Fermilab

Managed by Fermi Research Alliance, LLC for the U.S. Department of Energy Office of Science

Introduction to the Fermilab computing environment and setup for using *art*

James Amundson

art/LArSoft course

2015-08-03

The punchline

- I am a terrible comedian, so I will start with the punch line.
- You will be working on Exercise 1: Running Pre-built art Modules. In the process, you will type
  - *source /products/course_setup.sh*
  - *setup toyExperiment v0_00_29 -q$ART_WORKBOOK_QUAL:prof*
  - a bunch of shell commands to copy files, etc.
  - *art -c hello.fcl >& output/hello.log*

The goal of this lecture is to understand why you are typing these things.

Why?

- The Fermilab computing environment
  - Why does there need to be such a thing?
    - You will be dealing with a large set of interdependent software packages
      - External packages such as gcc, Python
      - HEP packages such as ROOT, CLHEP
      - *art*, LArSoft
      - Experiment-specific packages
    - A given version of the package *you* are developing will require a specific set of versions of dependent packages
    - Switching between multiple versions of sets of dependent packages is a complex problem
      - There is no industry-standard solution
        - Nor even HEP-standard solution
      - Our solution is particularly complete

UPS

- UPS (Unix Product Support) is our system for dealing with software packages and versions
  - See Chapter 7 of the *art* Workbook.
  - http://www.fnal.gov/docs/products/ups/ReferenceManual/
    - The manual tells you much more than you need to know
- UPS is very flexible and very complete
- It relies heavily on manipulating environment variables and the shell
  - It behooves us to recall a few things about dealing with the shell environment variables in a Unix-like environment
- The core ups command is *setup*

Shell commands

- There are many shells. *bash* is by far the most common.
  - If you don’t know what shell you are using, it is almost always bash.
  - Use bash.
- What happens when I type *cmd* in the (bash) shell?
  - Bash looks to see if *cmd* is an alias, shell function, or executable
    - It searches the directories listed in the environment variable PATH in the order given to find executables
  - Use the command *type* to see what will happen

```
|alcourse>type mkdir
mkdir is /bin/mkdir
|alcourse>type ls
ls is aliased to `ls --color=auto -CF'
```

An aside

- Use *type*, not *which*
  - Even though it can lead to “who’s on first” discussions when programming in pairs
  - *which* can be right or wrong because it only looks in PATH.
    - And, optionally, aliases…

Initializing the environment

- Typically, a single file will have to be sourced to get started

```
|alcourse>type setup
setup is /usr/bin/setup
|alcourse>source /products/course_setup.sh
|alcourse>type setup
setup is a function
setup ()
{
    . `/products/ups/v5_1_5/Linux64bit+2.6-2.12/bin/ups setup "$@"`
}
```

- Another aside
  - "*source*" and "." are the same thing
  - The latter is confusing in documentation…
- Why *source*? Why not just run an executable?
  - UPS is adding the *setup* function to your shell.
    - Executables can not do that.

Setting up your working environment

```
|alcourse>type art
-bash: type: art: not found
|alcourse>source /products/course_setup.sh
|alcourse>setup toyExperiment v0_00_29 -q$ART_WORKBOOK_QUAL:prof
|alcourse>type art
art is /products/art/v1_15_01/slf6.x86_64.e7.prof/bin/art
```

- The above is from a fresh login to a course machine.
- The *source* command only needs to be done once in a shell session.
- The environment variable `ART_WORKBOOK_QUAL` is provided by course_setup.
  - …as are many other variables.

A little more about UPS products

- To see which *toyExperiment*s we could have *setup*, use

  ups list –aK+ toyExperiment

```
|alcourse>ups list -aK+ toyExperiment
"toyExperiment" "v0_00_24" "Linux64bit+2.6-2.12" "debug:e7:s12" ""
…
"toyExperiment" "v0_00_29" "Linux64bit+2.6-2.12" "debug:e7:nu:s14" ""
"toyExperiment" "v0_00_29" "Linux64bit+2.6-2.12" "e7:prof:s14" ""
"toyExperiment" "v0_00_29" "Linux64bit+2.6-2.12" "e7:nu:prof:s14" ""
```

product *version* *flavor* *qualifier*

- We specified the qualifiers with

  -q$ART_WORKBOOK_QUAL:prof

```
|alcourse>echo $ART_WORKBOOK_QUAL
e7:s14
```

- The order of parts of qualifiers does not matter.
  - We got the second-to-last one.

A little more about environment variables

- The two most important variables modified by ups are PATH and LD_LIBRARY_PATH (DYLD_LIBRARY_PATH on OSX).
  - PATH determines where the shell finds executables.
  - LD_LIBRARY_PATH determines where the dynamic loader finds shared libraries.
- The output of *echo $PATH*, etc., can be hard to read

  ```
  echo $PATH | tr : \\n
  ```

  is a useful trick.
- *ldd* (*otool –L* on OSX) can show you which shared libraries will be used by an executable.
  - try

    ```
    ldd `type –p art`
    ```

Running *art*

- You will be running *art* with the following command

  art -c hello.fcl >& output/hello.log
- *hello.fcl* is a configuration file in the Fermilab Hierarchical Configuration Language (FHiCL)
- Basic syntax:
  - definition

    name : value
  - a group of definitions is a table, also called a parameter set
    - { *definitions* }
  - comments start with *//* or *#*
    - unless the first 8 characters on a line are *#include*.

Hello world in FHiCL

```
#include "fcl/minimalMessageService.fcl"

process_name : hello

source : {
  module_type : RootInput
  fileNames   : [ "inputFiles/input01.art" ]
}

services : {
  message : @local::default_message
}

physics :{
  analyzers: {
    hi : {
      module_type : HelloWorld
    }
  }

  e1        : [ hi ]
  end_paths : [ e1 ]

}
```

Conclusion

- In Exercise 1, you will initialize the UPS with the following command:

```
source /products/course_setup.sh
```

- You will then use UPS to setup the toyExperiment with

```
setup toyExperiment v0_00_29 -q$ART_WORKBOOK_QUAL:prof
```

- You will run the FHiCL version of Hello world with the following command

```
art -c hello.fcl >& output/hello.log
```

Get Started

- Work on Exercise 1 (Chapter 9) of the *art* Workbook

  https://web.fnal.gov/project/ArtDoc/Shared%20Documents/art-documentation.pdf