Experimental Techniques in Particle Physics

Geant4: Geometry Basics A

Andreas Nowack
nowack@physik.rwth-aachen.de
RWTH Aachen University

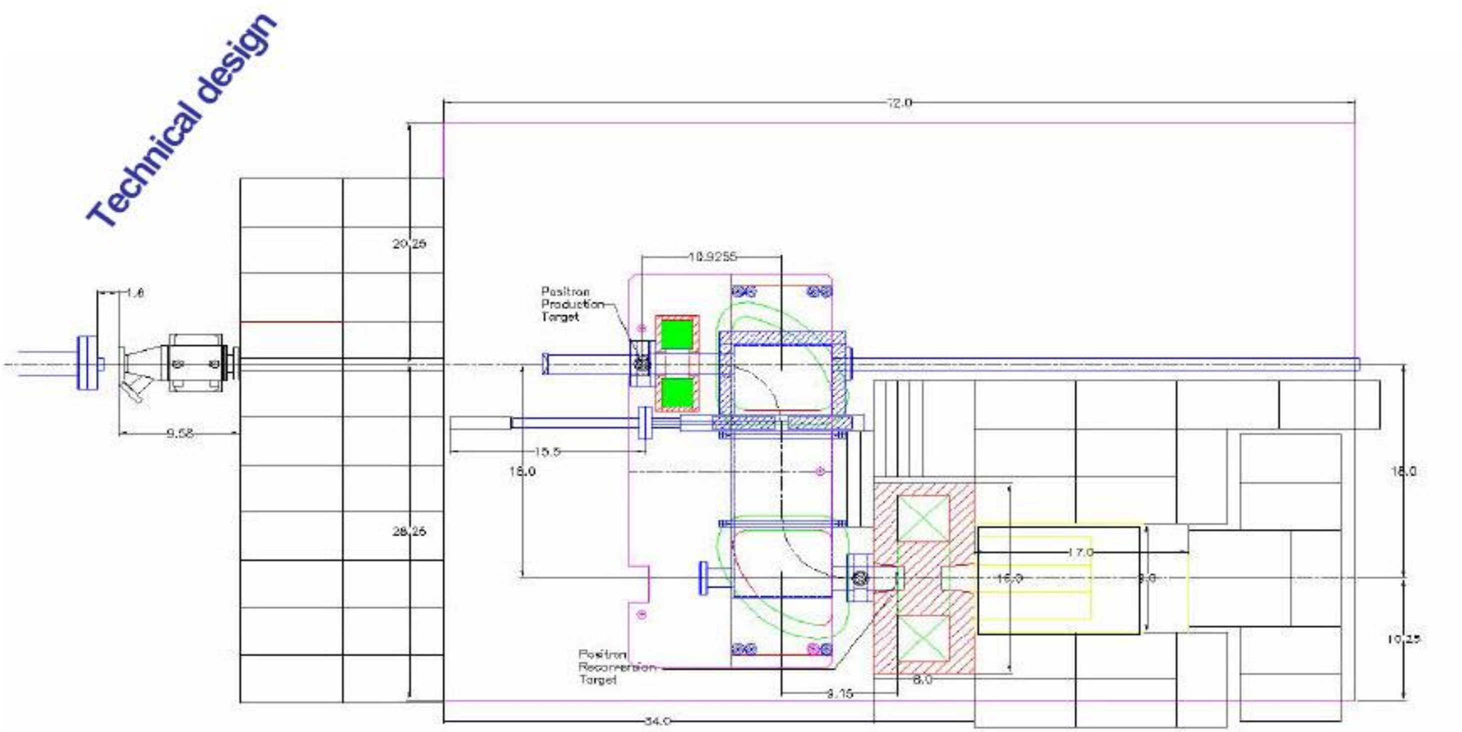
WS 2020/21

Quick Intro to Geant 4

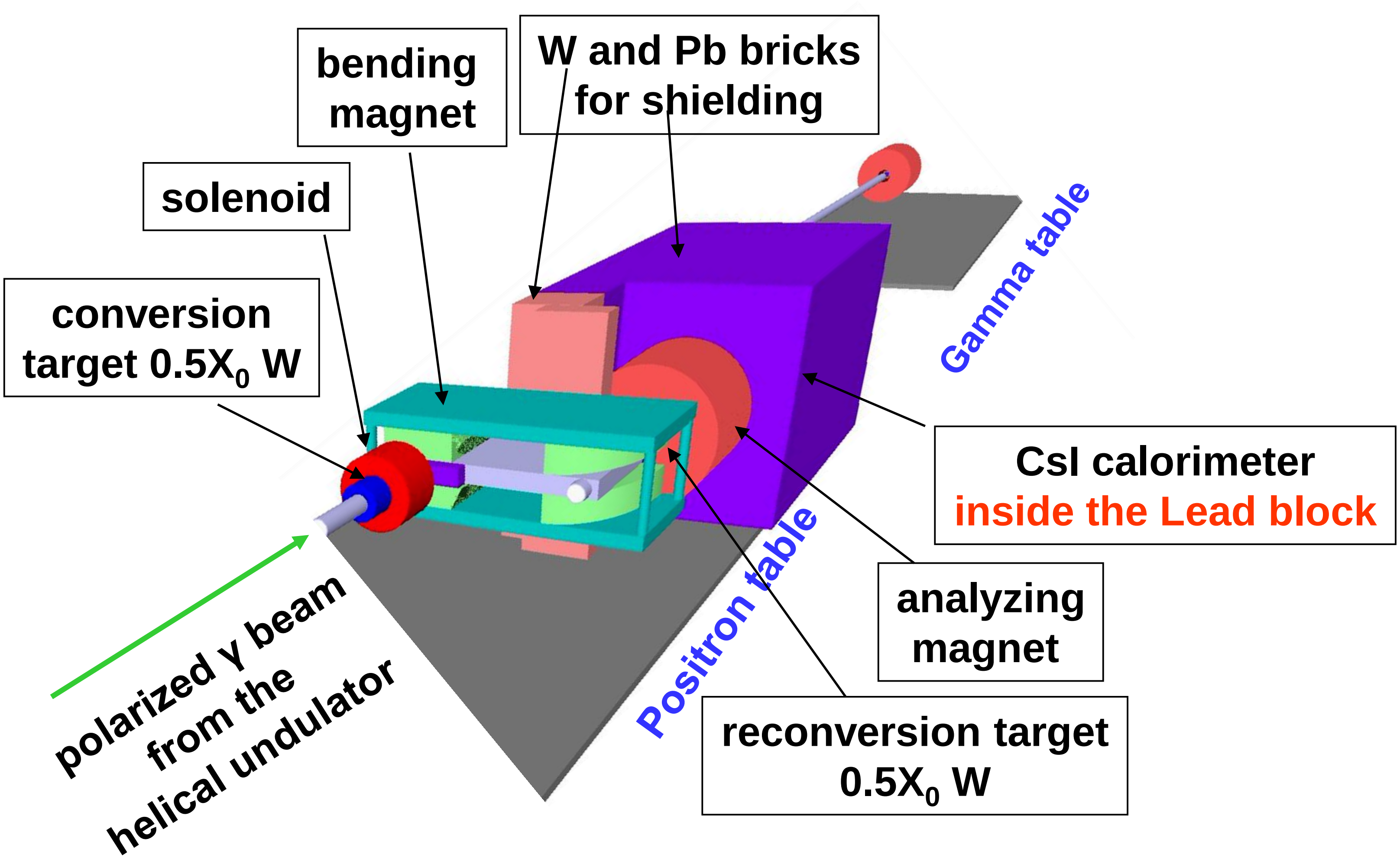
Example of an Experimental Setup

50 GeV e^- E - 166 10 MeV γ e^+ e^-
International Polarized Positron Collaboration

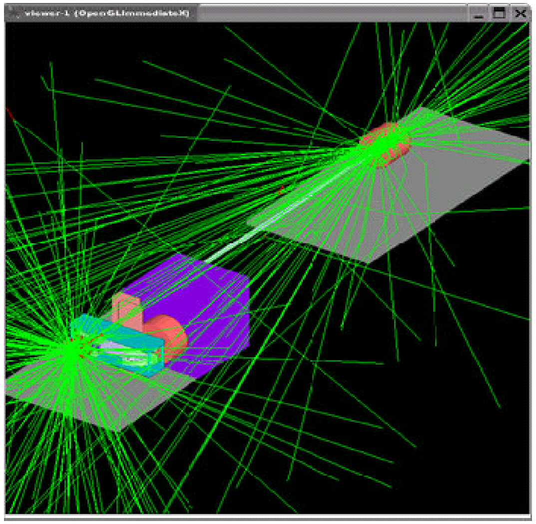
E166 Experimental setup
(Positron table at SLAC)

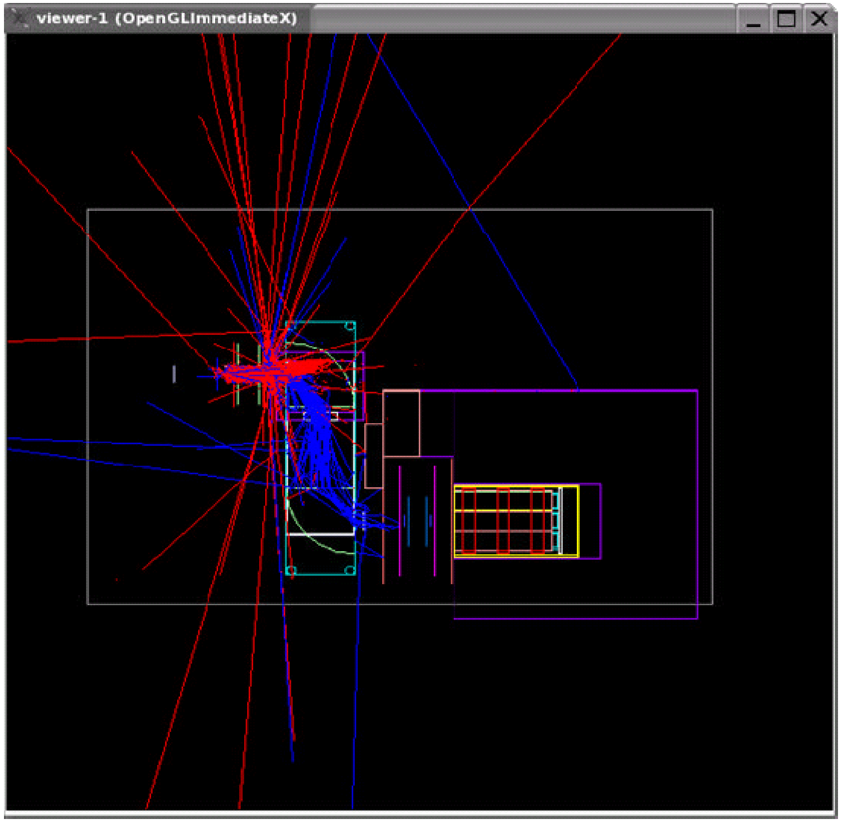


Example of an Experimental Setup.

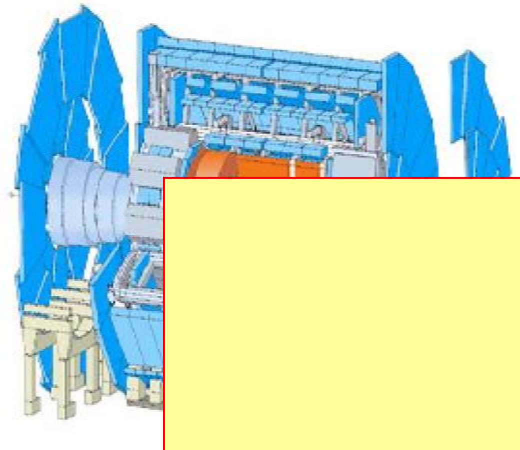


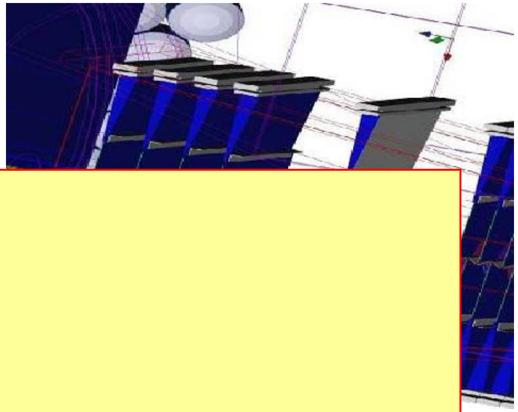
Example of an Experimental Setup. Test Run





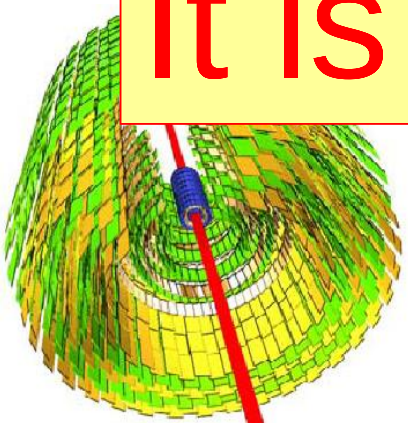
Geometry. Is it complicated?

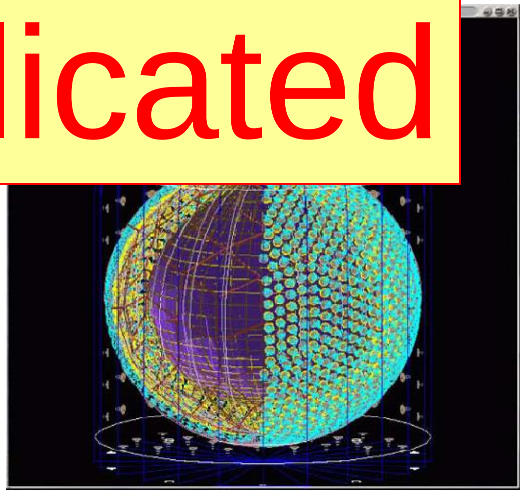




No !!

It is not complicated





How to Start with the Geometry?

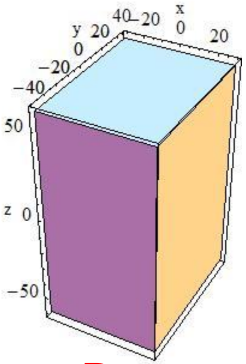
We are lucky: We have already

predefined

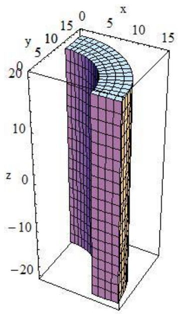
CSG Solids (geometries)
(Constructed Solid Geometry)

You just need to get used to its concept and syntax!
Then it’s easy!

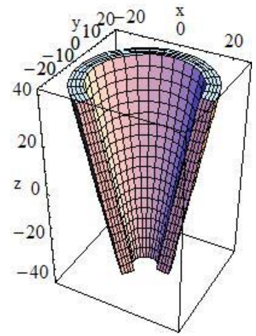
How Many Predefined CSG Do We Have?



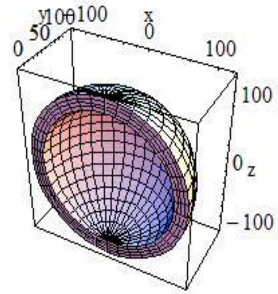
Box



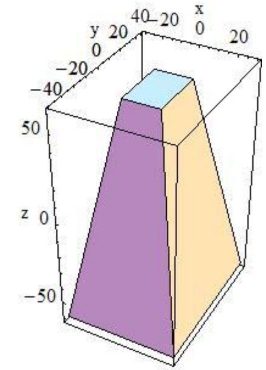
Tube



Cone

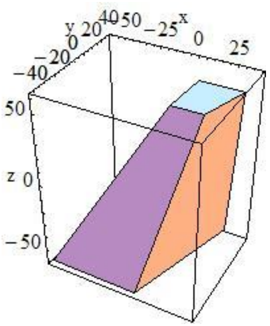


Sphere

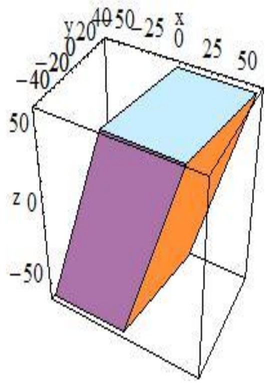


Trapezoid

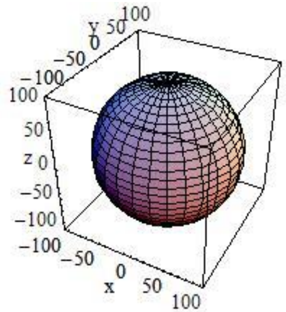
My preferred ones… very basic and simple… see exercise today



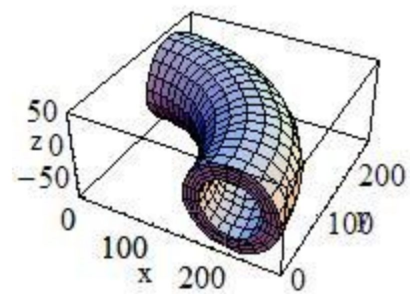
Generic Trapezoid



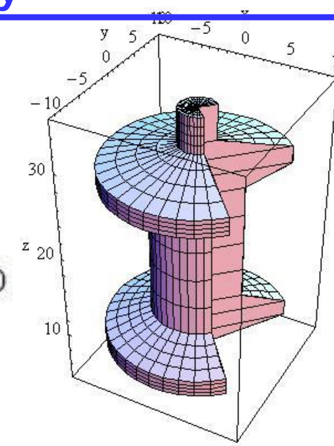
Parallelepiped



Solid Sphere

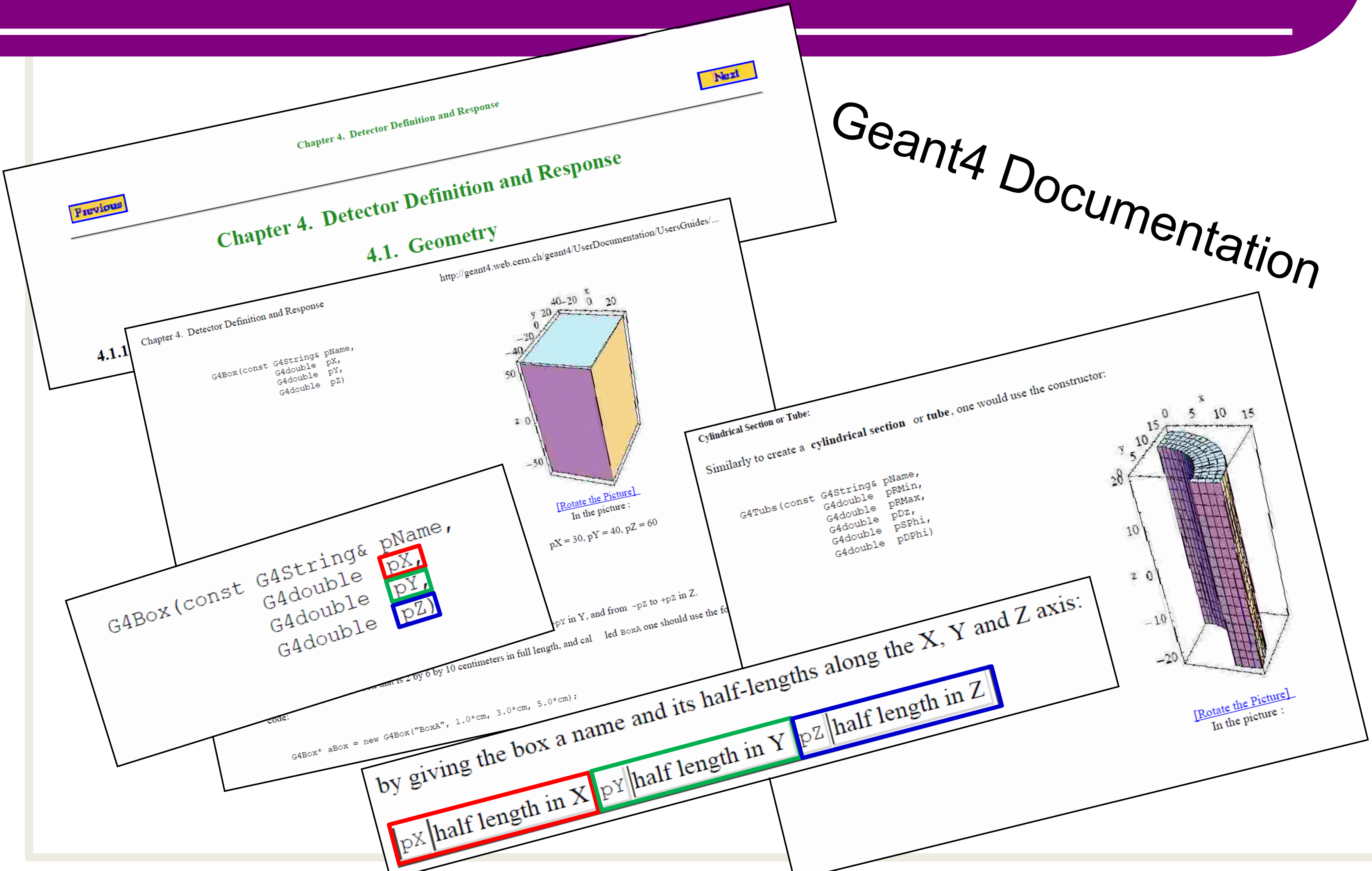


Torus



Polycons

Where to Find the CSG (Predefined Geometry)



Notion of World, Mother and Daughter Volumes

World volume (mandatory)

➡ **world volume is the mother volume of Volume 1 and 2**

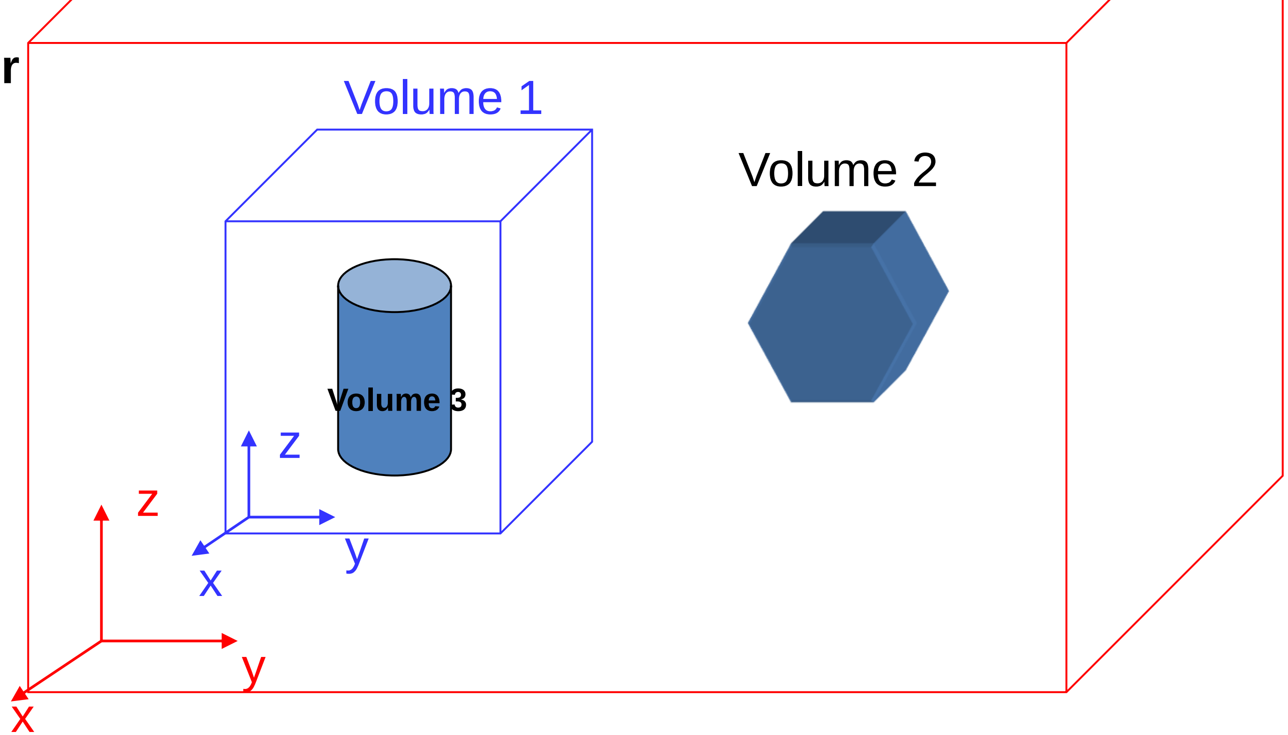
Volume 1 and 2 are daughters of world volume

Volume 1 is mother of volume 3

etc…

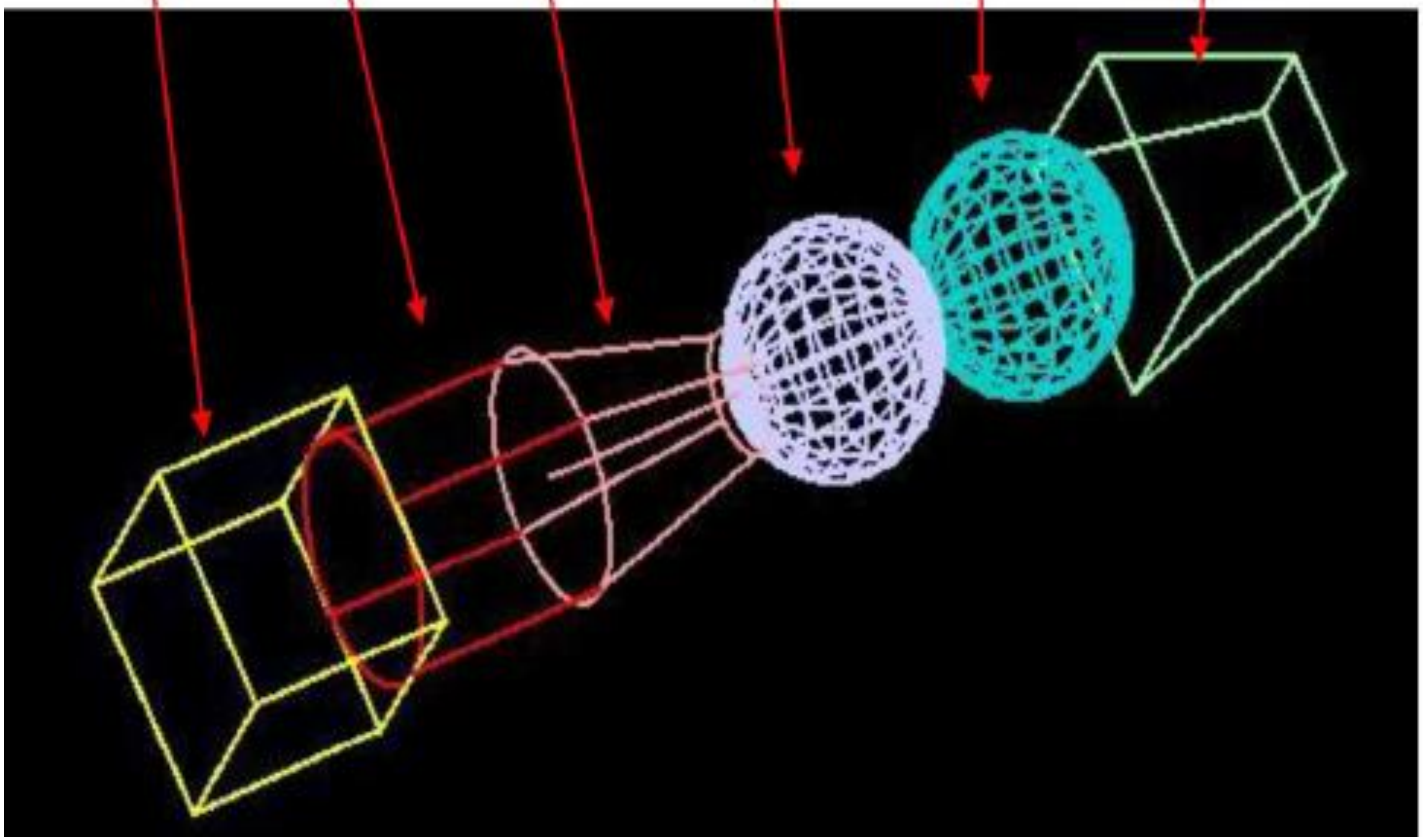
Important rules !

1. **No overlap**
2. **Fully contained**



Exercise of Today!

Box Cylinder Cone Sphere1 Sphere2 Trapezoid



Geometry in Three Steps

- **To have a volume implemented in Geant4 one has to go through three steps.**

Step 1

A *Solid* is used to describe a volume's mathematical shape. It is a geometrical object that has a shape and specific values for each of that shape's dimensions.

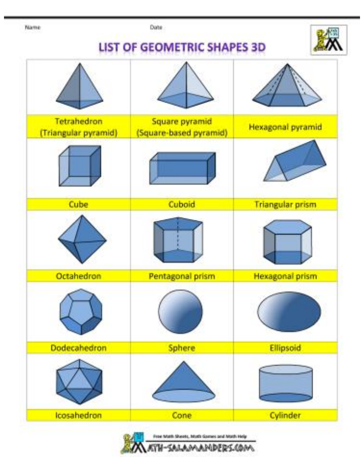
Step 2

A *Logical Volume* is used for describing a volume's full properties. It starts from its geometrical properties (the solid) and adds physical properties, like the material, the sensitivity, the magnetic/electric field, the color…

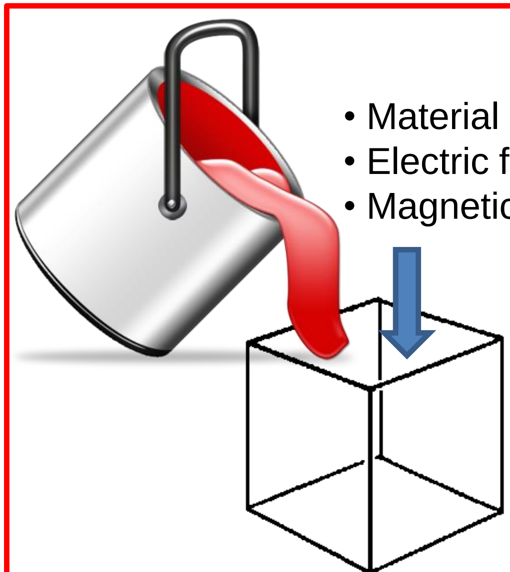
Step 3

What remains to describe is the position of the volume. For doing that, one creates a *Physical Volume* which places a copy of the logical volume inside a larger, containing volume.

Geometry in Three Steps

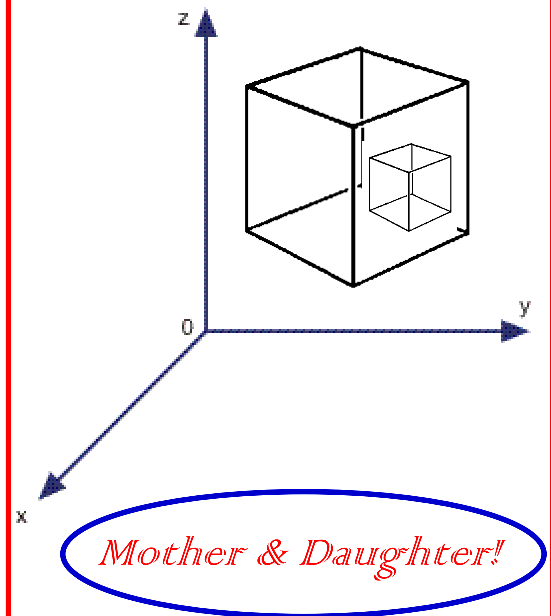


Mathematical shape
(Solid)



- Material
- Electric field
- Magnetic field

Logical Volume



Placement in (X, Y, Z)
Physical Volume

Geometry in Three Steps

- A detector geometry in Geant4 is made of a number of volumes.
- The largest volume is called the World volume. It must contain all other volumes in the detector geometry
- The other volumes are created and placed inside previous volumes, including the World.
- Each volume is created by describing its shape and its properties and characteristics then placing it inside a containing volume.
- The coordinate system used to specify where the daughter volume is placed is the one of the mother.

Geometry in Three Steps:

Detector Construction.

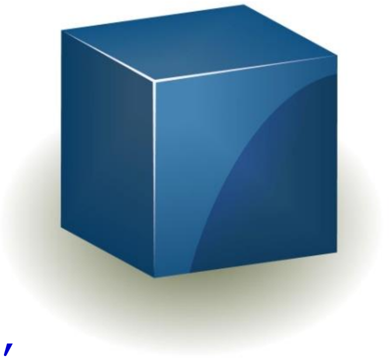
- G4VUserDetectorConstruction is one of the three mandatory classes in Geant4 the user MUST inherit from it to create his/her implementation of the detector geometry.
- G4VUserDetectorConstruction's method Construct() is invoked by the Run Manager to set up the detector geometry. Construct() should hence set up everything needed for the geometry definition.
- Construct() returns a pointer to the World's Physical Volume.

Geometry in Three Steps:

Solids.

- To create a simple box one has simply to define its name and its dimensions along each Cartesian axes:

```
#include "G4Box.hh"
...
G4double expHall_x=3.0*m;
G4double expHall_y=1.0*m;
G4double expHall_z=1.0*m;
...
G4Box* experimentalHall_box = new G4Box(
                                  "Exp._Hallbox",
                                  expHall_x,
                                  expHall_y,
                                  expHall_z);
```



Geometry in Three Steps:

Solids.

- The definition of a Box in Geant4 can be found in:
  /cvmfs/geant4.cern.ch/geant4/**<G4version>**/share/source/geometry/solids/CSG/include/G4Box.hh
- To create a box use its constructor:

```
G4Box (const G4String& pName, G4double pX, G4double pY, G4double pZ)
```

where:
pX: half length in X
pY: half length in Y
pZ: half length in Z

```
G4Box* a_box = new G4Box("My Box", 10.*cm, 0.5*m, 30.*cm);
```

Geometry in Three Steps:

Solids and Logical Volume.

- To create a Logical volume one must start from a solid and *a material should be defined*.

```
#include ”G4Box.hh”
#include ”G4LogicalVolume.hh”

...
G4Box* a_box = new G4Box(”A box”, dx, dy, dz );

G4LogicalVolume* a_box_log = new G4LogicalVolume(
                                   a_box,           (its solid)

                                   Lead,            (its material)

                                   ”a simple box”   (its name)
                             );
```

Geometry in Three Steps:
Logical and Physical Volume *(placement).*

- To position a volume, one must start with a logical volume and decide what volume (which must already exist) to place it inside, where to position it (with respect to the mother's reference system) and how to rotate it
- A physical volume is simply a positioned instance of a logical volume

```
#include "G4VPhysicalVolume.hh"
#include "G4PVPlacement.hh"
...
G4RotationMatrix *rm = new G4RotationMatrix();
rm->rotateX(30*deg);
```

```
G4VPhysicalVolume* a_box_phys =
  new G4PVPlacement(rm,                       // pointer to G4Rot.Matrix!
                    G4ThreeVector(1.0*m,0,0), // position
                    a_box_log,                // its logical volume
                    "a box",                  // its name
                    experimentalHall_log,     // its mother
                    false,                    // not used
                    1);                       // the copy nr.
```

Geometry in Three Steps

```
G4Tubs* SolidMyCylinder = new G4Tubs("SolidMyCylinder",
                                     Rmin ,
                                     Rmax,
                                     Lc/2. ,
                                     PhiMin1,
                                     PhiMax1);
```

```
G4LogicalVolume* LogicalMyCylinder = new G4LogicalVolume(SolidMyCylinder,   //its solid
                                                         Vaccum,            //its material
                                                         "LogicalMyCylinder"); //its name
```

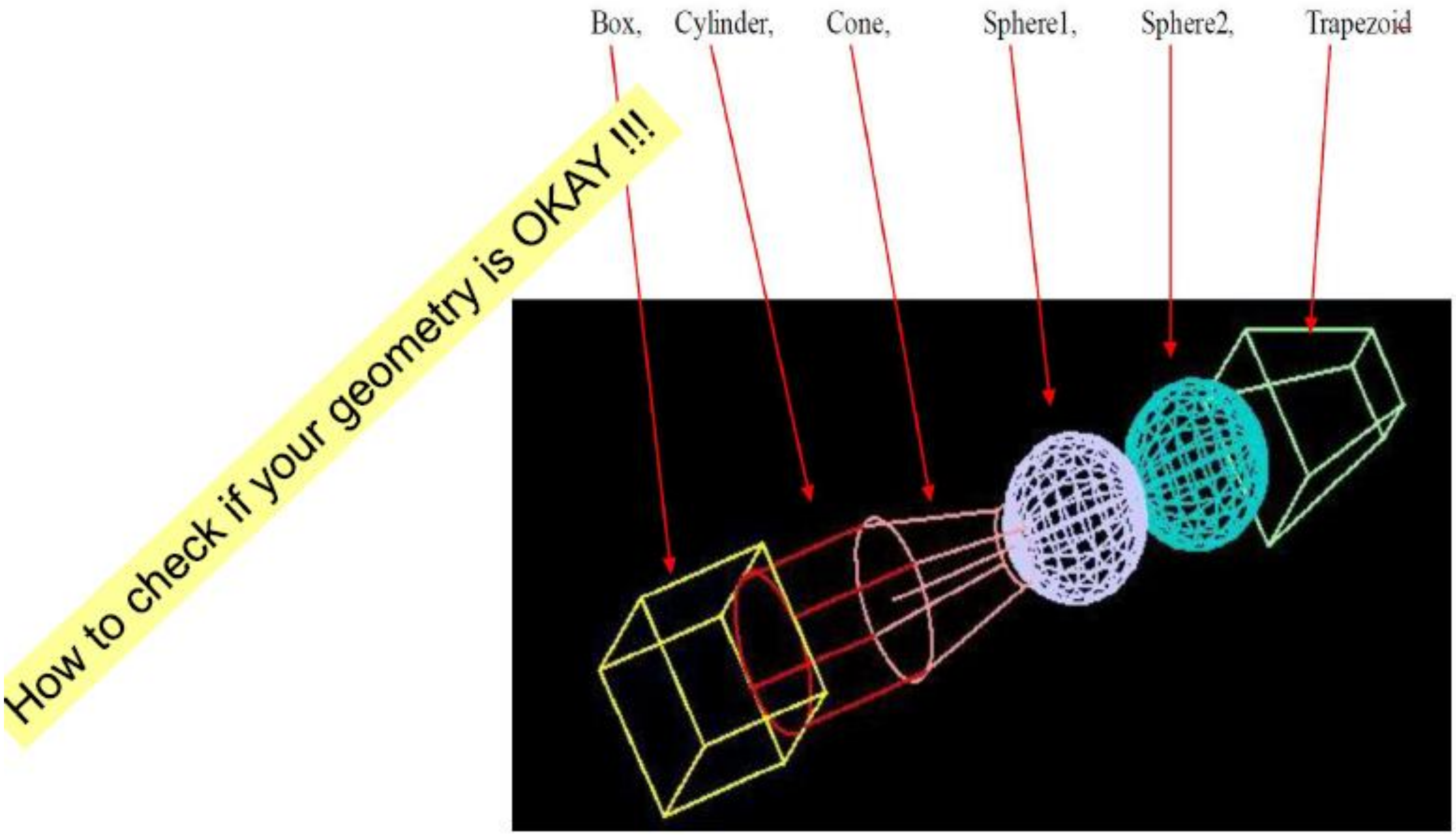
```
G4VPhysicalVolume* PhysicalMyCylinder = new G4PVPlacement(0,          //no rotation
                                     G4ThreeVector(0,0,0),  //at (0,0,0)
                                     "PhysicalMyCylinder",  //its name
                                     LogicalMyCylinder,     //its logical volume
                                     physiWorld,            //its mother volume
                                     false,                 //no Boolean operation
                                     0);                    //copy number
```

Example of 3D Rotation

```
// 1) Don’t forget to include
#include ”G4RotationMatrix.hh”

// Define Solid
// Define Logical Volume
...

// 2) Define 3D rotation
G4RotationMatrix* rot3D = new G4RotationMatrix();
rot3D->rotateX( 0.*deg);
rot3D->rotateY(10.*deg);
rot3D->rotateZ( 0.*deg);

// 3) Use the 3D rotation in the G4PVPlacement
G4VPhysicalVolume* a_box_phys =
  new G4PVPlacement(rot3D,                     // rotation
                    G4ThreeVector(1.0*m,0,0), // position
                    a_box_log,                // its logical volume
                    ”a box”,                  // its name
                    experimentalHall_log,     // its mother
                    false,                    // not used
                    1);                       // the copy nr.
```

Geometry Check

Session: **`/geometry/test/run`**



Program Template with Tasks

src/DetectorPhysDetectorConstruction.cc

```
...

//####
//#### Task A: Implement a Box at (0,0,0) (Do not forget to include the "G4Box.hh")
//####

//G4double ChamberGasBoxX = 10.*cm;
//G4double ChamberGasBoxY = 10.*cm;
//G4double ChamberGasBoxZ = 10.*cm;

//#### Task A-1: Once you are done, compile your c++ code (using "make")
//#### Task A-2: Then execute if it compiles. The executable is in the current directory (To execute u
//#### Task A-3: Check whether your geometry is OK using the command "/geometry/test/run"!

...

//####
//#### Task B: Implement a Cylinder (or Tube) (Do not forget to include the right ".hh" file. You can
//####         Make sure that this volume is adjacent to the previous volume as shown in the figure (s

//G4double Rmin    =  00.*cm;
//G4double Rmax    =   5.*cm;
//G4double Lc      =  10.*cm;
//G4double PhiMin1 =   0.*deg;
//G4double PhiMax1 = 360.*deg;

//#### Task B-1: Once you are done, compile your c++ code (using "make")
//#### Task B-2: Then execute if it compiles. The executable is in the current directory (To execute u
//#### Task B-3: Check whether your geometry is OK using the command "/geometry/test/run"!
...
```

Exercise

1. Download DetectorPhys_T1.tar.gz.
2. Decompress it (`tar xzvf DetectorPhys_T1.tar.gz`).
3. Edit the file DetectorPhysDetectorConstruction.cc (in the directory src/).
4. Follow the tasks step by step in order to be guided and build your geometry.
5. Compile your program (`cmake . ; make`).
6. Then run your simulation to visualize the geometry (`./DetectorPhys_T1`).
7. Check your geometry with: `/geometry/test/run`