User Interface I

Igor Semeniouk

LLR, CNRS - Ecole Polytechnique

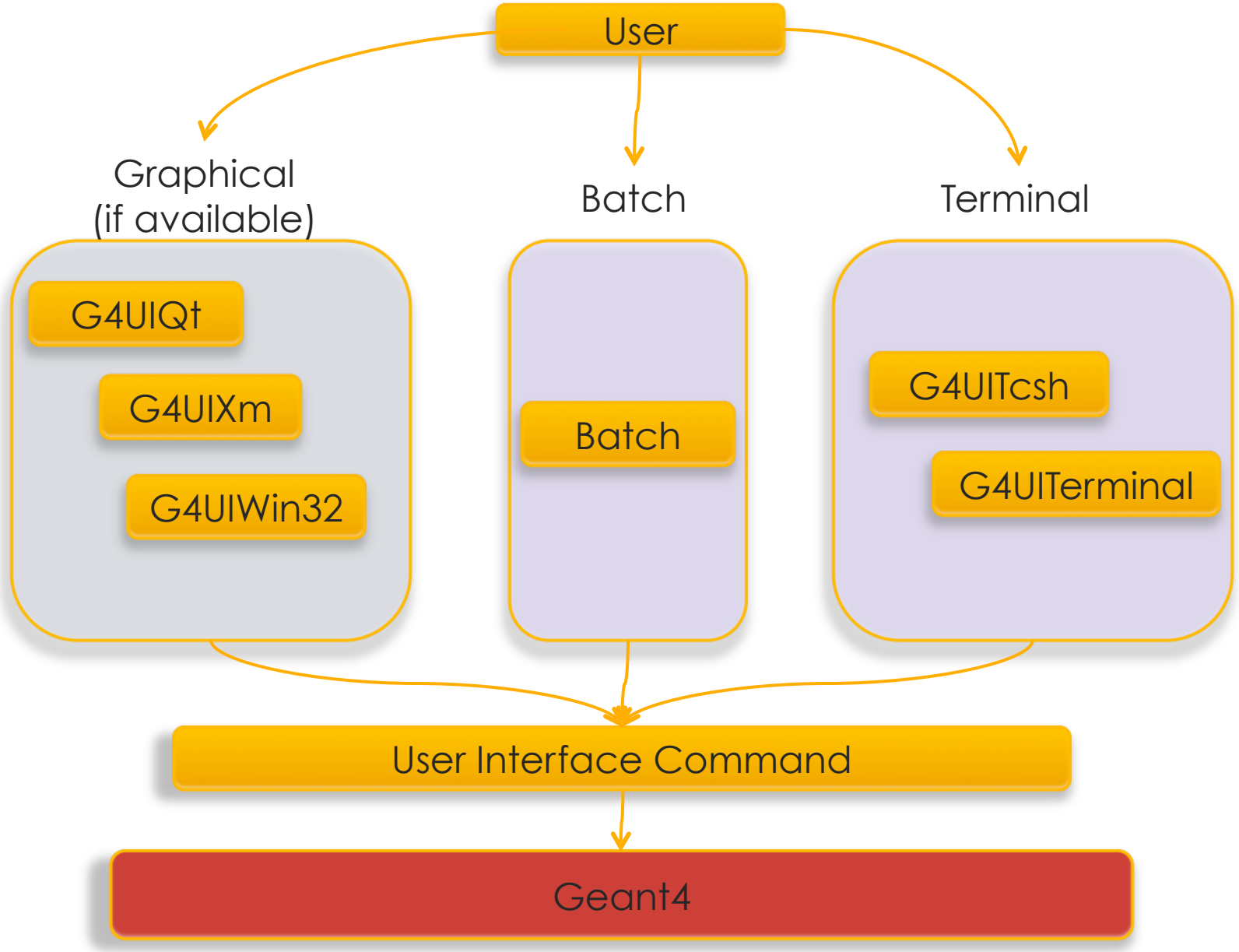
Slides from Laurent GARNIER, IRISA / INS2I / CNRS

Based on Makoto Asai (SLAC) slides

Contents

* Geant4 User Interfaces overview
* Command syntax
* Macro file
* G4UIExecutive

Geant4 UI overview



Geant4 UI overview

* Choosing your own user interface (if available)
  * By argument in command line

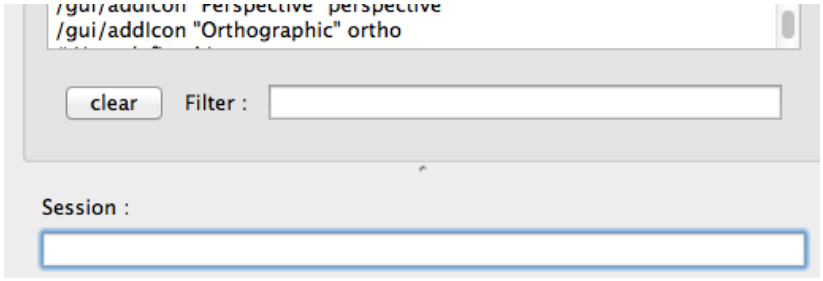
```
G4UIExecutive* ui = new G4UIExecutive(argc, argv, « qt »);
```

  * By setting session variable at compilation time (G4UI_USE_QT, G4UI_USE_TCSH…)
  * By ~/.g4session file

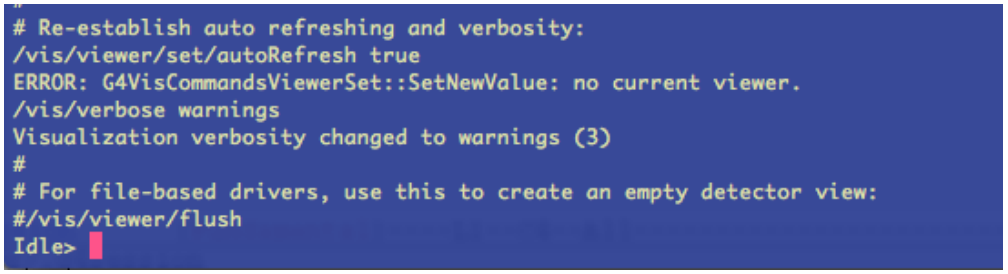
```
--**-XEmacs: .g4session      (Fundamental)----L1--
tcsh # default session
exampleN03 Qt # (application name / session type)
myapp tcsh
```

  * If nothing selected, Geant will guess the best available session for you

Qt session



tcsh



Geant4 UI command

* In a Geant4 interactive session, a command consists of
  * Command directory
  * Command
  * Parameter(s)

```
/run/verbose 1
/vis/viewer/flush
```

* A parameter can be a type of string, boolean, integer, double or G4ThreeVector.
  * Space is a delimiter.
  * Use double-quotes (“”) for string with space(s).

* A parameter may be “omittable”. If it is the case, a default value will be taken if you omit the parameter.
  * Default value is either predefined default value or current value according to its definition.
  * If you want to use the default value for your first parameter while you want to set your second parameter, use “!” as a place holder.

```
/dir/command ! second
```

Command submission

* Geant4 UI command can be issued by
  * (G)UI interactive command submission

```
Idle> /run/beamOn 1
```

  * Macro file

```
/control/execute file_name
```

  * Hard-coded implementation
    * Slow but no need for the targeting class pointer
    * Should not be used inside an event loop

```
G4UImanager* UI = G4UImanager::GetUIpointer();
UI->ApplyCommand("/run/verbose 1");
```

Command availability

* The availability of individual commands may vary according to the implementation of your application and may even vary dynamically during the execution of your job.
* Some commands are available only for limited Geant4 application state(s).
  * E.g. /run/beamOn is available only for Idle states.
* Command will be refused in case of
  * Wrong application state,
  * Wrong type of parameter, insufficient number of parameters, parameter out of its range (integer or double type parameter) or out of its candidate list (string type parameter)
  * Command not found

Macro file

- Macro file is an ASCII file containing UI commands.
- All commands must be given with their full-path directories.
- Use "#" for comment line.
  - First "#" to the end of the line will be ignored.
  - Comment lines will be echoed if /control/verbose is set to 2.
- Macro file can be executed
  - using the command :
    "/control/execute file_name"
  - hard-coded in c++:

```
G4UImanager* UI = G4UImanager::GetUIpointer();
UI->ApplyCommand("/control/execute file_name");
```

Macro file example

```
# Macro file for the visualization setting for the initialization phase
# of the B2 example when running in interactive mode


# Use these open statements to open selected visualization
# Use this open statement to create an OpenGL view:
/vis/open OGL
#
# Disable auto refresh and quieten vis messages whilst scene and
# trajectories are established:
/vis/viewer/set/autoRefresh false
/vis/verbose errors
#
# Draw geometry:
/vis/drawVolume
#
# Specify view angle:
/vis/viewer/set/viewpointThetaPhi 90. 180.
# Draw hits at end of event:
/vis/scene/add/hits
...
```

Available Commands

* You can get a list of available commands including your custom ones by

```
/control/manual [directory]
```

=> Plain text format to standard output

```
Idle > help
```

=> "help" command in user interface

* List of Geant4 built-in commands is also available in section 7.1 of *User's Guide For Application Developers.*

Alias & Loops

* Alias can be defined by

```
/control/alias [name] [value]
```

* Alias is to be used with other UI command.
  * Use curly brackets, { and }.
  * For example, frequently used lengthy command can be shortened by aliasing.

```
/control/alias tv /tracking/verbose
{tv} 1
```

* A set of commands or macros can be also called in a loop using **`/control/loop`** and **`/control/foreach`** commands

Batch mode / interactive mode

In your *main()*

```
int main(int argc, char** argv)
{
    ...
  if (argc != 1)
   { // batch mode
     G4String command = "/control/execute ";
     G4String fileName = argv[1];
     UImanager->ApplyCommand(command+fileName);
   }
  else
   {  // interactive mode : define UI session
     G4UIExecutive* ui = new G4UIExecutive(argc, argv);
     ui->SessionStart();
     delete ui;
   }
```

Call your executable

* Interactive mode

```
$>  my_application
```

* Batch mode

```
$>  my_application run1.mac
```

Terminal commands

- Interactive terminal supports some Unix-like commands for directory.
  - **`cd`**, **`pwd`** - change and display current command directory
    - By setting the current command directory, you may omit (part of) directory string.
  - **`ls`** - list available UI commands and sub-directories
- It also supports some other commands.
  - **`history`** - show previous commands
  - **`!`***`historyID`* - re-issue previous command
  - arrow keys and tab (TC-shell only)
  - **`?`***`UIcommand`* - show current parameter values of the command
  - **`help`** [*`UIcommand`*] - help
  - **`exit`** - job termination
- Above commands are interpreted in the interactive terminal and are not passed to Geant4 kernel. You cannot use them in a macro file.