```
// ==================== SteppingAction.cc Class
================================ //
// Developed by Bryan V. Egner, Darren E. Holland, and Julie V. Logan
// Modified by Darren Holland 2020-11-02
// ========================================================================
//
// This tracks the total energy deposited by a particle for each thread during
// a step
// ========================================================================
//
#include "B4aSteppingAction.hh"
#include "B4aEventAction.hh"
#include "B4DetectorConstruction.hh"
#include "G4UnitsTable.hh"
#include "G4SystemOfUnits.hh"
//
#include "G4Step.hh"
#include "G4RunManager.hh"
//
#include "Randomize.hh"
#include <iomanip>
#include <iostream>
#include <fstream>
#include <stdio.h>
#include <math.h>
#include <Settings.hh>
//
#include "G4UImanager.hh"
#include "G4UIcommand.hh"
#include <boost/filesystem.hpp>

using namespace std;

// Start step instance
B4aSteppingAction::B4aSteppingAction(B4aEventAction* eventAction) :
G4UserSteppingAction(),fEventAction(eventAction)
{}

// Destroy step instance when completed
B4aSteppingAction::~B4aSteppingAction()
{}

// During event add energy deposited for current step for current thread
void B4aSteppingAction::UserSteppingAction(const G4Step* theStep)
{
  // Get Event #
  G4int eID = 0;
  const G4Event* evt = G4RunManager::GetRunManager()->GetCurrentEvent();
  if(evt) eID = evt->GetEventID(); // Pulls the current ID number (particle
number) if it exists
  // Get volume of the event
  G4VPhysicalVolume* volume = theStep->GetPreStepPoint()->GetTouchableHandle()-
>GetVolume();
  // If energy is deposited in the detector volume then record
  if(theStep->GetPreStepPoint()->GetTouchableHandle()->GetVolume()->GetName() ==
"Detect") {
    // Get thread number
    std::vector<boost::filesystem::path> ThreadNum2 = fEventAction->ThreadNum;
    thread_local int Threadindex=-1;
    // If thread file exists (aka not first event+step) then change index number
    thread_local thread::id threadid = std::this_thread::get_id();
    thread_local stringstream tstr;
    tstr << threadid;
    thread_local string tstr_str = tstr.str();
```

```
    thread_local string tstr_str_cmp = "./" + tstr.str() + ".thread";
    // Match current thread to index (thread 1, 2, 3, etc)
    for (thread_local int ii = 0; ii < ThreadNum2.size(); ii++ ) {
       if ( tstr_str_cmp.compare(ThreadNum2[ii].string()) == 0 ) {
          Threadindex = ii;
       }
    }
    // Get energy deposited
    G4double edepStep = theStep->GetTotalEnergyDeposit();
    // Add energy to event for given thread (with position given by matched
index number)
    fEventAction->AddEdepDetector(edepStep,Threadindex);
  }
}
```