Version 11.3.2

Geant4 Tutorial Course

Jefferson Lab, Newport News, VA, USA

August 18-22 2025

Materials

Maurizio Ungaro

Jefferson Lab

Jefferson Lab
Thomas Jefferson National Accelerator Facility

U.S. DEPARTMENT OF ENERGY | Office of Science

JSA

Introduction

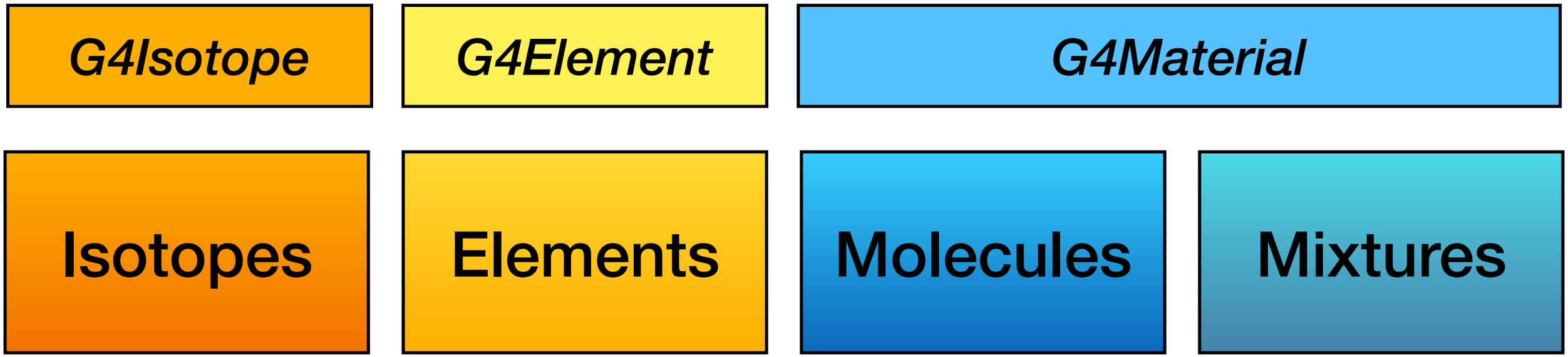
- **Materials in Geant4**
- **Defining a Material**
  - With elements
  - With molecular composition
  - With mixtures
  - With Isotopes
- **The NIST Material Database**

Material

describes the properties of atoms with unique name

describes the properties of elements with unique name, symbol

describes the macroscopic properties of matter with unique name



Note: a Global Material Table stores all pointers to these objects (unique indexes)

- Can be **solid**, **liquid** or **gaseous**
- Exist under various temperatures and pressure states

Material

- Mandatory parameter : **density**. Defined with **units**, typically `g/cm3`
- Optional Parameters:
  - State (`kStateSolid, kStateLiquid, kStateGas`), by default is solid or liquid *depending on the density*
  - Temperature, by default temperature is `T = 273.15 K`
  - Pressure, by default pressure is `100 kPascal ≃ 1 atm`

Single Element Material

Material Made of a single element (Fe, Pb, …)

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"

G4VPhysicalVolume* DetectorConstruction::Construct(){

    auto* pureIron = new G4Material("pureIron", 26, 55.845*g/mole, 7.87*g/cm3);

  }
```

iron atomic number (z) *iron atomic weight* *iron density g/cm3*

Single Element Material

Material Made of a single element (Fe, Pb, …)

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"

G4VPhysicalVolume* DetectorConstruction::Construct(){

    auto* pureIron = new G4Material("pureIron", 26, 55.845*g/mole, 7.87*g/cm3);

  }
```

iron atomic number (z) *iron atomic weight* *iron density g/cm3*

Notice the G4Element is NOT PROVIDED.

The corresponding G4Element object will be automatically created (with natural isotope abundance)

Single Element Material

Material Made of a single element (Fe, Pb, …)

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"

G4VPhysicalVolume* DetectorConstruction::Construct(){

     G4int z ;
     G4double d ;
     G4double a ;

     auto* pureIron = new G4Material("pureIron",
                                     z = 26,
                                     a = 55.845*g/mole,
                                     d = 7.87*g/cm3);
  }
```

Single Element Material

Material Made of a single element (Fe, Pb, …)

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"

G4VPhysicalVolume* DetectorConstruction::Construct(){

    G4int atomic_number = 26;                // iron atomic number (z)
    G4double density = 7.87*g/cm3;           // iron density g/cm3
    G4double atomic_weight = 55.845*g/mole;  // iron atomic weight 55.845

    auto* pureIron = new G4Material("pureIron", atomic_number, atomic_weight, density);

  }
```

Single Element Material

Material Made of a single element (Fe, Pb, …)

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"

G4VPhysicalVolume* DetectorConstruction::Construct(){

     G4int atomic_number = 26;                // iron atomic number (z)
     G4double density = 7.87*g/cm3;           // iron density g/cm3
     G4double atomic_weight = 55.845*g/mole;  // iron atomic weight 55.845

     auto* pureIron = new G4Material("pureIron", atomic_number, atomic_weight, density);
```

Public Member Functions

	G4Material (const G4String &name, G4double z, G4double a, G4double density, G4State state=kStateUndefined, G4double temp=NTP_Temperature, G4double pressure=CLHEP::STP_Pressure)
	G4Material (const G4String &name, G4double density, G4int nComponents, G4State state=kStateUndefined, G4double temp=NTP_Temperature, G4double pressure=CLHEP::STP_Pressure)
	G4Material (const G4String &name, G4double density, const G4Material *baseMaterial, G4State state=kStateUndefined, G4double temp=NTP_Temperature, G4double pressure=CLHEP::STP_Pressure)
void	**SetChemicalFormula** (const G4String &chF)
void	**SetFreeElectronDensity** (G4double val)
void	**ComputeDensityEffectOnFly** (G4bool val)
	G4Material (const G4Material &)=delete
const G4Material &	**operator=** (const G4Material &)=delete
void	**AddElementByNumberOfAtoms** (const G4Element *elm, G4int nAtoms)
void	**AddElement** (G4Element *elm, G4int nAtoms)
void	**AddElementByMassFraction** (const G4Element *elm, G4double fraction)
void	**AddElement** (G4Element *elm, G4double frac)
void	**AddMaterial** (G4Material *material, G4double fraction)
const G4String &	**GetName** () const
const G4String &	**GetChemicalFormula** () const
G4double	**GetFreeElectronDensity** () const
G4double	**GetDensity** () const
G4State	**GetState** () const
G4double	**GetTemperature** () const
G4double	**GetPressure** () const

Single Element Material

Material Made of a single element (Fe, Pb, …)

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"

G4VPhysicalVolume* DetectorConstruction::Construct(){

     G4int atomic_number = 26;                 // iron atomic number (z)
     G4double density = 7.87*g/cm3;            // iron density g/cm3
     G4double atomic_weight = 55.845*g/mole;   // iron atomic weight 55.845

     auto* pureIron = new G4Material("pureIron", atomic_number, atomic_weight, density);

     G4cout << pureIron->GetName()<<"->"<<'\n'<<
            "density: "<<pureIron->GetDensity()/(g/cm3)<<" g/cm3"<<'\n'<<
            "number of elements: "<<pureIron->GetNumberOfElements()<<'\n'<<
            "radiation length: "<<pureIron->GetRadlen()<<" mm"<<'\n'<<
            "nucl. int. length: "<<pureIron->GetNuclearInterLength()<<" mm"<<'\n'<<
            "e- per volume: "<<pureIron->GetTotNbOfElectPerVolume()<<" e-/mm3”<< G4endl;

}
```

Bash - execution

```
pureIron->
density: 7.87 g/cm3
number of elements: 1
radiation length: 17.5839 mm
nucl. int. length: 169.989 mm
e- per volume: 2.20655e+21 e-/mm3
```

Material: Molecule

A **molecule** is defined by (**N**>1) **elements** and its composition is specified with the AddElement method

Here we create new elements before adding them to the material

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"
#include "G4Element.hh"

G4VPhysicalVolume* DetectorConstruction::Construct() {

     G4double a   = 1.01 * g / mole;                          // Hydrogen(element): A
     G4int    z   = 1;                                        // Hydrogen(element): Z

     auto* elH = new G4Element(“ Hydrogen” , "H", z, a); // Hydrogen(element)

     a = 16.00 * g / mole; // Oxygen(element): A
     z = 8;                // Oxygen(element): Z

     auto* elO = new G4Element("Oxygen", "O", z, a);   // Oxygen (element)


     G4int       ncomp         = 2;              // water components
     G4double    water_density = 1.0 * g / cm3; // water density
     G4int nAtoms;

     auto* H2O = new G4Material(“Water”, water_density, ncomp); // water material

     H2O->AddElement(elH, nAtoms = 2); // add element by number of atoms
     H2O->AddElement(elO, nAtoms = 1); // add element by number of atoms

}
```

Material: Mixture

Similarly to combining elements in molecules, it is possible to combine materials and elements in mixtures

The percentages here refer to the **MASS FRACTION**

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"
#include "G4Element.hh"


G4VPhysicalVolume* DetectorConstruction::Construct() {

    // Simplified Air, mass fraction: 70% Nitrogen, 30% Oxygen

    G4int    z;
    G4double a;
    G4double density;
    G4int    ncomponents;
    G4String name;
    G4String symbol;
    G4double fracMass;

    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, fracMass = 70.0 * perCent); //add element by frac mass
    Air->AddElement(elO, fracMass = 30.0 * perCent); //add element by frac mass
}
```

Material: Mixture

Similarly to combining elements in molecules, it is possible to combine materials and elements in mixtures

Note that we are combining **AddElement** with **AddMaterial**

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"
#include "G4Element.hh"


G4VPhysicalVolume* DetectorConstruction::Construct() {

     // Simplified Aerogel material, frac mass: 62.5% SiO2, 37.4% H20
     […]

     G4Element* elC = ... ;   // define carbon element
     G4Material* H2O = ... ;  // define water molecule (previously done)
     G4Material* SiO2 = ... ; // define SiO2 molecule (left as exercise)

     density         = 0.20*g/cm3;
     G4Material* Aerog = new G4Material("Aerogel", density, ncomp); // Aereogel material
     Aerog->AddMaterial(SiO2, fracMass = 62.5*perCent); // Note that we are combining elements with materials
     Aerog->AddMaterial(H2O, fracMass = 37.4*perCent);  // Note that we are combining elements with materials
     Aerog->AddElement(elC, fracMass = 0.1*perCent);    // Note that we are combining elements with materials

}
```

Isotopes

- By default any G4Element is treated according to its **natural** isotopic abundance regardless of the atomic weight specified
- Hadronic processes only work with specific isotopes (i.e. exact exact A and Z.)*

What is the natural isotopic abundance of an element?

```
#include "G4Material.hh"
#include "G4SystemOfUnits.hh"

G4VPhysicalVolume* DetectorConstruction::Construct(){
   […]

   auto* pureIron = new G4Material("pureIron", atomic_number, atomic_weight, density);
   auto element = pureIron->GetElement(0);

   G4cout<< pureIron->GetName() << " # isotopes: “ << element->GetNumberOfIsotopes() << G4endl;

}
```

Bash - execution

```
pureIron # isotopes: 4
```

Iron has 4 naturally abundant isotopes: $^{54}Fe, ^{56}Fe, ^{57}Fe, ^{58}Fe$

* Note: in G4NISTManager, all the known isotopes are defined.

Isotopes

It is possible to create an element with non natural isotopic abundance assigning to it a list of G4Isotopes

Example: define “enriched Uranium” element as 10% of U235 and 90% U238

```
G4VPhysicalVolume* DetectorConstruction::Construct() {

    G4int      z_iso, a_iso; // Create uranium 235 and 238. Need to give Z and A

    auto* elF  = new G4Element("Fluorine", "F", 9, 18.998 * g / mole);
    auto* u235 = new G4Isotope("U235", z_iso = 92, a_iso = 235., 235.044 * g / mole);
    auto* u238 = new G4Isotope("U238", z_iso = 92, a_iso = 238., 238.051 * g / mole);

    G4int      u_comp; // Create enriched uranium
    G4double   u_abundance;

    auto* enrichedU = new G4Element("enrichedU", "eU", u_comp = 2);

    // Create the element by adding isotopes with their abundance
    enrichedU->AddIsotope(u235, u_abundance = 10.0 * perCent);  // add isotope to enrichedU
    enrichedU->AddIsotope(u238, u_abundance = 90.0 * perCent);  // add isotope to enrichedU

    auto* fuel = new G4Material("NuclearFuel",
                                5.09 * g / cm3,
                                ncomp = 2, // mandatory arguments
                               kStateGas,
                                640 * kelvin,
                                1.5e7 * pascal); // optional arguments
    // create fuel molecule
    fuel->AddElement(elF, 6);
    fuel->AddElement(enrichedU, 1);
}
```

The NIST material database

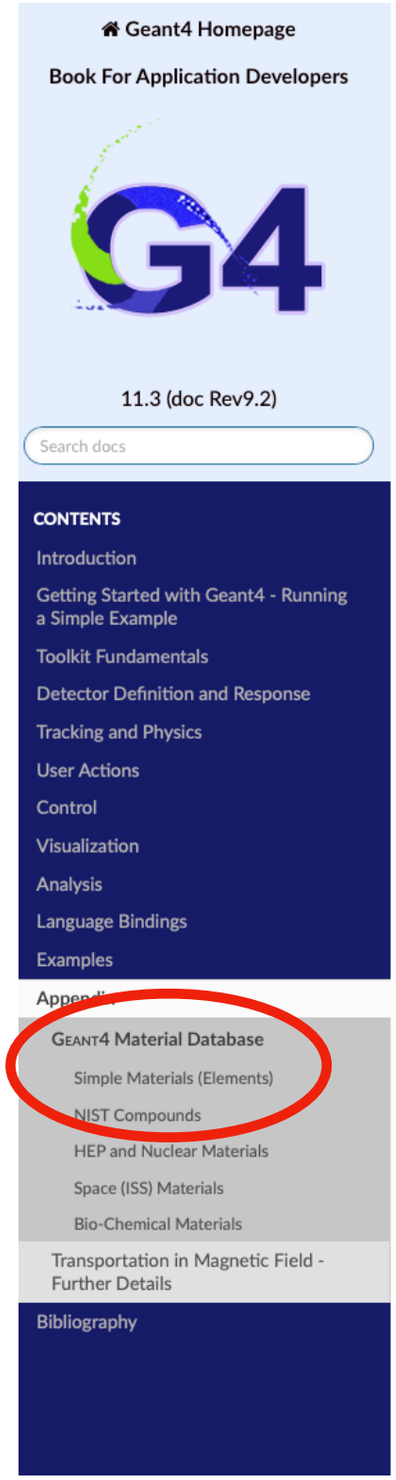
The National Institute of Standards and Technology has created a materials science data repository as part of an effort in coordination with the Materials Genome Initiative (MGI)

- The NIST material database includes
  - most of the materials useful in simulations (biomedical, space-science, …)
  - all elements with natural isotopic abundance
  - > 3,000 isotopes
- Using materials from the NIST database guarantees the best parameters for:
  - density
  - isotopic composition of elements
  - element composition of materials
  - mean ionization potential
  - chemical bonds

The NIST material database

The National Institute of Standards and Technology has created a materials science data repository as part of an effort in coordination with the Materials Genome Initiative (MGI)

- The NIST material database includes
  - most of the materials useful in simulations (biomedical, space-science, …)
  - all elements with natural isotopic abundance
  - > 3,000 isotopes
- Using materials from the NIST database guarantees the best parameters for:
  - density
  - isotopic composition of elements
  - element composition of materials
  - mean ionization potential
  - chemical bonds



Docs » Appendix » GEANT4 Material Database

GEANT4 Material Database

Simple Materials (Elements)

Z	Name	density(g/cm^3)	I(eV)
1	G4_H	8.3748e-05	19.2
2	G4_He	0.000166322	41.8
3	G4_Li	0.534	40
4	G4_Be	1.848	63.7
5	G4_B	2.37	76
6	G4_C	2	81
7	G4_N	0.0011652	82
8	G4_O	0.00133151	95
9	G4_F	0.00158029	115
10	G4_Ne	0.000838505	137
11	G4_Na	0.971	149
12	G4_Mg	1.74	156
13	G4_Al	2.699	166
14	G4_Si	2.33	173
15	G4_P	2.2	173
16	G4_S	2	180
17	G4_Cl	0.00299473	174
18	G4_Ar	0.00166201	188
19	G4_K	0.862	190
20	G4_Ca	1.55	191
21	G4_Sc	2.989	216
22	G4_Ti	4.54	233
23	G4_V	6.11	245
24	G4_Cr	7.18	257
25	G4_Mn	7.44	272

The NIST material database

The National Institute of Standards and Technology has created a materials science data repository as part of an effort in coordination with the Materials Genome Initiative (MGI)

- The NIST material database includes
  - most of the materials useful in simulations (biomedical, space-science, …)
  - all elements with natural isotopic abundance
  - > 3,000 isotopes
- Using materials from the NIST database guarantees the best parameters for:
  - density
  - isotopic composition of elements
  - element composition of materials
  - mean ionization potential
  - chemical bonds

Example: Print list of NIST materials

```
/material/nist/listMaterials   # UI command
```

Bash - execution

```
========================================================

###   Simple Materials from the NIST Data Base      ###

========================================================

 Z   Name   density(g/cm^3)  I(eV)

========================================================

 1   G4_H    8.3748e-05       19.2
 2  G4_He   0.000166322       41.8
 3  G4_Li         0.534         40
 4  G4_Be         1.848       63.7
 5   G4_B          2.37         76
 6   G4_C             2         81
 7   G4_N     0.0011652         82
 8   G4_O    0.00133151         95
 9   G4_F    0.00158029        115
10  G4_Ne   0.000838505        137
11  G4_Na         0.971        149
12  G4_Mg          1.74        156
13  G4_Al         2.699        166
14  G4_Si          2.33        173
15   G4_P           2.2        173
16   G4_S             2        180
17  G4_Cl    0.00299473        174
```

their name starts with the “**G4_**” **prefix**

The NIST material database

The National Institute of Standards and Technology has created a materials science data repository as part of an effort in coordination with the Materials Genome Initiative (MGI)

- The NIST material database includes
  - most of the materials useful in simulations (biomedical, space-science, …)
  - all elements with natural isotopic abundance
  - > 3,000 isotopes
- Using materials from the NIST database guarantees the best parameters for:
  - density
  - isotopic composition of elements
  - element composition of materials
  - mean ionization potential
  - chemical bonds

Example: Print list of NIST elements

`/material/nist/printElement`

Bash:

```
Nist Element: <H>  Z= 1  Aeff(amu)= 1.00794  6
isotopes:

            N: 1  2  3  4  5  6

          mass(amu): 938.783 1876.12 2809.76 3751.9
4694.3 5630.83

     abundance: 0.999885 0.000115 0 0 0 0
```

Directly from C++ Code:

```
G4VPhysicalVolume* DetectorConstruction::Construct(){

    auto* nist = G4NistManager::Instance();

    nist->PrintElement("Al")
    nist->ListMaterials("simple")

}
```

G4NistManager

Geant4 provides a (singleton) class, the **G4NistManager**, to get information and **objects** from the NIST database

DetectorConstruction.cc - Get element and material information directly from NIST database

```
G4VPhysicalVolume* DetectorConstruction::Construct(){

    auto* nist = G4NistManager::Instance();

    G4cout << "Natural uranium mass: “    << nist->GetAtomicMassAmu(92)<<" amu"<<'\n'<<
              "Hydrogen Nb of isotopes: “ << nist->GetNumberOfNistIsotopes(1) <<'\n'<<
              "Iron material density: “   << nist->GetNominalDensity(26)/(g/cm3) << " g/cm3” << G4endl;

    // Or print information per material/elements using
    nist->PrintElement(“Al");    // equivalent to UI command /material/nist/printElement Al
}
```

Bash - execution

```
Natural uranium mass: 238.029 amu

Hydrogen Nb of isotopes: 6

Iron material density: 7.874 g/cm3
```

Bash - execution

```
Nist Element: <Al>  Z= 13  Aeff(amu)= 26.9815  19 isotopes:

            N: 21  22  23  24  25  26  27  28  29  30  31  32  33
34  35  36  37  38  39

         mass(amu): 19587.5 20511 21431.2 22355.8 23278.4 24206.6
25133.1 26065 26995.1 27929 28861.3 29796.7 30730.8 31667.9
32602.2 33539.7 34474.9 35412.5 36348.7

    abundance: 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
```

G4NistManager

Geant4 provides a (singleton) class, the **G4NistManager**, to get information and **objects** from the NIST database

DetectorConstruction.cc - Get element and material objects from NIST database

```
G4VPhysicalVolume* DetectorConstruction::Construct() {

    // Get nist manager (singleton)
    auto* nist = G4NistManager::Instance();

    G4Element* elFe = nist->FindOrBuildElement(26);

    G4Material* fe    = nist->FindOrBuildMaterial("G4_Fe");

    G4Material* concrete    = nist->FindOrBuildMaterial("G4_CONCRETE");

}
```

The NIST material database

- Use these pre-defined materials whenever possible
  - guarantees high accuracy and consistency for many derived parameters
- 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.

Summary

- In Geant4, materials (**G4Material**) are defined in terms of elements (G4Element) and isotopes (G4Isotopes)
  - They can be made of single-elements
  - They can be mixtures of elements (forming molecules) and materials (by mass fraction)
- Every material exists under a certain **density** condition (**mandatory**) as well as *state, temperature and pressure* conditions (*optional*)
- Whenever possible, it is recommended to use the **G4NistManager** to retrieve both elements and materials from the NIST Material Database

GEANT4 Material Database

Simple Materials (Elements)

Z	Name	density(g/cm^3)	I(eV)
1	G4_H	8.3748e-05	19.2
2	G4_He	0.000166322	41.8
3	G4_Li	0.534	40
4	G4_Be	1.848	63.7
5	G4_B	2.37	76
6	G4_C	2	81
7	G4_N	0.0011652	82
8	G4_O	0.00133151	95
9	G4_F	0.00158029	115
10	G4_Ne	0.000838505	137
11	G4_Na	0.971	149