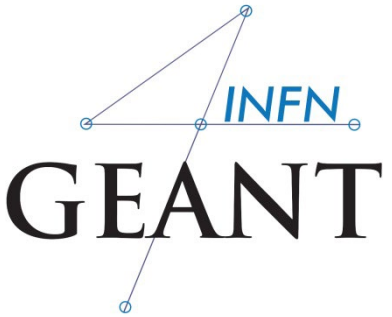


Custom User Commands

Luciano Pandola

INFN – Laboratori Nazionali del Sud

pandola@lns.infn.it

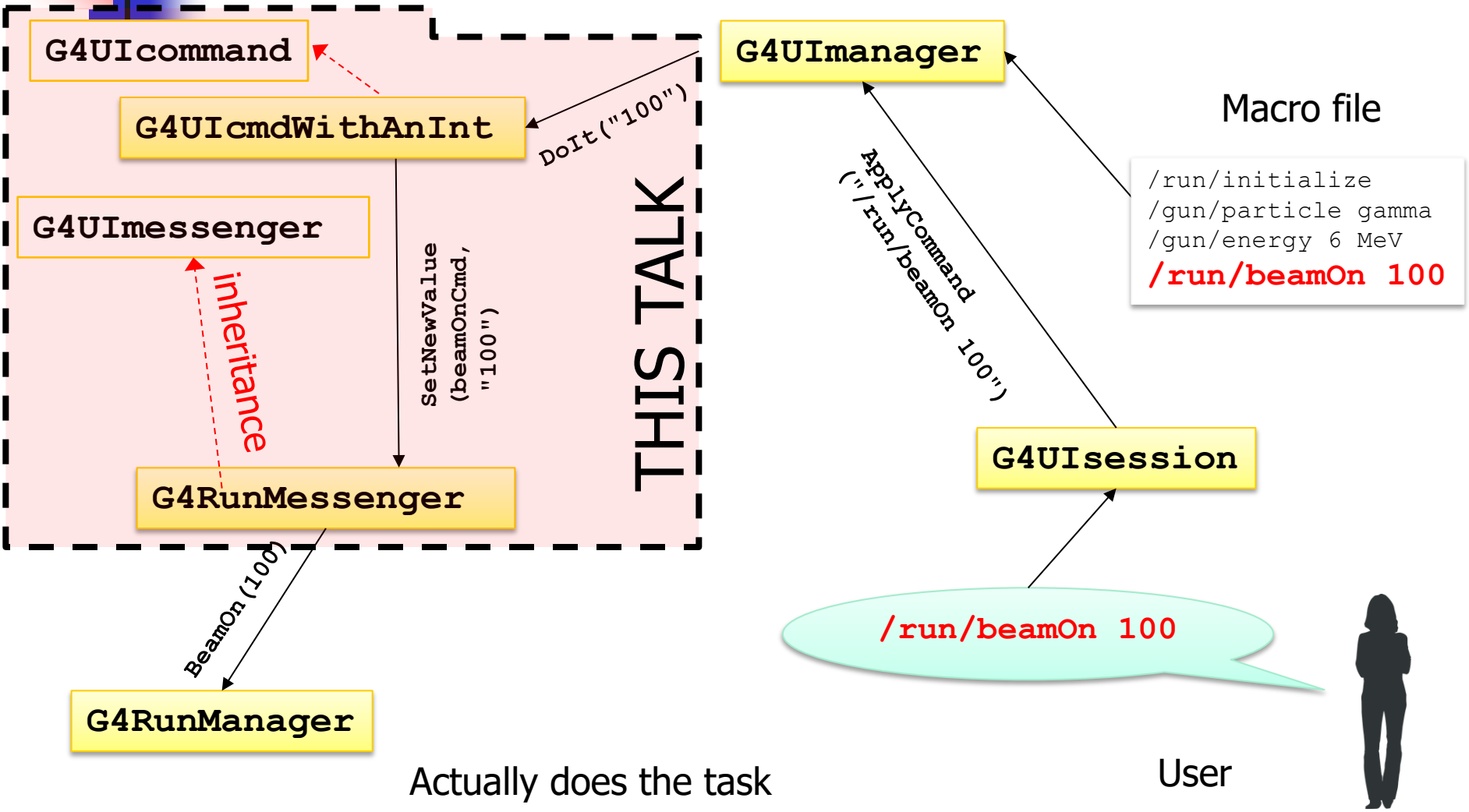
A lot of material by J. Pipek

The 2nd Geant4 School in China, Shandong University, Qingdao, March 25th- 29th, 2019

Reminder: a few **useful** built-in commands …

- **`/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

User interaction schema

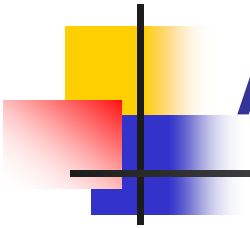


Example: G4RunMessenger

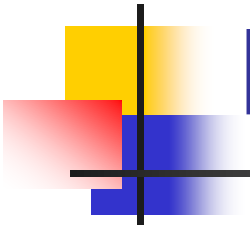
- Each command results in a method call (typically) of an **associated object** (e.g. **G4RunManager**)

```
/run/beamOn 100 ⇒ runManager->BeamOn(100);
/run/verbose 1 ⇒ runManager->SetVerboseLevel(1);
/run/initialize ⇒ runManager->Initialize();
/run/physicsModified ⇒ runManager
   ->PhysicsHasBeenModified();
...
/random/setSeeds 8 8 888 ⇒ G4Random::setTheSeeds({8,
88, 888});
```

Add custom macro commands



1. Create a **messenger class** for the commands
   - Can inherit from Geant4 base classes
2. Create an instance of **command directory** in the messenger constructor
3. Create the **command instances** inside the messenger constructor
   - → your custom commands (what's their name, etc)
4. Write the code that responds to the commands in the messenger **`SetNewValue`** method
   - Which kind of action is triggered by the command
5. Create an **messenger instance** at an appropriate place (ideally the associated object's constructor)



Built-in command types

with examples

- **`G4UIcmdWithoutParameter`**: /run/initialize
- **`G4UIcmdWithAnInteger`**: /run/beamOn 10
- **`G4UIcmdWithABool`**: /process/em/auger true*
- **`G4UIcmdWithADouble`**: /particle/property/decay/br 0.5
- **`G4UIcmdWithADoubleAndUnit`**: /gps/time 1.5 us
- **`G4UIcmdWithAString`**: /gps/particle e-
- **`G4UIcmdWith3Vector`**: /gps/direction 0.707 0.707 0.0
- **`G4UIcmdWith3VectorAndUnit`**:
  /gps/position 0.0 0.0 0.0 m

***Note:** The **true** values are **t**, **true**, **y**, **yes** and **1** (case insensitive); everything else is **false**.

New messenger: header file

MyClassMessenger.hh

```
#include <G4UImessenger.hh>
#include <G4UIdirectory.hh>
#include <G4UIcmdWithADouble.hh>
#include <G4UIcmdWithoutParameters.hh>

#include "MyClass.hh"

class MyClassMessenger : public G4UImessenger
{
public:
    MyClassMessenger(MyClass* myObject);

    ~MyClassMessenger();

    void SetNewValue(G4UIcommand* command, G4String newValue) override;

private:

    MyClass* fObject;
    G4UIdirectory* fDirectory;
    G4UIcmdWithADouble* fSomeParameterCommand;
    G4UIcmdWithoutParameters* fDoSomethingCommand;
};
```

Geant4 headers

Header to the class to manage

Inherit from **G4UImessenger**

Commands are defined in constructor

Method reacting to **all command calls** managed by this messenger

New messenger: source file (1)

MyClassMessenger.cc

Include the header

```
#include "MyClassMessenger.hh"

MyClassMessenger::~MyClassMessenger() {
    delete fSomeParameterCommand;
    delete fDoSomethingCommand;
    delete fDirectory;
}

MyClassMessenger::MyClassMessenger(MyClass* myObject)
    : fObject(myObject) {
    fDirectory = new G4UIdirectory("/myClass/");
    fDirectory->SetGuidance("Commands for MyClass");

    fSomeParameterCommand =
    // ...
```

UI command and directory pointers should be deleted!

Assign the pointer to the **associated** object

Create the command directory (with description)

To be continued in the next slide...

New messenger: source file (2)

MyClassMessenger.cc

Create a new command with description

```
    // ...
    fSomeParameterCommand = new G4UIcmdWithADouble("/myClass/setParameter", this);
    fSomeParameterCommand->SetGuidance("Set some parameter");
    fSomeParameterCommand->AvailableForStates(G4State_PreInit, G4State_Idle);
    // ... More details

    fDoSomethingCommand = new G4UIcmdWithoutParameters("/myClass/do", this);
}
```

Enable only in certain state(s)

Method reacting to command calls

Which command is it?

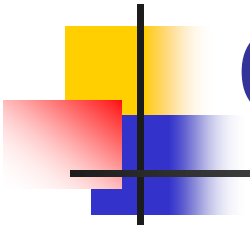
What are the command's parameters?

```
void MyClassMessenger::SetNewValue(G4UIcommand* cmd, G4String newValue) {
    if (cmd == fDoSomethingCommand) { fObject->DoSomething(); }
    else if (cmd == fSomeParameterCommand) {
        G4double value = fSomeParameterCommand->GetNewDoubleValue(newValue);
        fObject->SetParameter(newValue);
    }
}
```

Simple reaction

Use the parsed value

It is necessary to parse the parameters **`string`**!



G4GenericMessenger

Good news: we have alternative!

- no need to implement a new class ☺
- no need to instantiate a UI directory ☺
- no need to declare the comand objects ☺
- no need to parse values end write conditions ☺
- no need to delete so many pointers ☺
- without support for more complex commands ☹

It's the **`G4GenericMessenger`** class.

G4GenericMessenger - use

MyClass.hh

```
#include <G4GenericMessenger.hh>

class MyClass {
// ...
G4GenericMessenger* fMessenger;
};
```

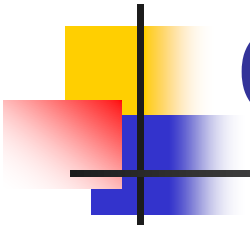
MyClass.cc

```
MyClass::MyClass() {
    // ...
    fMessenger = new G4GenericMessenger("/myClass/", this);
    fMessenger->SetGuidance("Commands for MyClass");
    fMessenger->DeclareMethod("setParameter", &MyClass::SetParameter)
        .SetStates(G4State_PreInit, G4State_Idle)
        .SetGuidance("Set some parameter");
    fMessenger->DeclareMethod("do", &MyClass::DoIt);
}

MyClass::~MyClass() {
    // ...
    delete fMessenger;
}
```

Method to be executed when the command is called

These few lines do exactly the same as previously defined **`MyClassMessenger`**. **Simpler?** ☺



Conclusions

- A number of general-purpose commands are provided by Geant4, but **users can define more**, according to their needs → flexibility!
- Defining new user commands is possible using one of the ways:
  - **more general** and **complex**: G4UImessenger, G4UIdirectory, G4UIcommand, ...
  - **easier** but **less flexible**: G4GenericMessenger