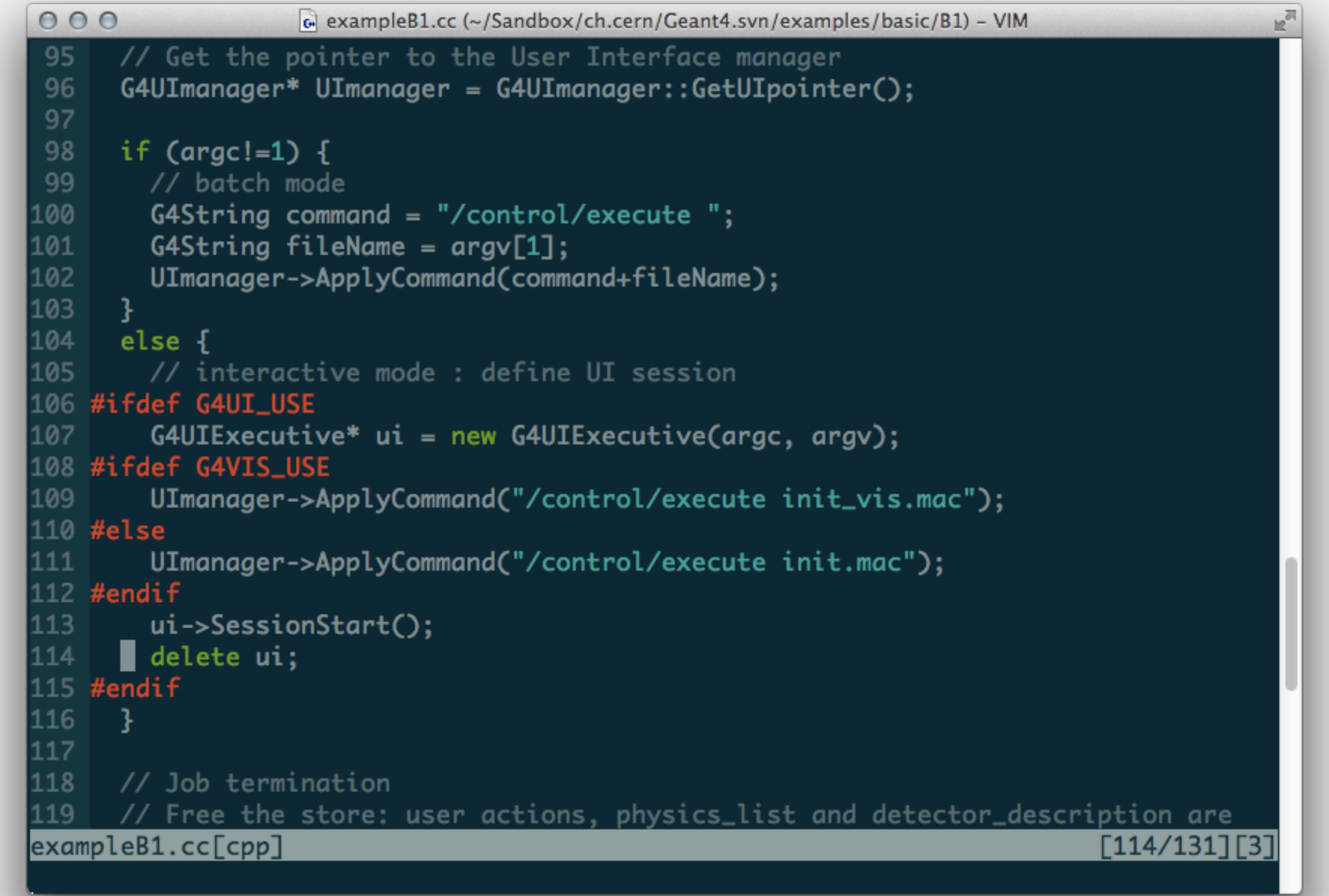
Macro and Data Paths

Ben Morgan

Problem 1: Hard Coded Script Paths


exampleB1.cc (~/Sandbox/ch.cern/Geant4.svn/examples/basic/B1) - VIM

```
  // Get the pointer to the User Interface manager
  G4UImanager* UImanager = G4UImanager::GetUIpointer();

  if (argc!=1) {
    // batch mode
    G4String command = "/control/execute ";
    G4String fileName = argv[1];
    UImanager->ApplyCommand(command+fileName);
  }
  else {
    // interactive mode : define UI session
#ifdef G4UI_USE
    G4UIExecutive* ui = new G4UIExecutive(argc, argv);
#ifdef G4VIS_USE
    UImanager->ApplyCommand("/control/execute init_vis.mac");
#else
    UImanager->ApplyCommand("/control/execute init.mac");
#endif
    ui->SessionStart();
    delete ui;
#endif
  }

  // Job termination
  // Free the store: user actions, physics_list and detector_description are
```

Why I View This as a Problem*

- Cannot run application unless scripts are in CWD
- *In the context of the examples, perhaps this is not important.
- But...
- Means we don’t demonstrate how to handle this case
    - *Again, perhaps this is not critical in the example context

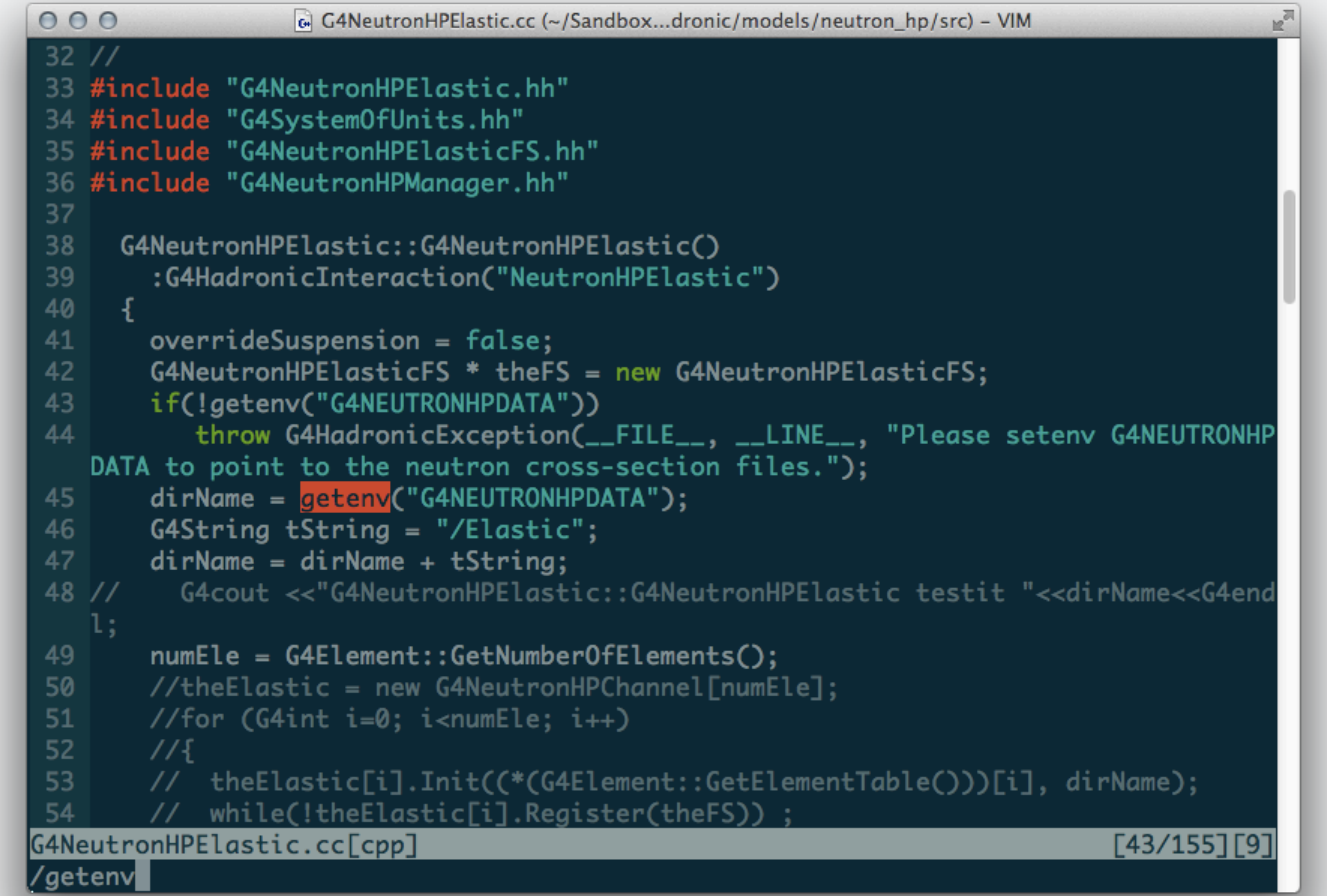
Finding the Application

- `/control/macroPath <path>` UI command?
  - Great feature, but where does the path come from?
- How does application know where its resources are?

Runtime Location of Executable?

- `binreloc` (see, e.g. https://github.com/drbenmorgan/Resourceful)
- `QCoreApplication::applicationDirPath()`
- `[NSBundle* pathForResource]`
- Win32 `GetModuleFileName()`
- User choice of technology, but could we demonstrate?

Problem 2: Locating Physics Data Resources


G4NeutronHPElastic.cc (~/Sandbox...dronic/models/neutron_hp/src) - VIM

```
//
#include "G4NeutronHPElastic.hh"
#include "G4SystemOfUnits.hh"
#include "G4NeutronHPElasticFS.hh"
#include "G4NeutronHPManager.hh"

  G4NeutronHPElastic::G4NeutronHPElastic()
    :G4HadronicInteraction("NeutronHPElastic")
  {
    overrideSuspension = false;
    G4NeutronHPElasticFS * theFS = new G4NeutronHPElasticFS;
    if(!getenv("G4NEUTRONHPDATA"))
       throw G4HadronicException(__FILE__, __LINE__, "Please setenv G4NEUTRONHPDATA to point to the neutron cross-section files.");
    dirName = getenv("G4NEUTRONHPDATA");
    G4String tString = "/Elastic";
    dirName = dirName + tString;
//    G4cout <<"G4NeutronHPElastic::G4NeutronHPElastic testit "<<dirName<<G4endl;
    numEle = G4Element::GetNumberOfElements();
    //theElastic = new G4NeutronHPChannel[numEle];
    //for (G4int i=0; i<numEle; i++)
    //{
    //  theElastic[i].Init((*(G4Element::GetElementTable()))[i], dirName);
    //  while(!theElastic[i].Register(theFS)) ;
G4NeutronHPElastic.cc[cpp]                                             [43/155][9]
/getenv
```

Why I View This as a Problem*

- N further environment variables to set
- Geant4/Data Resource versions must be compatible
- *Users/majority of developer want everything to work
- *Developers working with Data need flexibility
- How to balance these use cases?

Layered Configuration of Data Path?

- Access data resource via “`G4GetResource(name)`”
- Locates resource via Environment, then a path list
- The list can initially hold a runtime/hardcoded path
  - Runtime from location of “`G4global`”
  - Hardcoded from configure time value of `GEANT4_INSTALL_DATADIR`
- `binreloc/DllMain` => possible x-platform solution?

Summary

- Are these really problems in the context they occur in?

- `getenv()` for data certainly is, or at least an annoyance

- Thoughts, comments, questions?