EsbRootView on Apple/Metal

ESSnuSB WP5 video meeting
28 September 2020

2018 :-(

WWDC June 2018 : Apple, in a // session,
announced that their Apple/OpenGL is deprecated.

2018 :-(

- Bad news for people looking for a standard to do visualisation.
- Bad new for me, then EsbRootView, Geant4, and a lot of scientific software wanting to exploit macOS (and iOS).
- Due to the impact of Apple concerning interactivity, we can't ignore that…
- Apple promotes their proprietary Metal in remplacement of OpenGL on their devices. We have to look!

- (No date given about a strong removal of Apple/OpenGL on macOS and iOS)

inlib/exlib/sg scene graph logic

- In spirit, same logic as the great OpenInventor.
- A scene is described by a graph of "nodes" in which, for example, a rotated red cube is described by a matrix (rotation) node, a colour node and then a shape node.
- A graph is rendered on screen (or offscreen!) by using an implementation of a "renderer" for a given technology, for example OpenGL.
- See softinex at http://gbarrand.github.io

inlib/exlib/sg renderers

- GL-ES : it permits (today) with SAME CODE to visualise on Linux, macOS, Windows, iOS, Android.
- offscreen : to produce a .png, .jpeg, .ps, .pdf file without having to be tied to any graphics system (it is pure C++ code based on the std/stl libs). (Used in G4/g4tools offscreen plotting).
- wasm : a web assembly version (using WebGL). It permits to display in most web browser.
- Then I have to provide a renderer for Apple/Metal…

Not so easy to do !

- API is in Objective-C or in Swift (the “better than Python” Apple language, dixit… Apple).
- Apple examples are in Swift buildable from Xcode.
- Nothing in C++ buildable from a “simple make”.
- Stucked…

…up to the end of June 2020

- Some (despaired) googling gave a hit on GitHub : naleksiev/mtlpp
- mtlpp : a C++ wrapper around Metal
- With an example to draw a triangle buildable with make: bingo!

- (As says a famous quote : “give me a triangle and I visualise the world”).

Summer 2020 at the forge…

- After two months of very painful coding, I have now EsbRootView that works on macOS.
- And this by using straight the Objective-C Metal API from C++ (Apple clang permits to mix both languages).
- No extra libs involved.
- (It follows my “software least action principle”).
- Painful because the logic of Metal is not similar than GL-ES (even if ideas of rendering pipline, buffers, etc… are the same). We have to rethink a new renderer (which was not the case for offscreen and wasm ones).

Summer 2020 at the forge… (2)

- I have correct 3D rendering for basic primitives (points, lines, segments, triangles, triangle-fan and strip).
- I have lighting.
- I have texture mapping.
- With that I can my apps working on Metal.

And be sure it had not be easy to get !

Then "relief" !

- An iOS version has to be done (I am on it).
- In principle I am now ready for what Apple prepares for the future.
- (I strongly suspect that they are going to remove their OpenGL when the macOS major release, running on their own Ax processors, is going to come).

Conclusions

- I am very happy of this.
- (My R&D apps g4exa and g4view works also now with Metal. Then no problem with Geant4 libs).
- I am going to prepare a special “Metal” EsbRootView/3.1 release… (As for mtlpp, it may help others, in particular the G4/vis group)