Table of Contents

Adding histogram filling and saving to the duplicated Athena Hello World

Introduction

This page describes how to modify the Hello World algorithm in order to create a histogram filled with dummy values and save it to an output root file. We will make the changes to the duplicated Athena Hello World so that later on we can compare with the original Athena Hello World code and see the things we added.

First, log into your account and set up CMT (as described in WorkBookSetAccount). Assuming you are running in a folder with the same name as the Athena version and located in the ~/testarea, if you have a bash shell define this variable which will help as move from a folder to another folder easily while allowing us to use the same copy paste commands for different versions of Athena.

```
export ATHENA_VERSION=17.0.5.5.2
cd ~/testarea/$ATHENA_VERSION
asetup $ATHENA_VERSION,here
```

Edit the HelloAlgNEW files

```
cd ~/testarea/$ATHENA_VERSION/Control/AthenaExamples/AthExHelloWorld/src
```

In the HelloAlgNew.h add the following include statements

```
#include "GaudiKernel/ITHistSvc.h"
#include "TH1.h"
```

and the following private variables

```
  ITHistSvc *m_thistSvc;
  TH1D *m_hist_jetPt;
  double m_jetPt;
```

In the HelloAlgNew.cxx add the following code in the initialize() method to start the histogram service

```
  //NEW start
  StatusCode status;

  //service for histograms and trees (although the name includes just hist)
  status = service("THistSvc",m_thistSvc);
  //check if the status is successful
  if (status.isFailure()) {
    ATH_MSG_ERROR("No THistSvc!!!!!!!");
    return StatusCode::FAILURE;
  }
```

Please note that we have to check if the status is successful every time we use it. And just under it we define and initialize a histogram and again check the status.

```
  //define and initialize the histogram
  m_hist_jetPt = new TH1D("jetPt","Jet p_{T} ; p_{T} [GeV/c] ; Entries", 200, 0, 200);
  //book the histogram with the histogram/tree service
  status = m_thistSvc->regHist("/file1/jetPt", m_hist_jetPt);
  //check if the status is successful
  if (status.isFailure()) {
    ATH_MSG_ERROR("No THistSvc!!!!!!!");
    return StatusCode::FAILURE;
  }

  //NEW end
```

Here "file1" represents the name of the output file, which we will define later in the option file, where we will also introduce the histogram service. Also, jetPt represents the name of the variable from the output root file. Then, in the execute() method we fill the histogram for every event with a dummy value of 100 GeV/c. First here is an example of how you can add log messages

```
  //NEW start
  MsgStream log( messageService(), name() );
  log << MSG::DEBUG <<"Analysis execute()" << endreq;
  log << MSG::INFO <<"Analysis execute()" << endreq;
```

and then

```
m_jetPt = 100.0;
m_hist_jetPt->Fill(m_jetPt);

//NEW end
```

Compile the package with the algorithms

```
cd ../cmt
gmake
```

But the compilation fails when it tries to compile the include TH1.h statement, because we have not linked the ROOT libraries. For this, we need to edit the "requirements" file

```
emacs -nw requirements
```

to add the line

```
use AtlasROOT       AtlasROOT-*         External
```

Now we must update the configuration, so that it uses the latest "requirements" file.

```
cmt config
```

Now we are ready to compile again and this time it will work.

```
gmake
```

Modifying the option file and running

```
cd ~/testarea/$ATHENA_VERSION/PhysicsAnalysis/AnalysisCommon/UserAnalysis/run
emacs -nw HelloWorldOptionsNEW.py
```

Before the end of the job, add the following lines. It should look like this

```
from AthenaCommon.AppMgr import ServiceMgr

from GaudiSvc.GaudiSvcConf import THistSvc
ServiceMgr += THistSvc()

ServiceMgr.THistSvc.Output += ["file1 DATAFILE='output.root' OPT='RECREATE'"]

print theApp.TopAlg
print "TEST OPTION FILE END"

#==============================================================
#
# End of job options file
#
###############################################################
```

Somewhat at the beginning of the file also add the print statement to see when processing of the options file starts. It should look like this.

```
# Full job is a list of algorithms
from AthenaCommon.AlgSequence import AlgSequence
job = AlgSequence()

print theApp.TopAlg
print "LOG OPTIONS FILE START"
```

Now are are ready to run

```
athena.py HelloWorldOptionsNEW.py >& test_hist.log
```

As it ran successfully, we can open the output.root file and check that it has our jetPt histogram, which we can draw and we will see 10 entries (since our option file tells to run over 10 events - note that there is a loop with 10 steps even if no input file with real events is present) of values of 100.

```
root.exe output.root
root [1] .ls
root [2] jetPt->Draw()
```

Major updates:
-- AdrianBuzatu - 25-Jan-2012

This topic: Sandbox > AdrianBuzatuAthenaAthenaHelloWorldAddHistogram
Topic revision: r3 - 2012-01-27 - AdrianBuzatu