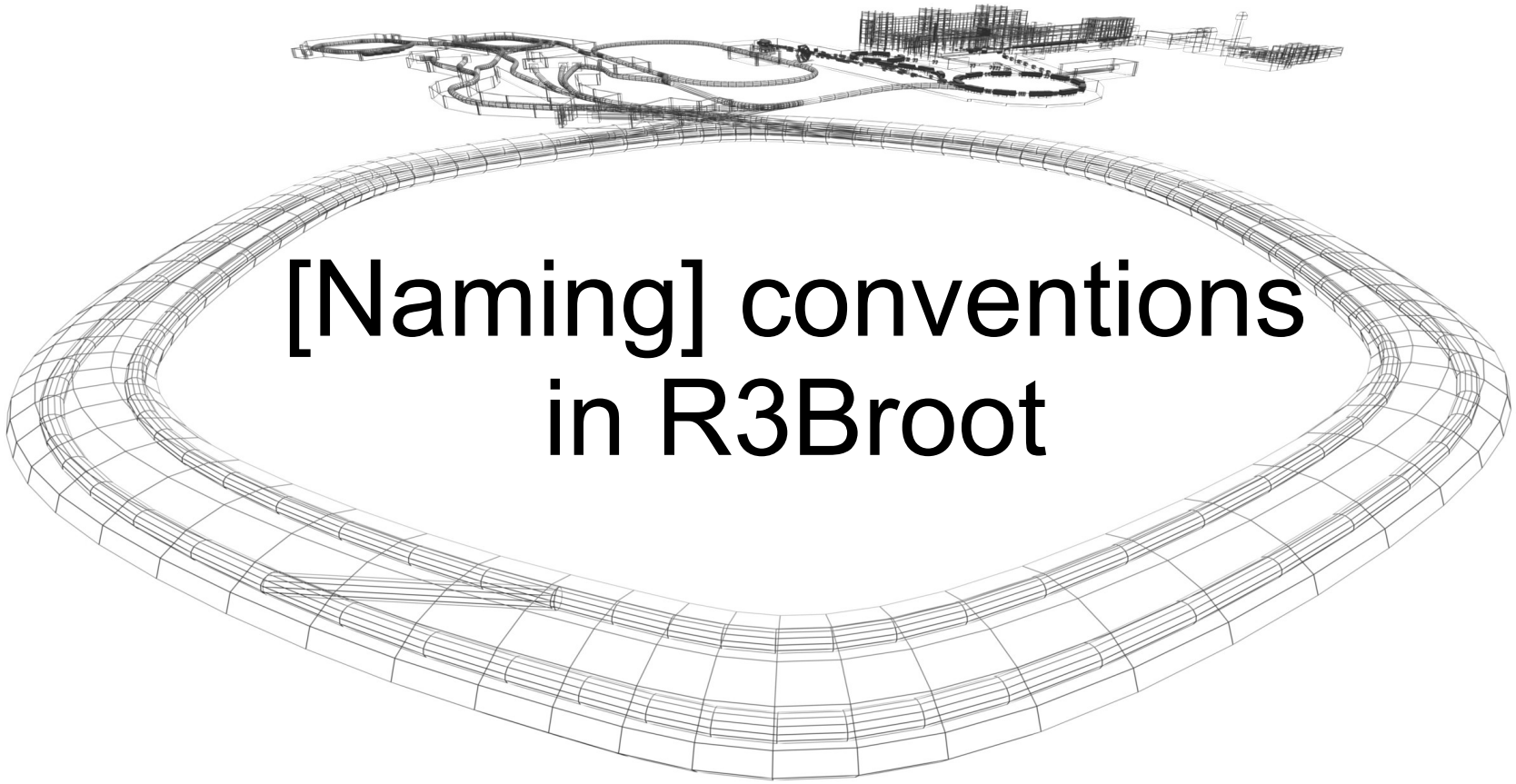


[Naming] conventions in R3Broot

GSI - March, 3rd 2017 - by Ralf Plag

Why & What

Why?

Allow useful information to be deduced from names.
Find and understand things faster.
To make life easier :-)

What?

Data levels, Units, Files, Classes, Branches, Containers, ...

Data levels (analysis)

Three **mandatory** data levels:

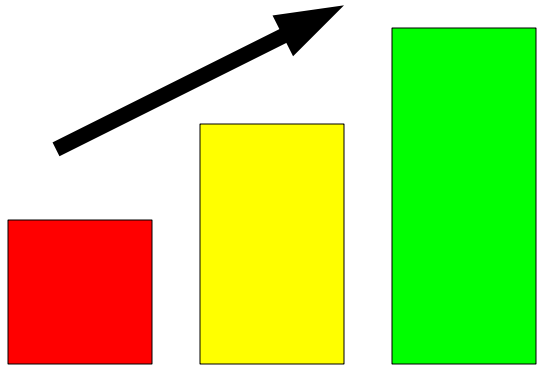
MAPPED
Mapped to plane, bar, pm, crystal..., measured values in channels, no calibration applied. Mapping is done by UCESB.

CAL
Times have been converted to ns, energies have been gain matched.

HIT
Reconstructed Hits with position in cm, time in ns, charge as atomic number, energy in MeV.

Additional levels can be introduced if needed.

Levels used in **simulations**?



Units

Use these most convenient (easiest) units to avoid mistakes:

Time:	ns
Energy:	MeV
Position:	cm
Charge:	atomic number
Mass:	u (or if convenient: number of nuclei A)
Speed:	1/c
Direction:	theta/phi or vector !?

Indices of planes, bars, photomultipliers, crystals, ...

are **always 1-based**: 1..n

if the are visible to the end-user.

Reason:
avoid confusion when looking for the
bad photomultiplier (or crystal).

Class names

Scheme: R3B[Detector]Purpose

[Detector] is only mandatory for classes that handle data of a certain detector.

Typical cases:	Examples:
Data containers: R3B[Detector][Level]Data	R3BTofdMappedData, R3BNeulandCalData.
Analysis tasks or converters, e.g. Conversion from Mapped to Cal or Cal to Hit R3B[Detector][FromLevel]2[ToLevel]	R3BNeulandMapped2Cal.
Parameter containers, e.g. For calibration parameters: R3B[Detector][Level]Par	R3BNeulandHitPar
Task which calculates parameters: R3B[Detector][FromLevel]2[ToLevel]Par	R3BNeulandCal2HitPar with output R3BNeulandHitPar.

Branch names

The **branch name** for a data object in the output file has to **match** the **class name** without prefix "R3B" and postfix “Data”.

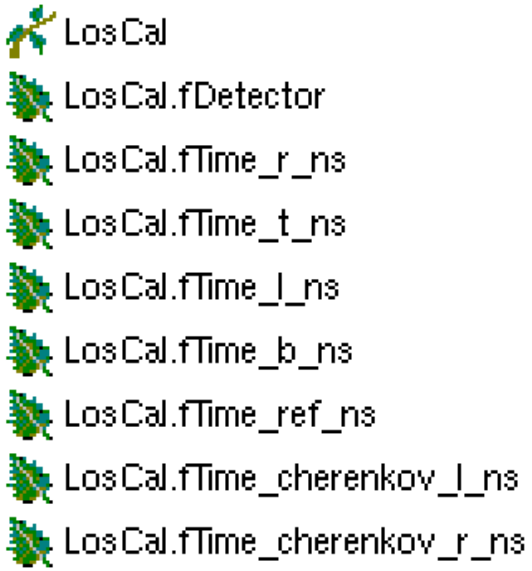
Example: **R3BLosCalData** is stored in the evt tree as **LosCal**.

```
InitStatus R3BLosMapped2Cal::Init()
{
    [...]

    mgr->Register("LosCal", "Land", fCalItems, kTRUE);

    return kSUCCESS;
}
```

Arrray of **R3BLosCalData**



File names

Every class should be defined in its own header and source file. Hence:

File name = class name

Example:

Class R3BLosMapped2Cal

Lives in files:

R3BLosMapped2Cal.cxx	(source)
R3BLosMapped2Cal.h	(header)

Directory structure

R3BRoot
- r3bsource
  - classes that read mapped data from UCESB, aka R3B[Detector]Reader
- r3bdata
  - [Detector]data
    - all data containers, aka R3B[Detector][Level]Data
- r3bdb
  - - under discussion -
- tcal
  - classes and tools for time calibration
- [Detector]
  - classes that analyse data, aka convert from one level to another
  - unpack - obsolete, don't use this folder

Coding rules

Types:
Begin with a capital letter, e.g. Int_t

Variables:
Begin with a lowercase letter
All class fields start with an f followed by a capitalized word, e.g. fEnergy

Indentation:
Tabs to indent, spaces to align. Reason: Tab-width doesn't matter.

```
{
→ Int_t····someInt;
→ Double_t·someDouble;
}
```

Learn the rules well,
so you can break them properly.

- Dalai Lama

See Root Coding Conventions:

https://root.cern.ch/coding-conventions
https://root.cern.ch/TaligentDocs/TaligentOnline/DocumentRoot/1.0/Docs/books/WM/WM_63.html