

New vis (and some GUI) features 2023

Scheduled for Geant4 11.2

These are summary slides for the plenary session
Please come to the vis parallel session this afternoon
for a full report and discussion

Contents

- G4debug—a new output stream
- A new scene tree—available to *any* viewer when using the Qt GUI
- /vis/open (without parameters)
 - Choose driver at run time
 - /vis/open (with parameters), e.g., /vis/open OGL, works as before
- New off-screen drivers with ToolsSG (TSG) and Vtk
 - Output to file—choice of formats
 - Any size—choose large size to get high resolution
- News

G4debug example using Qt GUI

In `B1::SteppingAction::UserSteppingAction` after testing for the scoring volume:

```
G4debug << "Deposited in scorer: "
```

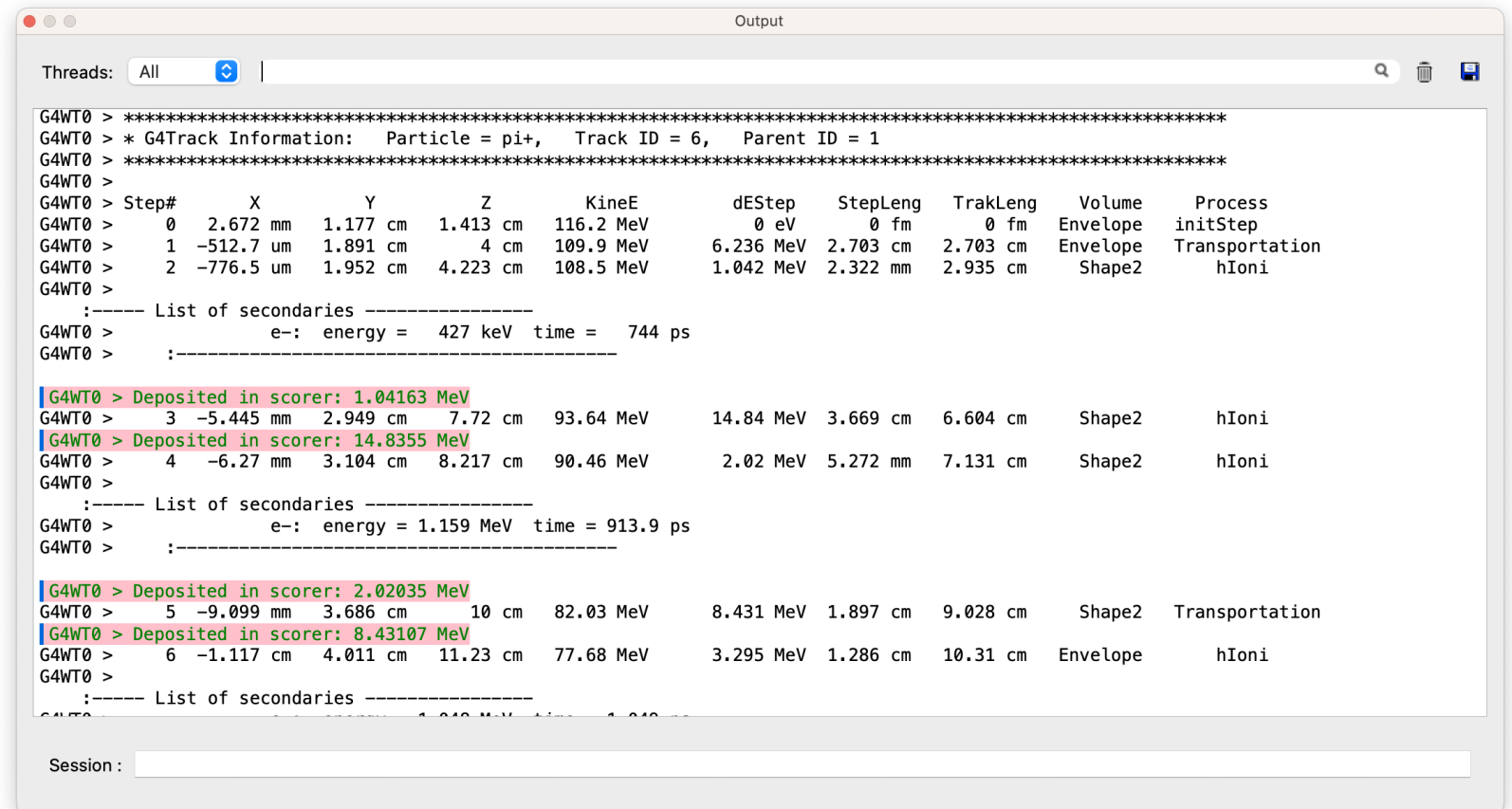
```
<< step->GetTotalEnergyDeposit()/CLHEP::MeV << " MeV" << G4endl;
```

To show occasional debug line in a mass of other output, run with:

```
/tracking/verbose 2  
/gun/particle proton  
/gun/energy 1 GeV  
/run/beamOn
```

The Qt GUI intercepts this Output stream and highlights it, making it easy to pick out the debug line.

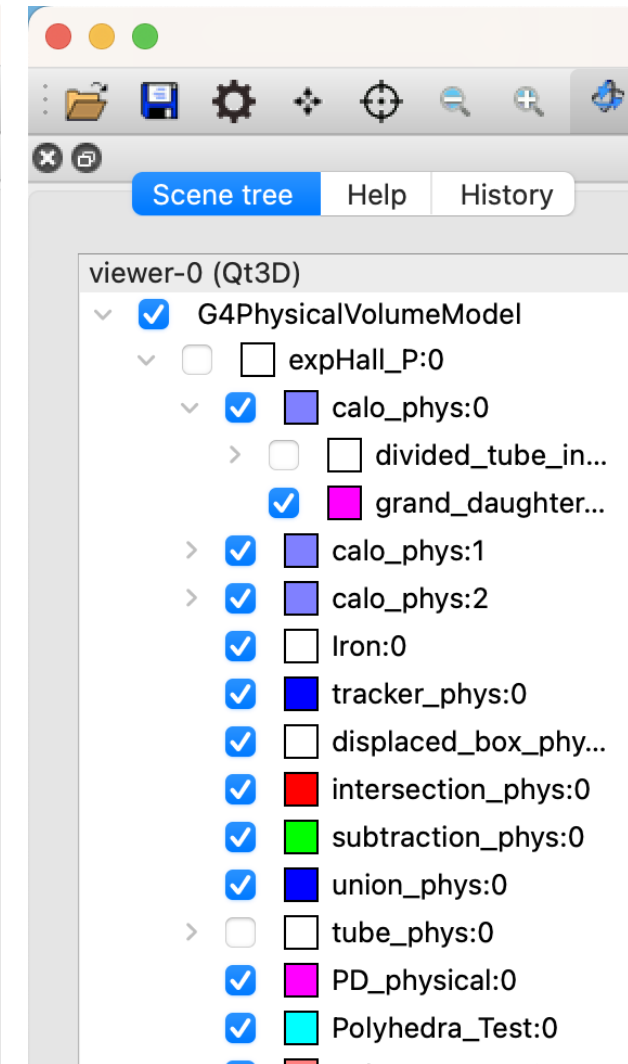
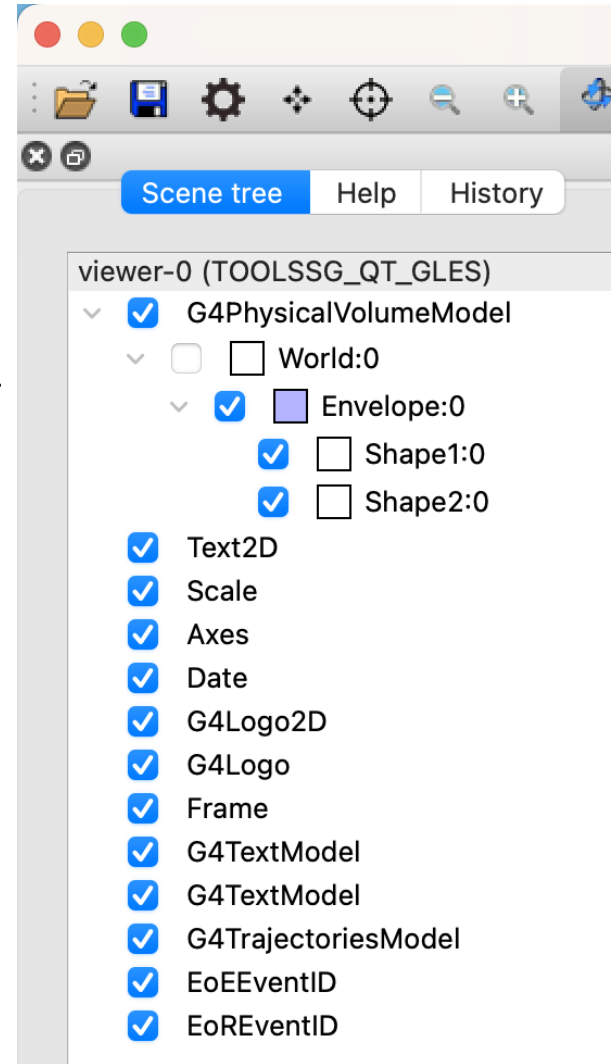
(Other sessions, e.g., tcsh, simply direct to `std::cout`.)



```
Output  
Threads: All  
G4WT0 > *****  
G4WT0 > * G4Track Information: Particle = pi+, Track ID = 6, Parent ID = 1  
G4WT0 > *****  
G4WT0 >  
G4WT0 > Step#      X          Y          Z          KineE      dEStep    StepLeng  TrakLeng  Volume  Process  
G4WT0 > 0    2.672 mm  1.177 cm  1.413 cm  116.2 MeV   0 eV      0 fm      0 fm      Envelope  initStep  
G4WT0 > 1   -512.7 um  1.891 cm   4 cm    109.9 MeV  6.236 MeV 2.703 cm  2.703 cm  Envelope  Transportation  
G4WT0 > 2   -776.5 um  1.952 cm  4.223 cm  108.5 MeV  1.042 MeV 2.322 mm  2.935 cm  Shape2    hIoni  
G4WT0 >  
:----- List of secondaries -----  
G4WT0 >          e-: energy = 427 keV time = 744 ps  
G4WT0 > :-----  
G4WT0 > Deposited in scorer: 1.04163 MeV  
G4WT0 > 3   -5.445 mm  2.949 cm  7.72 cm  93.64 MeV  14.84 MeV 3.669 cm  6.604 cm  Shape2    hIoni  
G4WT0 > Deposited in scorer: 14.8355 MeV  
G4WT0 > 4   -6.27 mm  3.104 cm  8.217 cm  90.46 MeV  2.02 MeV 5.272 mm  7.131 cm  Shape2    hIoni  
G4WT0 >  
:----- List of secondaries -----  
G4WT0 >          e-: energy = 1.159 MeV time = 913.9 ps  
G4WT0 > :-----  
G4WT0 > Deposited in scorer: 2.02035 MeV  
G4WT0 > 5   -9.099 mm  3.686 cm  10 cm    82.03 MeV  8.431 MeV 1.897 cm  9.028 cm  Shape2    Transportation  
G4WT0 > Deposited in scorer: 8.43107 MeV  
G4WT0 > 6   -1.117 cm  4.011 cm  11.23 cm  77.68 MeV  3.295 MeV 1.286 cm  10.31 cm  Envelope  hIoni  
G4WT0 >  
:----- List of secondaries -----  
G4WT0 >  
Session :
```

A new scene tree

- Available to *any* viewer when using the Qt GUI
 - Including non-Qt viewers, off-screen and file-writing viewers
 - A unique scene tree is held by each viewer and made available to the Qt GUI
 - The Qt GUI renders it to an interactive tree and notifies the interactions to the viewer
- Features
 - Hover to get dump of touchable
 - Click on blue check box to make invisible/visible
 - Click on chevron to hide/expose list of daughters in the scene tree
 - Double click on small square box to change colour
 - Right-click to get menu of actions



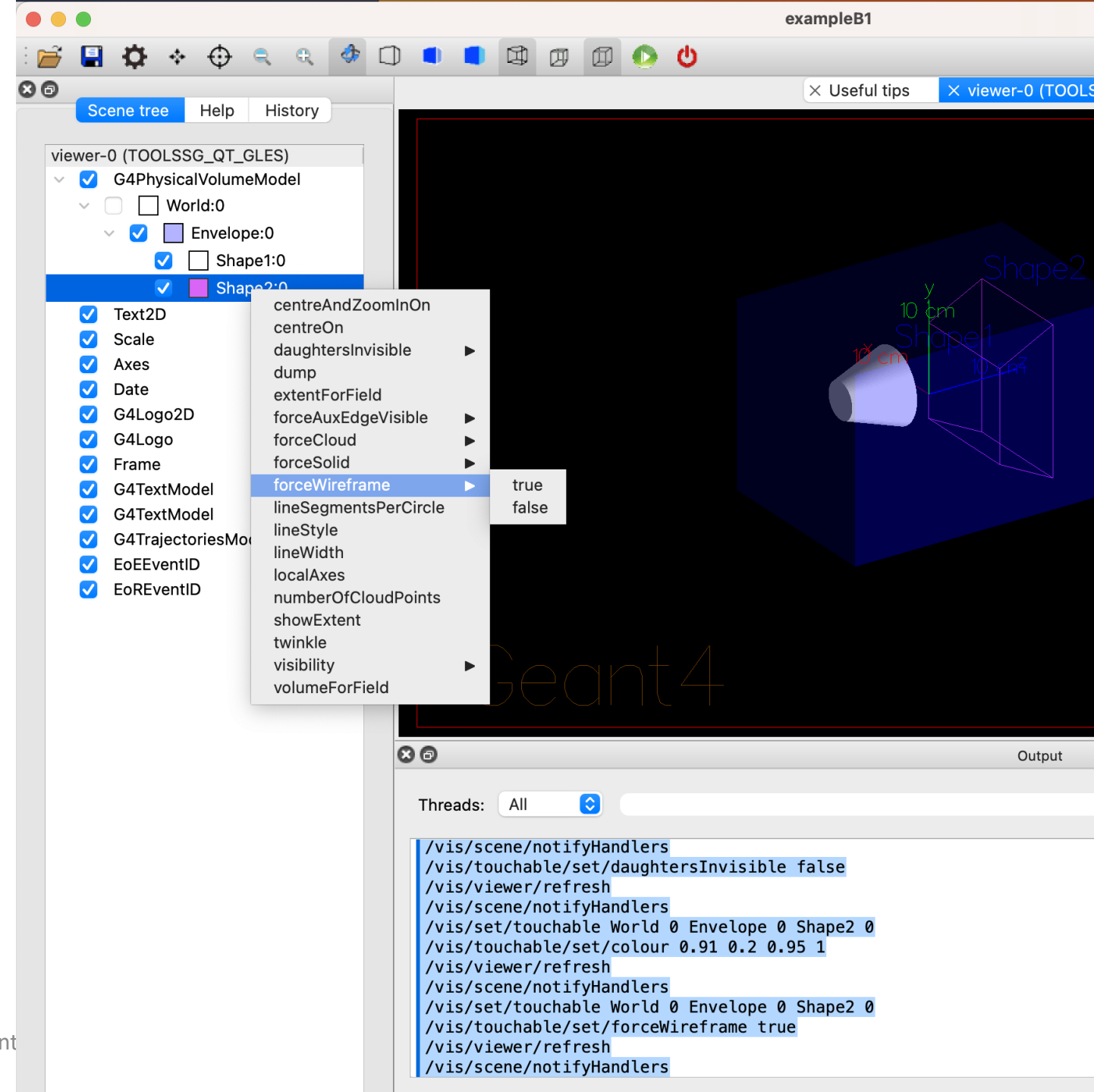
New scene tree (contd)

Double click on small square box to change colour

The screenshot displays the Geant4 visualization environment. On the left, the 'Scene tree' panel shows a hierarchy of objects under 'viewer-0 (TOOLSSG_QT_GLES)'. The 'G4PhysicalVolumeModel' is expanded to show 'World:0', 'Envelope:0', 'Shape1:0', and 'Shape2:0'. Below this, a list of visible objects includes Text2D, Scale, Axes, Date, G4Logo2D, G4Logo, Frame, G4TextModel, G4TrajectoriesModel, and two EoEventID objects. A 'Colors' dialog box is open in the center, showing a 'Spectrum' color picker with a rainbow gradient and an 'Opacity' slider set to 100%. The background shows a 3D visualization of a detector component with a blue cone and a purple rectangular block, labeled 'Shape1' and 'Shape2' with dimensions of 10 cm. The 'Geant4' logo is visible at the bottom of the viewer window. At the bottom right, the 'Output' panel shows a list of commands executed in the scene, such as '/vis/set/touchable World 0 Envelope 0 Shape2 0' and '/vis/touchable/set/visibility true'.

New scene tree (contd)

Right-click to get menu of actions



/vis/open (without parameters)

- This will be the recommended way of opening a viewer
 - Examples B1 and B2 have been changed
 - We plan to change all tests and examples for 11.2
 - A new constructor in all main programs:
`G4VisExecutive(argc, argv)`
 - All `/vis/open OGL` commands stripped of parameters in all `vis.mac` files
- Existing behaviour is preserved
 - The default default is OGL
 - You can still use the old constructor: `G4VisExecutive()`
 - You can still use `/vis/open OGL`
- The default can be changed at run time (without having to edit `vis.mac`!)
 - By programmed argument
 - By environment variable
 - By entry in a file in your home directory, `~/.g4session`
- Otherwise according to batch/interactive and build flags

New or improved and retired vis drivers

- New in Geant4 11.0 and further developed for 11.1 and 11.2
 - Qt3D (John Allison): limited functionality but nice
 - ToolsSG (TSG) (Guy Barrand): working nicely
 - Most features of the OpenGL drivers
 - Also supports plotting
 - Full-screen driver, TOOLSSG_OFFSCREEN—always built, default in batch mode
 - Open Inventor Qt (OIQt) (Fred Jones): Also very nice, requires users
 - Includes “bookmarking” and “navigation”
 - Vtk (Stewart Boogert, Laurie Nevay): Improved multi-featured version on the way
 - Interactive cutting and clipping
 - Export to GLTF (modern 3D object transfer protocol), interface to other packages
 - Export to web – scene can be rendered and manipulated in a webpage, “fantastic for manuals and documentation”
 - Off-screen rendering
- Retired (removed) in Geant4 11.1
 - HepRep/Wired (HepRepFile/HepRApp is retained)
 - VRML1 (VRML2 is retained)
 - The “network” drivers (those that communicate with their browser via BSD sockets”
 - VRML2 (VRMK2FILE is retained)
 - DAWN (DAWNFILE is retained)

vis/ToolsSG recap

- A vis driver introduced in 2021 based on a **scene graph logic** developed at LAL (now IJCLab) since 2010, and for which the code is now in **g4tools (under tools/sg)**.
- Good part of the code is on C++ standard libs.
- Some rather light code in **toolx** to bind to various renderer and windowing systems, today: OpenGL, X11, Xt, Windows, Qt (Qt5 and Qt6).
- Few code in visualization/ToolsSG: only the “glue” to the general/generic G4/vis system.
- It permits some **plotting** (see G4-Rennes slides).
- **Try to keep some free “academic way” to do visualisation.**

TSG_<windowing>_ZB

- In fact we can use also the g4tools/z-buffer to render in a screen window! All windowing systems permit to create a window/widget for “pixmap” rendering.
- Since done 100% on CPU, would it be fast enough for interactive visualisation?
- Experience show that on recent laptops it is workable! (in particular the Apple “M” ones). Then in G4/11.2 we propose the drivers: *TSG_[QT,X11,XT,WINDOWS]_ZB*
- Since not based on any external graphics package, they could be built by default, as soon as a windowing system is chosen.

Why VTK?

- Best in class scientific rendering
- KitWare world leading opensource scientific computing company (cf. Cmake)
- Trusted in medical applications, high end engineering, non-HEP scientific visualisation
- Large user community
- Stable and **documented** API in C++
- 1000s of C++ examples to base visualization code on
- Started in December 1993 (pre-dates Geant4)
- **Works exceptionally well for Geant4!**

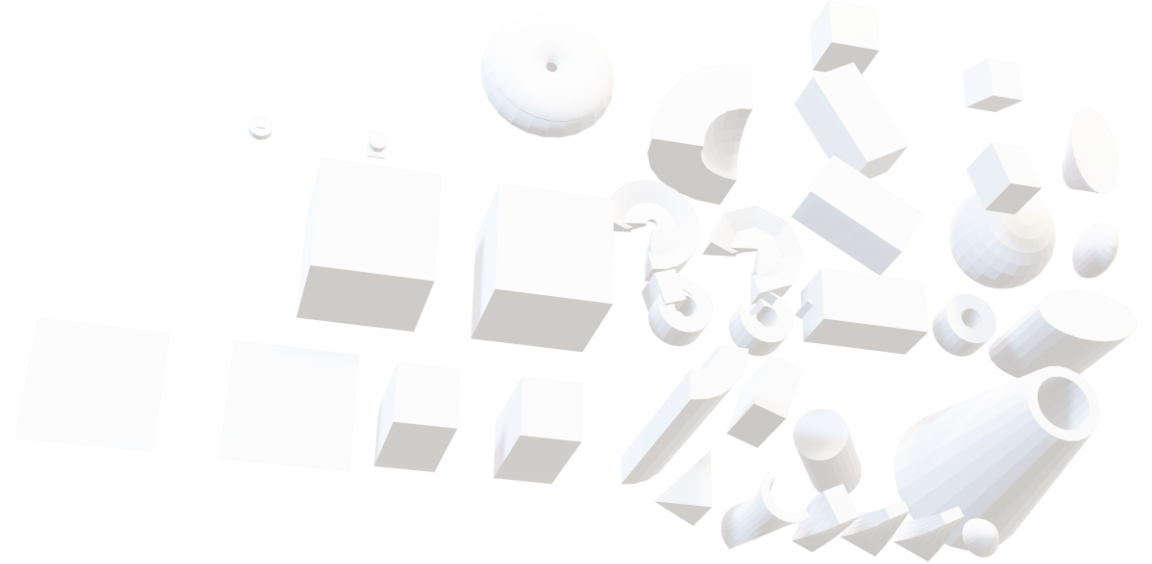
Development supported by



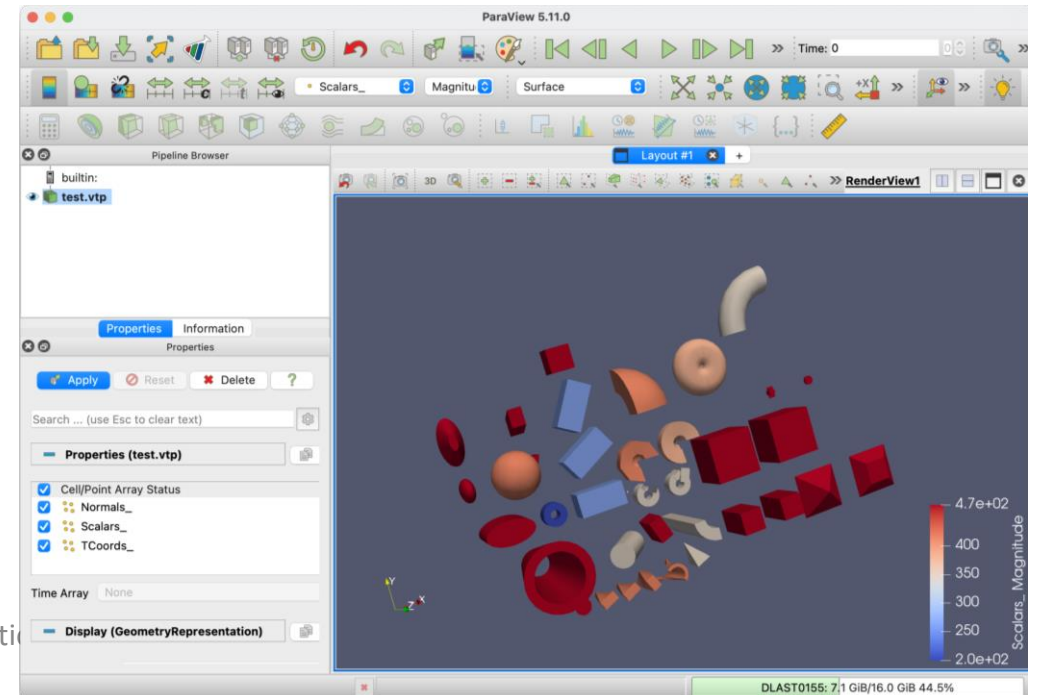
Vtk exports

- Export to almost all possible 3D file formats
 - VTU/VTP (vtk.js web model)
 - OBJ (convert to GLTF/USD web mode)
 - VRML
 - PLY
- Export screen grabs
 - PS
 - JPEG
 - TIFF
 - PNG
 - BMP
- Offscreen rendering and 3D conversion etc

Vtk → OBJ/GLTF → **Powerpoint**

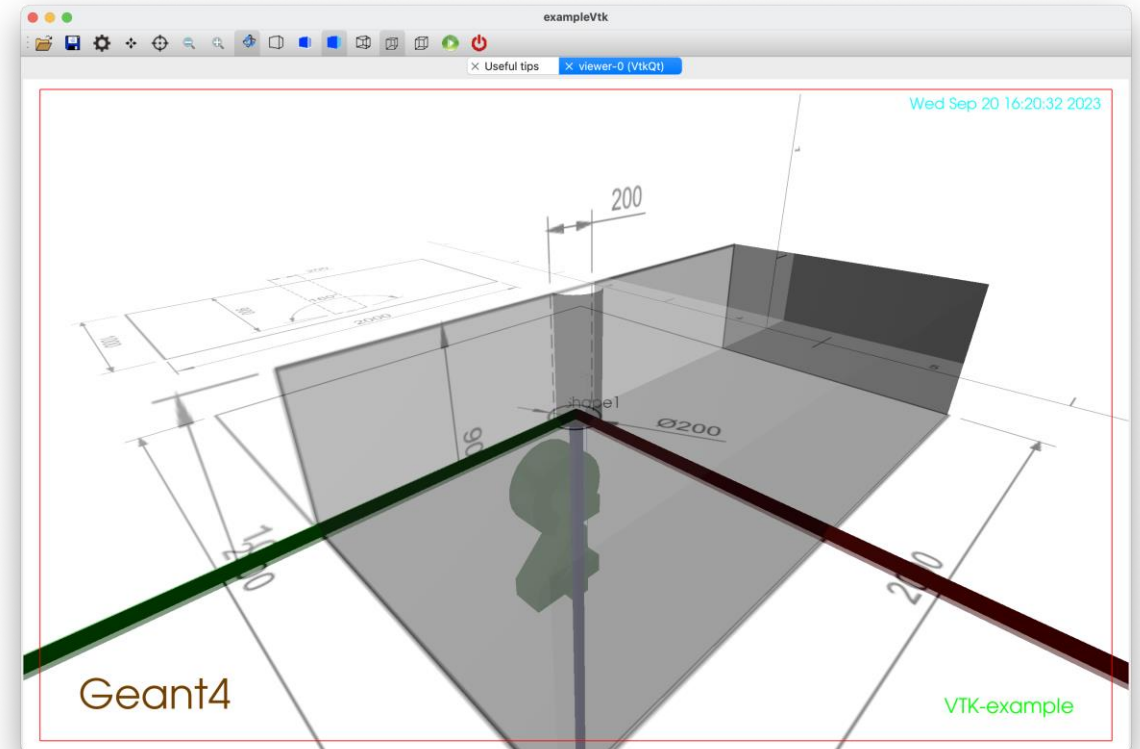


Vtk → VTP/VTP → **Paraview**



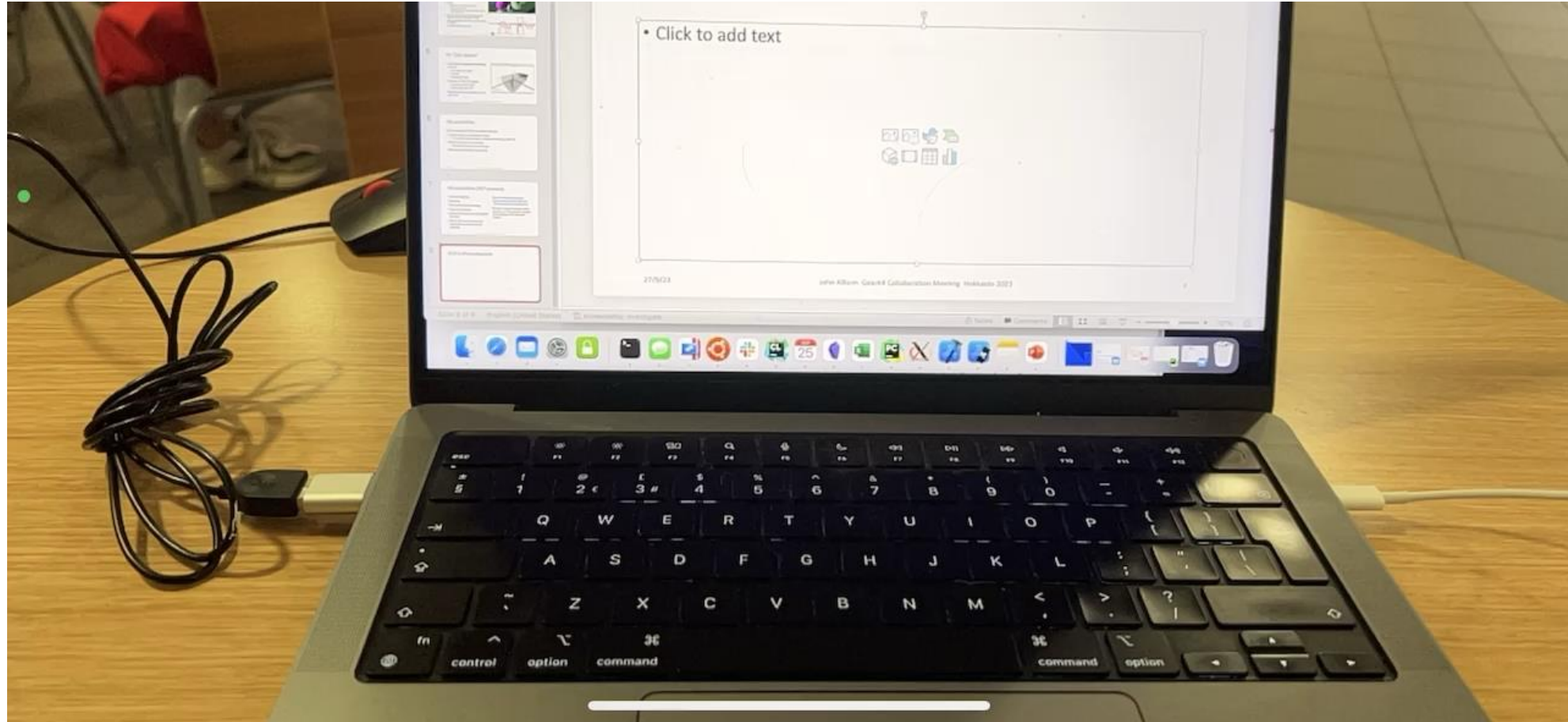
Vtk “side-loaders”

- Load data to augment modelling process
 - 2D image or 3D data
 - DICOM
 - Multiphysics data
- Example is CAD 2D images
 - exported as PDF->PNG
 - Side loaded in to VTK
- **Need to put into generic vis not just VTK**



GLTF in iPhone RealityKit

Vtk → OBJ/GLTF → iPhone App (RealityKit)



Vtk possibilities (HEP outreach)

- Complex lighting
- Shadows
- Physically based rendering
- Ray tracing Ospray
- OpenXR simple to enable (VR/AR for free)
- Vtk for iOS and Android and Javascript so mobile version possible.
- **Each of these (non-critical improvements) each will take ~40 hours work to implement**
- Boogert happy to support keen post-doc or PhD student wanted to develop an HEP outreach project

For more details, come to the vis parallel session at 1400.

G4 vis is an object-oriented multi-driver system. The core vis system converts the G4 scene (geometry, trajectories, etc.) into G4 graphics primitives. A driver renders the primitives and brings additional features, such as clipping, volume rendering, file export, etc.

You can choose your driver, even have different drivers in the same interactive session, and multiple views of the G4 scene.

Graphics is an ever-changing scene. It's hard to keep up. We want to exploit modern features to keep Geant4 looking smart and performant. In particular we are looking for people to take on the work of continuing development of the Vtk driver (under Stewart's supervision). We could do with someone with experience of Qt C++.

Come and join—even if part time—this fun and challenging working group.

Thankyou