

# GEANT4 BEGINNERS COURSE

GSSI, L'Aquila (Italy)

12<sup>nd</sup> May 2014

## How to install **Geant 4** and build an application

**Geant 4** tutorial course



# Outline

- Supported platforms & compilers
- Required software
- Where to download the packages
- Geant4 toolkit installation (*release 10*)
  - Using *Cmake, on Linux*
- Building a Geant4 application with CMake
- Example of a Geant4 application
- CLHEP full version installation (*optional*)

# Supported platforms & compilers

- Linux systems

- Scientific Linux CERN 6 with gcc 4.7.X, 64 bit

*Geant4 has also been successfully compiled on other Linux distributions, including Debian, Ubuntu and openSUSE (not officially supported)*



- MacOSX systems

- Mac OS X 10.8 (Lion with clang 3.3), 64bit

*Geant4 has also been successfully compiled on Mac OS X 10.7 (Lion) with clang 3.1 (Apple), and MacOS X 10.9 (Mavericks) with clang 2.79 (not officially supported)*



- Windows systems

- Windows 7 with Visual Studio 11 (VS2010).



# Required software

- The **Geant4** toolkit source code (10.00)
- **C++ compiler**
  - It is usually installed on your Linux. If not, you need to install it (*not shown here*)
- **CMake** 2.6.4 or higher
  - for clang, Cmake 2.8.2 or higher is required
- CLHEP library
  - an internal version is now supplied with the geant4 source (since 9.5 version)
- The Geant4 data files
  - an automatic procedure can retrieve them (with CMake)

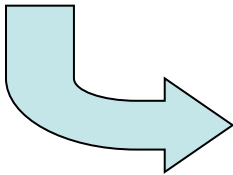
# External software packages

## Suggested tools (optional):

- X11 OpenGL Visualization (Linux and Mac OS X)
  - Requires: X11, OpenGL or MesaGL (headers and libraries).
- Qt User Interface and Visualization (All Platforms)
  - Requires: Qt4, OpenGL or MesaGL (headers and libraries).
- Motif User Interface and Visualization (Linux and Mac)
  - Requires: Motif and X11, OpenGL or MesaGL headers and libraries.
- Open Inventor Visualization (All Platforms)
- X11 RayTracer Visualization (Linux and Mac OS X)
- DAWN postscript renderer
- HepRApp Browser
- VRML browser
- WIRED4 JAS Plug-In
- GDML Support (All Platforms)
- AIDA (Abstract Interface for Data Analysis)

# Where to download the packages

- **Geant4**



## Geant4 10.0

released 6 December 2013

The Geant4 source code is freely available. See the [licence conditions](#).

Please read the [Release Notes](#) before downloading or using this release. It is required to apply a full rebuild of the libraries.

### Source files

Please choose the archive best suited to your system and archiving tool:

Download

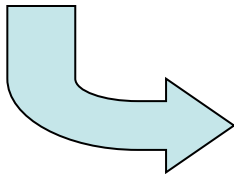
GNU or Linux tar format, compressed using gzip (29.4Mb, 30780131 bytes)  
After downloading, gunzip, then unpack using [GNU tar](#).

Download

ZIP format (41.4Mb, 43365939 bytes)  
After downloading, unpack using e.g. WinZip.

<http://geant4.cern.ch/support/download.shtml>

- **CLHEP**



## CLHEP - A Class Library for High Energy Physics

Shortcuts to: [Documentation](#) [Download](#) [CLHEP editors](#) [Mailing List](#) [CLHEP Workshops](#) [News and Bug Reports](#)

The **CLHEP project** was proposed by [Leif Lönnblad](#) at CERN 92. It is intended to be a set of HEP-specific [foundation](#) and [utility](#) classes such as random generators, physics vectors, geometry and linear algebra. CLHEP is structured in a set of [packages](#) independent of any external package (interdependencies within CLHEP are allowed under certain [conditions](#)).

A large fraction of contributions (mainly to the Random, Vector, Geometry and Matrix packages) came from using CLHEP within (in alphabetical order):

- the [BaBar experiment](#) @ [SLAC](#)
- the [Geant4](#) Collaboration
- the [ZOOM Project](#) @ [Fermilab](#)

### Latest Release:

The latest releases are:

- 2.1.0.1, released on November 11<sup>th</sup>, 2010.
- 1.9.4.7/2.0.4.7, released on July 2<sup>nd</sup>, 2010.

<http://proj-clhep.web.cern.ch>

# Downloading Geant4 and data files

## Source files

Please choose the archive best suited to your system and archiving tool:

- [Download](#) GNU or Linux tar format, compressed using gzip (29.4Mb, 30780131 bytes)  
*After downloading, gunzip, then unpack using [GNU tar](#).*
- [Download](#) ZIP format (41.4Mb, 43365939 bytes)  
*After downloading, unpack using e.g. WinZip.*

## Pre-compiled Libraries

These are compiled with Geant4 default settings and optimization turned on. Please choose according to your system/compiler:

- [Download](#) compiled using gcc 4.4.7 on Scientific Linux CERN 6 (SLC6, based on Redhat Linux Enterprise 6), 64 bits (15.0Mb, 15684036 bytes)
- [Download](#) compiled using gcc 4.2.1/clang-3.3 on Mac (MacOSX 10.9), 64 bits (13.6Mb, 14253160 bytes)
- [Download](#) compiled using VC++ 11.0 on Windows 7, 32 bits, zip file (48.3Mb, 50631960 bytes)
- [Download](#) compiled using VC++ 11.0 on Windows 7, 32 bits, executable installer (34.9Mb, 36606241 bytes)

**Geant4 source  
or  
pre-compiled  
libraries**

**data files**

## Data files (\*)

For specific, optional physics processes some of the following files are required. The file format is compatible with Unix, GNU, and Windows utilities.

- [Download](#) Neutron data files with thermal cross-sections - version 4.4 (402.0Mb, 421555304 bytes) **NEW**
- [Download](#) Data files for low energy electromagnetic processes - version 6.35 (18.2Mb, 19092577 bytes) **NEW**
- [Download](#) Data files for photon evaporation - version 3.0 (8.5Mb, 8864188 bytes) **NEW**
- [Download](#) Data files for radioactive decay hadronic processes - version 4.0 (962.4kb, 985509 bytes) **NEW**
- [Download](#) Data files from evaluated cross-sections in SAID data-base - version 1.1 (25.2kb, 25800 bytes)
- [Download](#) Data files for evaluated neutron cross-sections on natural composition of elements - version 1.4 (2.1Mb, 2249001 bytes) **NEW**
- [Download](#) Data files for nuclear shell effects in INCL/ABLA hadronic mode - version 3.0 (53.6kb, 54849 bytes) **NEW**
- [Download](#) Data files for shell ionisation cross-sections - version 1.3 (4.1Mb, 4293607 bytes)
- [Download](#) Optional data files for measured optical surface reflectance - version 1.0 (1.2Mb, 1257863 bytes)
- [Download](#) Data files for nuclides properties - version 1.0 (229.1kb, 234612 bytes) **NEW**

<http://geant4.cern.ch/support/download.shtml>

# Geant4 installation (10.0 version)

## Working area & installation area

- Why two different areas ?
  - To allow centralized installation of the Geant4 kernel libraries and related sources in a multi-user environment
  - To decouple user-developed code and applications from the kernel
  - To allow an easy integration of the Geant4 software in an existing software framework

## Two ways to proceed:

- Manually installing by env variables (*deprecated*)
- Using **CMake** (*recommended and officially supported*)

# Installing Geant4 with *CMake*

# CMake installation *(if not provided)*

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

In case it does not meet the minimum version requirement:

1. download the latest version (<http://www.cmake.org/>)
  2. unzip the tar-ball
  3. `./bootstrap, make, make install`
- On Mac: install it using the Darwin64 dmg installerpackage
  - On Windows: install it using the Win32 exe installerpackage

# Geant4 installation with CMake

- Unpack the geant4 source package geant4.10.00.tar.gz to a location of your choice:
  - ex.: /path/to/geant4.10.00 → source directory
- Create a directory in which to configure and run the build and store the build products (not inside the source dir!)
  - ex.: /path/to/geant4.10.10-build → build directory

```
$ cd /path/to
$ mkdir geant4.10.0-build
$ ls
geant4.10.00  geant4.10.0-build
```

- To configure, change into the build directory and run CMake:

```
$ cd /path/to/geant4.10.0-build
$ cmake -DCMAKE_INSTALL_PREFIX=/path/to/geant4.10.0-install /path/to/geant4.10.00
```

- CMAKE\_INSTALL\_PREFIX option is used to set the install directory
- The second argument to CMake is the path to the source directory.

# Geant4 installation with CMake

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

```
$ cmake -DCMAKE_INSTALL_PREFIX=/path/to/geant4.10.0-install /path/to/geant4.10.00
-- The C compiler identification is GNU
-- The CXX compiler identification is GNU
-- Check for working C compiler: /usr/bin/gcc
-- Check for working C compiler: /usr/bin/gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- setting default compiler flags for CXX
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Found EXPAT: /usr/lib64/libexpat.so
-- Looking for sys/types.h
-- Looking for sys/types.h - found
-- Looking for stdint.h
-- Looking for stdint.h - found
-- Looking for stddef.h
-- Looking for stddef.h -
```

*If you see that, you are successful !!!*



```
-- Configuring done
-- Generating done
-- Build files have been written to: /path/to/geant4.10.0-build
```

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



# Geant4 installation with CMake

- After the configuration has run, CMake have generated Unix Makefiles for building Geant4. To run the build, simply execute make in the build directory:

```
$ make -jN
```

- where N is the number of parallel jobs you require.
- The build will now run, and will output a progress report
- When build has completed, you can install Geant4 to the directory you specified earlier in CMAKE\_INSTALL\_PREFIX by running:

```
$ make install
```

# Geant4 installation with CMake

- Additional arguments can be passed to CMake to activate optional components of Geant4 (***standard*** and ***advanced*** options):
  - **-DGEANT4\_INSTALL\_DATA=ON** (*recommended*)  
the additional external data libraries are automatically downloaded
  - **-DGEANT4\_USE\_OPENGL\_X11=ON** (*recommended*)  
build the X11 OpenGL visualization driver
  - **-DGEANT4\_BUILD\_MULTITHREADED=ON** (*recommended*)  
build Geant4 libraries with support for multithreading

On Mac OSX the following two additional options are required:

  - **-DCMAKE\_C\_COMPILER=clang**
  - **-DCMAKE\_CXX\_COMPILER=clang++**
  - **-DGEANT4\_USE\_QT=ON** (*optional, but nice!!!*)  
build the Qt visualization driver
  - ...

# Geant4 installation with CMake

- If you want to *activate* additional options, simply rerun CMake in the build directory, passing it the extra options:

```
$ cd /path/to/geant4.10.0-build  
$ cmake -DGEANT4_INSTALL_DATA=ON .
```

- If you want to *deactivate* a previously selected option:

```
$ cmake -DCMAKE_INSTALL_PREFIX=/opt/geant4 -DGEANT4_USE_GDML=OFF \\  
/path/to/geant4.10.00
```

*You may also directly include the options since the beginning:*

```
cmake -DCMAKE_INSTALL_PREFIX=/path/to/geant4.9.10.00-install -DGEANT4_INSTALL_DATA=ON  
-DGEANT4_USE_OPENGL_X11=ON -DGEANT4_USE_QT=ON /path/to/geant4.9.10.00
```

# Geant4 installation with CMake

- The install of Geant4 is contained under the directory chosen (CMAKE\_INSTALL\_PREFIX), with the following structure:

```
+-- CMAKE_INSTALL_PREFIX
|
|-- bin/
|   |-- geant4-config      (UNIX ONLY)
|   |-- geant4.csh        (UNIX ONLY)
|   |-- geant4.sh         (UNIX ONLY)
|   |-- G4global.dll      (WINDOWS ONLY)
|   |-- ...
|
|-- include/
|   |-- Geant4/
|   |   |-- G4global.hh
|   |   |-- ...
|   |-- CLHEP/           (WITH INTERNAL CLHEP ONLY)
|   |-- tools/
```

- To make the Geant4 binaries and libraries available on your PATH and LIBRARY PATH and to set the variables for external data libraries:

```
$ . geant4.sh
```

*N.B.: each time you open a new shell remember to source the `geant4.sh` script before executing an application !!!*

- Alternatively, you may use the `geant4make.sh` (.csh) script to compile applications with GNUmakefile (*deprecated* → G4.10)

# Building an application with CMake

# Building an application with cmake

- To build an application using Geant4 toolkit, it is necessary to include Geant4 headers in the application sources and link the application to the Geant4 libraries:
  - using CMake → **Geant4Config.cmake** → writing a **CMakeLists.txt** script to locate Geant4 and describe the build of your application against it
- For instance: examples/basic/B1:

```
+-- B1/  
  +- CMakeLists.txt  
  +- exampleB1.cc  
  +- include/  
    | ... headers.hh ...  
  +- src/  
    ... sources.cc ...
```

CMakeLists.txt  
to be located in the root  
directory (B1)

exampleB1.cc contains main()

include/ and src/  
contain the  
implementation  
class headers and  
sources

# Building an application with cmake

```
# (1)
cmake_minimum_required(VERSION 2.6 FATAL_ERROR)
project(B1)

# (2)
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()

# (3)
include(${Geant4_USE_FILE})
include_directories(${PROJECT_SOURCE_DIR}/include)

# (4)
file(GLOB sources ${PROJECT_SOURCE_DIR}/src/*.cc)
file(GLOB headers ${PROJECT_SOURCE_DIR}/include/*.hh)

# (5)
add_executable(exampleB1 exampleB1.cc ${sources} ${headers})
target_link_libraries(exampleB1 ${Geant4_LIBRARIES})

# (6)
set(EXAMPLEB1_SCRIPTS
    exampleB1.in
    exampleB1.out
    init.mac
    init_vis.mac
    run1.mac
    run2.mac
    vis.mac
)

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

# (7)
install(TARGETS exampleB1 DESTINATION bin)
```

- The text file CMakeLists.txt is the CMake script containing commands which describe how to build the exampleB1 application
- Example of structure:
  1. Cmake minimum version and set project name
  2. Find and configure G4
  3. Configure the project to use G4 and B1 headers
  4. List sources
  5. Define and link the executable
  6. Copy any runtime script to the build directory of your application
  7. Install the executable

# Building an application with cmake

- First step: create a build directory for the specific application (suggestion: build that alongside the application source directory):

```
$ cd $HOME  
$ mkdir B1-build
```

- Change to this build directory and run CMake to generate the Makefiles needed to build the B1 application. Pass CMake two arguments:

```
$ cd $HOME/B1-build  
$ cmake -DGeant4_DIR=/home/you/geant4-install/lib64/Geant4-10.0.0 $HOME/B1
```

- CMake will now run to configure the build and generate Makefiles.:

```
$ cmake -DGeant4_DIR=/home/you/geant4-install/lib64/Geant4-10.0.0 $HOME/B1  
-- The C compiler identification is GNU  
-- The CXX compiler identification is GNU  
-- Check for working C compiler: /usr/bin/gcc  
-- Check for working C compiler: /usr/bin/gcc -- works  
-- Detecting C compiler ABI info  
-- Detecting C compiler ABI info - done  
-- Check for working CXX compiler: /usr/bin/c++  
-- Check for working CXX compiler: /usr/bin/c++ -- works  
-- Detecting CXX compiler ABI info  
-- Detecting CXX compiler ABI info - done  
-- Configuring done  
-- Generating done  
-- Build files have been written to: /home/you/B1-build
```

# Building an application with cmake

- The following files have been generated:

```
$ ls
CMakeCache.txt      exampleB1.in  init_vis.mac  run2.mac
CMakeFiles          exampleB1.out Makefile      vis.mac
cmake_install.cmake init.mac      run1.mac
```

- Once the Makefile is available we can do:

```
$ make -jN
```

- The following output should be displayed:

```
$ make
Scanning dependencies of target exampleB1
[ 16%] Building CXX object CMakeFiles/exampleB1.dir/exampleB1.cc.o
[ 33%] Building CXX object CMakeFiles/exampleB1.dir/src/B1PrimaryGeneratorAction.cc.o
[ 50%] Building CXX object CMakeFiles/exampleB1.dir/src/B1EventAction.cc.o
[ 66%] Building CXX object CMakeFiles/exampleB1.dir/src/B1RunAction.cc.o
[ 83%] Building CXX object CMakeFiles/exampleB1.dir/src/B1DetectorConstruction.cc.o
[100%] Building CXX object CMakeFiles/exampleB1.dir/src/B1SteppingAction.cc.o
Linking CXX executable exampleB1
[100%] Built target exampleB1
```

# Building an application with cmake

- List again the content of the build directory, you see the executable:

```
$ ls
CMakeCache.txt      exampleB1      init.mac      run1.mac
CMakeFiles          exampleB1.in  init_vis.mac  run2.mac
cmake_install.cmake exampleB1.out  Makefile      vis.mac
```

- Run the application, simply with `./exampleB1`, the following output should be displayed:

```
$ ./exampleB1

*****
Geant4 version Name: geant4-10-00-ref-00 [MT]    (6-December-2013)
<< in Multi-threaded mode >>
Copyright : Geant4 Collaboration
Reference : NIM A 506 (2003), 250-303
WWW : http://cern.ch/geant4
*****
```

- And that's all !!!
- If you don't want any UI or Visualization, you could rerun CMake as:

```
$ cmake -DWITH_GEANT4_UIVIS=OFF -DGeant4_DIR=/home/you/geant4-install/lib64/Geant4-10.0.0 $HOME/B1
```

# Building an application with cmake

- For further details have a look at the Installation guide:


## Geant 4

[Download](#) | [User Forum](#) | [Gallery](#)  
[Contact Us](#)

Search Geant4

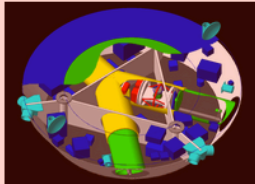
**Geant4** is a toolkit for the simulation of the passage of particles through matter. Its areas energy, nuclear and accelerator physics, as well as studies in medical and space science papers for Geant4 are published in *Nuclear Instruments and Methods in Physics Research* *IEEE Transactions on Nuclear Science* **53 No. 1 (2006) 270-278.**

### Applications



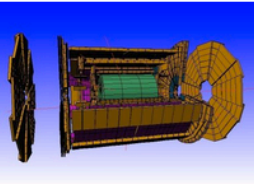
[A sampling of applications, technology transfer and other uses of Geant4](#)

### User Support




[Getting started, guides and information for users and developers](#)

### Publications



[Validation of Geant4, results from experiments and publications](#)

### Who



[Who](#)

### Events

- [Geant4 Beginners Course](#), Belfast (Northern Ireland), 20-24 January 2014.
- [SLAC Geant4 Tutorial Course](#), Jen-Hsun Huang Engineering Center, Stanford (US), 3-6 Mar
- 19<sup>th</sup> Geant4 Collaboration Meeting, Okinawa (Japan), 29 September - 4 October 2014.
- [Past events](#)

## Geant 4

[Home](#) > [User Support](#)

### User Support

- [Getting started](#)
- [Training courses and materials](#)
- Source code
  - [Download page](#)
  - [LXR code browser](#) -or- draft [doxygen documentation](#)
- [Frequently Asked Questions \(FAQ\)](#)
- [Bug reports and fixes](#)
- [User requirements tracker](#)
- [User Forum](#)
- [Documentation](#)
  - [Introduction to Geant4](#)
  - [Installation Guide](#)
  - [Application Developers Guide](#)
  - [Toolkit Developers Guide](#)
  - [Physics Reference Manual](#)
- [Examples](#)
- Physics lists
  - [Electromagnetic](#)
  - [Hadronic](#)
- User Aids
  - [Tips for improving CPU performance](#)

# **A Geant4 application**

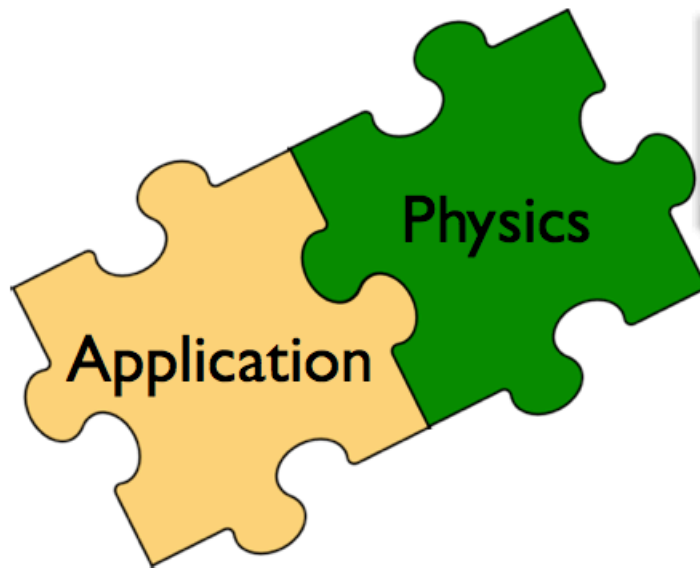
# A Geant4 application

- Geant4 is a **toolkit**: no “main” program
- User is responsible of building an application
- Increased flexibility, but...
  - ... more work to be done

# A Geant4 application



# A Geant4 application

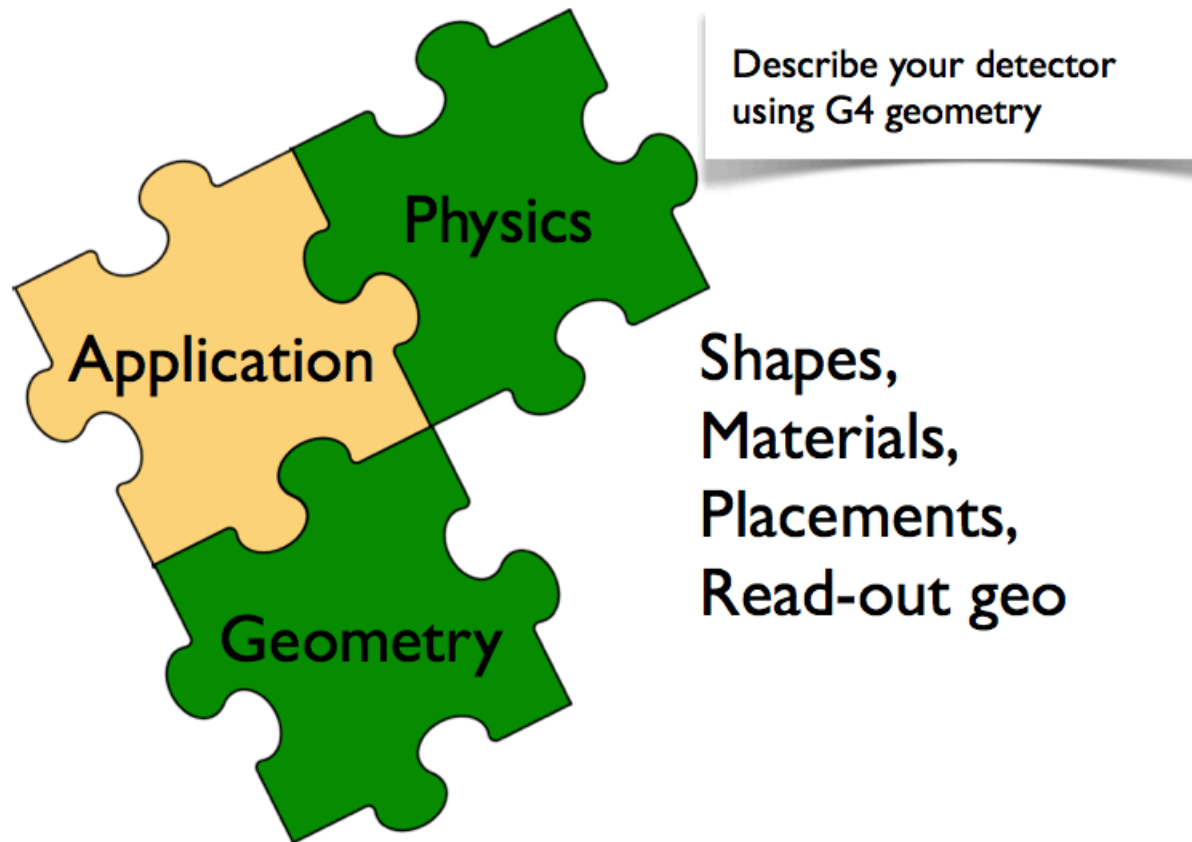


From Geant4:  
One of the provided Physics  
lists or build/tailor your own

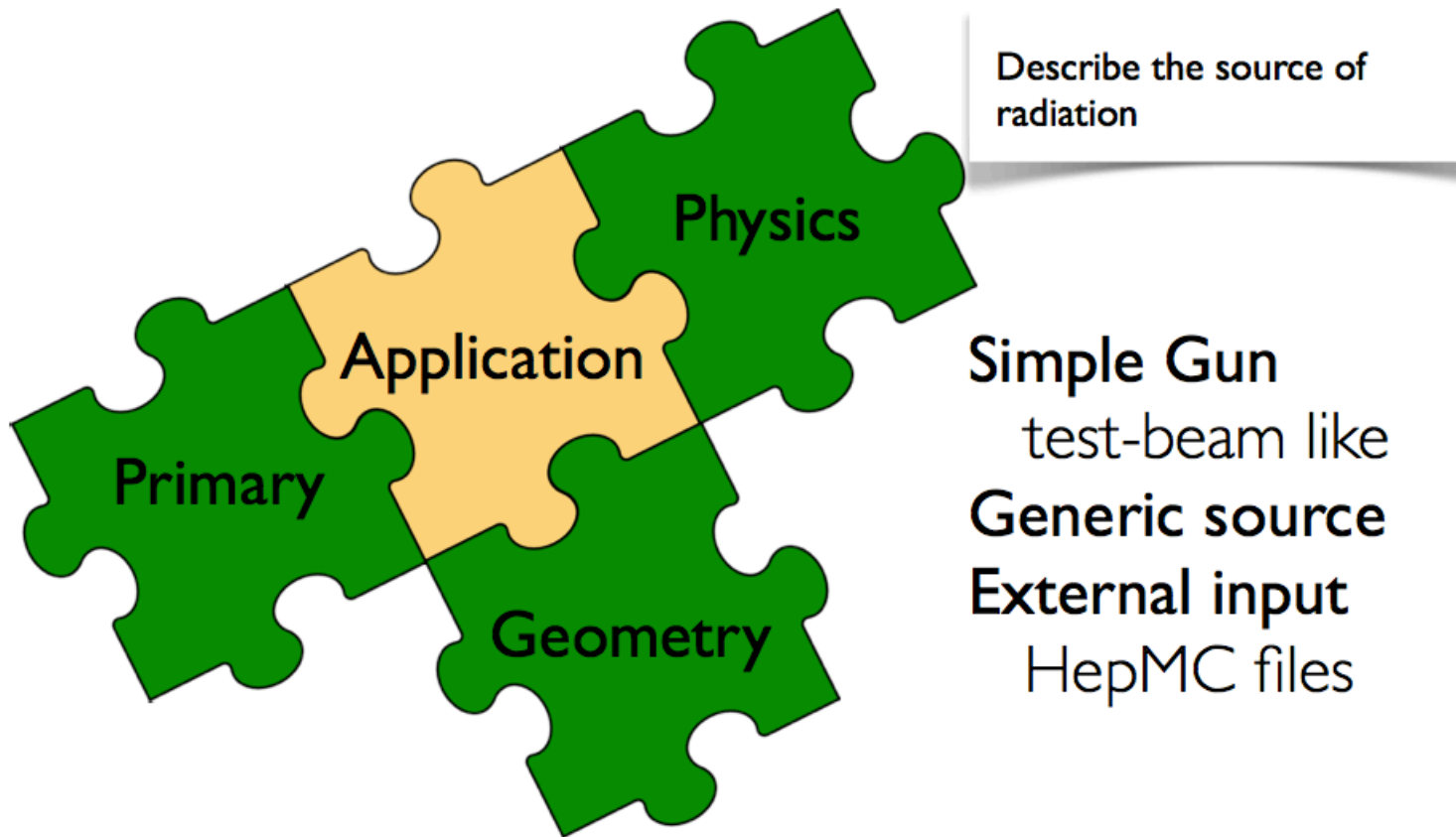
QGSP\_BERT  
FTFP\_BERT  
LHEP  
QGSP\_BIC  
CHIPS

....

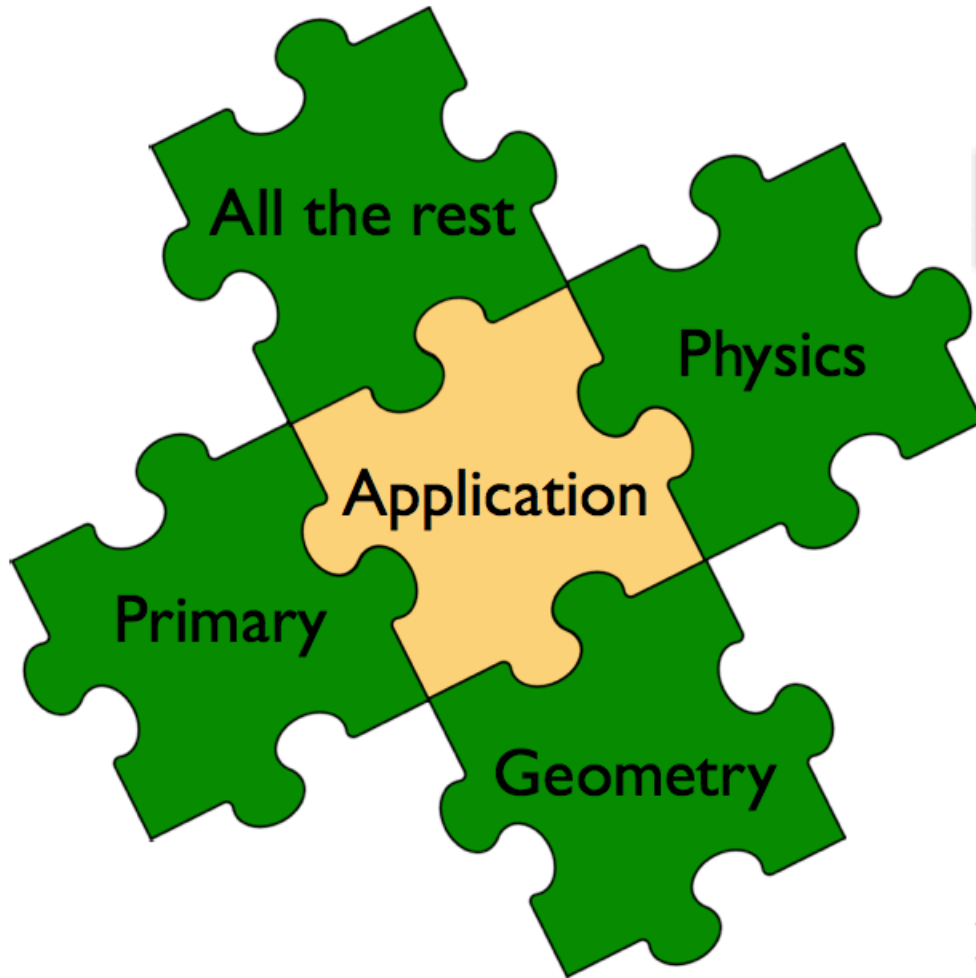
# A Geant4 application



# A Geant4 application



# A Geant4 application



Add all the rest

**G4UserActions**

interact with  
simulation

**G4Hits/Digits**

read-out

**Analysis**

**Visualization**

**Thanks for your attention**

# Installing CLHEP full version *(optional)*

- Create a directory for the installation procedure (ex.:clhep)

```
[geant4-tutorial] ~ >  
[geant4-tutorial] ~ >  
[geant4-tutorial] ~ >  
[geant4-tutorial] ~ >  
[geant4-tutorial] ~ > mkdir clhep  
[geant4-tutorial] ~ > cd clhep  
[geant4-tutorial] ~/clhep > █
```

- Move the downloaded tar-ball into this directory

```
[geant4-tutorial] ~/clhep >  
[geant4-tutorial] ~/clhep >  
[geant4-tutorial] ~/clhep >  
[geant4-tutorial] ~/clhep > mv ~/Desktop/clhep-2.0.3.2-src.tgz .  
[geant4-tutorial] ~/clhep > ls  
clhep-2.0.3.2-src.tgz  
[geant4-tutorial] ~/clhep > █
```

- Unzip the extract tar-ball into this directory

```
[geant4-tutorial] ~/clhep >  
[geant4-tutorial] ~/clhep >  
[geant4-tutorial] ~/clhep >  
[geant4-tutorial] ~/clhep > tar xzvf clhep-2.0.3.2-src.tgz  
2.0.3.2/  
2.0.3.2/CLHEP/  
2.0.3.2/CLHEP/CVS/  
2.0.3.2/CLHEP/CVS/Root  
2.0.3.2/CLHEP/CVS/Repository  
2.0.3.2/CLHEP/CVS/Entries  
2.0.3.2/CLHEP/CVS/Template  
2.0.3.2/CLHEP/CVS/Tag
```

- The extracted CLHEP package can be found in the subdirectory "2.0.3.2/CLHEP". Have a look at the content:

```
[geant4-tutorial] ~/clhep >
[geant4-tutorial] ~/clhep >
[geant4-tutorial] ~/clhep > ls
2.0.3.2 clhep-2.0.3.2-src.tgz
[geant4-tutorial] ~/clhep > ls 2.0.3.2/CLHEP
aclocal.m4      Evaluator      Matrix
autom4te.cache  Exceptions    missing
bootstrap      GenericFunctions Random
build-clheplib.in Geometry      RandomObjects
Cast           getObjectList.in README
ChangeLog      HepMC         ReadMe.cygwin-VC71
clhep-config.in HepPDT        RefCount
compilers.txt  INSTALL      setup.cygwin-VC71
config.guess   install-sh   StdHep
config.sub     makeBinaryTar.in Units
configure      Makefile.am  Utilities
configure.in   Makefile.in  Vector
CVS           makeSourceDist.in
```

*Have a look in the "INSTALL" file: It contains more details on the installation procedure*

- Create two directories (inside our "clhep" directory), which are used for building and installing the package:

```
[geant4-tutorial] ~/clhep >
[geant4-tutorial] ~/clhep > mkdir build
[geant4-tutorial] ~/clhep > mkdir install
[geant4-tutorial] ~/clhep > ls
2.0.3.2 build clhep-2.0.3.2-src.tgz install
[geant4-tutorial] ~/clhep > cd build
[geant4-tutorial] ~/clhep/build >
```

*NOTE: The package will be finally installed in the directory "~/clhep/install"*

- Inside the “build” directory, call the CLHEP configure script (which is contained in the “2.0.3.2/CLHEP” directory).

*NOTE: As argument you need to specify the directory, where CLHEP should be installed. Thus the full command to be called is: `../2.0.3.2/CLHEP/configure --prefix=/home/geant4-tutorial/clhep/install`*

```
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build > ../2.0.3.2/CLHEP/configure --prefix
x=/home/geant4-tutorial/clhep/install
checking build system type... i686-pc-linux-gnu
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for gawk... gawk
checking whether make sets $(MAKE)... yes
checking for a BSD-compatible install... /usr/bin/install -c
checking whether ln -s works... yes
checking for ranlib... ranlib
```

*Adapt prefix path according to your own installation directory!*

- The `configure` script checks for required programs and libraries, and creates some files, e.g. makefiles, and directories:

```
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build > ls
build-clheplib  Evaluator          makeBinaryTar      RandomObjects
Cast           Exceptions         Makefile           RefCount
clhep-config   GenericFunctions  makeSourceDist     Units
config.log     Geometry          Matrix             Vector
config.status  getObjectList     Random
[geant4-tutorial] ~/clhep/build > █
```

- If no error occurred in the configure process, one can start to build the CLHEP package using the “make” command:

```
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build > make
Making all in Units
make[1]: Entering directory `/home/geant4-tutorial/clhep/build/Units'
Making all in Units
make[2]: Entering directory `/home/geant4-tutorial/clhep/build/Units/Units'
make all-am
make[3]: Entering directory `/home/geant4-tutorial/clhep/build/Units/Units'
make[3]: Für das Ziel »all-am« ist nichts zu tun.
make[3]: Leaving directory `/home/geant4-tutorial/clhep/build/Units/Units'
make[2]: Leaving directory `/home/geant4-tutorial/clhep/build/Units/Units'
Making all in .
make[2]: Entering directory `/home/geant4-tutorial/clhep/build/Units'
/home/geant4-tutorial/clhep/2.0.3.2/CLHEP/Units/autotools/install-sh -d /home/
geant4-tutorial/clhep/build/Units/CLHEP;
make[3]: Entering directory `/home/geant4-tutorial/clhep/build/Units/Units'
install headers in /home/geant4-tutorial/clhep/build/Units/CLHEP/Units
make[3]: Leaving directory `/home/geant4-tutorial/clhep/build/Units/Units'
make[2]: Leaving directory `/home/geant4-tutorial/clhep/build/Units'
```

*This may take a while...*

*Only the initial and last output messages of the make command are shown*

```
liblist=`./getObjectList -static Units Vector Evaluator GenericFunct
ions Geometry Random Matrix RandomObjects RefCount Cast Exceptions`;
\
ar cru libCLHEP-2.0.3.2.a $liblist; ranlib libCLHEP-2.0.3.2.a
rm -f libCLHEP-2.0.3.2.so
liblist=`./getObjectList -shared Units Vector Evaluator Ge
ions Geometry Random Matrix RandomObjects RefCount Cast Ex
\
g++ -O -ansi -pedantic -Wall -D_GNU_SOURCE -g -O2 -o lib
3.2.so -shared -Wl,-soname,libCLHEP-2.0.3.2.so $liblist -o libCLHEP-
2.0.3.2.so
make[1]: Leaving directory `/home/geant4-tutorial/clhep/build'
[geant4-tutorial] ~/clhep/build > █
```

*Compiling was successful if  
“make” does not exit with error  
messages...*

- Once the package was compiled successfully, CLHEP can be installed using the “**make install**” command:

```
[geant4-tutorial] ~/clhep/build >
[geant4-tutorial] ~/clhep/build > make install
Making install in Units
make[1]: Entering directory `/home/geant4-tutorial/clhep/build/Units'
Making install in Units
make[2]: Entering directory `/home/geant4-tutorial/clhep/build/Units/Units'
make[3]: Entering directory `/home/geant4-tutorial/clhep/build/Units/Units'
make[3]: Für das Ziel »install-exec-am« ist nichts zu tun.
test -z "/home/geant4-tutorial/clhep/install/include/CLHEP/Units" || mkdir -p -- "/home/geant4-tutorial/clhep/install/include/CLHEP/Units"
/usr/bin/install -c -m 644 '../2.0.3.2/CLHEP/Units/Units/GlobalPhysicalConstants.h' '/home/geant4-tutorial/clhep/install/include/CLHEP/Units/GlobalPhysicalConstants.h'
/usr/bin/install -c -m 644 '../2.0.3.2/CLHEP/Units/Units/GlobalSystemOfUnits.h' '/home/geant4-tutorial/clhep/install/include/CLHEP/Units/GlobalSystemOfUnits.h'
/usr/bin/install -c -m 644 '../2.0.3.2/CLHEP/Units/Units/PhysicalConstants.h' '/home/geant4-tutorial/clhep/install/include/CLHEP/Units/PhysicalConstants.h'
```

- The CLHEP libraries are now installed in the directory “**~/clhep/install**”

(NOTE: We specified the installation directory in the configure process; see the previous slides)

```
[geant4-tutorial] ~/clhep/install >
[geant4-tutorial] ~/clhep/install >
[geant4-tutorial] ~/clhep/install >
[geant4-tutorial] ~/clhep/install >
[geant4-tutorial] ~/clhep/install > ls
bin  include  lib
[geant4-tutorial] ~/clhep/install > █
```

**Congratulations!**

- What do the subdirectories in “~/clhep/install” contain?
  - **include**: Contains (in a defined directory tree structure) the C++ header files of CLHEP
  - **lib**: Contains the (static and shared) CLHEP libraries
  - **bin**: Contains configure scripts
- Finally, to save some disk space, you can remove the “build” directory, as well as the tar-ball and the source package

```
[geant4-tutorial] ~/clhep > du -sh *  
27M      2.0.3.2  
93M      build  
4,9M     clhep-2.0.3.2-src.tgz  
53M      install  
[geant4-tutorial] ~/clhep > rm -r 2.0.3.2 build clhep-2.0.3.2-src.tgz  
[geant4-tutorial] ~/clhep > █
```