New EFL Debugging Probes

EFL Korea Community Seminar 2015

Seoul Conrad Hotel

2015-10-28

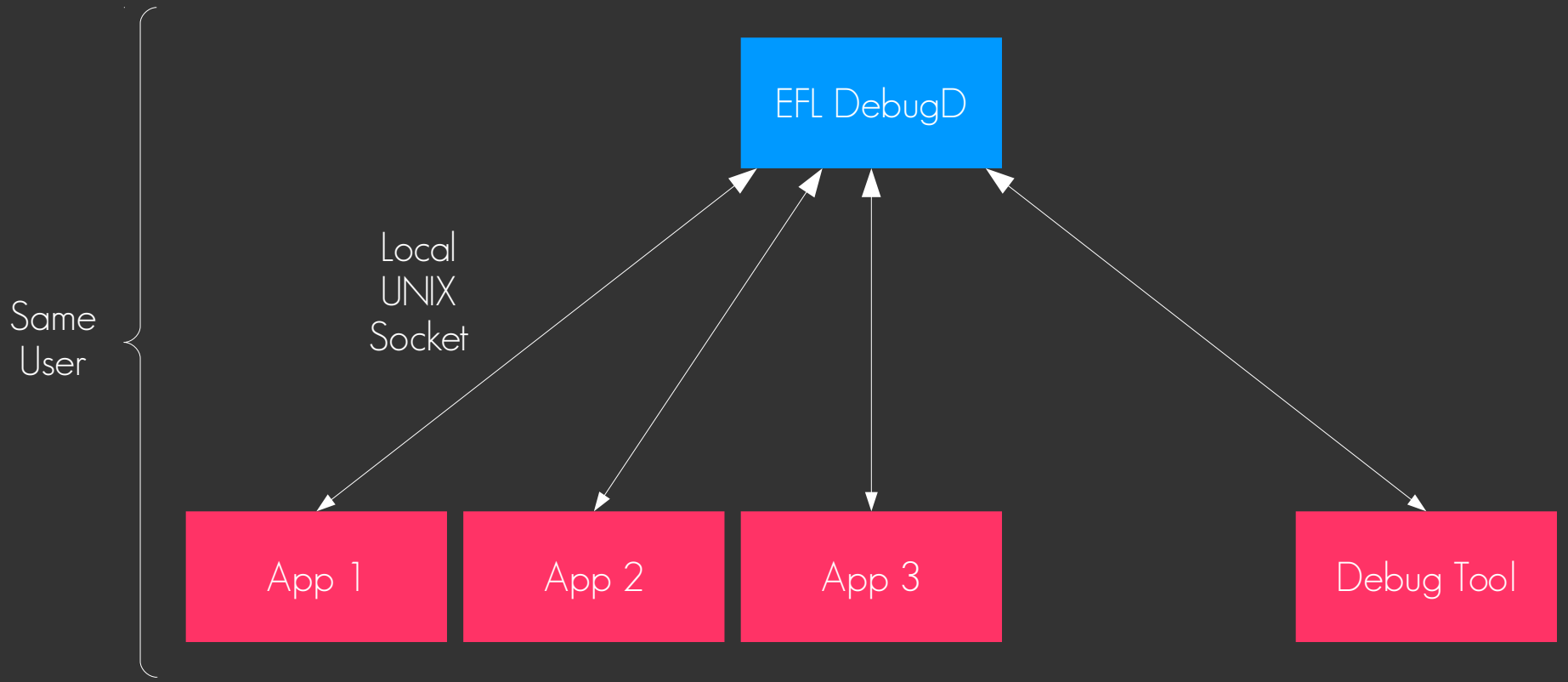
Carsten Haitzler <c.haitzler@samsung.com> <raster@rasterman.com>

Master Engineer Samsung Electronics, Korea
Enlightenment / EFL Founder

EFL DebugD

- New in EFL 1.16
- Not finalized and stable
  - Looking for input, usage, maturing of protocol
- User-level
  - No Root or kernel features needed
  - If run, ALL EFL apps connect to EFL DebugD by default
    - If connection fails, debugging not enabled
- Uses simple UNIX Sockets

EFL DebugD


EFL DebugD
Local
UNIX
Socket
Same
User
App 1
App 2
App 3
Debug Tool

Every EFL App Connects

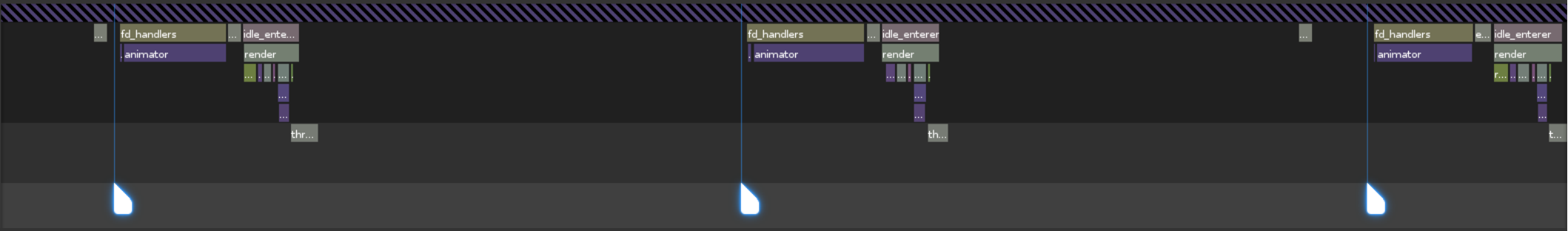
- **`eina_init()`** will:
  - Connect to EFL DebugD
    - If connect succeeds → app has debug controls
    - If connect fails → app has no debug controls
    - If connnection drops → app stops having debug controls
- **`efl_debug`** cmdline tool can:
  - **`list`** → list PIDs of debuggable clients
  - **`evlogon PID`** → start logging events from process PID
    - A file **`~/efl_debug_evlog-PID.log`** will be have all events appended to
  - **`evlogoff PID`** → stop logging events from process PID
    - Log file will be closed

Eina Evlog

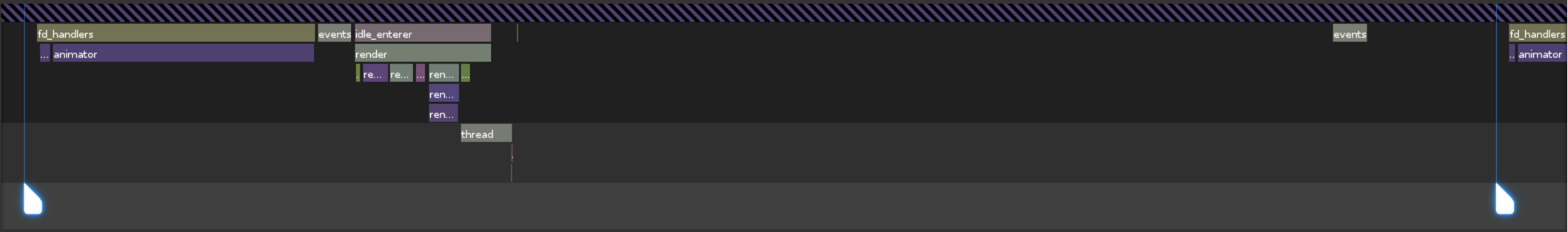
- Allows you to instrument events and timelines
  - Single events like “vsync event” or “interrupts”
  - States beginning and ending (begin animating, end animating)
  - Call ranges (begin doing x, end doing x)

```
eina_evlog("!EVENT", NULL, 0.0, NULL);
eina_evlog("+do_x", NULL, 0.0, NULL);
eina_evlog("-do_x", NULL, 0.0, NULL);
eina_evlog(">state_y", NULL, 0.0, NULL);
eina_evlog("<state_y", NULL, 0.0, NULL);
```

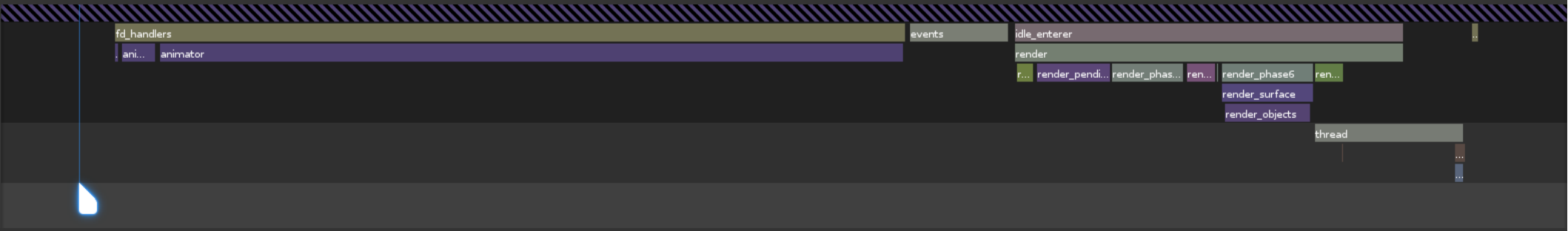
Evlog Visualized



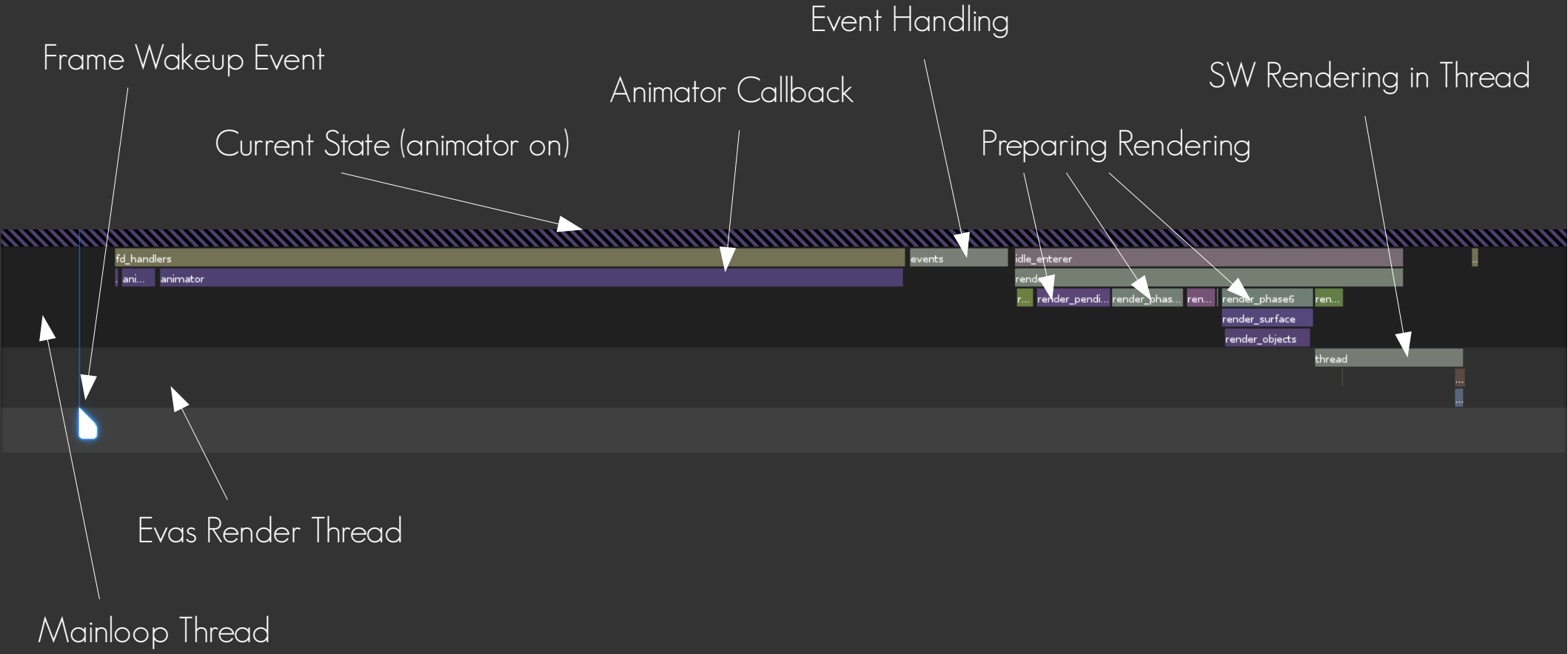
Evlog Visualized



Evlog Visualized



Evlog Visualized

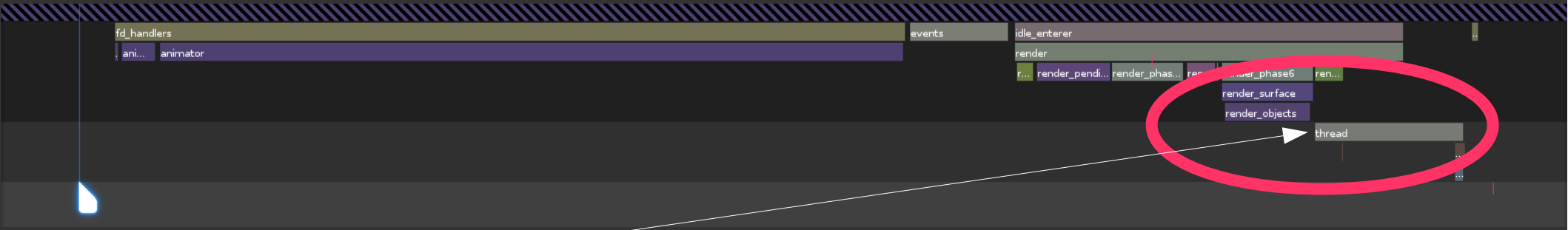


Surprise!

- We spend very little time rendering even in SW!
- Time spent manipulating objects in animator is huge!
- Rendering preparation is very heavy vs. rendering!

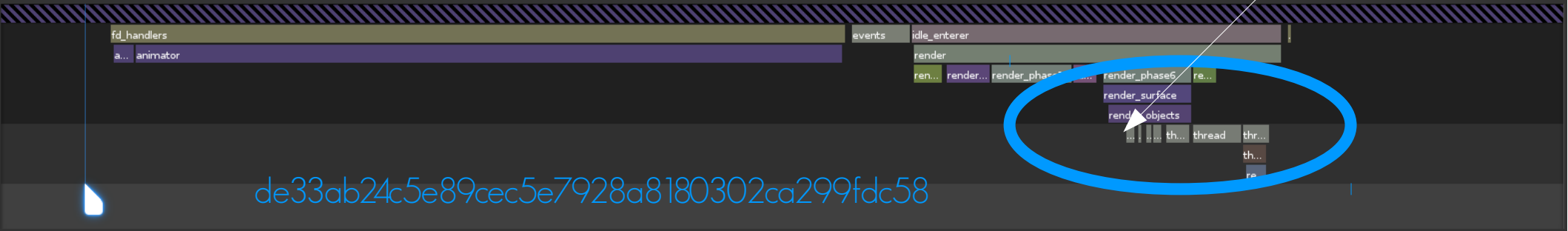
- Eina DebugD + Evlog is very useful
- Evlog can be used by apps too to trace their own code
- We need to add more and more Evlog probes in EFL to get more info
  - Help us do this if you write or debug any code in EFL!

A Fix Evlog found



Rendering doesn't start until all objects have been walked and commands generated

Rendering now begins WHILE walking objects



de33ab24c5e89cec5e7928a8180302ca299fdc58

Evlog Improvements to do

- Expand EFL DebugD client protocol to get Evlog streams directly
- Have Evlog Util talk directly to EFL DebugD
- Have Evlog Util allow selection otarget & turn on/off on the fly
- Add more Evlog probes everywhere
- Add more info like CPU usage per thread logs (and which CORE, frequency of the core ...)

https://git.enlightenment.org/devs/raster/evlog.git

EFL DebugD Improvements

- Protocol will expand/be improved
- Become extensible
  - Clouseau
  - Allow inspection of all eo objects at runtime
  - Improve existign backtrace infra
    - Can get a bt from every thread right now in tight polling
    - Not reported via EFL DebugD at the moment
- Find ways to make remote systems debuggable
  - Connect to a remote EFL DebugD

EFL DebugD Improvements

- Make far more complete Debugging UI and debug tools
  - Get more low level debug info like complete memory usage
    - Use malloc hooks, mmap wrappers etc.
  - Get rendering cost info (GL or SW etc.)
  - Get higher level memory usage
    - Images, fonts, etc. etc.
  - Get higher level data content
    - Images, fonts, etc. etc.

Current work underway

- Move Clouseau to DebugD
  - Add Clouseau support in DebugD protocol
    - Means making protocol extensible

Demo

QnA