How-to tb_recon

A refresher course in obtaining, compiling, and running centella

PLUS! How-to read in both raw & recon files at the same time!!!

Current Status

- We're on the brink of our first release!
- Latest Build: B071000
- MyAnalysis is now available in CVS
  Handles both raw & recon data

Before anything else:
Get TBEvent and compile it

- cvs co TBEvent
- Windows: open TBEvent.dsw and compile the Debug/Release version of TBEvent
- Unix: under the TBEvent directory
  autogen, configure, make (gmake)
- Create a TBEVENT_DIR environment variable
  see tb_recon documentation for details.

Now you're ready to get tb_recon

- Get the code via CVS

  cvs co tb_recon

  cvs co -r B071000 tb_recon

- Useful documentation is available in the tb_recon/documents directory

  Open centella.html to begin views docs.

Compiling tb_recon: *Windows*

Open tb_recon/centella1.dsw in VC++

Choose the “all” project, Debug or Release

Under the Build menu “Rebuild All”

This will build centella, centallaGUI, and Recon.dll

By default your executables & libraries will be located:

tb_recon\centella\Debug\centella.exe

tb_recon\centella\Release\centella.exe

tb_recon\centellaGUI\Debug\centellaGUI.exe

tb_recon\centellaGUI\Release\centellaGUI.exe

tb_recon\RootTree\Debug\Recon.dll

tb_recon\RootTree\Release\Recon.dll

Compiling tb_recon: *Linux*

Under the tb_recon directory:

autogen
configure
make

This will build centella, centellaGUI and libRecon.so (needed to read recon files in ROOT)

The executables & library are located:

tb_recon/centella/centella

tb_recon/centellaGUI/centellaGUI

tb_recon/RootTree/.libs/libRecon.so

Compiling tb_recon: *Solaris*

Under the tb_recon directory:

autogen
configure --disable-shared
gmake

From here it's the same as Linux.

NOTE: SLAC users, tb_recon is already available on your afs - the most recent version is B071000

Running tb_recon

- All options / parameters for tb_recon are controlled in the “centella.in” file
- Modify this file as you see fit. (within reason!)
- To run centella:
  centella centella.in
  centelleGUI centella.in

Now you have a Recon ROOT file

- Recall that MyEvent.c handles Raw tb files
- MyRecon.c handles Recon tb files
- Now there is MyAnalysis.c which can read in either raw and/or recon data.
  To get MyAnalysis.c:
  cvs co ROOTAnalysis
  The code is located under:
  ROOTAnalysis/MyAnalysis

Using MyAnalysis.c

- For more detailed instructions see:

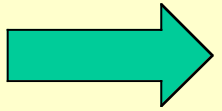
  http://www-sldnt.slac.stanford.edu/glast/ROOT/MyAnalysis/MyAnalysis.htm

- Modify startRecon.c

  The TBRECON_DIR environment variable is optional

startRecon.c loads the TBEvent & Recon libraries:

```
{
 new TBrowser;
 gSystem->Load("$TBEVENT_DIR/bin/Debug/TBEvent.dll");
 gSystem->Load("$TBRECON_DIR/RootTree/debug/Recon.dll");
}
```



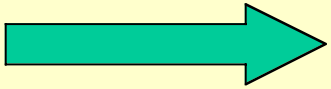
Using MyAnalysis Cont.

- Modify runAnalysis
  runAnalysis sets up your MyAnalysis object:

```
{
   gROOT->Reset();    // roll back CINT context to last Save
   gROOT->LoadMacro("MyAnalysis.c");  // load your code
   // Update for your location of Root files
   MyAnalysis* m= new MyAnalysis(
    "I:/Data/rootfiles/run943.root",
    "I:/CentellaB071000/tb_recon/output/reconrun943.root");
}
```

The MyAnalysis contructor accepts up to 2 arguments

MyAnalysis(“raw.root”, “recon.root”)

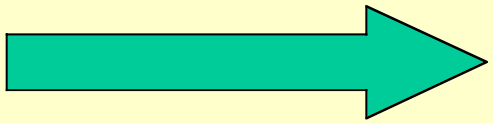


Using MyAnalysis Cont.

- If interested in only one “type” of file, you may set either filename to the null (“”) string.
  Ex) MyAnalysis(“”, “myRecon.root”);
- Now fire up ROOT

```
root [0] .x startRecon.c
root [1] .x runAnalysis
```

You are now ready to start processing your ROOT file(s).

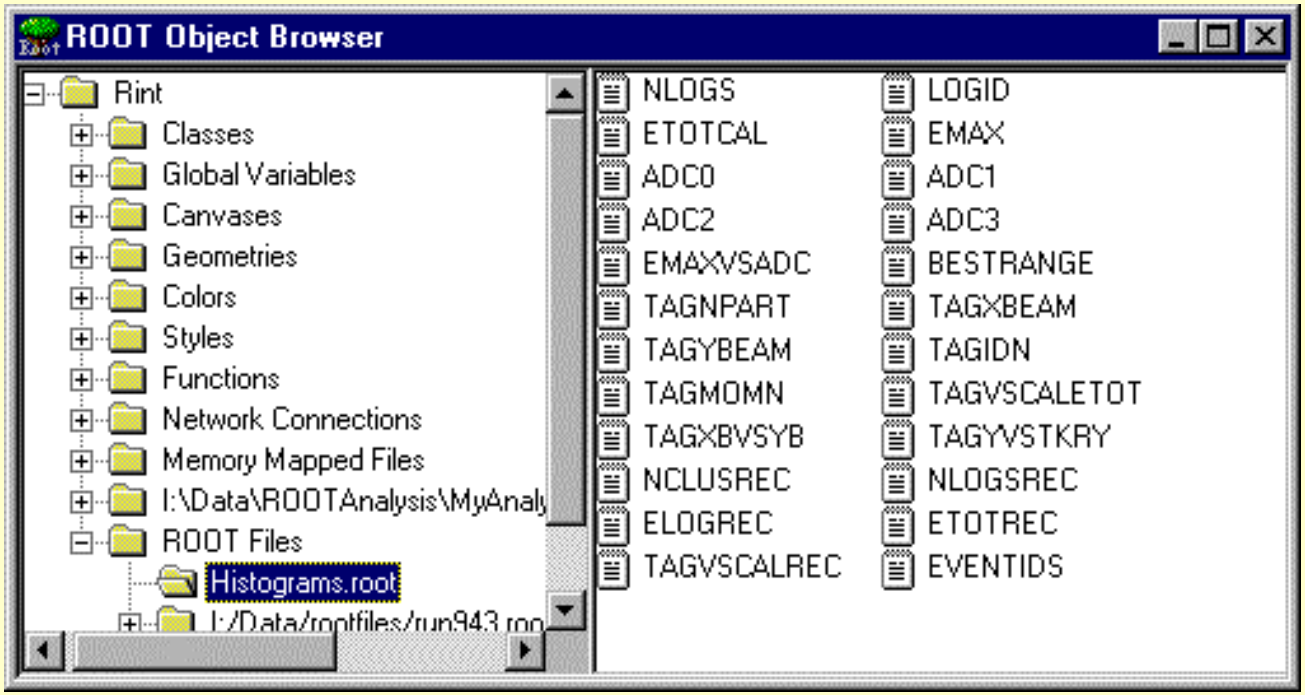


Processing Events

- By executing runAnalysis, a new object called "m" was created.
- To Read & process 100 events:

  m->Go(100)

  View the generated histograms in the TBrowser window.



Processing Events Cont.

- To “rewind” back to the beginning of the file(s):
  m->Rewind()
- To setup our starting event number:
  m->StartWithEvent(int eventNum)
- To clear histograms:
  m->HClr()
- To open up new files:
  m->Init(“raw.root”, “recon.root”)
  NOTE: This does not clear the histograms!

Modifying MyAnalysis

- Only necessary to modify MyAnalysis.c, not .h
- To add your own histograms:

  Add a histogram definition to the MyAnalysis::HistDefine() function,

  ex) TH1F* myHist = new TH1F("ID", "Title", nBins,
  Bin1_lowerEdge, BinN_upperEdge);

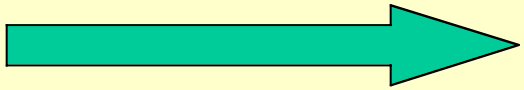
  Next modify MyAnalysis::Go()

  Under // refresh your histogram pointers here insert a line such as:

  TH1F* myHist = (TH1F*) histFile->Get("ID");

  Then somewhere in the analysis loop you'll fill the histogram:

  myHist->Fill(myData);



Modifying MyAnalysis Cont.

- Analysis portion of the event loop has 3 parts:
  1.) raw-only processing
  2.) recon-only processing
  3.) raw & recon processing
- There are examples of each type in MyAnalysis.c.
- To create a histogram that uses raw & recon data:
  Create the histogram object as outlined.
  Fill the histogram as you would any other:
  EVENTIDS->Fill(float(rawEventID), float(reconEventID));

That's It!

- Just save your changes to MyAnalysis.c, go back to Root and run your new version.