XV Seminar on Software for Nuclear, Subnuclear and Applied Physics

Alghero (ITALY)

27th May – 1st June 2018

Geant4 Installation

Pietro Pisciotta

Geant4 tutorial

Installation process

1) Check that you meet all the requirements
2) Download Geant4 source code
3) Configure the build using CMake
4) Make & install
5) Configure your environment to use Geant4

https://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/InstallationGuide/html/index.html

① Supported platforms & requirements

- **Operating system**
  - "recent" Linux (e.g. CentOS 7), best support
  - macOS 10.10+
  - Windows 7+ (limited support, not recommended)
- **Compilers**
  - C++11 compliance
  - such as GCC 4.8.5+, clang 3.6+, Visual C++ 14.0 (2015)
- **CMake** (configuration generation tool) 3.3+
- System libraries (as development packages):
  - expat, xerces-c

Virtual Machine:
CentOS 7 with gcc 4.8.5

These may or may not be necessary. Just keep this in mind when compilation fails.

CMake installation (if not provided)

- Depending on the OS installation, CMake may not be installed by default. In that case you have to install it:
  - **Linux**: it is recommended to use the CMake provided by the package management system of your distribution.

    If version 3.3+ is not available:
    1. download the latest version *(http://www.cmake.org/)*
    2. unzip the tar-ball
    3. **./bootstrap, make, make install**
  - **macOS**: install it using the Darwin64 dmg installerpackage
  - **Windows**: install it using the Win64/32 exe installerpackage

Note: You may also want to install **ccmake** and/or **cmake-gui** tools for user-friendly configuration

Optional libraries

- **X11** for simple graphical user interface and ray-tracing
- **OpenGL** for visualization
- **Qt4** or **Qt5** for graphical user interface
- **ROOT** for data analysis (even inside Geant4)

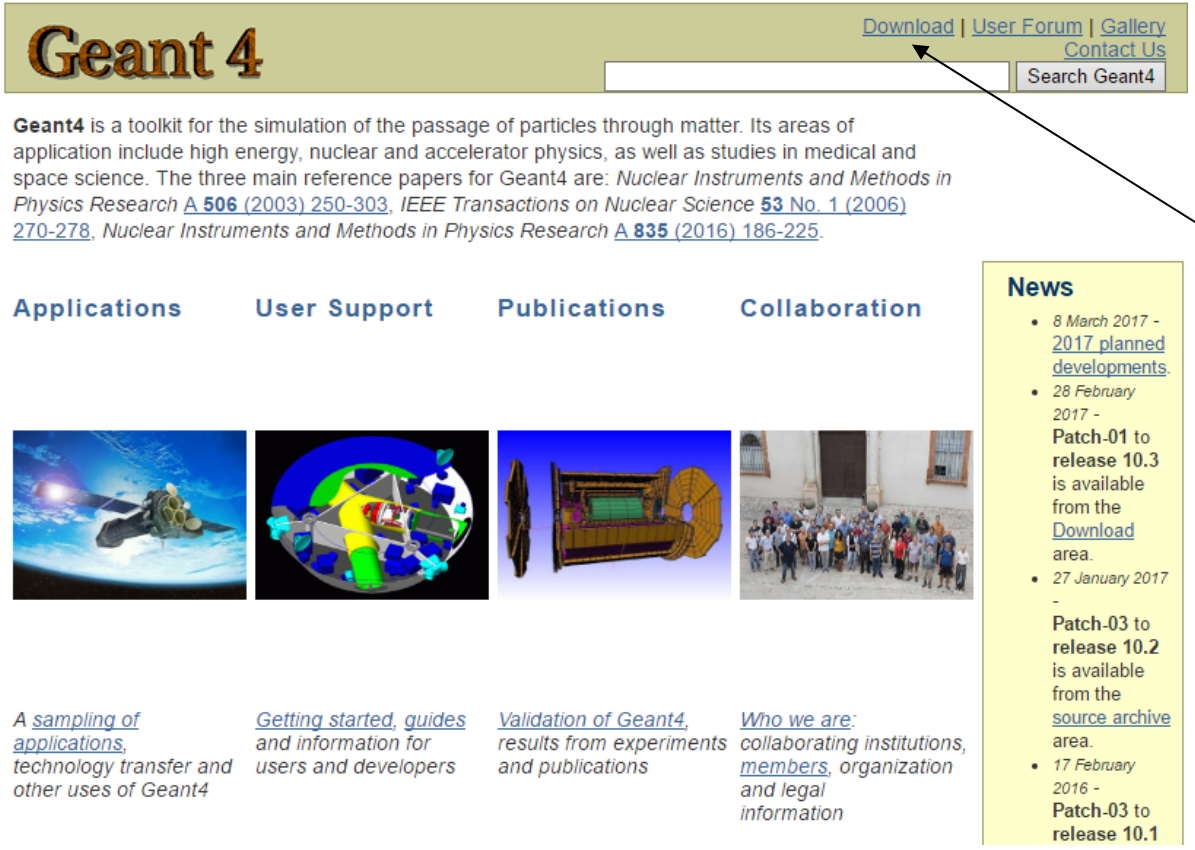
Less frequently used libraries/tools:

Motif, OpenInventor, DAWN, RayTracer X11, HepRApp, WIRED JAS Plug-in, AIDA, VRML browser, (external) CLHEP, Wt...

② Download Geant4…

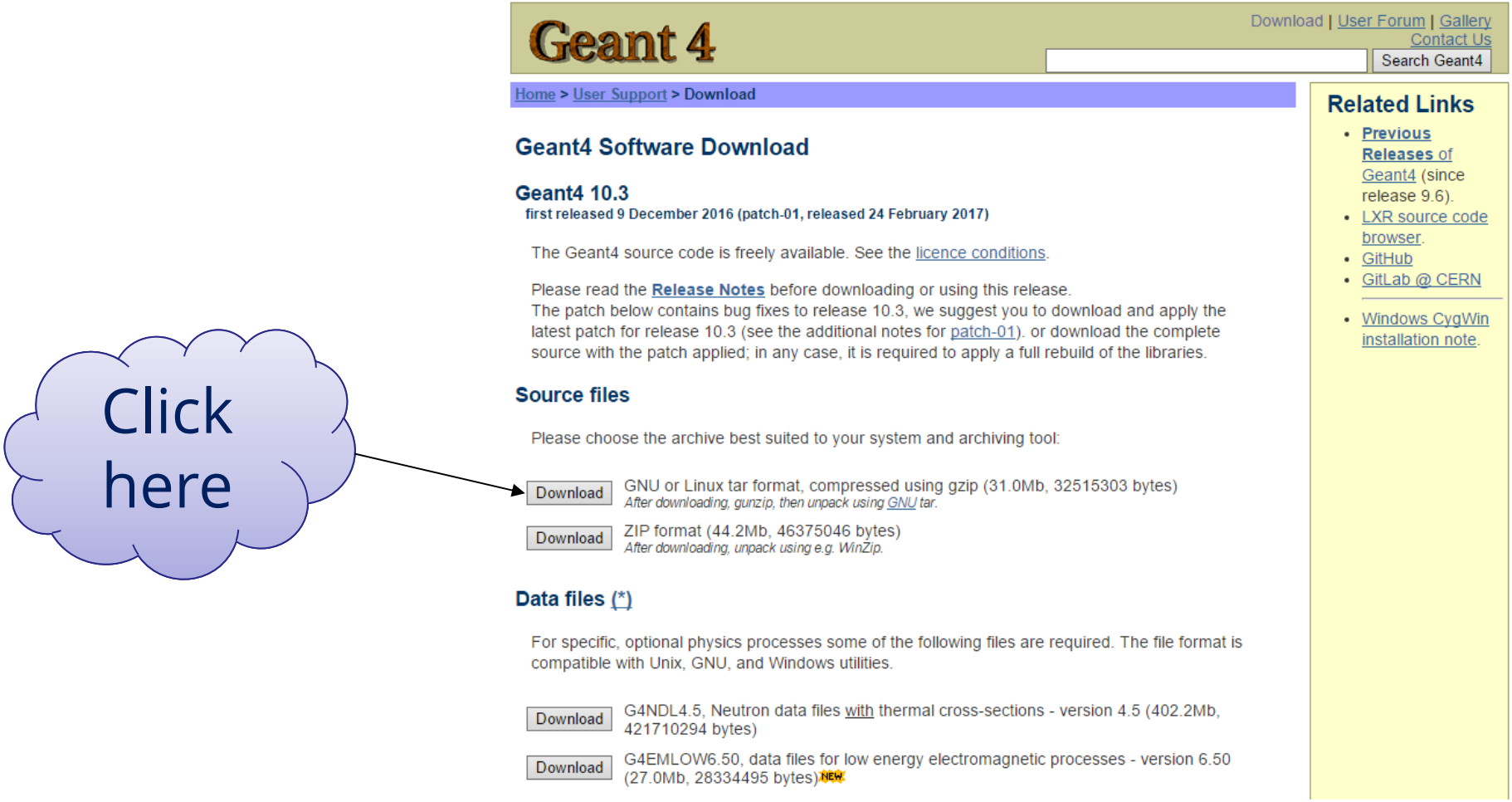
- Go to the Geant4 webpage:

  http://geant4.web.cern.ch/geant4/



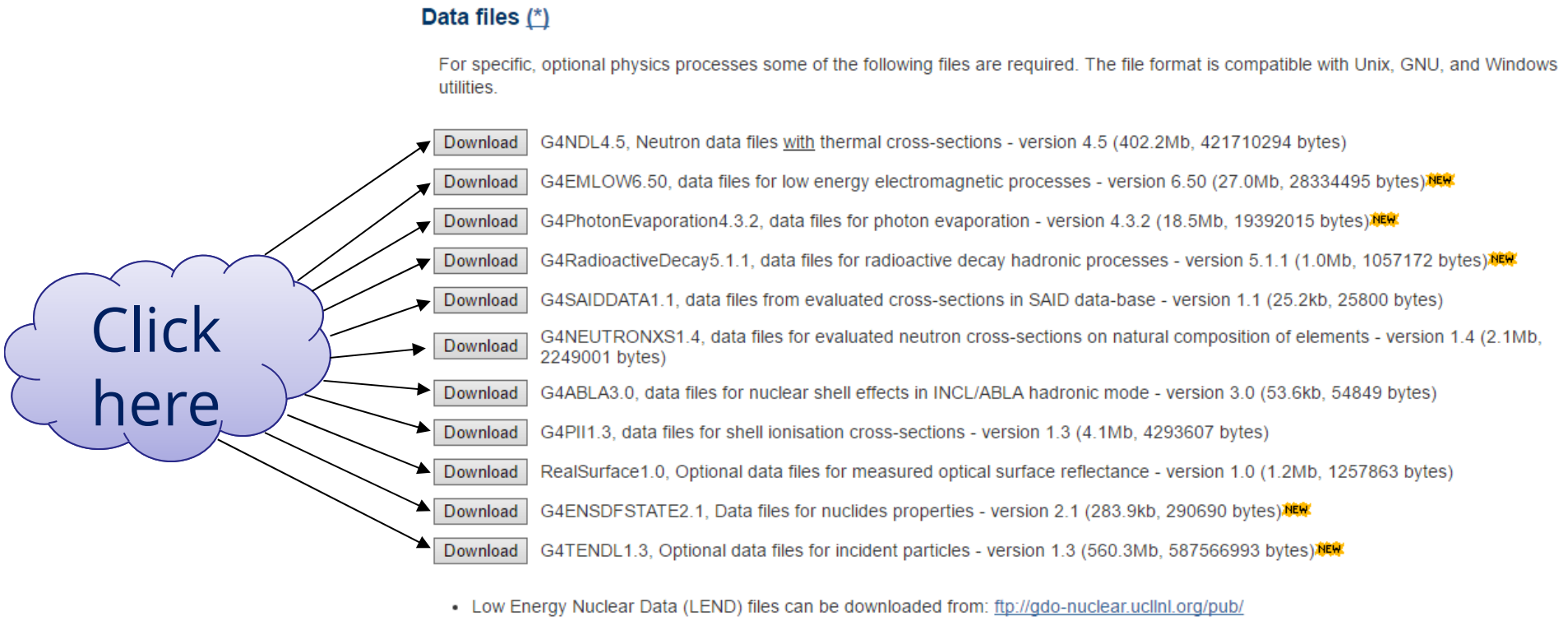
…download Geant4…

http://geant4.web.cern.ch/geant4/support/download.shtml



Download data (optional)

Alternative 1: download everything (slow connections)



Alternative 2: use **CMake** to download data automatically (preferred)

③ Configuration with CMake

- Extract the package into **source directory**

```
tar xzvf geant4.10.04.tar.gz
```

- Create the **build directory**

```
mkdir geant4-build
```

- Run CMake in the **build directory**

```
cd geant4-build
cmake [options...] ../geant4.10.04
```

See next slide for directory explanation...

Choose name to your liking

Directories for installation

Source directory: where you unpack the source

`/usr/local/geant4/geant4.10.04` VM

Build directory: where you run CMake and build Geant4 (“working directory”)

`/usr/local/geant4/geant4.10.04-build` VM

Installation directory: where you install Geant4 to and which the applications compile against

`/usr/local/geant4/geant4.10.04-install` VM

Only the installation dir is necessary to compile & run user apps.

CMake configuration options

Important options:

- -DCMAKE_INSTALL_PREFIX=*...installation_path...*
- -DGEANT4_INSTALL_DATA=ON/OFF
- -DGEANT4_BUILD_MULTITHREADED=ON/OFF

Further options:

- -DGEANT4_USE_OPENGL_X11=ON/OFF
- -DGEANT4_USE_QT=ON/OFF
- ...

https://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/InstallationGuide/html/ch02s03.html

Running CMake

CMake configures the build and generates Unix **Makefiles** to perform the actual build:

```
cmake -DGEANT4_INSTALL_DATA=ON -DGEANT4_BUILD_MULTITHREADED=ON -DCMAKE_INSTALL_PREFIX=
/usr/local/geant4/geant4.10.04-install /usr/local/geant4/geant4.10.04
```

```
-- The C compiler identification is GNU 4.8.5
-- The CXX compiler identification is GNU 4.8.5
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc – works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
...(~50 lines)...
-- Configuring done
-- Generating done
-- Build files have been written to: /usr/local/geant4/geant4.10.04-build
```

If you see that, you are successful !!!

If you see errors at this point, carefully check the messages output by CMake

④ Compile…

- Run `make` (and get a cup of coffee)

Tip: If you have a multi-core machine, you can run the compilation in parallel using multiple jobs. Just add the `-jN` parameter, where N is the required number of jobs (it is recommended to set this to the number of your processor's cores), e.g.:

```
make -j2
```

```
Scanning dependencies of target G4ENSDFSTATE
Scanning dependencies of target G4NDL
[  0%] Creating directories for 'G4ENSDFSTATE'
[  0%] Creating directories for 'G4NDL'
[  0%] Performing download step (download, verify and extract) for 'G4NDL‘
...(4029 lines, ~1 hour of execution)
[100%] Built target G4visXXX
[100%] Building CXX object
source/visualization/gMocren/CMakeFiles/G4GMocren.dir/src/G4GMocrenIO.cc.o
[100%] Building CXX object
source/visualization/gMocren/CMakeFiles/G4GMocren.dir/src/G4GMocrenMessenger.cc.o
[100%] Linking CXX shared library ../../../BuildProducts/lib64/libG4GMocren.so
[100%] Built target G4GMocren
```

If you see that, you are successful !!!

...and install

- Run `make install` (this takes much less time)

```
make install
```

```
[  0%] Built target G4ENSDFSTATE
[  0%] Built target G4NDL
[  0%] Built target PhotonEvaporation
[  0%] Built target RadioactiveDecay
[  0%] Built target G4ABLA
...(42830 lines, ~2 minute of execution)
-- Installing: /usr/local/geant4/geant4.10.04-install/include/Geant4/G4VModelCommand.hh
-- Installing: /usr/local/geant4/geant4.10.04-install/include/Geant4/G4VModelFactory.hh
-- Installing: /usr/local/geant4/geant4.10.04-install/include/Geant4/G4VTrajectoryModel.hh
-- Installing: /usr/local/geant4/geant4.10.04-install/include/Geant4/G4VisTrajContext.hh
-- Installing: /usr/local/geant4/geant4.10.04-install/include/Geant4/G4VisTrajContext.icc
```

Tip: If you want to combine the two steps (compilation + installation) into one, you can leave out the first step.

(Random) installation notes

- **Windows:** See the installation guide

https://geant4.web.cern.ch/geant4/UserDocumentation/UsersGuides/InstallationGuide/html/ch02s02.html

- **Binary packages:** Installation without compiling Geant4 is possible (but not recommended)
- **Data packages:** If you haven't used CMake to download them, unpack the downloaded files in the share/Geant4-10.4.0/data/ sub-directory of your installation

⑤ Set-up environment

- Geant4 need properly set environment variables:

```
G4ABLADATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/G4ABLA3.0"
G4ENSDFSTATEDATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/G4ENSDFSTATE2.1"
G4LEDATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/G4EMLOW6.50"
G4LEVELGAMMADATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/PhotonEvaporation4.3.2"
G4NEUTRONHPDATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/G4NDL4.5"
G4NEUTRONXSDATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/G4NEUTRONXS1.4"
G4PIIDATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/G4PII1.3"
G4RADIOACTIVEDATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/RadioactiveDecay5.1.1"
G4REALSURFACEDATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/RealSurface1.0"
G4SAIDXSDATA="/usr/local/geant4/geant4.10.04-install/share/Geant4-10.4.0/data/G4SAIDDATA1.1"
LD_LIBRARY_PATH="...:/usr/local/geant4/geant4.10.04-install/lib64"
PATH="...:/usr/local/geant4/geant4.10.04-install/bin"
```

- To set them up properly in your shell, run the script in Geant4 installation directory:

```
source /usr/local/geant4/geant4.10.04-install/bin/geant4.(c)sh
```

- You can put this line your `~/.bashrc` file (or similar for other shells)

Your Geant4 is ready now.

Can we continue to…
build an application?

Build a Geant4 application

Application build process

1) Properly organize your code into directories
2) Prepare a CMakeLists.txt file
3) Create a build directory and run CMake
4) Compile (make) the application
5) Run the application

Note: Recommended, not enforced!

① Application source structure in Geant4

Official basic/B1 example:

```
2,4K  4 Dic 14:48 CMakeLists.txt
475B  4 Dic 14:48 GNUmakefile
2,8K  4 Dic 14:48 History
7,5K  4 Dic 14:48 README
4,0K  4 Dic 14:48 exampleB1.cc
226B  4 Dic 14:48 exampleB1.in
 35K  4 Dic 14:48 exampleB1.out
272B  4 Dic 14:49 include
338B  4 Dic 14:48 init_vis.mac
553B  4 Dic 14:48 run1.mac
448B  4 Dic 14:48 run2.mac
272B  4 Dic 14:49 src
3,8K  4 Dic 14:48 vis.mac
```

The text file CMakeLists.txt is the CMake script containing commands which describe how to build the exampleB1 application

contains main() for the application

Header files

```
2,2K  4 Dic 14:48 B1ActionInitialization.hh
2,4K  4 Dic 14:48 B1DetectorConstruction.hh
2,4K  4 Dic 14:48 B1EventAction.hh
2,7K  4 Dic 14:48 B1PrimaryGeneratorAction.hh
2,5K  4 Dic 14:48 B1RunAction.hh
2,4K  4 Dic 14:48 B1SteppingAction.hh
```

Source files

```
2,9K  4 Dic 14:48 B1ActionInitialization.cc
7,7K  4 Dic 14:48 B1DetectorConstruction.cc
2,6K  4 Dic 14:48 B1EventAction.cc
4,3K  4 Dic 14:48 B1PrimaryGeneratorAction.cc
5,8K  4 Dic 14:48 B1RunAction.cc
3,2K  4 Dic 14:48 B1SteppingAction.cc
```

Macro file containing the commands

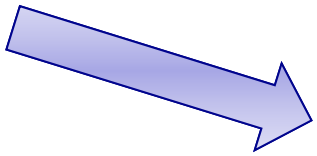
② CMake (again)

- **CMake** is a build configuration tool
  - it takes configuration file (CMakeLists.txt)
  - it finds all dependencies (in our case, Geant4)
  - creates **Makefile** to run the compilation itself
- You have to write this CMakeLists.txt file
  - take inspiration in examples directories
  - be sure to set the name of your application correctly
  - specify all auxiliary files you need

Note: It is possible but **discouraged** to base build on GNU make instead of CMake.

CMakeLists.txt

```
cmake_minimum_required(VERSION 2.6 FATAL_ERROR)
project(B1)
option(WITH_GEANT4_UIVIS "Build example with Geant4 UI and Vis drivers" ON)
if(WITH_GEANT4_UIVIS)
  find_package(Geant4 REQUIRED ui_all vis_all)
else()
  find_package(Geant4 REQUIRED)
endif()

include(${Geant4_USE_FILE})
include_directories(${PROJECT_SOURCE_DIR}/include)

file(GLOB sources ${PROJECT_SOURCE_DIR}/src/*.cc)
file(GLOB headers ${PROJECT_SOURCE_DIR}/include/*.hh)

add_executable(exampleB1 exampleB1.cc ${sources} ${headers})
target_link_libraries(exampleB1 ${Geant4_LIBRARIES})

set(EXAMPLEB1_SCRIPTS
  exampleB1.in
  exampleB1.out
  init_vis.mac
  run1.mac
  run2.mac
  vis.mac
  )

foreach(_script ${EXAMPLEB1_SCRIPTS})
  configure_file(
    ${PROJECT_SOURCE_DIR}/${_script}
    ${PROJECT_BINARY_DIR}/${_script}
    COPYONLY
    )
```



File structure

1) Cmake minimum version and **project name**
2) Find and configure G4
3) Configure the project to use G4 and B1 headers
4) List the **sources**
5) Define and link the **executable**
6) Copy any macro files to the build directory

③ Build directory and CMake

1) If modifying the Geant4 examples, copy them to your $HOME first:

```
cp -r /usr/local/geant4/geant4.10.04/examples/basic/B1 ~
```

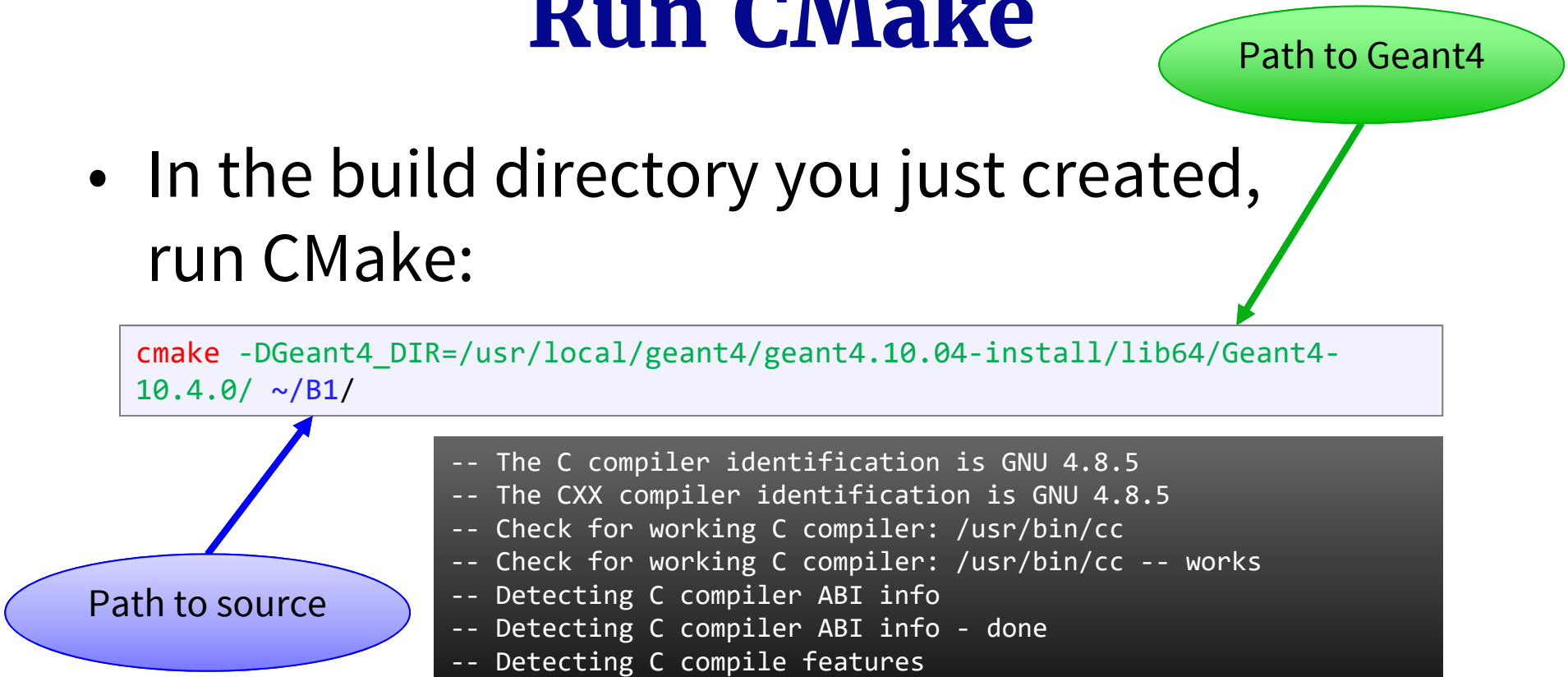
2) Create a **build directory***, where the compiled application will be put:

```
mkdir -p ~/B1-build
cd ~/B1-build
```

***Note:** It is possible (though not recommended) to compile **inside** source directory.

Run CMake

- In the build directory you just created, run CMake:



④ Compilation

- In the build directory, run `make`

  (and don't get a cup of coffee)

  - You have only a couple of files, it should be ready in a minute or two
  - An **executable** with the name of your application is created (e.g. exampleB1) in build directory
  - **Macros** and other auxiliary files are copied into build directory

`make -j2`

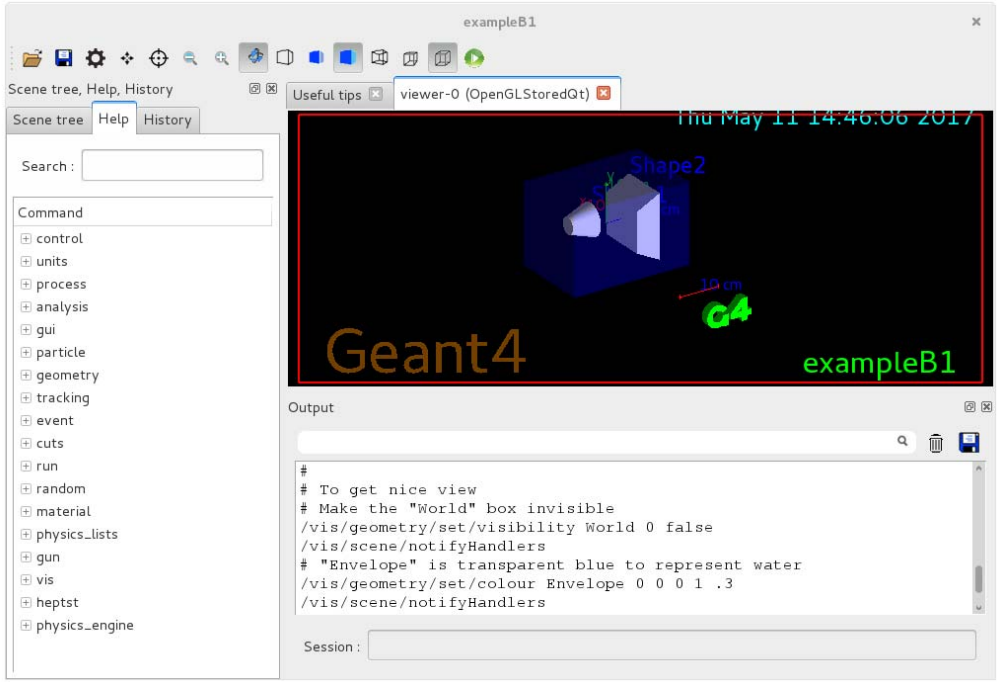
```
Scanning dependencies of target exampleB1
[ 12%] Building CXX object CMakeFiles/exampleB1.dir/exampleB1.cc.o
[ 25%] Building CXX object CMakeFiles/exampleB1.dir/src/B1RunAction.cc.o
[ 37%] Building CXX object CMakeFiles/exampleB1.dir/src/B1SteppingAction.cc.o
[ 50%] Building CXX object CMakeFiles/exampleB1.dir/src/B1DetectorConstruction.cc.o
[ 62%] Building CXX object
CMakeFiles/exampleB1.dir/src/B1PrimaryGeneratorAction.cc.o
[ 75%] Building CXX object CMakeFiles/exampleB1.dir/src/B1EventAction.cc.o
[ 87%] Building CXX object CMakeFiles/exampleB1.dir/src/B1ActionInitialization.cc.o
[100%] Linking CXX executable exampleB1
[100%] Built target exampleB1
```

⑤ Run the application – GUI

- Just type the name of your application, including the **./** identifier of current directory (e.g. ./exampleB1)
- By default, **graphical user interface** is started*

```
./exampleB1
```

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



***Note:** Depends on your application main(), Geant4 configuration, etc.

Conclusion

Building an application is easy ☺