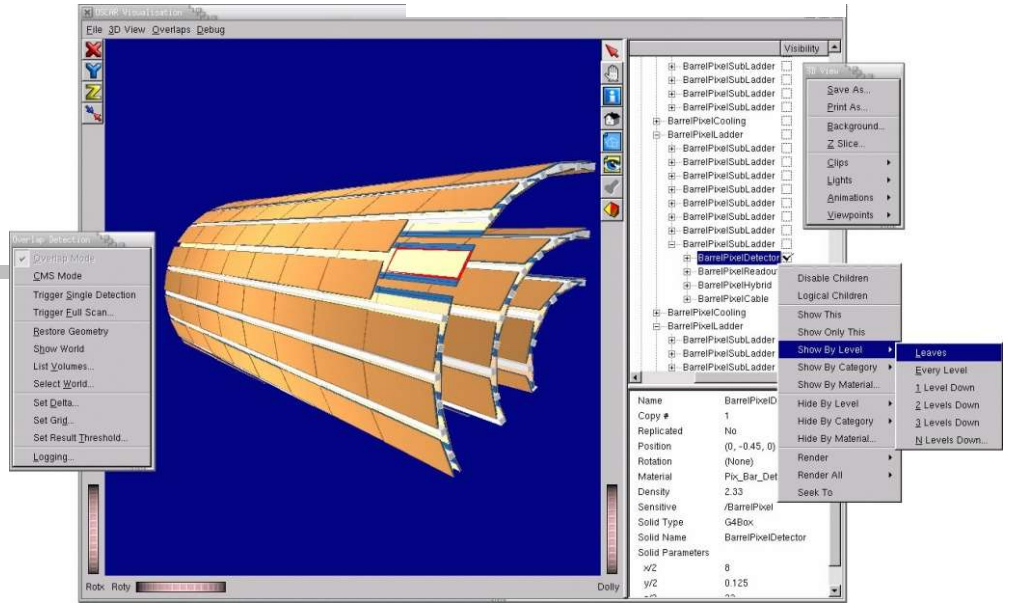
Geant4 (G)UI

Luciano Pandola

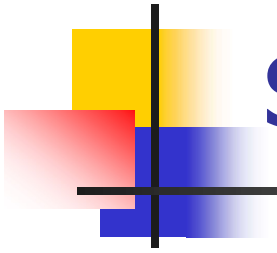
INFN – Laboratori Nazionali del Sud



A lot of material by J. Pipek

Geant4 Course
at the VIEnna Workshop on Simulations (VIEWS24)
Vienna, April 22nd- 25th, 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());

    delete runManager;
}
```

Insert the control code here!

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

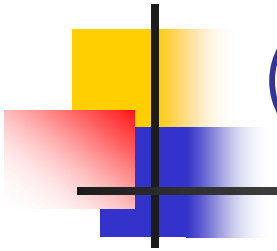
exampleB1.mac

```
# 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

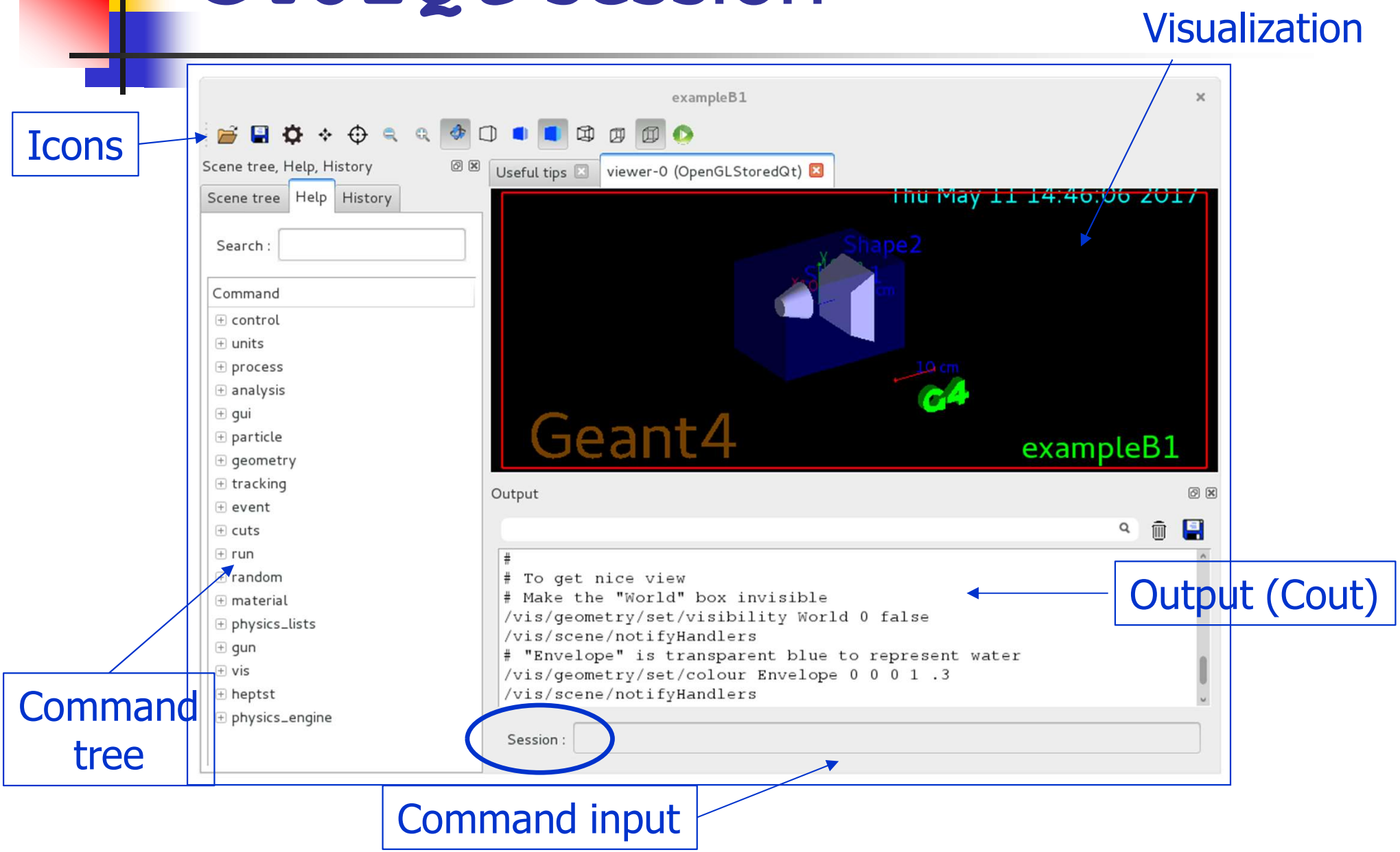
```
// ...

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



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*

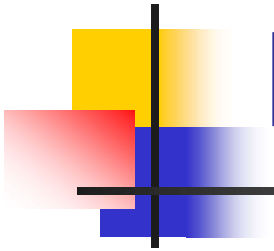
```
// ...
 G4UIExecutive* ui = new G4UIExecutive(argc, argv);
 ui->SessionStart();
 delete ui;
// ...
```

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



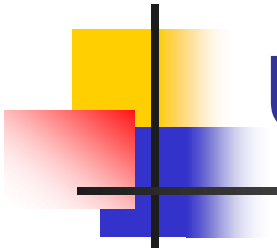
Executing macro commands

- **Hard-coded** (!)

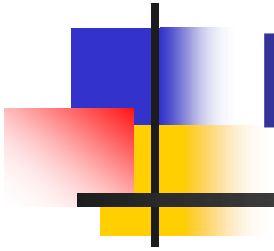
```
// ...
G4UImanager* UImanager = G4UImanager::GetUIpointer();
G4String command = "put your command here";
UImanager->ApplyCommand(command);
// ...
```

- **Batch** session
  - put the command in the macro file
- **Interactive** session
  - just type the command in the window or in the terminal line

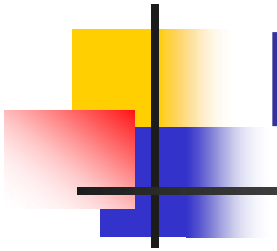
Example UI commands: a few useful ones…



- **`/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://indico.cern.ch/event/1275551`**
- Let us start with the exercises:
  - **`http://geant4.lns.infn.it/vienna2024/introduction`**
  - Bookmark this link: all exercises will be uploaded here
  - Now **task0** available