

Geant 4

Software Installation

<http://cern.ch/geant4>

Outline

- Supported platforms & compilers
- External software packages and tools
- Working area and installation area
- Toolkit installation
 - Using the *Configure* installation script
 - Configuring the environment manually
- Building an executable
- Integrating Geant4 in a software framework

Supported platforms & compilers

(release 9.3)

- MacOSX systems

- MacOS 10.6 and g++ gcc 4.2
 - G4SYSTEM: Darwin-g++



- Linux systems

- Scientific Linux SLC5, g++ gcc 4.1.2/4.3 or *icc-11.1*
 - G4SYSTEM: Linux-g++ or *Linux-icc*



- Windows systems

- Win/XP & Cygwin32, MSVC++ 9.0 .NET
 - G4SYSTEM: WIN32-VC



- Other systems, configured but no longer supported

- SUN-SunOS v.5.8, CC v.5.5
 - G4SYSTEM: SUN-CC



Required software

- A UNIX shell and related basic UNIX commands
- C++ compiler
 - Optional: FORTRAN compiler (*gfortran*) is required for building utility tools in the *g3tog4* module
- GNU Make
 - GNU g++ is required for dependencies pre-processing
- CLHEP library (<http://cern.ch/clhep>)
- Geant4 toolkit

External software packages - 1

Visualization/GUI tools (optional):

- X Windows
- OpenGL or MesaGL
- Qt graphics toolkit
- VRML browser
- DAWN (PostScript renderer)
 - DAVID (Geometry debugging tool based on DAWN)
- Open Inventor or HEP Inventor
 - requires OpenGL/MesaGL
- Open Scientist
 - interactive environment, including GUI
- Momo
 - Java-based GUI environment
 - GGE, GPE graphics editors
- HepRApp or WIRED4 JAS Plug-In
 - Uses the HepRep built-in graphics driver

External software packages - 2

XML parser for GDML (optional)

- Xerces-C++ (XML parser)

Module and Tools for analysis (optional)

- AIDA (Abstract Interfaces for Data Analysis)
 - JAS (Java Analysis Studio)
 - PI (Physicist Interfaces for AIDA Interactive Analysis)
 - Open Scientist (Interactive Analysis Environment)
 - Any other analysis tool compliant with AIDA interfaces ...

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
 - To allow multiple installations of the kernel and user code
- Working and Installation area can be the same
- Are controlled by two environment variables
 - **G4WORKDIR** and **G4INSTALL**

CONFIGURE SCRIPT FOR AUTOMATIC INSTALLATION

Using the **Configure** script for installation & configuration

- The **Configure** script guides through the whole installation process described so far by defining the proper environment and triggering the actual build of the libraries:

```
./Configure -build
```

- Once the environment has been configured and libraries built, **Configure** stores the current installation setup, and can then be used to install the libraries in the specified “installation area”:

```
./Configure -install
```

- The installation setup will become the default for the current installation, in case future changes to the installation are necessary
- Once the installation is complete, **Configure** can be used to generate shell scripts for configuring the user environment to build a Geant4 application according to the current installation:

```
./Configure
```

- Generates `env[.sh/.csh]` scripts in the user’s current directory
 - Scripts must be sourced each time a new shell/terminal is opened
 - It assumes the user specifies a working directory (`G4WORKDIR`)
 - In case not, the user’s home directory is set as default `G4WORKDIR` path

**ALTERNATIVE: MANUAL
APPROACH FOR INSTALLATION**

Configuring the environment:

the manual approach for installation

- Identify the system used for the installation
 - G4SYSTEM
- Identify the area of installation (i.e. path where the source code and the kernel libraries should be based)
 - G4INSTALL
 - Optionally, specify a different path for the kernel libraries and/or the temporary object files
 - G4LIB, G4TMP
 - Optionally, specify a different path for exporting of source header files
 - G4INCLUDE

Configuring the environment: the manual approach for installation

- Specify the path of installation for CLHEP
 - CLHEP_BASE_DIR
 - should point to the area where `include/` and `lib/` are placed from the standard CLHEP installation procedure
 - Paths can be customised: CLHEP_INCLUDE_DIR, CLHEP_LIB_DIR
 - the CLHEP library name is assumed to be: `[lib]CLHEP[.a/.lib/.dylib]`
 - A different name can be explicitly specified: CLHEP_LIB
- Specify the graphics/UI drivers to install
 - G4VIS_BUILD_<name>_DRIVER
 - G4UI_BUILD_<name>_DRIVER
 - the path to the related graphics/(G)UI packages, if required

Configuring the environment: the manual approach for installation

- Specify installation specific attributes
 - G4DEBUG
 - To build libraries including debug symbolic information
 - By default, optimised mode is selected
 - G4LIB_BUILD_SHARED
 - To specify if to build kernel libraries as shared libraries
 - Static archive libraries are built by default
 - Adding also G4LIB_BUILD_STATIC will build both
 - G4_NO_VERBOSE
 - For better performance, verbosity code can be left out by defining this flag (i.e. no verbosity will be possible). The default is with verbosity on

Starting the installation

- Choose the installation layout
 - Maximum granularity of libraries (*granular* libraries)
 - Ideal for developers and local installations
 - Link list of libraries automatically generated
 - Triggered with “make” from `$G4INSTALL/source`
 - Category compound libraries (*global* libraries)
 - Convenient for a centralized multi-users installation
 - Default for shared libraries builds
 - Triggered with “make global” from `$G4INSTALL/source`
- Installing source header files
 - `G4INCLUDE` defines the installation path
 - Triggered by “make includes” from `$G4INSTALL/source`
- Installing the hadronic physics-lists
 - Triggered by “make” from `$G4INSTALL/hadronic_lists/lists`
 - **NOTE:**
 - Can only be built as *static* libraries if *granular shared* libraries are used !

Configuring the environment to use Geant4

- Specify the working area: `G4WORKDIR`
 - If not, Geant4 assumes `G4INSTALL` as the working area
 - Products of application builds are placed in **`$G4WORKDIR`**
 - Binaries in **`$G4WORKDIR/bin`**
 - Object files and other temporary files in **`$G4WORKDIR/tmp`**
- Specify which graphics drivers, (G)UI drivers you want to use from the current installation
 - `G4VIS_USE_<name>`
 - `G4UI_USE_<name>`
- Specify the path where to retrieve data-files for specific simulations

Building an executable

- Configure the environment according to the current installation
 - Source or integrate the shell script generated by `Configure`
- Define the working area (`G4WORKDIR`)
- Build any of the available examples:

```
cp -r $G4INSTALL/examples $G4WORKDIR  
cd $G4WORKDIR/examples/novice/N01  
make
```

Building DLLs on Windows

- DLLs (Dynamic Link Libraries) on Windows can be built for global compound libraries only
 - Using the **Configure** script
 - Follow the steps till explicitly asked for
 - Or manually with “make dll” from `$G4INSTALL/source`
- Build any of the available examples by setting **G4LIB_USE_DLL** first in your environment
- Add to PATH the path where libraries are installed and run your application, e.g.:

```
export PATH=$PATH:/usr/local/geant4/lib/$G4SYSTEM
```

Integrating Geant4 in a framework

- Consider Geant4 as an external software package
- Well define its area of installation
 - For global libraries and source header files
- Choose an installation setup which best matches the project needs
- Adopt or integrate a configuration script reflecting the current installation