Material definition

Mihaly Novak (CERN, EP-SFT)

Geant4 Tutorial at Jefferson Lab

26 March 2024

- The Geant4 material model
- Material definition
- The NIST material data base
- Questions

Material definition

THE GEANT4 MATERIAL MODEL

- The Geant4 **material** model follows the natural one: materials are made of **element**s and elements are made of **isotope**s
- The 3 main classes to describe these objects are
  - *G4Isotope:* describes the properties of atoms (Z - atomic number, N - number of nucleons and A - molar mass) with unique `name` and `index`
  - *G4Element:* describes the properties of elements (Z - effective atomic number, N - effective number of nucleons and A - effective molar mass, number of isotopes, etc.) with unique `name,` `symbol` and `index`
  - *G4Material:* describes the macroscopic properties of matter (**density**, **state**, **temperature**, **pressure**, etc.) with unique `name` and `index`
- Unique index: a pointer to the created object is automatically stored in global table (isotope, element and material tables)

- The **material density** must be set (> zero) by the user at definition (except NIST materials) !
- The **material temperature** and **pressure** can optionally be set:
  - default: Normal Temperature and Pressure(NTP) 293.15 [k], 1 [atm] = 101.325 [kPascal]
- The **material state** can be **solid**, **liquid** or **gas**:
  - default is either solid or gas depending on the density (`kGasThreshold` = 10 [mg/cm^3])
  - non-crystalline (i.e. amorphous) solid by default (special extension for incorporating some information on the crystal structure)
- Special set of pre-defined materials: NIST material composition data base with some frequently used HEP materials
- **Geant4 material documentation: Material Documentation**

Material definition
MATERIAL DEFINITION

- Elements and isotopes:

  - *G4Element* object without specifying the isotope composition:

```
// simple way of Carbon element definition
G4Element* elC  = new G4Element(name="Carbon", symbol="C", z = 6., a = 12.01*g/mole);
```

   - need to give: `name`, `symbol`, `Z` and `A` (effective atomic number and molar mass)
   - isotopes will be automatically added with natural abundances (`A` won't be updated)

  - *G4Element* object by specific (non-natural) isotope composition:

```
// Define "enriched uranium" element as 90 % of U235 and 10 % of U 238:
//
// create the isotopes: iz = number of protons and n = number of nucleons
G4Isotope* U5 = new G4Isotope(name="U235", iz=92, n=235);
G4Isotope* U8 = new G4Isotope(name="U238", iz=92, n=238);
// create the element and build up by adding the isotopes with their abundance
G4Element* elU=new G4Element(name="enriched uranium",symbol="U",numisotopes=2);
elU->AddIsotope(U5, abundance= 90.*perCent);
elU->AddIsotope(U8, abundance= 10.*perCent);
```

   - element object must be created: `name`, `symbol`, `number of isotopes`
   - isotope objects must be created: `name`, `number of protons` and `nucleons`
   - isotopes need to be added by their relative abundance

- Simple *G4Material* object definition:

  - "simple": the material contains only one element and the corresponding *G4Element* object is not provided:

```
// single element "Uranium" material without giving the uranium element object
G4Material* matU = new G4Material(name    = "Uranium",
                                  z       = 92.0,
                                  a       = 238.03 * g/mole,
                                  denisty = 18.950 * g/cm3);
```

- the corresponding *G4Element* object will be automatically created (with natural isotope abundance)

- need to give: `name, density` of the material, `Z` and `A` (effective atomic number and molar mass) of the single *G4Element*

- what happens if we want the single element to have non-natural isotope abundance e.g. the previously created `enriched uranium` (see later)

- **G4Material** object definition as chemical molecule:
  - molecules build up from (several) elements with composition specified by the number of element (e.g. water = H_2O)
  - accordingly, **G4Material** object can be created by adding **G4Element** objects to it together with their composition number:

```cpp
// Create water material as molecule based on its chemical formula (H2O)
//
// create the necessary H and O elements (natural isotope abundance):
G4Element* elH  = new G4Element(name   = "Hydrogen",
                                symbol = "H",
                                z      =  1.00,
                                a      =  1.01 * g/mole);
G4Element* elO  = new G4Element(name   = "Oxygen",
                                symbol = "O",
                                z      =  8.00,
                                a      = 16.00 * g/mole);
// create the water material (name, density, number of components):
G4Material* matH2O = new G4Material(name        = "Water",
                                    density     = 1.0 * g/cm3,
                                    ncomponents = 2);
// add the elements to the material with their composition number
matH2O->AddElement(elH, numberOfatoms = 2);
matH2O->AddElement(elO, numberOfatoms = 1);
```

- *G4Material* object definition as mixture:

  - mixture of elements (*G4Element*), mixture of other materials (*G4Material*) or even mixture of elements and materials

  - similar to molecules with the differences:
    - components can be other materials not only elements
    - the ratio of the components must be given as "**fractional mass**" not as "**number of atoms**"

  - **mixture of elements** example: using the `AddElement` method

```
// Create air material as 70-30 % mixture of N and O elements
// (assuming that N and O Geant4 elements have already been created as
//   elN and elO)
//
// create the air material (name, density, number of components):
G4Material* matAir = new G4Material(name         = "Air",
                                    density       =  1.290 * mg/cm2,
                                    ncomponents =  2);
// add the elements to the material with their fractional mass
matAir->AddElement(elN, fractionmass = 0.7);
matAir->AddElement(elO, fractionmass = 0.3);
```

- *G4Material* object definition as mixture:

  - **mixture of element(s) and material(s)** example: using the `AddElement` and `AddMaterial` methods

```cpp
// Create aerogel material as 62.5 % silicon dioxide (SiO2), 37.4 % water (H2O)
// materials and 0.1 % carbon elemet. Assumimg that the materials (matSiO2 and
// matH2O) as well as the carbon elemnt (elC) have already been created.
//
// create the aerogel material (name, density, number of components):
G4Material* matAerog = new G4Material(name        = "Aerogel",
                                      density      =  0.2 * g/cm2,
                                      ncomponents  =  3);
// add the elements to the material with their fractional mass
matAerog->AddMaterial(matSiO2, fractionmass = 62.5 * perCent);
matAerog->AddMaterial(matH2O , fractionmass = 37.4 * perCent);
matAerog->AddElement (elC     , fractionmass =  0.1 * perCent);
```

Material definition

THE NIST MATERIAL DATA BASE

- The data base includes more than 3000 **isotopes**

- Isotopic composition of **elements** (Z = [1-108]) with their **natural isotopic abundance**: using the **NIST Atomic Weights and Isotopic Compositions** data base

- **NIST elements** can be obtained easily from the Geant4 NIST data base **by** using **their** `symbol` or `Z-atomic number`:

  - the corresponding *G4Isotope* objects will be automatically built

  - "find or build" i.e. avoids duplication of element objects

```
// get the carbon G4Element object from the NIST data base: by its symbol
G4Element* elC  = G4NistManager::Instance()->FindOrBuildElement("C");
// get the silicon G4Element object from the NIST data base: by its Z
G4Element* elSi = G4NistManager::Instance()->FindOrBuildElement(14);
```

- Large collection of **pre-defined materials**:

  - pre-defined: density, elemental composition (with the pre-defined natural isotopic composition), mean ionization energy, density effect parameters, etc.

- **Use these pre-defined materials whenever possible**:
  - guaranties high accuracy for many derived parameters (consistency)
- **NIST** and more pre-defined **materials** (318 at the moment)**:**
  - **single element NIST** materials with Z = [1-98] and named after the atomic symbol:
    - aluminum (`"G4_Al"`), silicon (`"G4_Si"`), gold (`"G4_Au"`), etc.
  - **compound NIST** materials:
    - `"G4_AIR"`, `"G4_ALUMINUM_OXIDE"`, `"G4_MUSCLE_SKELETAL_ICRP"`, etc.
  - **HEP** and nuclear materials:
    - liquid argon `"G4_lAr"`, lead tungstate `"G4_PbWO4"`, `"G4_STAINLESS-STEEL"`, etc.
  - **space** materials:
    - `"G4_KEVLAR"`, `"G4_NEOPRENE"`, etc.
  - **bio-chemical** materials:
    - the DNA bases `"G4_ADENINE"`, `"G4_GUANINE"`, `"G4_CYTOSINE"`, `"G4_THYMINE"`, etc.

- How to access these pre-define materials:
  - can be obtained from the Geant4 NIST data base **by** using **their name**
  - their name starts with the "`G4_`" **prefix** (see in the previous

```
// Use the NIST data base to get predefined materials: carbon, silicon
//
// get the simple pre-defined carbon material from the NIST data base
G4Material* matC  = G4NistManager::Instance()->FindOrBuildMaterial("G4_C");
// get the simple pre-defined silicon material from the NIST data base
G4Material* matSi = G4NistManager::Instance()->FindOrBuildMaterial("G4_Si");
```

```
// Use the NIST data base to get pre-defined materials:
//
// get the NIST manager (just to simplify)
G4NISTManager* nistMGR = G4NistManager::Instance();
// get the pre-defined liquid argon ("G4_lAr") from the NIST DB
G4Material* matLAr     = nistMGR->FindOrBuildMaterial("G4_lAr");
// get the pre-defined concerete ("G4_CONCRETE") from the NIST DB
G4Material* matConcr   = nistMGR->FindOrBuildMaterial("G4_CONCRETE");
```

- List available pre-define NIST elements, materials from the data base with their composition:

  - user interface command:
    - `/material/nist/printElement <SYMBOL>`
    - `/material/nist/listMaterials <CATEGORY>`

  - directly from C++ code:

```cpp
// List the pre-defined NIST ELEMENT(S) with its(their) isotope composition:
//
// element name can be: the element SYMBOL i.e. "Al" or "all"
const G4String nistElementName = "Al";
G4NistManager::Instance()->PrintElement(nistElementName);
//
// List the pre-defined NIST MATERIALS with their element composition:
//
// category name can be: "simple", "compound", "hep", "space", "bio", "all"
const G4String nistMatCategoryName = "simple";
G4NistManager::Instance()->ListMaterials(nistMatCategoryName);
```

Material definition
QUESTIONS