

X. Rouby

Delphes

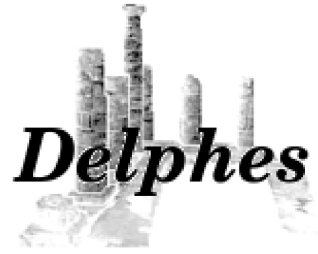
A framework for fast simulation of a
generic collider experiment

Xavier Rouby^(a), Séverine Oryn

Université catholique de Louvain, Belgium
Center for Particle Physics and Phenomenology (CP3)

*(a) now in Physikalisches Institut Albert-Ludwigs-
Universität Freiburg*

References



X. Rouby

Website :

<http://www.fynu.ucl.ac.be/delphes.html>

News / Download / User manual / FAQ

Paper + User manual :

[arXiv:0903.2225\[hep-ph\]](https://arxiv.org/abs/0903.2225)

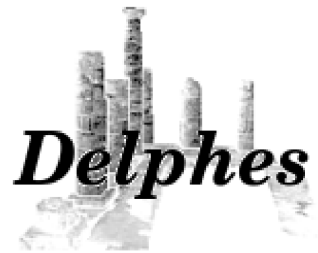
Delphes, a framework for fast simulation of a generic collider experiment,
S. Ovyn, X. Rouby and V. Lemaître

Contacts :

severine.ovyn@uclouvain.be

xavier.rouby@cern.ch

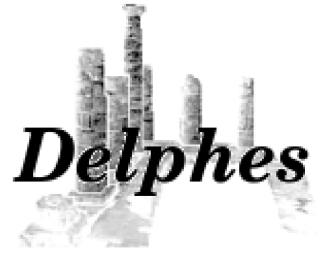
Tutorial session



X. Rouby

Getting started...

Getting started : download



X. Rouby

Code download

from the website : « *download* » link

Delphes tar-ball is self-sufficient, it contains every dependencies needed for the physics.

Or from a command line :

```
wget http://www.fynu.ucl.ac.be/users/s.ovyn/Delphes/files/Delphes_V_1.7.tar.gz
```

Requirements

A recent working ROOT version (<http://root.cern.ch>)

ROOT: R. Brun, F. Rademakers, [NIM A 389 \(1997\) 81-86](#).

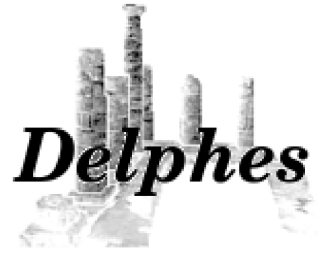
Delphes has been developed on ROOT > 5.18 on Linux with GNU gcc/g++ > 4.1.2, but any recent version should be fine.

For Mac-OSX users:

In **Delphes'** genMakefile.tcl, you should add "-Dmacos" in the CXXFLAGS definition (line 219):

```
CXXFLAGS += $(ROOTCFLAGS) -Dmacos -DDROP_CGAL -I. ...
```

Getting started : download



X. Rouby

Requirements

Checking ROOT installation

```
echo $ROOTSYS
```

If empty, check that the environment variables are defined.

In *bash* shell, check that these variables are in the *.bashrc*:

(e.g. assuming ROOT is in */usr/bin/root*)

```
export ROOTSYS=/usr/bin/root
export PATH=$PATH:$ROOTSYS/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ROOTSYS/lib
```

Test it :

```
root
```

If the FROG event display is to be run:

3D-OpenGL libraries are not included in the `tar.gz`, but required only if FROG is used. These libraries can be downloaded from here: <http://curl.haxx.se/download.html>

More on FROG requirements:

<http://projects.hepforge.org/frog/index.php?page=Starting.php>

Getting started : compile



Untar – decompress the code sources

```
tar -xzf Delphes_V_1.7.tar.gz
```

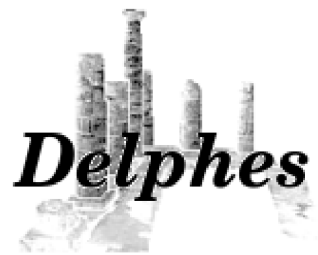
Compile the sources

```
cd Delphes_V_1.7
./genMakefile.tcl > Makefile
make
>> Compiling tmp/Utilities/ExRootAnalysis/src/BlockClassesDict.cc
>> Compiling tmp/src/TreeClassesDict.cc
...
>> Building Analysis_Ex
Delphes has been compiled
Ready to run
```

Many lines are printed during the compilation.

In particular, the dependencies (like FastJet, mcfio, stdhep) lead to a few warning messages. This is normal and harmless.

Getting started : compile



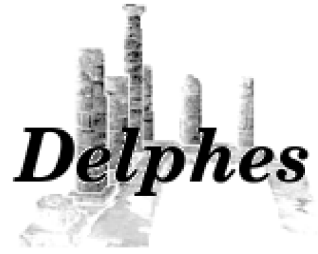
If you are running from the server...

Thanks to your comments this morning, Delphes is running smoothly on your server. If you plan to use this version during this tutorial, should still need to copy the `data` folder into your home:

```
mkdir myDelphes  
cd myDelphes  
cp -r /usr/local/Delphes_V_1.5/data .
```

This simply copies all the detector cards, trigger card, beam files so that you can play with them...

X. Rouby



Input files from MC generator

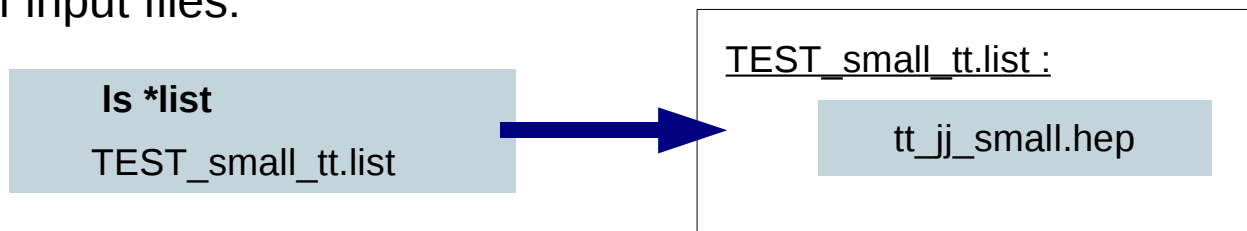
Suggested samples for this introduction:

```
wget http://www.fynu.ucl.ac.be/users/s.ovyn/Delphes/files/tt_jj_small.hep.tar.gz
tar -xzf tt_jj_small.hep.tar.gz
mv samples/* .
```

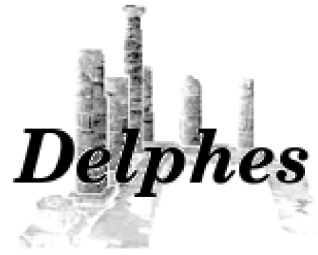
These events are $\gamma p \rightarrow ttX$,

- generated with MadGraph/MadEvent and
- hadronised with Pythia
- saved into StdHEP file format (*.hep).

List of input files:



- text file containing one input data file per line
- all data files must be of the same type



X. Rouby

Running *Delphes* :

`./Delphes`

Usage: `./Delphes input_file output_file [detector_card] [trigger_card]`

`input_list` - list of files in Ntpl, StdHep or LHEF format,

`output_file` - output file.

`detector_card` - Datacard containing resolution variables for the detector simulation (optional)

`trigger_card` - Datacard containing the trigger algorithms (optional)

Main things needed:

- `input_list` : e.g. TEST_small_tt.list
- `output_file` : e.g. test.root

List of input MC files

Output ROOT filename

Some options:

- `detector_card` : e.g. data/DetectorCard_CMS.dat
- `trigger_card` : e.g. data/TriggerCard_CMS.dat

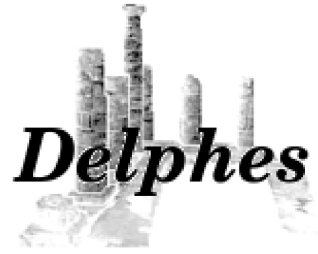
Detector parameters

Trigger definitions

Try it:

```
./Delphes TEST_small_tt.list test.root
```

Getting started : run



Delphes

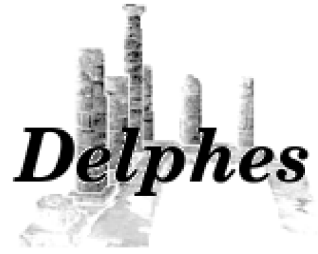
X. Rouby

Running *Delphes* :

```
./Delphes TEST_small_tt.list test.root
```

```
*****
*****
**
**                                     **
**                               Welcome to                               **
**                                     **
**                                     **
**                                     **
**      .ddddddd-                lL                hH                **
**      -Dd` `dD:                Ll                hH`                **
**      dDd  dDd    eeee.  lL  .pp+pp    Hh+hhh`    -eeee- `sssss **
**      -Dd  `DD    ee. ee  Ll  .Pp. PP    Hh. HH.    ee. ee  sSS    **
**      dD`  dDd    eEeee:  lL.  pP.  pP    hH  hH`  eEeee:` -sSSsS. **
**      .Dd  :dd    eE.      LL  PpppPP    Hh  Hh    eE          sSS    **
**      dddddd:.    eee+:  lL.  pp.        hh.  hh    eee+  ssssss **
**                                     Pp                                     **
**                                     **
**      Delphes, a framework for the fast simulation                    **
**      of a generic collider experiment                                **
**      arXiv:0903.2225v1 [hep-ph]                                     **
**                                     **
**      ...                                                             **
**                                     **
**                               Exiting Delphes ...                    **
**                                     **
*****
*****
```

Getting started : run



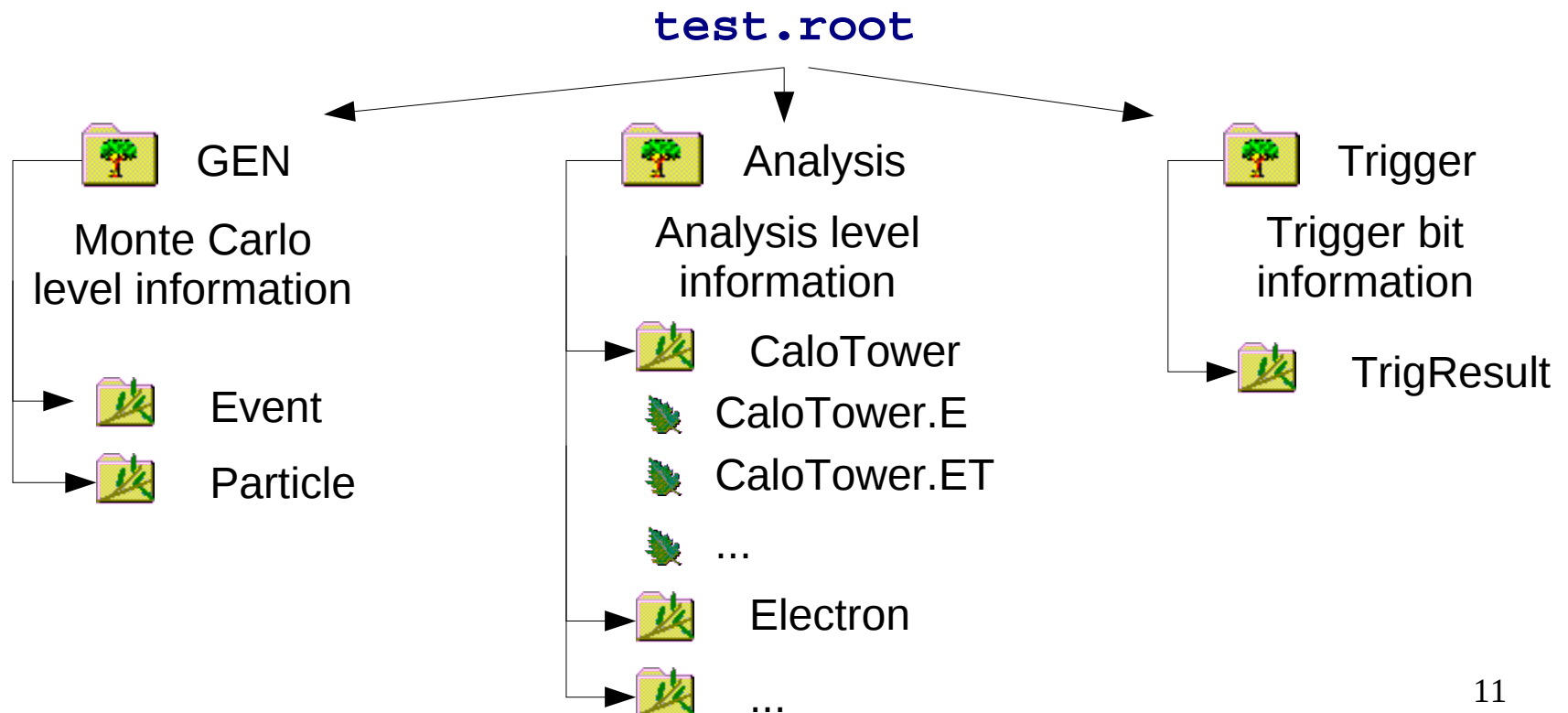
Running *Delphes* :

```
./Delphes TEST_small_tt.list test.root
```

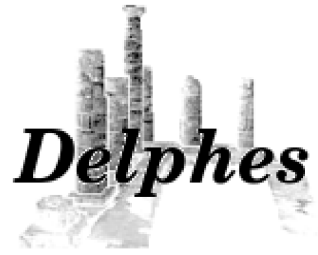
This creates the following output file test.root

```
root -l test.root
root [0]
Attaching file test.root as _file0...
Warning in <TClass::TClass>: no dictionary for class TRootGenEvent is available
...
root [1] Tbrowser t
```

X. Rouby



Getting started : run



Running *Delphes* :

```
./Delphes TEST_small_tt.list test.root
```

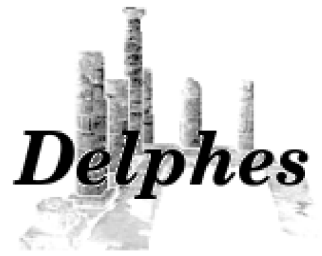
This creates the following output file test.root

```
root -l test.root
root [0]
Attaching file test.root as _file0...
Warning in <TClass::TClass>: no dictionary for class TRootGenEvent is available
...
root [1] Tbrowser t
```

A log file is also created, with all parameters test_run.log

```
# Input PDG table : data/particle.tbl *
* *
#***** *
# Central detector characteristics *
#***** *
* *
* Maximum tracking system: 2.5 *
* Maximum central calorimeter: 3 *
... *
```

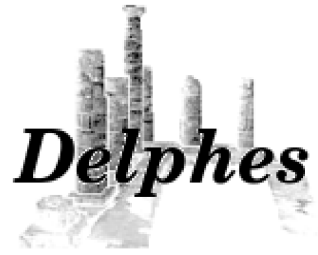
This log file contains the input parameters from data card, and triggers.



X. Rouby

Interlude: ROOT...

Getting started : ROOT



X. Rouby

Some useful commands in ROOT:

```
root -l test.root
root [0]
Attaching file test.root as _file0...
Warning in <TClass::TClass>: no dictionary for class TRootGenEvent is available
...
```

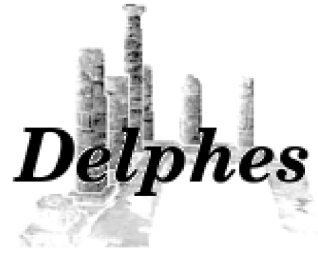
This command opens ROOT and at the same time opens the test.root file

```
root [1] TBrowser t ← Browser for the file contents
root [2] .ls
root [3] Analysis->GetEntries() ← Number of events in the tree
root [4] GEN->GetListOfBranches()->ls() ← Branch/leaf contents
root [5] Analysis->GetListOfLeaves()->ls("Electron")
```

Quickly drawing distributions:

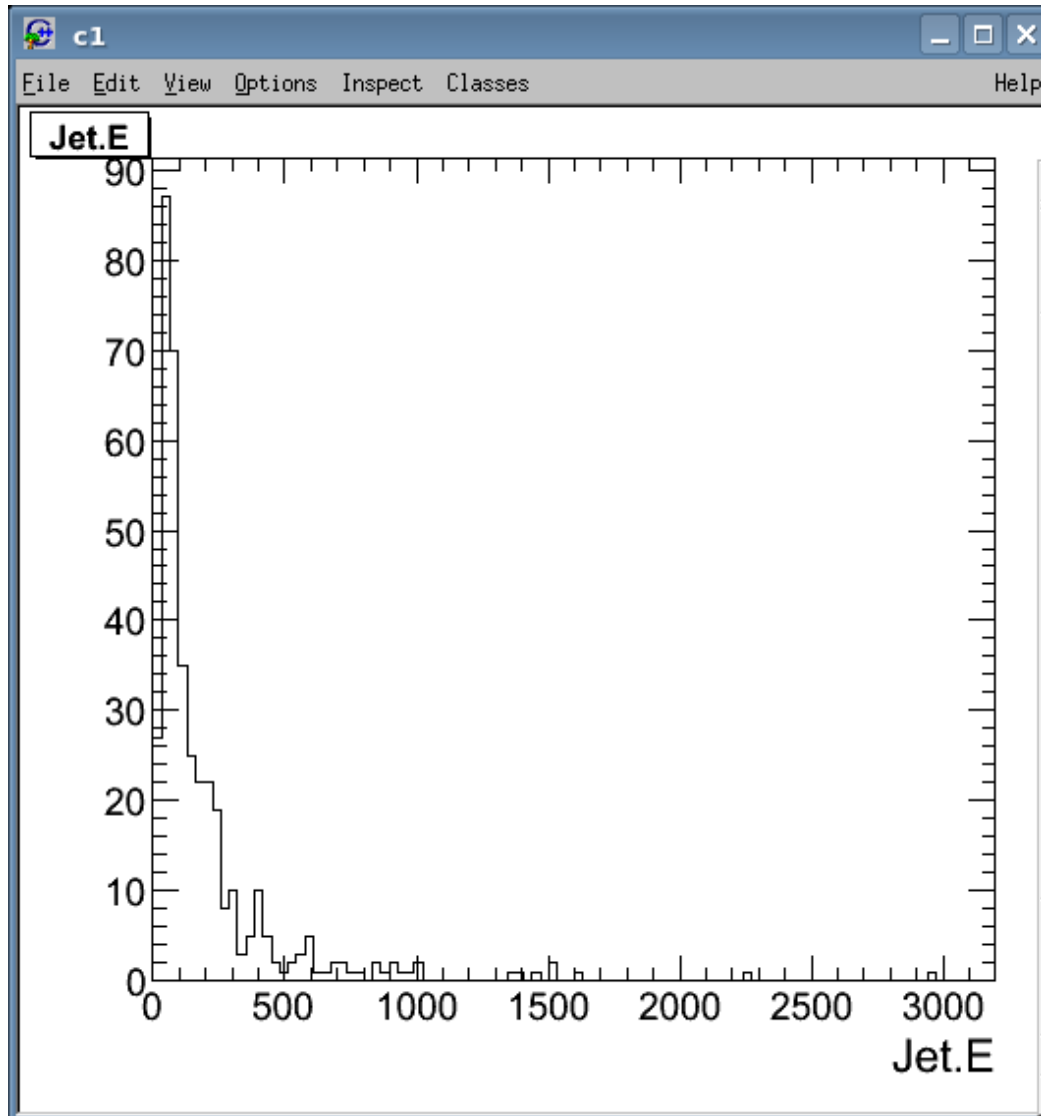
```
root [6] Analysis->Draw("Jet.E") Energy
root [7] Analysis->Draw("Jet.Pt : Jet.Eta") Pt vs Eta
root [8] Analysis->Draw("Photon.Phi","Photon.E>15") Phi with a cut on energy
root [9] Trigger->Draw("TrigResult.Accepted") Trigger result
```

Getting started : ROOT



X. Rouby

```
root [6] Analysis->Draw("Jet.E")
```



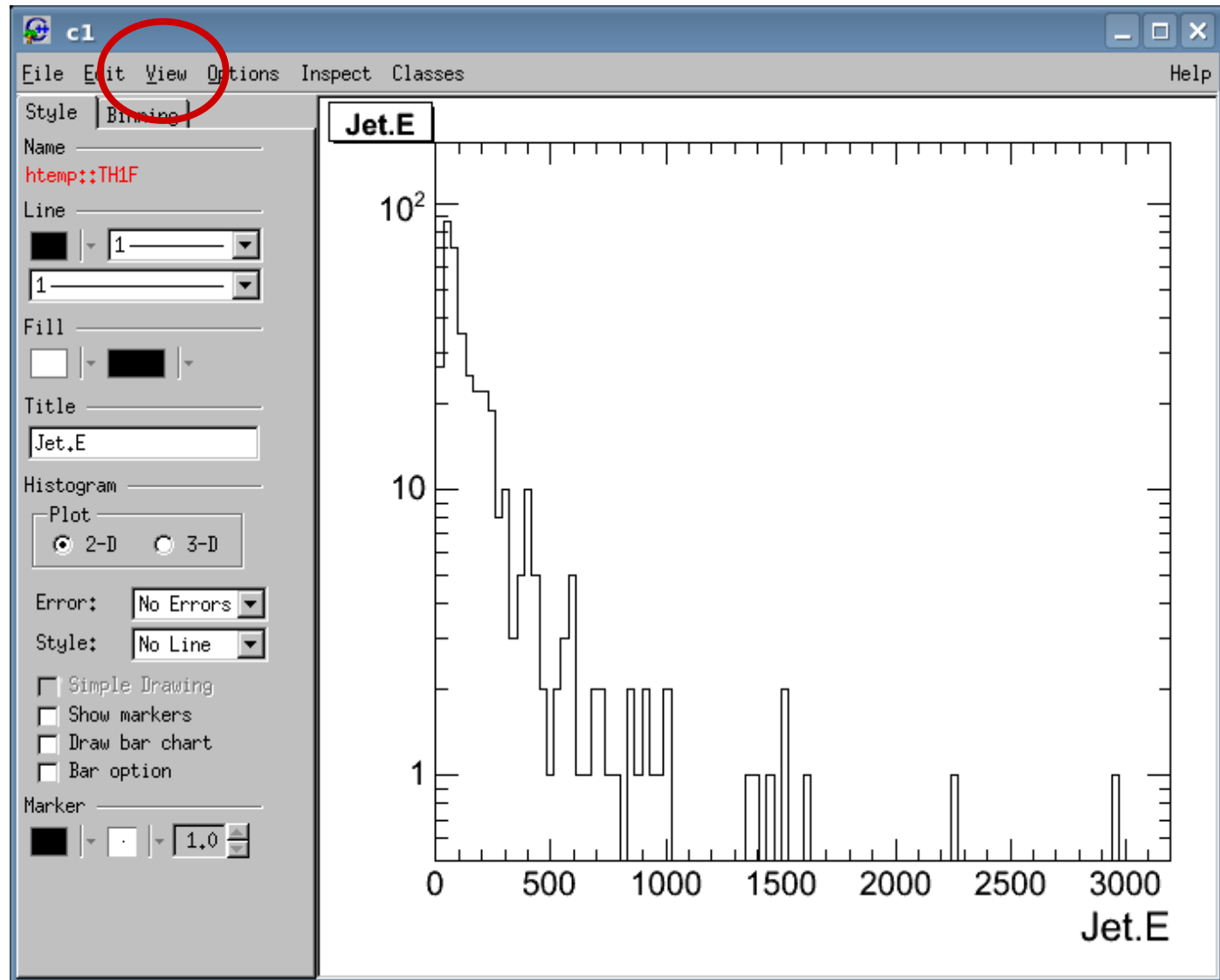
Right-click in the white edge

Getting started : ROOT



```
root [6] Analysis->Draw("Jet.E")
```

View → Editor



X. Rouby

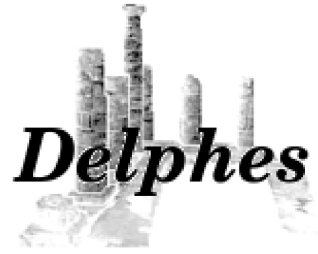


Delphes

X. Rouby

*Back to Delphes
parameters...*

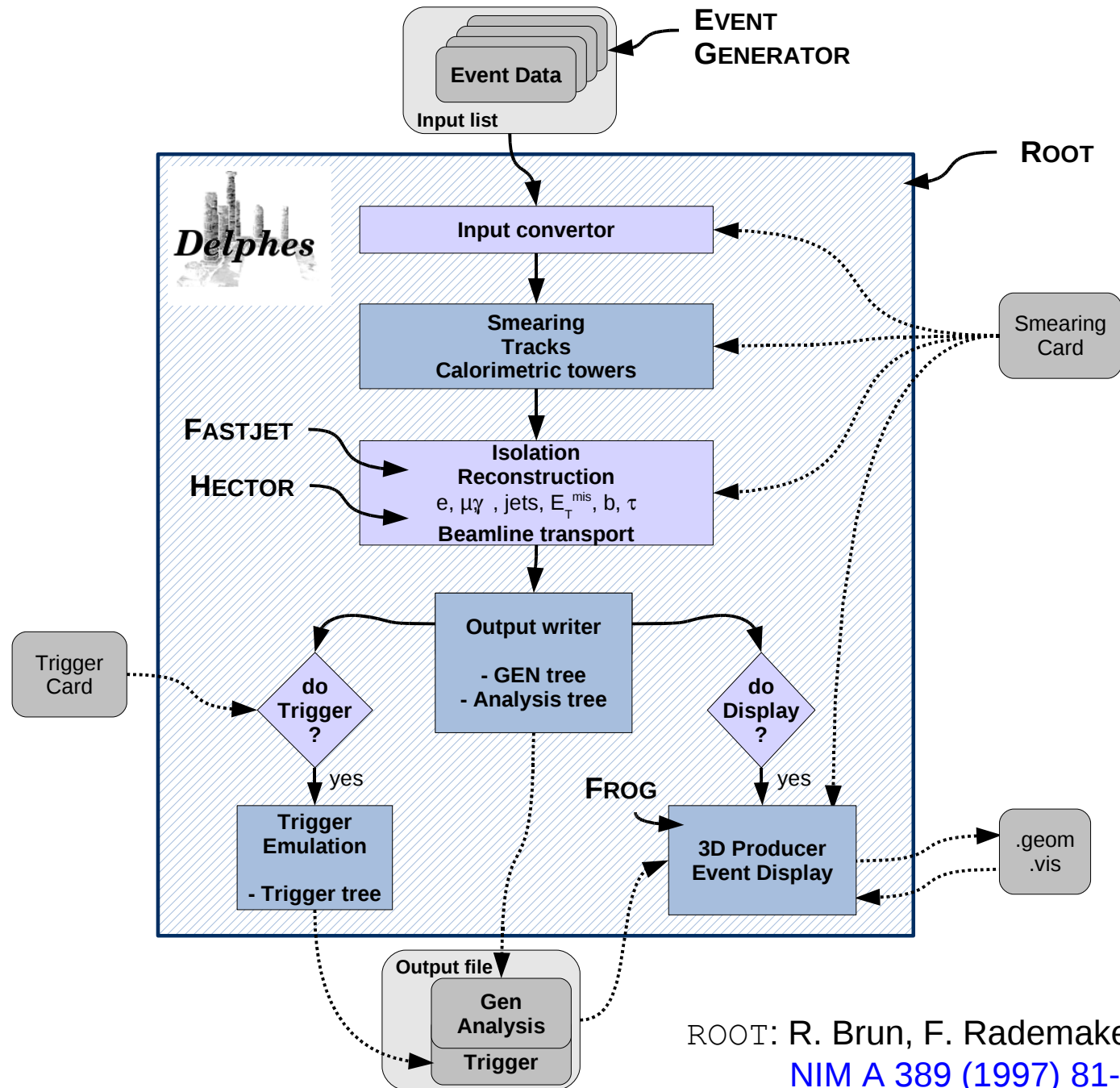
C++/ROOT implementation



X. Rouby

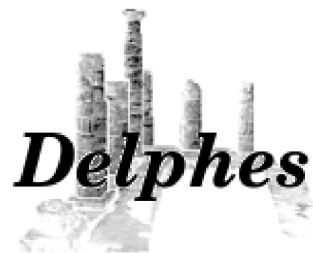
Motivations
Simulation
BUT also
Conclusion

21/05/2009



ROOT: R. Brun, F. Rademakers,
NIM A 389 (1997) 81-86.

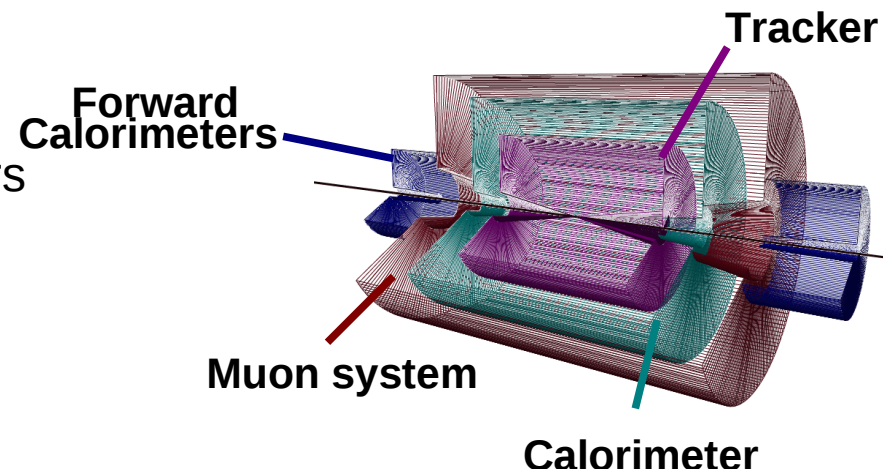
Detector Card : layout



Delphes simulates a generic HEP detector:

All parameters describing the detector are defined in the **detector card**:

- Subdetector extensions/parameters
- Expected resolutions
- Paths to external input files (beam optics, PDG particles)
- ...



X. Rouby

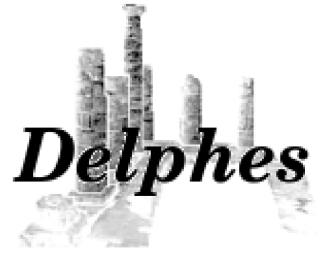
Central detector components:

```
DETECTOR CARD          # DO NOT REMOVE THIS IS A TAG!  
  
# Detector extension, in pseudorapidity units  
CEN_max_tracker        2.5    // Maximum tracker coverage  
CEN_max_calor_cen      3.0    // central calorimeter coverage  
CEN_max_calor_fwd     5.0    // forward calorimeter pseudorapidity coverage  
CEN_max_mu             2.4    // muon chambers pseudorapidity coverage
```

Symmetries :

$\eta > 0 \leftrightarrow \eta < 0$ (pseudorapidity)
All- ϕ (azimuth)

Detector Card : layout



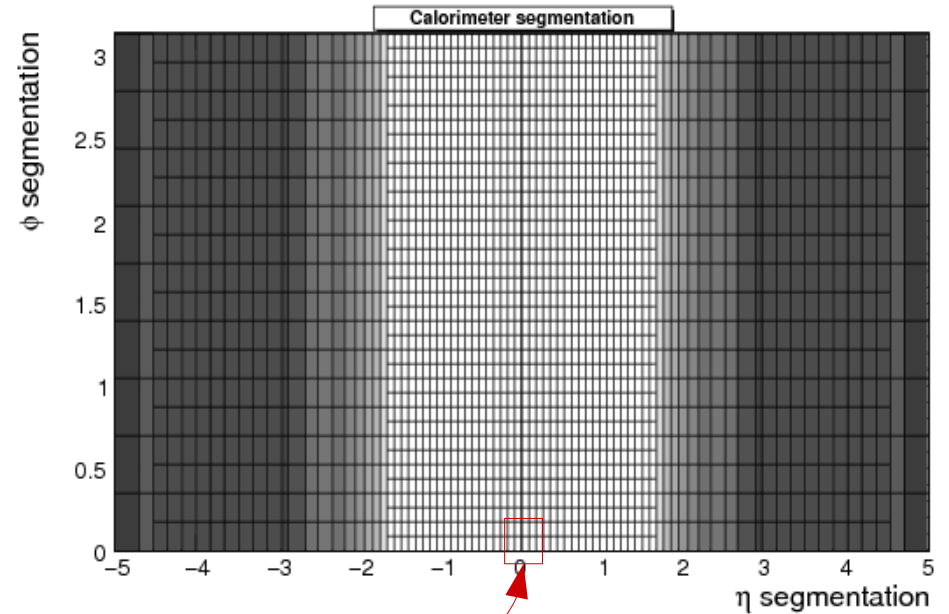
X. Rouby

Central detector components:

calorimetric tower segmentation

List of calotowers:

- edges in eta
- size in phi (degrees)



```
# Calorimetric towers
TOWER_number 40
TOWER_eta_edges 0. 0.087 0.174 0.261 0.348 0.435 0.522 0.609
0.696 0.783 0.870 0.957 1.044 1.131 1.218 1.305 1.392 1.479
1.566 1.653 1.740 1.830 1.930 2.043 2.172 2.322 2.500 2.650
2.868 2.950 3.125 3.300 3.475 3.650 3.825 4.000 4.175 4.350
4.525 4.700 5.000
## list of the edges of each tower in eta for eta>0 assuming
a symmetric detector in eta<0
TOWER_dphi 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 20 20
### list of the tower size in phi (in degrees)
```

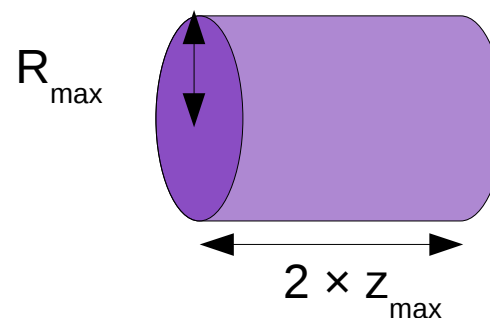
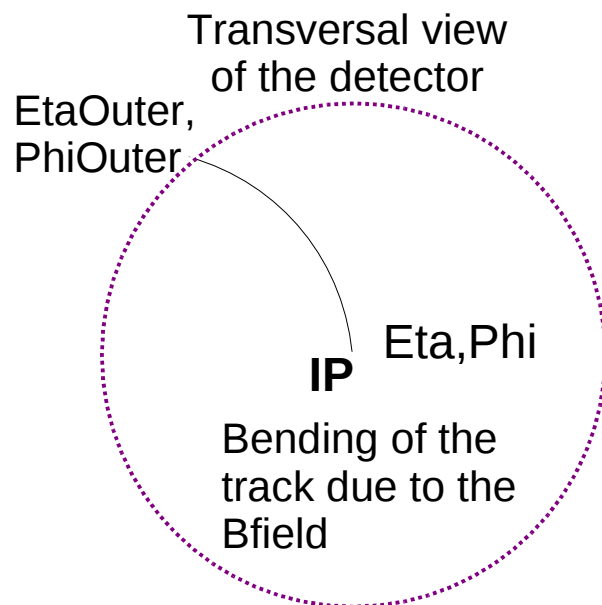
Detector Card : layout



Central detector components:

Solenoidal magnetic field parameters, for the tracking system:

```
# In case BField propagation allowed  
  
TRACK_radius      129 //radius of the BField coverage, in cm  
TRACK_length      300 //length of the BField coverage, in cm  
TRACK_bfield_x    0    //X component of the BField, in T  
TRACK_bfield_y    0    //Y component of the BField, in T  
TRACK_bfield_z    400 //Z component of the BField, in T
```



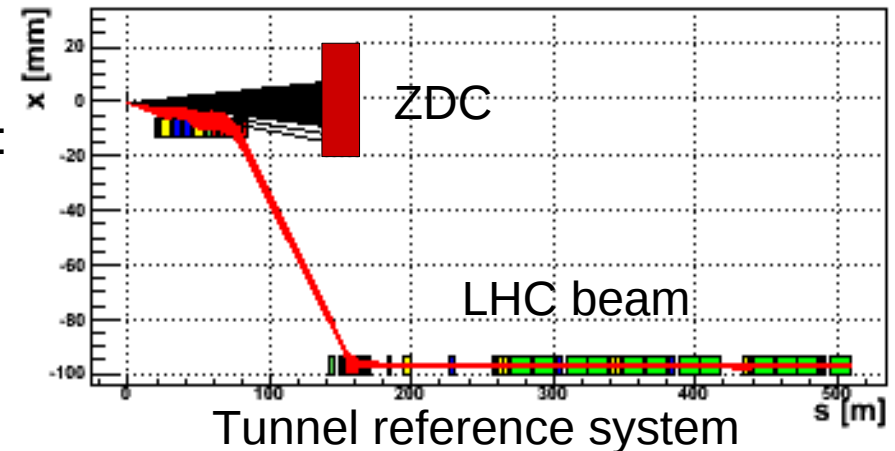
Detector Card : layout



Forward detectors: zero-degree calorimeters (ZDC)

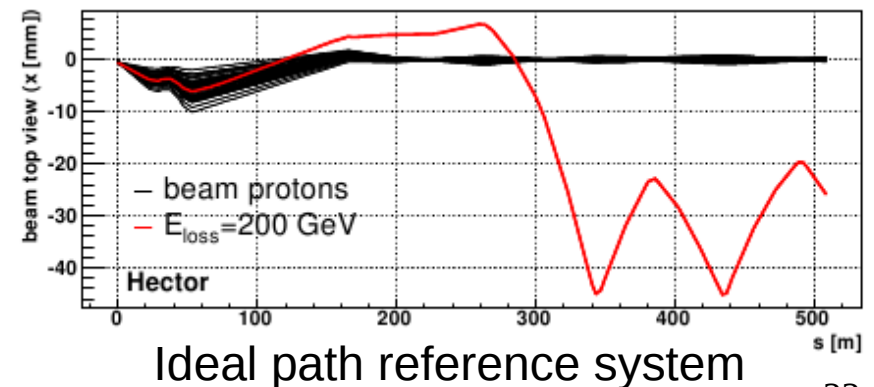
```
VFD_min_zdc  8.3  // zero-degree neutral calorimeter coverage, in eta  
VFD_s_zdc    140  // distance to the IP, in meters
```

Zero degree calorimeters are located after the beam separation : see all neutrons/photons directly coming from the IP



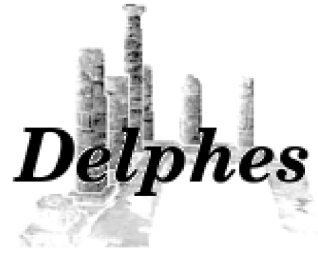
Forward detectors: roman pots (RP220) / taggers (FP420)

Roman pots and forward taggers see beam particles *elastically* scattered with some energy loss, at the IP



X. Rouby

Detector Card : layout



Forward detectors: roman pots/taggers

X. Rouby

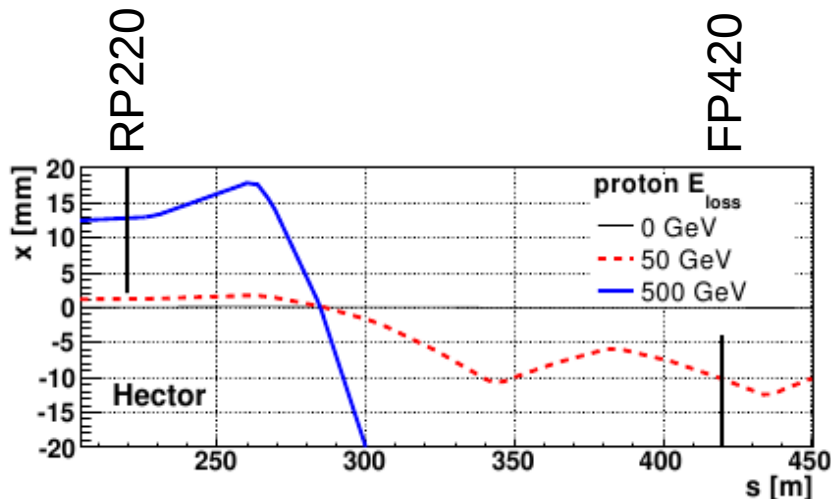
```
#Hector parameters
RP_220_s      220    // distance of the RP to the IP, in meters
RP_220_x      0.002  // distance of the RP to the beam, in meters

RP_offsetEl_x 0.097  // hor. separation between both beam, in meters
RP_offsetEl_y 0      // ver. separation between both beam, in meters
RP_offsetEl_s 120    // distance of beam separation point, from IP
RP_cross_x    -500   // IP offset in horizontal plane, in micrometers
RP_cross_y    0      // IP offset in vertical plane, in micrometers
RP_cross_ang_x 142.5 // half-crossing angle in hor. plane, in microrad
RP_cross_ang_y 0     // half-crossing angle in ver. plane, in microrad
```

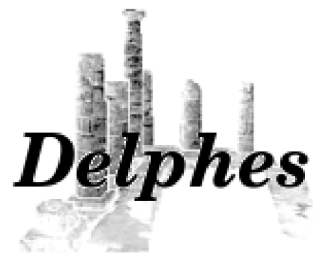
Parameters for Hector

Time resolution

```
ZDC_T_resolution 0 // in s
RP220_T_resolution 0 // in s
RP420_T_resolution 0 // in s
```



Detector Card : layout

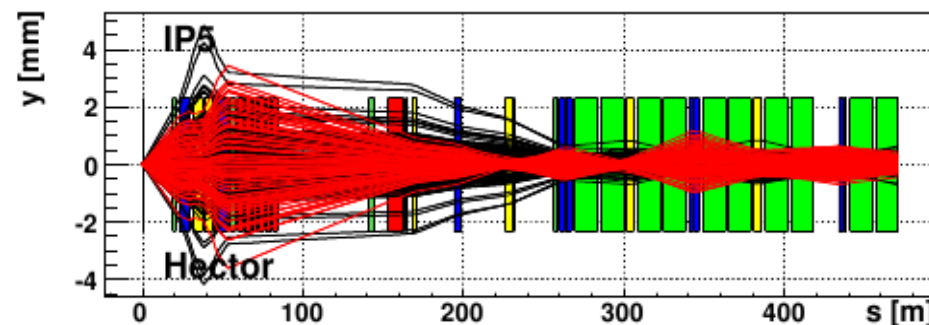
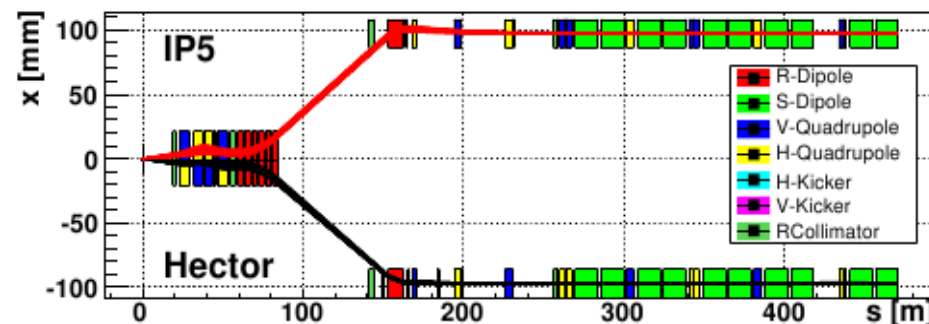


X. Rouby

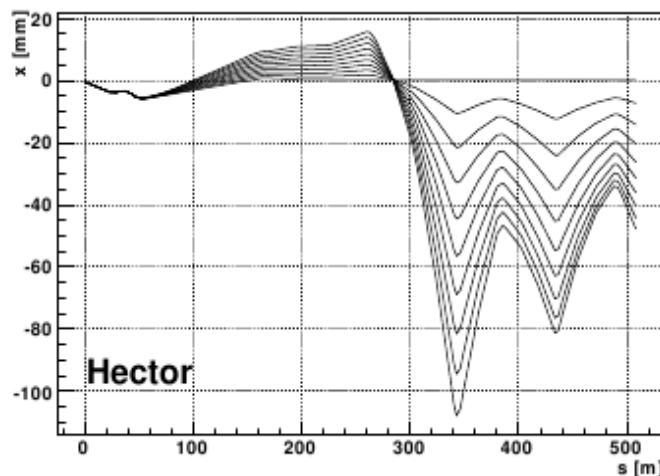
Forward detectors: roman pots/taggers

List of beam magnets:

- effective length,
- field strength,
- position,
- geometrical aperture

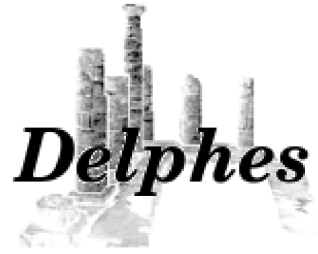


```
RP_beam1Card      data/LHCB1IR5_v6.500.tfs // beam optics file, beam 1
RP_beam2Card      data/LHCB2IR5_v6.500.tfs // beam optics file, beam 2
RP_IP_name        IP5      // tag for IP in Hector ; 'IP1' for ATLAS
```



LHC beam 1 or 2
Interaction region 1 (ATLAS)
or 5 (CMS)
7 TeV proton beams
in collision mode

Detector Card : resolutions



All calorimeters are simulated according to their **resolution**, parametrised with this formula:

$$\frac{\sigma}{E} = \frac{S}{\sqrt{E}} \oplus \frac{N}{E} \oplus C$$

Stochastic term Noise term Constant term

X. Rouby

```
# Energy resolution for electron/photon
# \sigma/E = C + N/E + S/\sqrt{E}, E in GeV
ELG_Scen      0.05      // S term for central ECAL
ELG_Ncen      0.25      // N term for central ECAL
ELG_Ccen      0.005     // C term for central ECAL
ELG_Sfwd      2.084     // S term for FCAL
ELG_Nfwd      0.0       // N term for FCAL
ELG_Cfwd      0.107     // C term for FCAL
ELG_Szdc      0.70      // S term for ZDC
ELG_Nzdc      0.0       // N term for ZDC
ELG_Czdc      0.08      // C term for ZDC
```

```
# Energy resolution for hadrons in ecal/hcal/hf
# \sigma/E = C + N/E + S/\sqrt{E}, E in GeV
HAD_Shcal     1.5       // S term for central HCAL
HAD_Nhcal     0.        // N term for central HCAL
HAD_Chcal     0.05      // C term for central HCAL
HAD_Shf       2.7       // S term for FCAL
HAD_Nhf       0.        // N term for FCAL
HAD_Chf       0.13      // C term for FCAL
HAD_Szdc      1.38      // S term for ZDC
HAD_Nzdc      0.        // N term for ZDC
HAD_Czdc      0.13      // C term for ZDC
```

electromagnetic sections:

- Central
- Forward
- ZDC

hadronic sections:

- Central
- Forward
- ZDC

Detector Card : resolutions



X. Rouby

Time resolution for forward detectors

```
# Time resolution for ZDC/RP220/RP420
ZDC_T_resolution 0 // in s
RP220_T_resolution 0 // in s
RP420_T_resolution 0 // in s
```

In reality, neither RP220 (Totem/IP5) nor ALFA (Atlas/IP1) can measure time of flight

Pt resolution on muons

```
MU_SmearPt 0.01 // in GeV
```

It is foreseen to have input distributions instead of flat probabilities for the smearing in future versions of *Delphes*.

Tracking efficiencies

```
# Tracking efficiencies
TRACK_ptmin 0.0 // minimal pt needed to reach the calorimeter in GeV
TRACK_eff 100 // efficiency associated to the tracking (%)
```

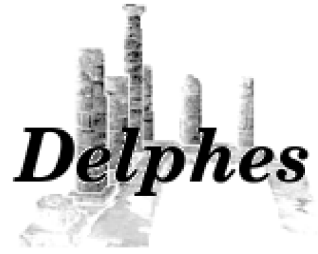
Reconstruction thresholds (Pt or E)

```
PTCUT_elec 10.0 // in GeV
PTCUT_muon 10.0
PTCUT_jet 20.0
PTCUT_gamma 10.0
PTCUT_taujet 10.0

ZDC_gamma_E 20
ZDC_n_E 50
```

Real experiments can *not* reconstruct complex objects at a too small scale (Pt / E).

Detector Card : misc.



X. Rouby

Various flags to switch on/off some components

```
FLAG_bfield      1  //1 to run the bfield propagation else 0
FLAG_vfd         1  //1 to run the very forward detectors else 0
FLAG_RP          1  //1 to run the very forward detectors else 0
FLAG_trigger     1  //1 to run the trigger selection else 0
FLAG_frog        0  //1 to run the FROG event display
FLAG_lhco        0  //1 to run the LHCO
```

It is possible *a posteriori* to run the trigger, to create the LHCO output file or to generate the input files for FROG

./Trigger_Only

Usage: `./Trigger_Only input_file output_file [detector_card]`
`[trigger_card]`
input_file - file in Delphes root format,
trigger_card - Datacard containing the trigger algorithms
(optional)

./LHCO_Only

Usage: `./LHCO_Only input_file [runlog_file]`
input_file - file in Delphes root format,
[runlog_file] - the corresponding log file (optional)

./Frog_on_analysis_output

Usage: `./Frog_on_analysis_output input_file [N_events]`
input_file - root file containing the events to display
detector_card - Datacard containing resolution variables for the
detector simulation (optional)

Detector Card : misc.



X. Rouby

Similarly, it is possible *a priori* to run the converter stage, to translate the input data (*hep, *hepmc, *root, *lhe) into a **Delphes** ROOT file (GEN tree).

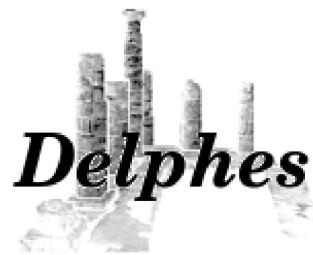
./Convertors_Only

```
Usage: ./Convertors_Only input_file output_file PDG_Table
input_list - list of files in Ntpl, StdHep or LHEF format,
output_file - output file,
PDG_Table - file with PDG particle table
```

./Convertors_Only TEST_small_tt.list test.root data/particle.tbl

```
...
root -l test.root
...
.ls
TFile**          test.root
TFile*           test.root
KEY: TTree       GEN;1  Analysis tree
```

Jet algorithms.



X. Rouby

Many parameters are used for the jet definition/tuning

```
JET_coneradius 0.7 // generic jet radius ; not for tau's !!!  
JET_jetalgo    1    // 1 for Cone algorithm, 2 for MidPoint algorithm,  
                // 3 for SIScone algorithm, 4 for kt algorithm  
                // 5 Cambridge/Aachen, 6 anti-kt  
JET_seed       1.0  // minimum seed to start jet reconstruction, in GeV  
JET_Eflow      1    // Perfect energy assumed in the tracker coverage
```

1 is on (Energy flow) ; 0 is off

b-tagging / misidentification: flat distributions

```
BTAG_b         40   // b-tag efficiency (%)  
BTAG_mistag_c  10   // mistagging (%)  
BTAG_mistag_l  1    // mistagging (%)
```

Leptons + misc.



X. Rouby

Lepton isolation

```
# Charged lepton isolation. Pt and Et in GeV
ISOL_PT          2.0          //minimal pt of tracks for isolation criteria
ISOL_Cone        0.5          //Cone for isolation criteria
ISOL_Calo_ET     2.0
//minimal tower transverse energy for isolation criteria. 1E99 means "off"
ISOL_Calo_Grid   3           //Grid size (N x N) for calorimetric isolation
```

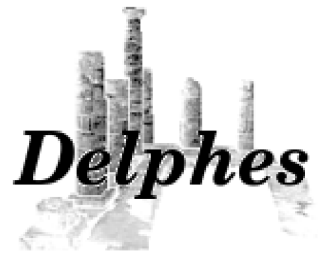
For the leptons, a tracking isolation is applied, with such parameters.

In particular, for muons, some information is also available on the Calorimetric isolation.

Others...

```
PdgTableFilename data/particle.tbl
NEvents_Frog      100
```

Trigger Card

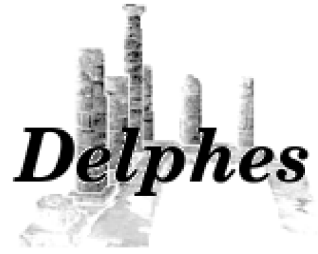


X. Rouby

```
# trigger_name      >> algorithm      #comments
Inclusive electron  >> ELEC1_PT:      '29'
di-electron         >> ELEC1_PT:      '17'    &&    ELEC2_PT:      '17'
Inclusive Photon    >> GAMMA1_PT:     '80'
di-Photon           >> GAMMA1_PT:     '40'    &&    GAMMA2_PT:     '25'
Inclusive muon      >> MUON1_PT:      '19'
di-muon             >> MUON1_PT:      '7'     &&    MUON2_PT:      '7'
Taujet and ETmis    >> TAU1_PT:       '86'    &&    ETMIS_PT:      '65'
di-Taujets          >> TAU1_PT:       '59'    &&    TAU2_PT:       '59'
Jet and ETmis       >> JET1_PT:       '180'   &&    ETMIS_PT:      '123'
Taujet and electron >> TAU1_PT:       '45'    &&    ELEC1_PT:      '19'
Taujet and muon     >> TAU1_PT:       '40'    &&    ELEC1_PT:      '15'
Inclusive b-jet     >> Bjet1_PT:      '237'
Inclusive 1 jet     >> JET1_PT:       '657'
Inclusive 3 jets    >> JET1_PT:       '247'   &&    JET2_PT:       '247'
&&      JET3_PT:   '247'
```

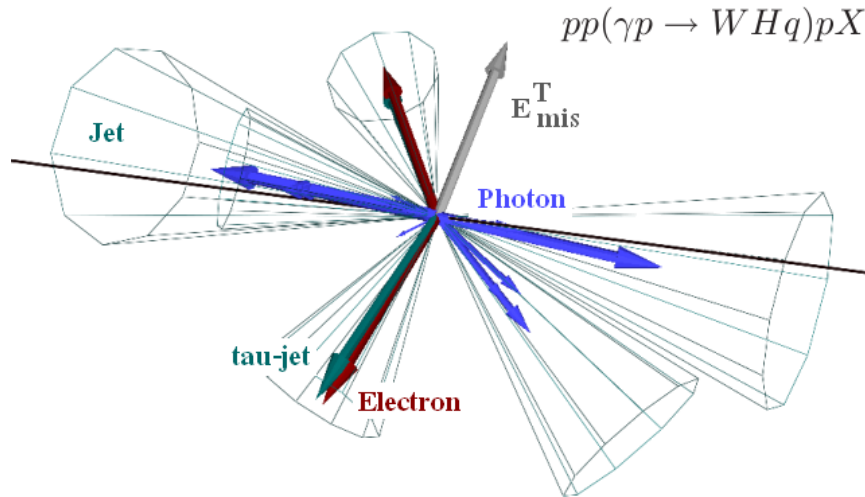
	<i>Trigger code</i>	<i>Corresponding object</i>
Cuts on Pt in GeV	ELEC_PT	electron
	IElec_PT	isolated electron
logical AND operator	MUON_PT	muon
	IMuon_PT	isolated muon
	JET_PT	jet
	TAU_PT	τ -jet
	ETMIS_PT	missing transverse energy
	GAMMA_PT	photon
	Bjet_PT	b-jet

FROG



3D Event Display

FROG interfaced to *Delphes*



FROG: L. Quertenmont, V. Roberfroid,
[arXiv:0901.2718v1\[hep-ex\]](https://arxiv.org/abs/0901.2718v1)

To run FROG, some libraries are needed, as it uses the OpenGL free libraries, which are not in Delphes

```
OpenGL:
xorg-x11-Mesa-libGL-6.8.2-1.EL.33.0.2
xorg-x11-Mesa-libGLU-6.8.2-1.EL.33.0.2
GLUT:
freeglut-2.2.0-14
freeglut-devel-2.2.0-14
X11-devel:
xorg-x11-devel-6.8.2-1.EL.33.0.2
CURL:
libcurl
```

FROG runs on two files : *geom and *vis, which are created if the flag is ON

```
FLAG_frog      1  //1 to run the FROG event display
```

Then compile the source (NOT done automatically when Delphes is compiled)

```
cd Utilities/FROG
make
...
cd ../..
./Utilities/FROG/frog
```

Compile it...

...Run it

X. Rouby

FROG



X. Rouby

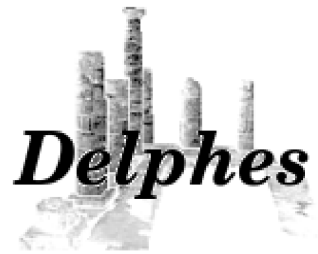
FROG Displayer Press F1 for Help
130 FPS

#Run 1 #Event 33 (33/99)
No TimeStamp

The interface displays three views of the detector geometry: a top-down circular view, a side-view cross-section, and a large 3D perspective view. The 3D view shows a complex cylindrical structure with internal components, overlaid with a network of colored lines representing particle tracks. A legend in the bottom right corner provides control instructions.

F1 : help
m : zoom out
p : zoom in
1 : without detector display
0 : with detector display
right arrow : next event
left arrow : previous event
ESC : exit

A small green cartoon frog is positioned in the bottom left corner of the window. The word "FROG" is written in yellow capital letters on the frog's back.



X. Rouby

That's all for today...

*All comments and feedback
are welcome!*