Apple/Metal at the Orsay forge

Geant4 collaboration meeting Sep 2020

G.Barrand, CNRS/IN2P3/IJCLAB

2018 :-(

- WWDC June 2018 : Apple, in a // session, announced that the Apple/ OpenGL is deprecated.
- Bad news for people looking for a standard to do visualisation.
- Bad new for me and Geant4, and a lot of scientific software.
- 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 scene graph logic as the great OpenInventor.
- 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
- g4tools/plotting done with that by using some offscreen renderers.
- GL-ES renderer. It permits (today) with SAME CODE to visualise on Linux, macOS, Windows, iOS, Android.
- Then I have to provide a renderer for Apple/Metal…

Not so easy to do !

- API is in Objective-C or in Swift.
- 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 googling gave a hit on GitHub : naleksiev/mtlpp which is a C++ wrapper around Metal with an example to draw a triangle buildable with make : bingo!

Summer 2020 at the forge…

- After two months of very painful coding, I have now one C++ app (a display for ESSnu) that works on macOS by using Cocoa and Metal.
- And this by using straight the Objective-C Metal API from C++ (Apple clang permits to mix both languages).
- 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 or WebGL 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 have my apps working on Metal.

And be sure it had not be easy to get !

Can it help for Geant4 ?

- My R&D apps g4exa, g4view should run with Metal and I am going to release versions of these.
- But it is not based on the “G4 vis system” largely used now.
- The G4 vis system is in principle designed to handle multiple heterogenous graphics systems (= drivers).
- For example there is an OpenInventor driver and some offscreen ones (HepRep, VRML).

Can it help for Geant4 ? (2)

- Right now what is promoted by Geant4 is Qt for the GUI and OpenGL for the graphics.
- Qt comes now with Qt3D to do 3D rendering. Qt people are going probably to provide an Apple/Metal version of it. J.Allison made progresses around a G4 Qt3D vis driver.
- Anyway, I would strongly suggest to G4 to not put all its eggs in the same Qt-basket and still maintain an "academic way" to do GUI and graphics. If so, the effort done at Orsay may help in handling Metal for such G4 vis driver.
- (A g4tools (= part of inlib/exlib) G4 vis driver ? Not yet mature, but the idea makes its way…)

The X11 way on Macs

- Anyway now X11, OpenMotif, an X11 server, OpenGL/ Mesa, are available in a consistent way (for exemple through MacPorts) on macOS, then we can always run on Macs this way without any GUI and graphics Apple software.
- (Avoid to mix with XQuartz libs !).
- (Would Qt/X11 with Mesa/GL be running on these ?)

- (In fact the same is true on Windows by using CYGWIN).