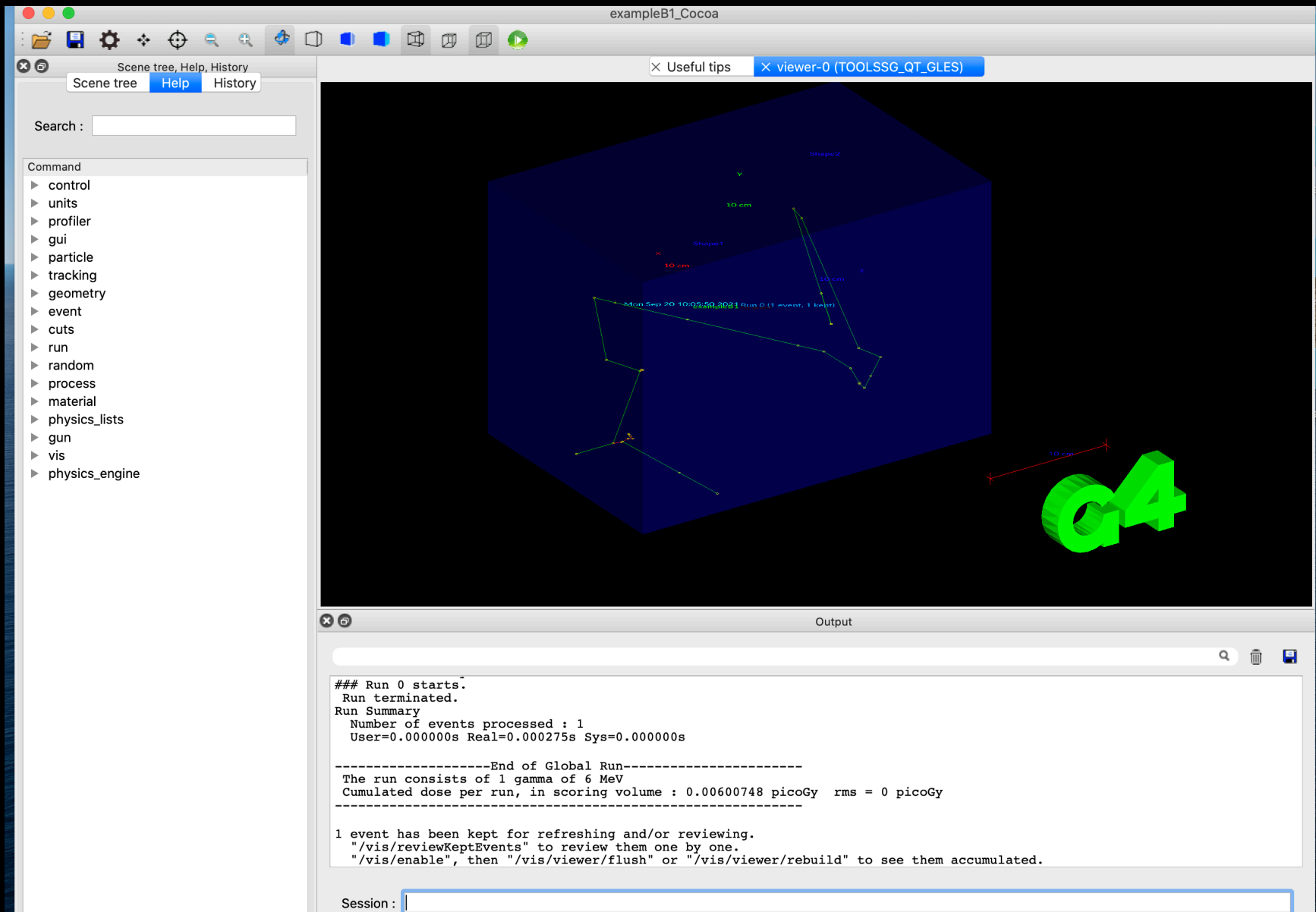


g4tools @ vG4workshop-2021

G4/vis/ToolsSG

- Have the G4/vis “ToolsSG” vis drivers based on the g4tools/sg “scene graph” logic (similar to the “So” one of Inventor).
- Rendering of a scene graph is abstracted. Have concrete renderer for OpenGL, but also for “offscreen” by using gl2ps.
- Good part of the code is already here in g4tools because of offscreen plotting.
- In G4-11, have code in g4tools to tie to “windowing systems” as straight X11, Windows but also for OpenMotif and Qt.
- Arrange to have in G4/vis/ToolsSG itself, only the code needed to tie to the G4/vis logic.
- Still push, as much as possible, an “academic way” in G4/vis.

ToolsSG/Qt/OpenGL driver on B1



The screenshot displays the ToolsSG/Qt/OpenGL driver interface. The main window, titled "exampleB1_Cocoa", shows a 3D visualization of a particle track within a blue rectangular volume. The track is represented by a series of green points connected by lines, with labels "Step01" and "Step02" indicating different stages of the track. A red "X" marks a point on the track. A 3D "G4" logo is visible in the bottom right corner of the main window, with a red double-headed arrow indicating a distance of "10 cm".

The interface includes a "Scene tree, Help, History" panel on the left with a search field and a "Command" list:

- control
- units
- profiler
- gui
- particle
- tracking
- geometry
- event
- cuts
- run
- random
- process
- material
- physics_lists
- gun
- vis
- physics_engine

The "Output" panel at the bottom displays the following text:

```
### Run 0 starts.  
Run terminated.  
Run Summary  
Number of events processed : 1  
User=0.000000s Real=0.000275s Sys=0.000000s  
  
-----End of Global Run-----  
The run consists of 1 gamma of 6 MeV  
Cumulated dose per run, in scoring volume : 0.00600748 picoGy rms = 0 picoGy  
  
1 event has been kept for refreshing and/or reviewing.  
"/vis/reviewKeptEvents" to review them one by one.  
"/vis/enable", then "/vis/viewer/flush" or "/vis/viewer/rebuild" to see them accumulated.
```

A "Session:" input field is located at the bottom of the output panel.

g4tools in G4-11



- Have a MR to prune g4tools to have in it only what is really used in G4/analysis and now G4/vis/ToolsSG. Based on the #63 issue of Ben to ease the maintenance and integration of g4tools, see: <https://gitlab.cern.ch/geant4/geant4-dev/-/issues/63>
- (A lot of files (around 100) here for g4tools/tests, but these can stay in gbarrand/github/g4tools).
- ? Have a MR to have “toolx” namespaced code for code related to “externals” (now needed for X11, Windows, GL, png, jpeg). It should ease Ben-#63.

G4/vis/plotting in G4-11?

- All is here to have interactive plotting.
- (We have already tools/sg offscreen plotting in G4/analysis).
- What is the right way to do it in G4? Through G4UI or G4/vis?
- I had (since 2018) a “G4UI way”, but it is quite not satisfactory.
- A G4/vis way would be better since the logic of attaching a windowing and a rendering system, creating viewers and scenes is already here.
- After all a plot is nothing more than the visualisation of an histogram. And then some “G4VSceneHandler::AddPrimitive(<histo>)” should be able to do it!
- Relationship with G4/analysis?
- Have to be discussed with John and Ivana...

As a reminder...

- g4tools is an automatic extraction of some code found in the softinex/inlib,exlib and namespaced “g4tools” for an embedding in Geant4.
- Pure header code. Highly portable (including iOS, Android and now WebAssembly). Easily embeddable (no “config.h” or specific build tool in the way).
- Strongly OO. No implicit management. Layered.
- Thread safe (no writable statics).
- See <http://gbarrand.github.io>