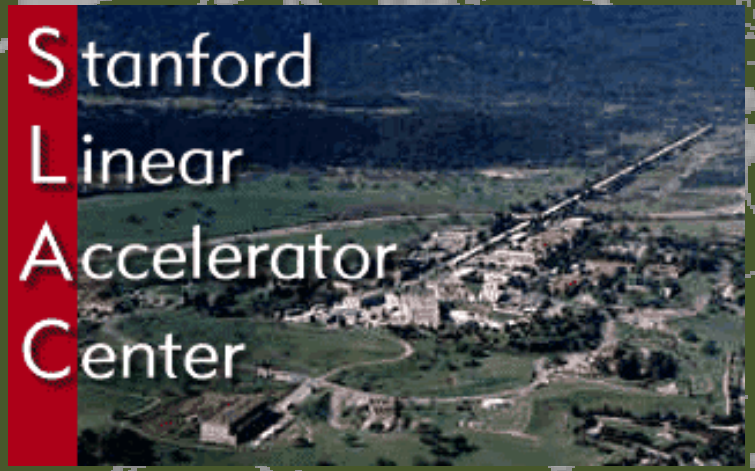


UI command

Makoto Asai (SLAC)

Geant4 Tutorial Course @ Bordeaux

November 2005

Geant4

Contents

- Command syntax
- G4UIterminal
- Alias and loop
- Mechanism of UI command
- Messenger class
- Defining a command
- Implementing a messenger

Geant4 UI command

- 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 or double.
  - 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
  - Macro file
  - 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");
```

- The availability of individual command, the ranges of parameters, the available candidates on individual command parameter 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 refusal

- Command will be refused if
  - Wrong application state
  - Wrong type of parameter
  - Insufficient number of parameters
  - Parameter out of its range
    - For integer or double type parameter
  - Parameter out of its candidate list
    - For string type parameter
  - Command not found

Macro file

- Macro file is an ASCII file contains 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
  - interactively or in (other) macro file

    `/control/execute` ***`file_name`***
  - hard-coded

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

Available Commands

- You can get a list of available commands including your custom ones by
  `/control/manual [directory]`
  - Plane text format to standard output
  `/control/createHTML [directory]`
  - HTML file(s)
- List of built-in commands is also available in section 7.1 of *User's Guide For Application Developers.*

G4UIterminal

- G4UIterminal is a concrete implementation derived from G4UIsession abstract class. It provides character-base interactive terminal functionality to issue Geant4 UI commands.
  - C-shell or TC-shell (Linux only)

```
new G4UIterminal(new G4UItcsh);
```

- It 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 (TC-shell only)
  - **?***UIcommand* - show current value
  - **help** [*UIcommand*] - help
  - **exit** - job termination
- Above commands are interpreted in G4UIterminal and are not passed to Geant4 kernel. You cannot use them in a macro file.

Alias

- Alias can be defined by

  /control/alias [name] [value]

  - It is also set with **/control/loop** and **/control/foreach** commands
  - Aliased value is always treated as a string even if it contains only numbers.
- 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 trv1 "/tracking/verbose 1"
{trv1}
```

  - Aliases can be used recursively.

```
/control/alias file1 /diskA/dirX/fileXX.dat
/control/alias file2 /diskB/dirY/fileYY.dat
/control/alias run 1
/myCmd/getFile {file{run}}
```

Loop

- **`/control/loop`** and **`/control/foreach`** commands execute a macro file more than once. Aliased variable name can be used inside the macro file.
- **`/control/loop [`*`macroFile`*`] [`*`counterName`*`]`**
  `[`*`initialValue`*`] [`*`finalValue`*`] [`*`stepSize`*`]`
  - ***`counterName`*** is aliased to the number as a loop counter
- **`/control/foreach [`*`macroFile`*`] [`*`counterName`*`] [`*`valueList`*`]`**
  - ***`counterName`*** is aliased to a value in *`valueList`*
  - ***`valueList`*** must be enclosed by double quotes (" ")

- on UI terminal or other macro file

```
/control/loop myRun.mac Ekin 10. 20. 2.
```

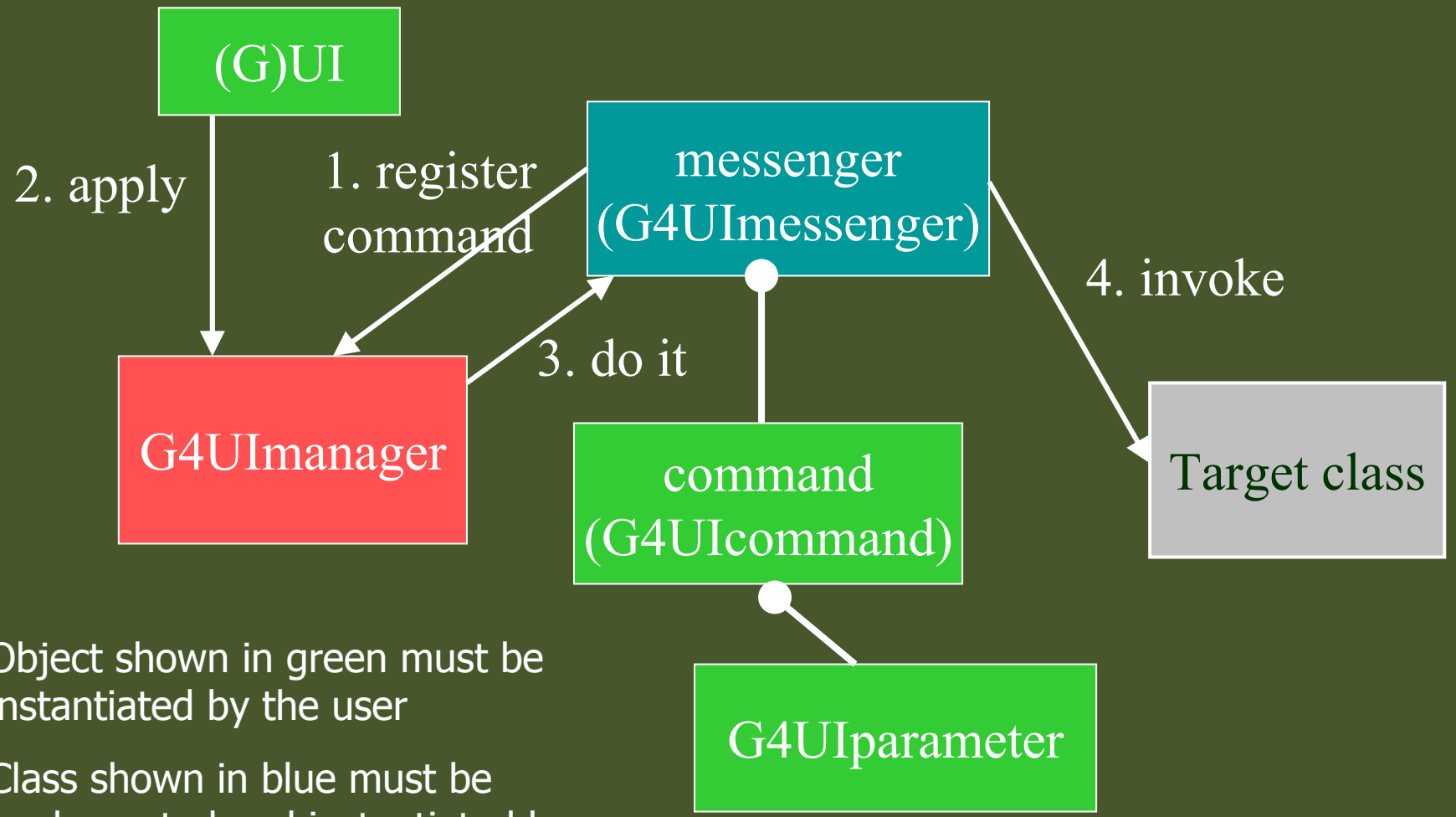
- in myRun.mac

```
/control/foreach mySingleRun.mac pname “proton pi+ mu+”
```

- in mySingleRun.mac

```
/gun/particle {pname}
/gun/energy {Ekin} GeV
/run/beamOn 100
```

Mechanism of UI command



Object shown in green must be instantiated by the user

Class shown in blue must be implemented and instantiated by the user

Messenger class

- Each messenger class must be derived from G4UImessenger base class. A messenger class can handle more than one UI commands.
- A messenger class should be instantiated by the constructor of the target class to which commands should be delivered, and should be deleted by the destructor of the target class.
- Methods of messenger class
  - Constructor
    - Define (instantiate) commands / command directories
  - Destructor
    - Delete commands / command directories
  - void SetNewValue(G4UIcommand* command,G4String newValue)
    - Convert "newValue" parameter string to appropriate value(s) and invoke a method of the target class
    - This method is invoked when a command is issued.
  - G4String GetCurrentValue(G4UIcommand* command)
    - Access to a get-method of the target class and convert the current values to a string
    - This method is invoked when the current value(s) of parameter(s) of a command is asked by (G)UI.

Definition (instantiation) of a command

- To be implemented in the constructor of a messenger class.

```
A01DetectorConstMessenger::A01DetectorConstMessenger
(A01DetectorConstruction* tgt)
:target(tgt)
{
  mydetDir = new G4UIdirectory("/mydet/");
  mydetDir->SetGuidance("A01 detector setup commands.");

  armCmd = new
   G4UIcmdWithADoubleAndUnit("/mydet/armAngle",this);
  armCmd->SetGuidance("Rotation angle of the second arm.");
  armCmd->SetParameterName("angle",true);
  armCmd->SetRange("angle>=0. && angle<180.");
  armCmd->SetDefaultValue(30.);
  armCmd->SetDefaultUnit("deg");
}
```

- Guidance can (should) be more than one lines. The first line is utilized as a short description of the command.

G4UIcommand and its derivatives

- G4UIcommand is a class which represent a UI command. G4UIparameter represents a parameter.
- G4UIcommand can be directly used for a UI command. Geant4 provides its derivatives according to the types of associated parameters.
  - G4UIcmdWithoutParameter
  - G4UIcmdWithAString
  - G4UIcmdWithABool
  - G4UIcmdWithAnInteger
  - G4UIcmdWithADouble, G4UIcmdWithADoubleAndUnit
  - G4UIcmdWith3Vector, G4UIcmdWith3VectorAndUnit
  - G4UIdirectory
  - A UI command with other type of parameters must be defined by G4UIcommand base class with G4UIparameter.

Parameter name(s)

```
void SetParameterName(
    const char*parName,
    G4bool omittable,
    G4bool currentAsDefault=false);
void SetParameterName(
    const char*nam1, const char*nam2, const char*nam3,
    G4bool omittable,
    G4bool currentAsDefault=false);
```

- Parameter names are used in help, and also in the definition of parameter range.
- If "omittable" is true, the command can be issued without this particular parameter, and the default value will be used.
- If "currentAsDefalult" is true, current value of the parameter is used as a default value, otherwise default value must be defined with SetDefaultValue() method.

Range, unit and candidates

`void SetRange(const char* rangeString)`

- Available for a command with numeric-type parameters.
- Range of parameter(s) must be given in C++ syntax.

```
aCmd->SetRange("x>0. && y>z && z>(x+y)");
```

- Not only comparison with hard-coded number but also comparison between variables and simple calculation are available.
- Names of variables must be defined by SetParameterName() method.

`void SetDefaultUnit(const char* defUnit)`

- Available for a command which takes unit.
- Once the default unit is defined, no other unit of different dimension will be accepted.
- Alternatively, you can define a dimension (unit category) without setting a default unit.

```
void SetUnitCategory(const char* unitCategory)
```

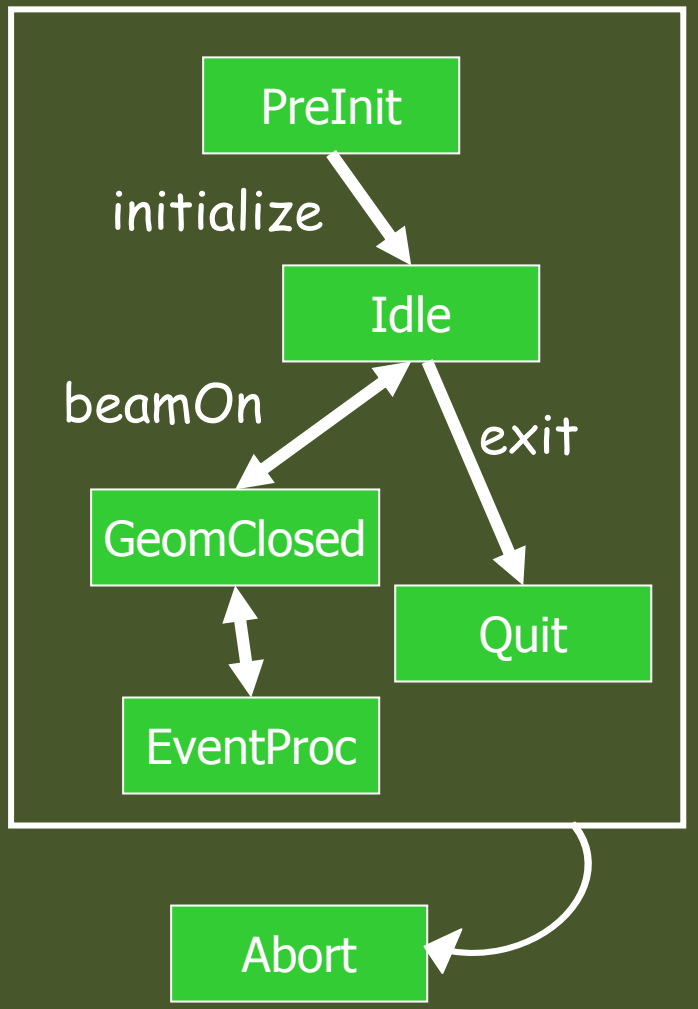
`void SetCandidates(const char* candidateList)`

- Available for a command with string type parameter
- Candidates must be delimited by a space.
- Candidates can be dynamically updated.

Available state

```
void AvailableForStates(G4ApplicationState s1,…)
```

- Define command's applicability for Geant4 application states.
- Geant4 has six application states.
  - G4State_PreInit
    - Material, Geometry, Particle and/or Physics Process need to be initialized
  - G4State_Idle
    - Ready to start a run
  - G4State_GeomClosed
    - Geometry is optimized and ready to process an event
  - G4State_EventProc
    - An event is processing
  - G4State_Quit, G4State_Abort
    - UI command unavailable



Converting between string and values

- Derivatives of G4UIcommand with numeric and boolean parameters have corresponding conversion methods.
- From a string to value

```
G4bool GetNewBoolValue(const char*)
G4int GetNewIntValue(const char*)
G4double GetNewDoubleValue(const char*)
G4ThreeVector GetNew3VectorValue(const char*)
```

  - To be used in SetNewValue() method in messenger.
  - Unit is taken into account automatically.
- From value to string

```
G4String ConvertToString(...)
G4String ConvertToString(...,const char* unit)
```

  - To be used in GetCurrentValue() method in messenger.

SetNewValue and GetCurrentValue

```
void A01DetectorConstMessenger
::SetNewValue(G4UIcommand* command,G4String newValue)
{
  if( command==armCmd )
  { target->SetArmAngle(armCmd->GetNewDoubleValue(newValue)); }
}

G4String A01DetectorConstMessenger
::GetCurrentValue(G4UIcommand* command)
{
  G4String cv;
  if( command==armCmd )
  { cv = armCmd->ConvertToString(target->GetArmAngle(),"deg"); }
  return cv;
}
```

- G4ParticleGunMessenger is a good example of implementing a messenger and defining various types of commands.