it means more battery drain.
- https://developer.android.com/reference/android/app/job/JobScheduler.html
- https://developer.android.com/reference/android/app/job/JobInfo.Builder.html


QUESTIONS?