8 Utility Tools

8.1 Debug tool

Debug tool

Goal: Provide user friendly dump of the event

- Work on both reconstructed and MC data.
- Side by side print out of reconstructed and MC data.
- Informations can be selected by the user.
- Flat or Tree display.

Debug tool usage

Getting the tool

Get the definition `#include "DaVinciMCTools/IDebugTool.h"`

Declare your instance `IDebugTool *m_debug`

Get your instance `toolSvc()->retrieveTool("DebugTool", m_debug)`

Using the tool

Event as trees `m_debug->printEventAsTree( mcparts [, assoc] )`

Particle decay as tree `m_debug->printTree( part [, depth] )`

Event as a flat list `m_debug->printEventAsList( parts [, assoc] )`

Ancestors `m_debug->printAncestor( mcpart )`

Debug tool output

```
<---------------------------------- MCParticle ---------------------------------->
                Name          E          M          P         Pt        phi         Vz
                            GeV        GeV        GeV        GeV       mrad         cm
B0                       29.828      5.279     29.358      7.666    -29.331     -1.780
+-->J/psi(1S)            29.119      3.097     28.954      7.707      4.943     -1.450
|+-->mu+                 23.835      0.106     23.835      7.396     69.765     -1.450
|+-->mu-                  5.284      0.106      5.283      0.580   -967.540     -1.450
| +-->nu_e                0.046     -0.000      0.046      0.041     58.550   1932.276
| +-->e-                  0.039      0.001      0.039      0.038  -2650.137   1932.276
| +-->nu_e                0.021     -0.000      0.021      0.017   1996.055   1932.276
+-->KS0                   0.710      0.498      0.506      0.267  -1736.262     -1.450
 +-->pi+                  0.280      0.140      0.243      0.167   2955.913     -0.167
 |+-->mu+                 0.110      0.106      0.030      0.019  -1545.998    234.307
 ||+-->nu_e               0.017      0.000      0.017      0.007   2418.076    234.307
 ||+-->e+                 0.036      0.001      0.036      0.030   2629.653    234.307
 ||+-->nu_e               0.052     -0.000      0.052      0.037   -554.517    234.307
 |+-->nu_e                0.030      0.000      0.030      0.019   1595.595    234.307
 +-->pi-                  0.429      0.140      0.406      0.317  -1182.472     -0.167
```

Debug tool side by side output

```
<------------- MCParticle -------------><--------- Particle --------->
               Name          P         Pt       Name          P         Pt
                           GeV        GeV                   GeV        GeV
B0                      29.358     7.666  No associated particle
+-->J/psi(1S)           28.954     7.707  No associated particle
|+-->mu+                23.835     7.396  No associated particle
|+-->mu-                 5.283     0.580 mu-              5.272      0.578
| +-->nu_e               0.046     0.041  No associated particle
| +-->e-                 0.039     0.038  No associated particle
| +-->nu_e               0.021     0.017  No associated particle
+-->KS0                  0.506     0.267  No associated particle
 +-->pi+                 0.243     0.167  No associated particle
  |+-->mu+               0.030     0.019  No associated particle
  ||+-->nu_e             0.017     0.007  No associated particle
  ||+-->e+               0.036     0.030  No associated particle
  ||+-->nu_e             0.052     0.037  No associated particle
  |+-->nu_e              0.030     0.019  No associated particle
 +-->pi-                 0.406     0.317  No associated particle
```

Debug tool List and Ancestors output

Flat list

```
<------------------- Particle ------------------->
               Name         Vz         Vz         Vz
                            cm         cm         cm
pi+                      0.645      0.537       64.9
pi-                      0.385     -0.729       28.9
e-                      -0.735        7.2        484
pi+                      0.393      0.728       28.9
mu-                       5.97       15.5        487
pi+                       6.01       15.2        488
pi-                     -0.293      0.899       15.4
e-                       -3.85      -6.23        232
mu-                      0.664     -0.854       7.93
gamma                     -205       -295   1.26e+03
```

Ancestors

```
pi0 -> gamma -> e+
```

Debug tool configuration

In the *jobOption* file:

- User can select the informations to dump.
- Width of the columns can be adjusted.
- Numerical precision can also be tuned.
- Tree depth can be limited in general.

Available informations are:

`Name`	The particle name (plus the tree drawing)
`E`	The energy
`M`	The mass
`P`, `Pt`, `Px`, `Py`, `Pz`	The momentum
`Vx`, `Vy`, `Vz`	The position of the first measured point
`theta`, `phi`	The spherical angles
`eta`	The pseudo-rapidity

Debug tool example jobOption

```
// Defaults for DebugTool
MyToolOwner.DebugTool.PrintDepth = 999;
MyToolOwner.DebugTool.TreeWidth = 20;
MyToolOwner.DebugTool.FieldWidth = 10;
MyToolOwner.DebugTool.FieldPrecision = 3;
MyToolOwner.DebugTool.Informations = "Name E M P Pt phi Vz";
```

8.2 (MC)DecayFinder

(MC)DecayFinder

Goal: Find any inclusive or exclusive decay in an event

- Work at the particle ID level.
- Work on both reconstructed and MC data.
- Find multiple instances of the decay.
- Use a simple description of the decay.

(MC)DecayFinder usage

Getting the tool

Get the definition `#include "DaVinciMCTools/I(MC)DecayFinder.h"`

Declare your instance `I(MC)DecayFinder *m_finder`

Get your instance

```
toolSvc()->retrieveTool("(MC)DecayFinder",
                        m_finder)
```

Using the tool

```
const (MC)Particle *result = NULL;
while( m_finder->findDecay( (mc)parts, result ) )
{
// The decay has been found
  m_debug->printTree( result );
}
```

Or to simply test for the presence of the decay

```
bool found = m_finder->hasDecay( (mc)parts )
```

(MC)DecayFinder configuration

Only two parameters exists:

`Decay` **The decay we are looking for. Must be set in the jobOption file.**

`ResonanceThreshold` **The lifetime under which a particle is considered a resonance. Default provided.**

(MC)DecayFinder decay grammar

A *decay* is a mother or

mother -> daughter$_1$ daughter$_2$ ...

Use => instead of -> if you want to skip the resonances.

A *mother* is either

- a particle name or !name or ?
- (name)
- [name]os
- [name$_1$, name$_2$]cc
- {mother$_1$, mother$_2$,...}
- pp

A *daughter* is either

- a particle name or !name or ?
- (decay)
- [name$_1$, name$_2$]cc
- {mother$_1$, mother$_2$,...}
- {(decay$_1$), (decay$_2$),...}
- ...

(MC)DecayFinder grammar examples

`B0`	any B^0
`(B0)`	any stable B^0
`(pi+)`	any stable π^+
`J/psi(1S)`	any stable $J\Psi$
`pi0 -> gamma gamma`	any π^0 decay to 2 γ
`[B0,B+]cc`	any B^0, $\bar{B}^0$, B^+ or B^-
`pp => [K-]cc`	
`B0 -> (J/psi(1S) -> mu+ mu- ) (KS0 -> pi- pi+)`	
`B0 -> (J/psi(1S) -> mu+ mu- {gamma,}) (KS0 -> pi- pi+)`	
`B0 => mu+ mu- gamma (KS0 -> pi+ pi-)`	

8.3 Gaudi utilities: a reminder

Printing

To print out informations use the *MessageService* and not `cout`. Because:

- It works like `cout`.
- It adds a severity tag to your message.
- It tells the user from where the message is coming.
- It can be filtered based on severity.

Printing How-To

- **Get the definition of this facility.**

```
#include "GaudiKernel/MsgStream.h"
```

- **Create a stream.**

```
MsgStream log(msgSvc(), name())
```

- **Print!**

```
log << MSG::DEBUG << "Hello World!" << endreq
```

Severity & Notes

The available severity levels are (in increasing order):

- `MSG::DEBUG`
- `MSG::INFO`
- `MSG::WARNING`
- `MSG::ERROR`
- `MSG::FATAL`

One *request* to the Message Service can be split. You just need to start it with a severity tag and end it with a `endreq`**.**

```
log << MSG::INFO << "Momentum along x: " << mypart->momentum().px()/GeV << endl;
double ptx = sqrt(pow(mypart->momentum().py(),2)+pow(mypart->momentum().pz(),2));
log << "Momentum in yz plane: " << ptx/GeV << endreq;
```

Ntuple

To use a Ntuple you have to

1. Declare the variables of your ntuple.
2. Create the ntuple.
3. Register the ntuple.
4. Register your variables to your ntuple.
5. Fill the variables & commit.
6. Adjust the `NtupleSvc.Output` in your jobOption file.

Note that step 2. could fail if the ntuple already exists.

Ntuple variables declaration

First get the definition of what kind of items can be put in the ntuple.

```
#include "GaudiKernel/NTupleItems.h"
```

Than declare your variables with the appropriate type.

```
NTuple::Item<long> m_nPart;
NTuple::Array<float> m_px, m_py, m_pz;
NTuple::Matrix<float> m_trackEnds_x, m_trackEnds_y, m_trackEnds_z;
```

`Array` and `Matrix` can only be used with a column wise ntuple.

Ntuple creation & booking

First check if your ntuple has already been registered.

```
NTuplePtr MyNtuple(ntupleSvc(), "MyFileKey/MyDirectory/MyID");
```

If not (*MyNtuple* == 0) then create and book your ntuple.

```
MyNtuple = ntupleSvc()->book ("MyFileKey/MyDirectory", MyID,
                              CLID_ColumnWiseTuple, "MyTitle");
```

Here it was created in *MyDirectory* as *MyID* in the file associated to *MyFileKey*.

Ntuple setup

Attach the variables to the ntuple.

```
status = nt->addItem ("NParts",  m_nPart, 0, 5000);
if( status.isSuccess() )
  status = nt->addIndexedItem ("px", m_nPart, m_px);
if( status.isSuccess() )
  status = nt->addIndexedItem ("vx", m_nPart, 2, m_trackEnds_x);
...
```

Or if it already exists, reattach the variables.

```
status = nt->item ("NParts",  m_nPart);
if( status.isSuccess() )  status = nt->item ("px", m_px);
if( status.isSuccess() )  status = nt->item ("vx", m_trackEnds_x);
...
```

Ntuple filling

Use the variables you associated to the ntuple as usual. When you are ready to write the row of the ntuple out, simply call the `write` method.

```
ntuple->write()
```

After that call, all the variables will be reset to zero.

Ntuple jobOption settings

Be sure to have the NTuple service loaded (it is by default in DaVinci).

```
ApplicationMgr.ExtSvc += { "NTupleSvc" };
```

Then to have your ntuple saved to disk you need to say what kind of persistence format you want.
For the traditional HBOOK format:

```
NTupleSvc.Output={"MyFileKey DATAFILE=’MyFileName.hbook’ TYP=’HBOOK’ OPT=’NEW’"};
```

For the more recent ROOT format:

```
NTupleSvc.Output={"MyFileKey DATAFILE=’MyOtherFileName.rt’ TYP=’ROOT’ OPT=’NEW’"};
```

Histogram

To use the histogram facility you have to

1. Select the kind of persistence you want (Hbook or Root).
2. Adjust the jobOption file.
3. Create & register your histograms.
4. Fill them.

Histogram persistence

Histogram persistence can be achieved with either Hbook or Root.
To change the default of Hbook to Root you must change the requirement file of DaVinci to

```
#use HbookCnv              v12r0
use RootHistCnv          v6r0
```

You also need to change the jobOption file to

```
\\#include "$STDOPTS/Hbook.opts"
\\HistogramPersistencySvc.OutputFile = "Histos.hbook";
#include "$STDOPTS/RootHist.opts"
HistogramPersistencySvc.OutputFile = "Histos.rt";
```

Histogram creation

Get the headers defining the histograms.

```
#include "GaudiKernel/IHistogramSvc.h"
#include "AIDA/IHistogram1D.h"
```

Declare the variable which will contain your histogram.

```
IHistogram1D *m_hB0Mass
```

Create and book your histogram.

```
m_hB0Mass = histoSvc()->book("MyDirectory", MyID, "MyTitle",
                             NBins, Min, Max);
if( 0 == m_hB0Mass) BUG();
```

Histogram filling

Histogram filling is straightforward.

```
m_hB0Mass->fill(candB0.mass()/GeV, 1.)
```

Always divide the value by the unit so you don’t have to remember the default units.