Histograms

Booking 1-d Histograms

Through the Histogram service

```
IHistogram1D* m_pH1dElePt =
  histoSvc()->book("/stat/1Dhist/1", //HBOOK
```

Title	"ElePt",
Number of bins (X)	100,
Low Edge (X)	0.0,
High Edge (X)	50.0);

Booking 2-d Histograms

Through the Histogram service

```
IHistogram2D* multiplicityVsEnergyH2D =
histoSvc()->book("/stat/myhist/", 2,
```

Title	"MultiplictyvsEnergy(GeV)",
Number of bins (X)	100,
Low Edge (X)	0.0,
High Edge (X)	1000.0,
Number of bins (Y)	50,
Low Edge (Y)	0.0,
High Edge (Y	500.0);

Filling Histograms

1-D Histograms

```
multiplicityH1D->fill(
    X - value        mult,
    Weight           1.0);
```

2-D Histograms

```
multiplicityVsEnergyH2D->fill(
    X - value        mult,
    Y - value        energy,
    Weight           1.0);
```

Calls similar to HBOOK HF1 & HF2

Histogram Persistency

In jobOptions file

- **Load the relevant shared library**

  ApplicationMgr.DLLs += {“HbookCnv”}/ {“RootHistCnv”};

- **Specify persistency (accordingly to shared lib)**

  ApplicationMgr.HistogramPersistency =

  “HBOOK” / “ROOT” / “NONE” (default);

- **Specify name of output file, e.g.,**

  HistogramPersistencyService.OutputFile = “atlfast.hbook” / “atlfast.rt”;

Ntuples

Booking A Ntuple

For CBNT this is taken care of by the infrastructure...

Through the Ntuple service

```
NTuplePtr nt( ntupleSvc(), "/NTUPLES/FILE1/100”);
if ( !nt )  { //check if already booked
        nt = ntupleSvc()->book (
                ”/NTUPLES/FILE1/100",
                CLID_ColumnWiseTuple,
                ”My Ntuple");
        }
if ( nt )    { use it }
```

Location on store
Row- or Column wise ?
Title

Define Ntuple Columns

For column-wise, variable must first be declared as data members of the class

- **Item, Array, Matrix of type bool, float, long**

```
NTuple::Item<long>    m_nZee;
NTuple::Array<float>  m_mass;
NTuple::Tuple* nt;
```

After the Ntuple is booked

```
if ( nt )    {
    status = nt->addItem("ZEE/z_ncand", m_nZee, 0,5);
    status = p_nt1->addItem("ZEE/z_mass",m_nZee, m_mass)
}
```

See CBNT_Zee.h/cxx

Filling Ntuples

Items behave like a number

- Just assign values:

```
m_nZee = container()->size();
```

- Same for arrays

```
m_mass[i] = Zee.mass();
```

Write record

```
status = ntupleSvc()->writeRecord(m_ntuple);
   if ( status.isFailure() )  {
     log << MSG::ERROR << "Cannot fill Ntuple" <<
  endreq;
   }
```

Ntuple Persistency

In job options specify output files

```
NTupleSvc.Output = { ”FILE1
            DATAFILE=’atlfast.ntup'
            OPT='NEW'
            TYP=’HBOOK'" };
```

Technology: HBOOK

```
NTupleSvc.Output = { ”FILE1
            DATAFILE=’atlfast.ntup'
            OPT='NEW'
            TYP='ROOT'" };
```

Technology: ROOT

Documentation

Athena User/Developers Guides v2.0.x (et al.) - & release notes

- Both from http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/architecture/General/index.html

Gaudi Developers Guide

- http://proj-gaudi.web.cern.ch/proj-gaudi/GDG/v2

Athena Tutorials

- http://atlas.web.cern.ch/Atlas/GROUPS/SOFTWARE/OO/architecture/General/Tutorial/

Web video stream

- http://www.wlap.org

AthASK& PythonScripting

- http://wlav.home.cern.ch/wlav/athena/