

# Geant 4

*Detector Description - Materials*

<http://geant4.cern.ch>

PART I

---

# Geant4 system of units

# System of Units (1)

- Geant4 has **no default unit**. To give a number, unit must be “multiplied” to the number.
  - for example:

```
G4double width = 12.5*m;  
G4double density = 2.7*g/cm3;
```
  - If no unit is specified, the *internal* G4 unit will be used, but **this is discouraged!**
  - Almost all commonly used units are available.
  - The user can define new units.
  - Refer to CLHEP: `SystemOfUnits.h`
- Divide a variable by a unit you want to get:

```
G4cout << dE/MeV << “ (MeV)” << G4endl;
```

# System of Units (2)

- System of units are **defined in CLHEP**, based on:
  - millimetre (**mm**), nanosecond (**ns**), Mega-eV (**MeV**), positron charge (**epplus**) degree Kelvin (**kelvin**), the amount of substance (**mole**), luminous intensity (**candela**), radian (**radian**), steradian (**steradian**)
- All other units are computed from the basic ones
- In output, Geant4 **can choose the most appropriate unit** to use. Just specify the *category* for the data (Length, Time, Energy, etc...):

```
G4cout << G4BestUnit(StepSize, "Length");
```

StepSize will be printed in km, m, mm or ... fermi, depending on its value

# System of Units (3)

---

- **New units** can be defined directly as constants, or (suggested way) via `G4UnitDefinition`
  - `G4UnitDefinition` ( name, symbol, category, value )
- Example (mass thickness):
  - `G4UnitDefinition` ("grammpercm2", "g/cm2", "MassThickness", g/cm2);
  - The new category "MassThickness" will be registered in the kernel in **G4UnitsTable**
- To print the list of units:
  - From the code  
`G4UnitDefinition::PrintUnitsTable();`
  - At run-time, as UI command:  
`Idle> /units/list`

## PART II

---

# Definition of materials and compounds

# Definition of Materials

---

- Different **kinds** of **materials** can be defined:
  - isotopes < > G4Isotope
  - elements < > G4Element
  - { ■ molecules < > G4Material
  - { ■ compounds and mixtures < > G4Material
- **Attributes** associated to materials:
  - **density** (mandatory), temperature, pressure, state (liquid, gas, etc.)

# Isotopes, Elements and Materials

---

- **G4Isotope** and **G4Element** describe the properties of the *nuclei* and *atoms*:
  - Atomic number, number of nucleons, mass of a mole, shell energies, etc...
  - Cross-sections per atoms, etc...
  - **Elements** are possibly **built up by isotopes**
- **G4Material** describes the ***macroscopic*** properties of the matter:
  - temperature, pressure, state, density
  - Radiation length, absorption length, etc...
  - **Materials** are **built up by elements**



# G4Elements (1)

---

- Isotopes can be assembled into elements

```
G4Isotope (const G4String& name,  
          G4int      z,      // number of atoms  
          G4int      n,      // number of nucleons  
          G4double   a );   // mass of mole
```

- ... building elements as follows:

```
G4Element (const G4String& name,  
          const G4String& symbol, // element symbol  
          G4int      nIso );     // # of isotopes  
G4Element::AddIsotope(G4Isotope* iso, // isotope  
                     G4double relAbund); // relative abundance
```

# G4Elements (2)

---

- Alternatively, if the element under consideration has the *natural isotopic abundance*, it can be defined using:

```
G4Element(const G4String& name, //its name const
          G4String& symbol, //its symbol
          G4double Zeff, //atomic number
          G4double Aeff); //mass of mole
```

- For instance:

```
G4Element* elOxygen =
    new G4Element("Oxygen", symbol="O", z=8., a=16*g/mole);
```

# G4Elements (3)

---

- Elements can also be defined using the **internal Geant4 database**:

```
#include "G4NistManager.hh"
[...]  
G4NistManager* man = G4NistManager::Instance();  
// define elements  
G4Element* elCarbon = man->FindOrBuildElement("C");
```

- To print information on a constituent element:

```
G4cout << elCarbon << G4endl;
```

# Material: single element

---

- **Single-element** materials can be defined in a quick way as follows:

```
G4double density = 1.390*g/cm3;
```

```
G4double a = 39.95*g/mole;
```

```
G4Material* lAr =
```

```
  new G4Material("liquidArgon", z=18., a, density);
```

- Materials composed by **many elements** (molecules or compounds) have to be defined through their **constituent elements**
- Prefer **low-density** material to vacuum

# Material: molecule

- A **Molecule** is made of several elements (composition by **number of atoms**):

define **elements**  
define **molecule**

```
a = 1.01*g/mole;  
G4Element* e1H =  
    new G4Element("Hydrogen", symbol="H", z=1., a);  
a = 16.00*g/mole;  
G4Element* e1O =  
    new G4Element("Oxygen", symbol="O", z=8., a);  
density = 1.000*g/cm3;  
G4Material* H2O =  
    new G4Material("Water", density, ncomp=2);  
H2O->AddElement(e1H, natoms=2);  
H2O->AddElement(e1O, natoms=1);
```

number of  
components


# Material: compound

- **Compound**, made by several elements:  
composition by **fraction of mass**

define **compound**    define **elements**

```
a = 14.01*g/mole;  
G4Element* elN =  
    new G4Element(name="Nitrogen",symbol="N",z= 7.,a);  
a = 16.00*g/mole;  
G4Element* elO =  
    new G4Element(name="Oxygen",symbol="O",z= 8.,a);  
density = 1.290*mg/cm3;  
G4Material* Air =  
    new G4Material(name="Air",density,ncomponents=2);  
Air->AddElement(elN, 70.0*perCent);  
Air->AddElement(elO, 30.0*perCent);
```

number of  
components



# Material: mixture

- Composition of **compound materials**

```
G4Element* elC = ...; // define "carbon" element
G4Material* SiO2 = ...; // define "quartz" material
G4Material* H2O = ...; // define "water" material

density = 0.200*g/cm3;
G4Material* Aerog =
    new G4Material("Aerogel",density,ncomponents=3);
Aerog->AddMaterial(SiO2,fractionmass=62.5*perCent);
Aerog->AddMaterial(H2O ,fractionmass=37.4*perCent);
Aerog->AddElement(elC ,fractionmass= 0.1*perCent);
```

- Mixtures could also be defined using their **elemental mass composition** (as described in the previous slide)

# Material: NIST database

- Materials can also be defined using the [internal Geant4 database](#), based on NIST.

```
#include "G4NistManager.hh"
G4NistManager* man = G4NistManager::Instance();
// define pure NIST materials
G4Material* Cu = man->FindOrBuildMaterial("G4_Cu");
// define NIST materials
G4Material* H2O = man->FindOrBuildMaterial("G4_WATER");
```

- The list of [available material names](#) is extended permanently. It can be accessed at [run-time](#):

```
Idle> /material/nist/listMaterials all
```

or [from the code](#)

```
G4NistManager::Instance()->ListMaterials("all");
```



## PART III

---

**A few examples  
and information**

# Example: define a gas

- It may be necessary to specify temperature and pressure (dE/dx computation affected). For density  $< 10 \text{ mg/cm}^3$  the material is automatically considered a gas (kStateGas)

```
G4double density = 27.*mg/cm3;
```

```
G4double temperature = 325.*kelvin;
```

```
G4double pressure = 50.*atmosphere;
```

```
G4Material* CO2 =
```

```
    new G4Material("CarbonicGas", density, ncomponents=2  
                  kStateGas, temperature, pressure);
```

```
CO2->AddElement(C,natoms = 1);
```

```
CO2->AddElement(O,natoms = 2);
```

# Example: define "vacuum"

- **Absolute vacuum** does **not exist**. It is a gas at **very low density** (e.g. transparent to particles being tracked)

**Cannot define materials with  $\rho = 0$**

```
G4double atomicNumber = 1.;
G4double massOfMole = 1.008*g/mole;
G4double density = 1.e-25*g/cm3; ←
G4double temperature = 2.73*kelvin;
G4double pressure = 3.e-18*pascal; ←
G4Material* Vacuum =
    new G4Material("interGalactic", atomicNumber,
                  massOfMole, density, kStateGas,
                  temperature, pressure);
```

# Print material information

---

- To print **information** on a given user-defined G4Material:

```
G4cout << Air << G4endl;
```

or at **run-time** via the UI command

```
Idle> /material/g4/printMaterial materialName
```

- To print the **full list** of user-defined materials:

```
G4cout << *(G4Material::GetMaterialTable()) << G4endl;
```