h AFPEventSelectorEventSaver.h 5.36 KiB

```
#ifndef AFPEVENTSELECTOREVENTSAVER_H_
#define AFPEVENTSELECTOREVENTSAVER_H_

#include "TopAnalysis/EventSaverFlatNtuple.h"
#include "TRandom3.h"
#include "TH2.h"
#include "TopEventSelectionTools/PlotManager.h"
#include <AfpAnalysisTools/IAfpAnalysisTool.h>
#include "xAODForward/AFPSiHit.h"
#include "xAODForward/AFPSiHitContainer.h"
#include "xAODForward/AFPStationID.h"
#include "AsgTools/AsgTool.h"
#include "AsgTools/ToolHandle.h"
#include "AsgTools/ToolHandleArray.h"
#include "AsgTools/AnaToolHandle.h"

//#include "Math/LorentzVector.h"

using namespace ROOT::Math;

struct AFPCluster {

  AFPCluster(float x_, float y_, float z_, int s, int l):
    x {x_},
    y {y_},
    z {z_},
    station {s},
    layer {l} {}

    float x;
    float y;
    float z;
    int station;
    int layer;
};

inline bool operator==(const AFPCluster& lhs, const AFPCluster& rhs) {

  if (lhs.x != rhs.x) return false;
  if (lhs.y != rhs.y) return false;
  if (lhs.z != rhs.z) return false;
  if (lhs.station != rhs.station) return false;
  if (lhs.layer != rhs.layer) return false;

  return true;
}


struct AFPTrack {

  AFPTrack(float x_, float y_, int s, std::array<int, 4> a):
    x {x_},
    y {y_},
    station {s},
    layerClusters {std::move(a)} {}

    float x;
    float y;
    int station;
    std::array<int, 4> layerClusters;
};


class AFPEventSelectorEventSaver : public top::EventSaverFlatNtuple {

public:

  ///-- Default - so root can load based on a name --///
  AFPEventSelectorEventSaver();

  ///-- Exposition of base class initializers (to avoid [-Woverloaded-virtual] hidden warnings --///
  using top::EventSaverFlatNtuple::initialize;

  ///-- Run once at the start of the job --///
  void initialize(std::shared_ptr<top::TopConfig> config, TFile* file, const std::vector<std::string>& extraE

  ///-- For parton level objects --///
```