```
//
// ********************************************************************
// * License and Disclaimer                                           *
// *                                                                  *
// * The  Geant4 software  is  copyright of the Copyright Holders  of *
// * the Geant4 Collaboration.  It is provided  under  the terms  and *
// * conditions of the Geant4 Software License,  included in the file *
// * LICENSE and available at  http://cern.ch/geant4/license .  These *
// * include a list of copyright holders.                             *
// *                                                                  *
// * Neither the authors of this software system, nor their employing *
// * institutes,nor the agencies providing financial support for this *
// * work  make  any representation or  warranty, express or implied, *
// * regarding  this  software system or assume any liability for its *
// * use.  Please see the license in the file  LICENSE  and URL above *
// * for the full disclaimer and the limitation of liability.         *
// *                                                                  *
// * This  code  implementation is the result of  the  scientific and *
// * technical work of the GEANT4 collaboration.                      *
// * By using,  copying,  modifying or  distributing the software (or *
// * any work based  on the software)  you  agree  to acknowledge its *
// * use  in  resulting  scientific  publications,  and indicate your *
// * acceptance of all terms of the Geant4 Software license.          *
// ********************************************************************
//
//
// $Id$
//
//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......
//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

#include "HEPDSWDetectorMessenger.hh"

#include <sstream>

#include "HEPDSWDetectorConstruction.hh"
#include "G4UIdirectory.hh"
#include "G4UIcommand.hh"
#include "G4UIparameter.hh"
#include "G4UIcmdWithAnInteger.hh"
#include "G4UIcmdWithADoubleAndUnit.hh"
#include "G4UIcmdWithoutParameter.hh"
#include "G4UIcmdWithABool.hh"
#include "G4UIcmdWithAString.hh"

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

HEPDSWDetectorMessenger::HEPDSWDetectorMessenger(HEPDSWDetectorConstruction * De
t)
:Detector(Det)
{
  G4UIparameter* param;
  fHepdDir = new G4UIdirectory("/hepd/");
  fHepdDir->SetGuidance("UI commands specific to this example");

  fWorldSizeCmd = new G4UIcommand("/hepd/setWorldSize",this);
  fWorldSizeCmd->SetGuidance("Set world dimension");
  fWorldSizeCmd->AvailableForStates(G4State_PreInit,G4State_Idle);
  param = new G4UIparameter("X",'d',false);
  param->SetGuidance("X dimension");
  param->SetParameterRange("X>0");
  fWorldSizeCmd->SetParameter(param);
  param = new G4UIparameter("Y",'d',false);
  param->SetGuidance("Y dimension");
  param->SetParameterRange("Y>0");
  fWorldSizeCmd->SetParameter(param);
  param = new G4UIparameter("Z",'d',false);
  param->SetGuidance("Z dimension");
  param->SetParameterRange("Z>0");
```

```
  fWorldSizeCmd->SetParameter(param);
  param = new G4UIparameter("unit",'s',false);
  param->SetGuidance("length unit");
  fWorldSizeCmd->SetParameter(param);

  fCaloActivateCmd = new G4UIcmdWithABool("/hepd/ActivateCalo",this);
  fCaloActivateCmd->SetGuidance("Enable or disable the calorimeter");
  fCaloActivateCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fTrackerActivateCmd = new G4UIcmdWithABool("/hepd/ActivateTracker",this);
  fTrackerActivateCmd->SetGuidance("Enable or disable the tracker");
  fTrackerActivateCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

//   fScintillatorActivateCmd = new G4UIcmdWithABool("/hepd/ActivateScintillator
",this);
//   fScintillatorActivateCmd->SetGuidance("Enable or disable the scintillator")
;
//   fScintillatorActivateCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fSatelliteActivateCmd = new G4UIcmdWithABool("/hepd/ActivateSatellite",this);
  fSatelliteActivateCmd->SetGuidance("Enable or disable the satellite components");
  fSatelliteActivateCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fHEPDBoxActivateCmd = new G4UIcmdWithABool("/hepd/ActivateHEPDBox",this);
  fHEPDBoxActivateCmd->SetGuidance("Enable or disable the HEPD Box components");
  fHEPDBoxActivateCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fCaloConfigCmd = new G4UIcmdWithAString("/hepd/CaloConfiguration",this);
  fCaloConfigCmd->SetGuidance("Select the calorimeter configuration");
  fCaloConfigCmd->SetParameterName("CaloConfiguration",false);
  fCaloConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

//   fScintConfigCmd = new G4UIcmdWithAString("/hepd/ScintillatorConfiguration",
this);
//   fScintConfigCmd->SetGuidance("Select the scintillator configuration");
//   fScintConfigCmd->SetParameterName("ScintConfiguration",false);
//   fScintConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fTrackerConfigCmd = new G4UIcmdWithAString("/hepd/TrackerConfiguration",this);
  fTrackerConfigCmd->SetGuidance("Select the tracker configuration");
  fTrackerConfigCmd->SetParameterName("TrackerConfiguration",false);
  fTrackerConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fSatelliteConfigCmd = new G4UIcmdWithAString("/hepd/SatelliteConfiguration",this);
  fSatelliteConfigCmd->SetGuidance("Select the satellite configuration");
  fSatelliteConfigCmd->SetParameterName("SatelliteConfiguration",false);
  fSatelliteConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fHEPDBoxConfigCmd = new G4UIcmdWithAString("/hepd/HEPDBoxConfiguration",this);
  fHEPDBoxConfigCmd->SetGuidance("Select the HEPD Box configuration");
  fHEPDBoxConfigCmd->SetParameterName("HEPDBoxConfiguration",false);
  fHEPDBoxConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

//   fCaloCaloMatConfigCmd = new G4UIcmdWithAString("/hepd/Calorimeter/Calorimet
erMaterialConfiguration",this);
//   fCaloCaloMatConfigCmd->SetGuidance("Set the calorimeter material");
//   fCaloCaloMatConfigCmd->SetParameterName("Calorimeter Material",false);
//   fCaloCaloMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fCaloCalo2MatConfigCmd = new G4UIcommand("/hepd/Calorimeter/CalorimeterTwoMaterialConfigur
ation",this);
  fCaloCalo2MatConfigCmd->SetGuidance("Set the calorimeter material");
  param = new G4UIparameter("ScintillatorMaterial",'s',false);
  param->SetGuidance("Scintillator Material");
  fCaloCalo2MatConfigCmd->SetParameter(param);
  param = new G4UIparameter("CrystalMaterial",'s',false);
  param->SetGuidance("Crystal Material");
  fCaloCalo2MatConfigCmd->SetParameter(param);
  fCaloCalo2MatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);
  fCaloVetoMatConfigCmd = new G4UIcmdWithAString("/hepd/Calorimeter/VetoMaterialConfiguratio
n",this);
  fCaloVetoMatConfigCmd->SetGuidance("Set the veto material");
  fCaloVetoMatConfigCmd->SetParameterName("Veto Material",false);
  fCaloVetoMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fCaloPoronMatConfigCmd = new G4UIcmdWithAString("/hepd/Calorimeter/PoronMaterialConfigura
tion",this);
  fCaloPoronMatConfigCmd->SetGuidance("Set the poron material");
  fCaloPoronMatConfigCmd->SetParameterName("Poron Material",false);
  fCaloPoronMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fCaloCarbonFiberMatConfigCmd = new G4UIcmdWithAString("/hepd/Calorimeter/CarbonFiberM
aterialConfiguration",this);
  fCaloCarbonFiberMatConfigCmd->SetGuidance("Set the carbon fiber material");
  fCaloCarbonFiberMatConfigCmd->SetParameterName("CarbonFiber Material",false);
  fCaloCarbonFiberMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle)
;

  fCaloHoneyCombMatConfigCmd = new G4UIcmdWithAString("/hepd/Calorimeter/HoneyCombMate
rialConfiguration",this);
  fCaloHoneyCombMatConfigCmd->SetGuidance("Set the honey comb material");
  fCaloHoneyCombMatConfigCmd->SetParameterName("HoneyComb Material",false);
  fCaloHoneyCombMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fCaloCrystalLayerConfigCmd = new G4UIcmdWithAnInteger("/hepd/Calorimeter/CrystalLayerN
umberConfiguration",this);
  fCaloCrystalLayerConfigCmd->SetGuidance("Set the number of crystal layer calorimeter");
  fCaloCrystalLayerConfigCmd->SetParameterName("Number of Crystal LAyer",false);
  fCaloCrystalLayerConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

//   fScintScintMatConfigCmd = new G4UIcmdWithAString("/hepd/Scintillator/Scinti
llatorMaterialConfiguration",this);
//   fScintScintMatConfigCmd->SetGuidance("Set the scintillator material");
//   fScintScintMatConfigCmd->SetParameterName("Scintillator Material",false);
//   fScintScintMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

//   fScintPoronMatConfigCmd = new G4UIcmdWithAString("/hepd/Scintillator/PoronM
aterialConfiguration",this);
//   fScintPoronMatConfigCmd->SetGuidance("Set the poron material");
//   fScintPoronMatConfigCmd->SetParameterName("Poron Material",false);
//   fScintPoronMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

//   fScintCarbonFiberMatConfigCmd = new G4UIcmdWithAString("/hepd/Scintillator/
CarbonFiberMaterialConfiguration",this);
//   fScintCarbonFiberMatConfigCmd->SetGuidance("Set the carbon fiber material")
;
//   fScintCarbonFiberMatConfigCmd->SetParameterName("CarbonFiber Material",fals
e);
//   fScintCarbonFiberMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_I
dle);

  fTrackerSiliconMatConfigCmd = new G4UIcmdWithAString("/hepd/Tracker/SiliconMaterialConfi
guration",this);
  fTrackerSiliconMatConfigCmd->SetGuidance("Set the silicon material");
  fTrackerSiliconMatConfigCmd->SetParameterName("Silicon Material",false);
  fTrackerSiliconMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fTrackerKaptonMatConfigCmd = new G4UIcmdWithAString("/hepd/Tracker/KaptonMaterialConfig
uration",this);
  fTrackerKaptonMatConfigCmd->SetGuidance("Set the kapton material");
  fTrackerKaptonMatConfigCmd->SetParameterName("Kapton Material",false);
  fTrackerKaptonMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fTrackerCarbonFiberMatConfigCmd = new G4UIcmdWithAString("/hepd/Tracker/CarbonFiberM
aterialConfiguration",this);
  fTrackerCarbonFiberMatConfigCmd->SetGuidance("Set the carbon fiber material");
```

```
  fTrackerCarbonFiberMatConfigCmd->SetParameterName("CarbonFiber Material",false);
  fTrackerCarbonFiberMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Id
le);

  fHEPDBoxMLBlanketMatConfigCmd = new G4UIcommand("/hepd/HEPDBox/MLBlanketMaterialConfi
guration",this);
  fHEPDBoxMLBlanketMatConfigCmd->SetGuidance("Set the Thermal blanket material");
  param = new G4UIparameter("Mylar",'s',false);
  param->SetGuidance("Mylar");
  fHEPDBoxMLBlanketMatConfigCmd->SetParameter(param);
  param = new G4UIparameter("MylarCoating",'s',false);
  param->SetGuidance("Mylar Coating");
  fHEPDBoxMLBlanketMatConfigCmd->SetParameter(param);
  param = new G4UIparameter("Dracon",'s',false);
  param->SetGuidance("Dracon");
  fHEPDBoxMLBlanketMatConfigCmd->SetParameter(param);
  param = new G4UIparameter("Kapton",'s',false);
  param->SetGuidance("Kapton");
  fHEPDBoxMLBlanketMatConfigCmd->SetParameter(param);
  fHEPDBoxMLBlanketMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle
);

  fHEPDBoxBlanketMatConfigCmd = new G4UIcmdWithAString("/hepd/HEPDBox/BlanketMaterialC
onfiguration",this);
  fHEPDBoxBlanketMatConfigCmd->SetGuidance("Set the hepd blanket material");
  fHEPDBoxBlanketMatConfigCmd->SetParameterName("Blanket Material",false);
  fHEPDBoxBlanketMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fHEPDBoxWallMatConfigCmd = new G4UIcmdWithAString("/hepd/HEPDBox/WallMaterialConfigur
ation",this);
  fHEPDBoxWallMatConfigCmd->SetGuidance("Set the hepd box wall material");
  fHEPDBoxWallMatConfigCmd->SetParameterName("Wall Material",false);
  fHEPDBoxWallMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fHEPDBoxWallTwoMatConfigCmd = new G4UIcommand("/hepd/HEPDBox/WallTwoMaterialConfigurat
ion",this);
  fHEPDBoxWallTwoMatConfigCmd->SetGuidance("Set thehepd box  wall material");
  param = new G4UIparameter("WallBox",'s',false);
  param->SetGuidance("CF");
  fHEPDBoxWallTwoMatConfigCmd->SetParameter(param);
  param = new G4UIparameter("WallCore",'s',false);
  param->SetGuidance("AlHC");
  fHEPDBoxWallTwoMatConfigCmd->SetParameter(param);
  fHEPDBoxWallTwoMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

  fSatelliteBlanketMatConfigCmd = new G4UIcmdWithAString("/hepd/Satellite/BlanketMaterial
Configuration",this);
  fSatelliteBlanketMatConfigCmd->SetGuidance("Set the satellite blanket material");
  fSatelliteBlanketMatConfigCmd->SetParameterName("Blanket Material",false);
  fSatelliteBlanketMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle
);

  fSatelliteWallMatConfigCmd = new G4UIcmdWithAString("/hepd/Satellite/WallMaterialConfigur
ation",this);
  fSatelliteWallMatConfigCmd->SetGuidance("Set the satellite wall material");
  fSatelliteWallMatConfigCmd->SetParameterName("Wall Material",false);
  fSatelliteWallMatConfigCmd->AvailableForStates(G4State_PreInit,G4State_Idle);

}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

HEPDSWDetectorMessenger::~HEPDSWDetectorMessenger()
{
  delete fWorldSizeCmd;
  delete fCaloActivateCmd;
  delete fTrackerActivateCmd;
  //  delete fScintillatorActivateCmd;
  delete fSatelliteActivateCmd;
  delete fHEPDBoxActivateCmd;
  delete fCaloConfigCmd;
  //  delete fScintConfigCmd;
  delete fTrackerConfigCmd;
  delete fSatelliteConfigCmd;
  delete fHEPDBoxConfigCmd;
  delete fHepdDir;
  //  delete fCaloCaloMatConfigCmd;
  delete fCaloCalo2MatConfigCmd;
  delete fCaloVetoMatConfigCmd;
  delete fCaloPoronMatConfigCmd;
  delete fCaloCarbonFiberMatConfigCmd;
  delete fCaloHoneyCombMatConfigCmd;
  delete fCaloCrystalLayerConfigCmd;
  //  delete fScintScintMatConfigCmd;
  //  delete fScintPoronMatConfigCmd;
  //  delete fScintCarbonFiberMatConfigCmd;
  delete fTrackerSiliconMatConfigCmd;
  delete fTrackerKaptonMatConfigCmd;
  delete fTrackerCarbonFiberMatConfigCmd;
  delete fHEPDBoxMLBlanketMatConfigCmd;
  delete fHEPDBoxBlanketMatConfigCmd;
  delete fHEPDBoxWallMatConfigCmd;
  delete fHEPDBoxWallTwoMatConfigCmd;
  delete fSatelliteBlanketMatConfigCmd;
  delete fSatelliteWallMatConfigCmd;

}
//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......

void HEPDSWDetectorMessenger::SetNewValue(G4UIcommand* command,G4String newValue
)
{
  if (command == fWorldSizeCmd)
    {
      G4double Xdim,Ydim,Zdim;
      G4String unit;
      std::istringstream is(newValue);
      is >> Xdim >> Ydim >> Zdim >> unit;
      Xdim*= G4UIcommand::ValueOf(unit);
      Ydim*= G4UIcommand::ValueOf(unit);
      Zdim*= G4UIcommand::ValueOf(unit);
      Detector->SetWorldDimensions(Xdim/2.,Ydim/2.,Zdim/2.);
    }
  if (command == fCaloActivateCmd)
    Detector->SetCalorimeterDetector(fCaloActivateCmd->GetNewBoolValue(newValue)
);
  if (command == fTrackerActivateCmd)
    Detector->SetTrackerDetector(fTrackerActivateCmd->GetNewBoolValue(newValue))
;
  // if (command == fScintillatorActivateCmd)
//     Detector->SetScintillatorDetector(fScintillatorActivateCmd->GetNewBoolVal
ue(newValue));
  if (command == fSatelliteActivateCmd)
    Detector->SetSatelliteDetector(fSatelliteActivateCmd->GetNewBoolValue(newVal
ue));
  if (command == fHEPDBoxActivateCmd)
    Detector->SetHEPDBoxDetector(fHEPDBoxActivateCmd->GetNewBoolValue(newValue))
;
  if (command == fCaloConfigCmd)
    Detector->SetCaloConfiguration(newValue);
//   if (command == fScintConfigCmd)
//     Detector->SetScintConfiguration(newValue);
  if (command == fTrackerConfigCmd)
    Detector->SetTrackerConfiguration(newValue);
  if (command == fSatelliteConfigCmd)
    Detector->SetSatelliteConfiguration(newValue);
  if (command == fHEPDBoxConfigCmd)
    Detector->SetHEPDBoxConfiguration(newValue);
```

```

//   if(command == fCaloCaloMatConfigCmd)
//     Detector->CaloSetCaloMaterial(newValue);
  if(command == fCaloCalo2MatConfigCmd){
    G4String mat1,mat2;
    std::istringstream is(newValue);
    is >> mat1 >> mat2;
    Detector->CaloSetCaloMaterial(mat1,mat2);
  }
  if(command == fCaloVetoMatConfigCmd)
    Detector->CaloSetVetoMaterial(newValue);
  if(command == fCaloPoronMatConfigCmd)
    Detector->CaloSetPoronMaterial(newValue);
  if(command == fCaloCarbonFiberMatConfigCmd)
    Detector->CaloSetCarbonFiberMaterial(newValue);
  if(command == fCaloHoneyCombMatConfigCmd)
    Detector->CaloSetHoneyCombMaterial(newValue);
  if(command == fCaloCrystalLayerConfigCmd)
    Detector->CaloSetNumberOfCrystalLayer(fCaloCrystalLayerConfigCmd->GetNewIntV
alue(newValue));

//   if(command == fScintScintMatConfigCmd)
//     Detector->ScintillatorSetScintillatorMaterial(newValue);
//   if(command == fScintPoronMatConfigCmd)
//     Detector->ScintillatorSetPoronMaterial(newValue);
//   if(command == fScintCarbonFiberMatConfigCmd)
//     Detector->ScintillatorSetCarbonFiberMaterial(newValue);


  if(command == fTrackerSiliconMatConfigCmd)
    Detector->TrackerSetSiliconMaterial(newValue);
  if(command == fTrackerKaptonMatConfigCmd)
    Detector->TrackerSetKaptonMaterial(newValue);
  if(command == fTrackerCarbonFiberMatConfigCmd)
    Detector->TrackerSetCarbonFiberMaterial(newValue);

  if(command == fHEPDBoxMLBlanketMatConfigCmd){
    G4String mat1,mat2,mat3,mat4;
    std::istringstream is(newValue);
    is >> mat1 >> mat2 >> mat3 >> mat4;
    Detector->HEPDBoxSetBlanketMaterial(mat1,mat2,mat3,mat4);
  }
  if(command == fHEPDBoxWallMatConfigCmd)
    Detector->HEPDBoxSetWallMaterial(newValue);
  if(command == fHEPDBoxBlanketMatConfigCmd)
    Detector->HEPDBoxSetBlanketMaterial(newValue);
  if(command == fHEPDBoxWallTwoMatConfigCmd){
    G4String mat1,mat2;
    std::istringstream is(newValue);
    is >> mat1 >> mat2;
    Detector->HEPDBoxSetWallMaterial(mat1,mat2);
  }

  if(command == fSatelliteWallMatConfigCmd)
    Detector->SatelliteSetWallMaterial(newValue);
  if(command == fSatelliteBlanketMatConfigCmd)
    Detector->SatelliteSetBlanketMaterial(newValue);

}

//....oooOO0OOooo........oooOO0OOooo........oooOO0OOooo........oooOO0OOooo......
```