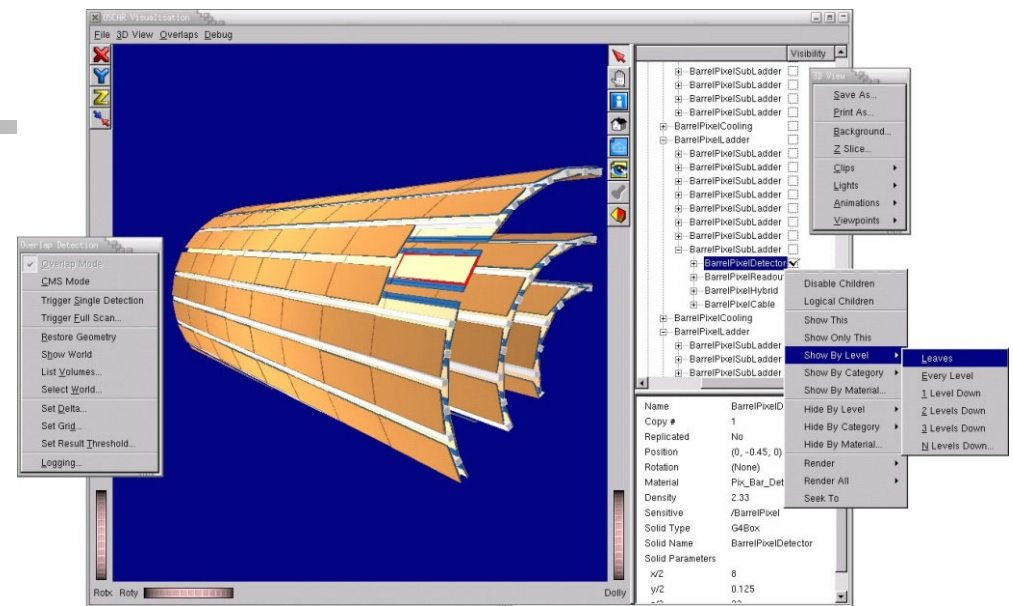


# Geant4 (G)UI

Luciano Pandola  
INFN – Laboratori  
Nazionali del Sud



A lot of material by J. Pipek

XI International Geant4 School  
Pavia, January 14<sup>th</sup>- 19<sup>th</sup>, 2024



# Three ways of steering the simulation

---

## 1) **hard-coded** application

- no user interaction
- everything specified in the C++ source
- re-compile needed to apply changes

## 2) **batch** session

- commands in external macro file

## 3) **interactive** session

- real-time command input by user
- textual, graphical, (network-based)



# Select the way of control

main.cc

```
int main(int argc, char** argv) {
    auto* runManager = G4RunManagerFactory::CreateRunManager();
    runManager->SetUserInitialization(new MyDetectorConstruction());

    // Physics list
    G4VModularPhysicsList* physicsList = new MyPhysicsList;
    physicsList->SetVerboseLevel(1);
    runManager->SetUserInitialization(physicsList);

    // User actions initialization
    runManager->SetUserInitialization(new MyActionInitialization());
```

**Insert the control code here!**

```
delete runManager;
```

```
}
```



# ① Hard-coded C++

```
// ...  
// User actions initialization  
runManager->SetUserInitialization(new MyActionInitialization());  
  
runManager->Initialize();  
runManager->BeamOn(1000);  
  
// ...  
delete runManager;  
}
```

- You must **initialize** and **start** the run by issuing “beam on”
- Even the **number of events** has to be specified!

## ② Batch session

```
// ...
// User actions initialization
runManager->SetUserInitialization(new MyActionInitialization());

G4UImanager* UImanager = G4UImanager::GetUIpointer();
G4String command = "/control/execute ";
G4String fileName = argv[1]; ←
UImanager->ApplyCommand(command + fileName);

// ...
delete runManager;
}
```

- This example gets the **file name of the macro** from the command-line argument:

```
./myApplication my-macro.mac
```

# Macro file: example basic/B1

exampleB1.mac

```
# Initialize the run (necessary)
/run/initialize
```

```
# gamma 6 MeV
/gun/particle gamma
/gun/energy 6 MeV
/run/printProgress 100
/run/beamOn 1000
```

```
# proton 210 MeV
/gun/particle proton
/gun/energy 210 MeV
```

```
/run/beamOn 1000
```

This is a comment line  
starting with hash, #

Meaning of these  
commands will be  
explained later 😊

The contents of the file, excluding **#comments**, are executed line by line in the application (→ previous slide)



## ③ Interactive session

---

- Many different session types, inheriting from **G4UIsession** class:
  - command-line based (dumb terminal)
  - graphical
  - special
  - your own? 😊
- **G4UIExecutive** class enabling to select the *appropriate session at runtime*, based on the environment variables (recommended)

# 3a: Concrete UI session

```
// ...
```

```
G4UIsession* session = new G4UITerminal();  
session->StartSession();  
delete session;
```

```
// ...
```

Select one of the  
concrete clases

## Session types:

- **G4UITerminal** – command-line (like C-shell)
- **G4UI(t) csh** – csh- or tcsh-like specific terminal
- **G4UIQt** – modern graphical UI (recommended)
- **G4UIWin32** – for windows only
- **G4UIWt** – experimental web-browser based
- **G4UIGAG** – for GAG java UI
- ...



# G4UIQt session

Visualization

Icons

The screenshot shows the G4UIQt interface for a session named 'exampleB1'. The top toolbar contains various icons for file operations, navigation, and execution. Below the toolbar are tabs for 'Scene tree, Help, History' and 'Useful tips'. The 'Scene tree' tab is active, displaying a search bar and a list of commands such as 'control', 'units', 'process', 'analysis', 'gui', 'particle', 'geometry', 'tracking', 'event', 'cuts', 'run', 'random', 'material', 'physics\_lists', 'gun', 'vis', 'heptst', and 'physics\_engine'. The central visualization window shows a 3D model of a detector component with labels 'Shape2' and '10 cm'. The text 'Geant4' is displayed in large orange letters, and 'exampleB1' is in green. A timestamp 'Thu May 11 14:46:06 2017' is visible in the top right of the viewer. Below the viewer is an 'Output' window containing the following text:

```
#  
# To get nice view  
# Make the "World" box invisible  
/vis/geometry/set/visibility World 0 false  
/vis/scene/notifyHandlers  
# "Envelope" is transparent blue to represent water  
/vis/geometry/set/colour Envelope 0 0 1 .3  
/vis/scene/notifyHandlers
```

At the bottom of the interface is a 'Session:' input field, which is circled in red.

Command tree

Output (Cout)

Command input

## 3b G4UIExecutive

- **G4UIExecutive** *behaves like a G4UISession*, but it selects the **most appropriate** concrete session:
  - from constructor argument
  - from **environment** variables: G4UI\_USE\_QT, ...
  - from \$HOME/.g4Session file
  - from the **list** (first that applies):

Available UI session types: [Qt, tcsh, csh]

→ See from *hands-on session*

```
// ...  
G4UIExecutive* ui = new G4UIExecutive(argc, argv);  
ui->SessionStart();  
delete ui;  
// ...
```

You may add a third argument here, i.e. the session name

# ②③ Universal

## batch/interactive approach

```
int main(int argc, char** argv) {
    // ...
    if (argc == 1) { ←————— No argument
        // Interactive mode
        G4UIExecutive* ui = new G4UIExecutive(argc, argv);
        ui->SessionStart();
        delete ui;
    } else { ←————— One argument (or more)
        // Batch mode
        G4UImanager* UImanager = G4UImanager::GetUIpointer();
        G4String command = "/control/execute ";
        G4String fileName = argv[1];
        UImanager->ApplyCommand(command + fileName);
    }
    // ...
}
```

- Mode selected based on application argument:
  - **No** argument = **interactive** mode
  - **One** argument = **batch** mode



# Executing macro commands

---

- **Hard-coded (!)**

```
// ...  
G4UImanager* UImanager = G4UImanager::GetUIpointer();  
G4String command = "put your command here";  
UImanager->ApplyCommand(command);  
// ...
```

- **Batch session**

- put the command in the **macro file**

- **Interactive session**

- just **type the command** in the window or in the terminal line



# Example UI commands: a few useful ones...

---

- `/run/verbose 1` – sets how much output the run manager will print (similar for other classes)
- `/run/initialize` – *initializes* the run (constructing the geometry, physics and preparing the user actions)
- `/run/beamOn 100` – starts a run with 100 events
- `/control/execute macroName` – run all commands contained in a macro file
- A complete list of built-in commands is available in the Geant4 Application Developers Guide, Chapter 7.1



Hands-on: ready to start!

---



# Hands-on

---

- **All slides** (so far) available in
  - <https://agenda.infn.it/event/37741/>
- Let us start with the **exercises**:
  - <http://geant4.lns.infn.it/pavia2024/introduction>
  - Bookmark this link: all exercises will be uploaded here
  - Now **task0** available