Setting up your workspace

Heavily inspired from previous tutorials (in particular Iker’s from last year)

Goals for this session

1) Setup and compile sbndcode for use in later sessions. Once this has been done (successfully), don’t touch it again. Make sure this has been done as you’ll need it later - ask if you need help.

2) Compile another copy of sbndcode where we can do *fun* things.

Before we start

- Things that are preceded with a $ symbol usually mean insert your own name for this e.g. your specific username or whatever you called that directory ($username=larsoft00).
- They could also refer to environmental variables which should already be set.
- The distinction will hopefully be obvious.

- For the setup I've included screenshots of what you should hopefully see so you can be sure you're on the right track.

➔ A blue box like this indicates commands you type in the terminal

- An orange box is just some extra information
- e.g. it might explain some of the commands or arguments you're using

Step 1. Login

Using a VNC:

- Go to this link https://phcomputeppe01.ph.ed.ac.uk/guacamole
  (or maybe this one if on eduroam
  https://guac-proxy.edi.scotgrid.ac.uk/guacamole/#/)
- Login
- Open a terminal

Or ssh directly:

➔ ssh $username@phcomputeppe01.ph.ed.ac.uk

Optional things to make life easier

- These are based on my preferences.. Feel free to ignore / do something else.

- Change your terminal prompt. Go from this `-bash-4.2$` to this `[15:37]~/Documents$`
- Helpful so we instantly know what directory we're in and don't have to PWD all the time.
- Make a file called ".bash_profile" - note the preceding .
- Put the following lines in your .bash_profile file:
  export PS1="[\A]\w$ "
  PS1='${debian_chroot:+($debian_chroot)}\[\033[01;32m\][\A]\w\[\033[00m\]$ '
- And then do source ~/.bash_profile

- Copy & paste instructions for the vnc:
  https://www.mfdemoportal.com/forum/blueshift-tips-and-tricks-staff-user/guacamole-cut-n-paste-functionality

Step 2. Make your larsoft directory and move to it

➔ mkdir $larsoft_workdir
➔ cd $larsoft_workdir

Step 3. Setup sbndocde

➔ source /cvmfs/sbnd.opensciencegrid.org/products/sbnd/setup_sbnd.sh

```
[04:12]/sbnd/app/users/tham/larsoft_workshop_v09_32_00$ source /cvmfs/sbnd.opensciencegrid.org/products/sbnd/setup_sbnd.sh
Setting up LArSoft from "CVMFS":
 - executing '/cvmfs/larsoft.opensciencegrid.org/products/setup'
 - appending '/cvmfs/fermilab.opensciencegrid.org/products/common/db'
Setting up artdaq from "CVMFS":
 - appending '/cvmfs/fermilab.opensciencegrid.org/products/artdaq'
Setting up sbn from "CVMFS":
 - appending '/cvmfs/sbn.opensciencegrid.org/products/sbn'
Setting up SBND from "CVMFS":
 - prepending '/cvmfs/sbnd.opensciencegrid.org/products/sbnd'
```

Step 4. Make a new development area

➔ mrb newDev -v v09_32_00 -q prof:e20

- -q; is for qualifiers
- prof; compile with optimisation and limited debugging.
- Could use ‘debug’ instead of ‘prof’ for full debugging.
- e20; Sets the compiler (ask someone more knowledgeable than me)

```
[04:12]/sbnd/app/users/tham/larsoft_workshop_v09_32_00$ mrb newDev -v v09_32_00 -q prof:e20

building development area for larsoft v09_32_00 -q prof:e20


The following configuration is defined:
  The top level directory is .
  The source code directory will be under .
  The build directory will be under .
  The local product directory will be under .

MRB_BUILDDIR is /sbnd/app/users/tham/larsoft_workshop_v09_32_00/build_slf7.x86_64
MRB_SOURCE is /sbnd/app/users/tham/larsoft_workshop_v09_32_00/srcs
INFO: copying /cvmfs/larsoft.opensciencegrid.org/products/larsoft/v09_32_00/releaseDB/base_dependency_database
/cvmfs/larsoft.opensciencegrid.org/products/mrb/v5_19_04/libexec/newDev.sh: line 192: unsetup: command not found

IMPORTANT: You must type
    source /sbnd/app/users/tham/larsoft_workshop_v09_32_00/localProducts_larsoft_v09_32_00_prof_e20/setup
NOW and whenever you log in
```

Step 5. Source local products

- Do as you're told and source the local products.
- We don't need the whole path since we're already in the larsoft directory.

➔ source localProducts_larsoft_v09_32_00_prof_e20/setup

- These set some variables so if you type e.g. "echo $MRB_PROJECT" it will print "larsoft".
- Can be useful to help navigate many layers of directories quickly.

```
[04:14]/sbnd/app/users/tham/larsoft_workshop_v09_32_00$ source localProducts_larsoft_v09_32_00_prof_e20/setup

MRB_PROJECT=larsoft
MRB_PROJECT_VERSION=v09_32_00
MRB_QUALS=prof:e20
MRB_TOP=/sbnd/app/users/tham/larsoft_workshop_v09_32_00
MRB_SOURCE=/sbnd/app/users/tham/larsoft_workshop_v09_32_00/srcs
MRB_BUILDDIR=/sbnd/app/users/tham/larsoft_workshop_v09_32_00/build_slf7.x86_64
MRB_INSTALL=/sbnd/app/users/tham/larsoft_workshop_v09_32_00/localProducts_larsoft_v09_32_00_prof_e20

PRODUCTS=/sbnd/app/users/tham/larsoft_workshop_v09_32_00/localProducts_larsoft_v09_32_00_prof_e20:/cvmfs/sbnd.opensci
encegrid.org/products/sbnd:/cvmfs/larsoft.opensciencegrid.org/products:/cvmfs/fermilab.opensciencegrid.org/products/c
ommon/db:/cvmfs/fermilab.opensciencegrid.org/products/artdaq:/cvmfs/sbn.opensciencegrid.org/products/sbn
CETPKG_INSTALL=/sbnd/app/users/tham/larsoft_workshop_v09_32_00/localProducts_larsoft_v09_32_00_prof_e20
```

Step 6. Clone sbndcode from github

- ➔ cd $MRB_SOURCE
- ➔ mrb g sbndcode
- ➔ cd sbndcode
- ➔ git checkout origin/uk_larsoft_workshop_2021 -b uk_larsoft_workshop_2021

- May get prompted with some *key fingerprints* Just type “yes” and continue

- This is the name of the remote sbndcode branch we want.
- This will be the name of the branch on our machine (could call it anything, but to avoid confusion let’s give it the same name)

```
[04:14]/sbnd/app/users/tham/larsoft_workshop_v09_32_00$ mrb g sbndcode

Cloning into 'sbndcode'...
X11 forwarding request failed on channel 0
remote: Enumerating objects: 34141, done.
remote: Counting objects: 100% (3399/3399), done.
remote: Compressing objects: 100% (1422/1422), done.
remote: Total 34141 (delta 2273), reused 2939 (delta 1973), pack-reused 30742
Receiving objects: 100% (34141/34141), 132.83 MiB | 21.43 MiB/s, done.
Resolving deltas: 100% (20033/20033), done.
Updating files: 100% (831/831), done.
NOTICE: Adding sbndcode to CMakeLists.txt file
```

- You can clone the repository *normally* with “git clone …”, but the CMakeLists.txt file would not be updated automatically.
- “mrb uc” will update the CMakeLists.txt file if you do things this way.

Step 7. Setup environment

➔ mrbsetenv

```
[04:35]/sbnd/app/users/tham/larsoft_workshop_v09_32_00/srcs/sbndcode$ mrbsetenv
The working build directory is /sbnd/app/users/tham/larsoft_workshop_v09_32_00/build_slf7.x86_64
The source code directory is /sbnd/app/users/tham/larsoft_workshop_v09_32_00/srcs
----------- check this block for errors -----------------------
INFO: mrb v5_19_04 requires cetmodules >= 2.29.01 to run: attempting to configure...v2_29_06 OK
-------------------------------------------------------------
To inspect build variable settings, execute /sbnd/app/users/tham/larsoft_workshop_v09_32_00/build_slf7.x86_64/cetpkg_
info.sh

Please use "buildtool" (or "mrb b") to configure and build MRB project "larsoft", e.g.:

  buildtool -vTl [-jN]

See "buildtool --usage" (short usage help) or "buildtool -h|--help"
(full help) for more details.
```

Step 8. Build and Install

➔ cd $MRB_BUILDDIR
➔ mrb install -j3

- -j sets the number of threads to use i.e. speeds up the build process.
- The “lscpu” command will give info about how many cores etc you have available.
- Don’t use more than 3 in this case, otherwise we might kill the system with everyone running this at the same time.

- Don’t wait for this to finish, move to the next slide.

Step 8.5. While we wait..

- While we wait for that to finish, let’s start compiling another copy of sbndcode where we can play around a little and not worry if we break something.

- We aren’t quite done with the first copy of sbndcode however, so don’t forget to come back (step 9) to finish it off.

- Open a new terminal (don’t close the current one since it’s still running).

- Starting from step 1, repeat everything again (in step 2, you will need to make a “larsoft_workdir” with a different name.)

- While this 2nd copy of sbndcode is compiling let’s finish the first one (hopefully it’s built by now)

Step 9. Setup installation

- Hopefully it will have installed successfully and you'll see the following message;

```
------------------------------------
INFO: stage install SUCCESS for MRB project larsoft v09_32_00
------------------------------------
```

➔ mrbslp

```
[05:48]/sbnd/app/users/tham/larsoft_workshop_v09_32_00/build_slf7.x86_64$ mrbslp
local product directory is /sbnd/app/users/tham/larsoft_workshop_v09_32_00/localProducts_larsoft_v09_32_00_prof_e20
---------- this block should be empty -----------------
-------------------------------------------------------
```

- Double check things are working (check we have the "lar" command).

➔ cd ..
➔ lar

```
[05:49]/sbnd/app/users/tham/larsoft_workshop_v09_32_00$ lar
OptionsHandler caught a cet::exception calling art::BasicOptionsHandler::doCheckOptions()
---- Configuration BEGIN
  No configuration file given.
---- Configuration END

Art has completed and will exit with status 89.
```

Next steps

- Once your first copy of sbndcode has been properly compiled (steps 1-9), don’t touch it again for now. You’ll need a working copy for the upcoming sessions.

- We’ll cover a bit more explanation and setup and then look at generating a few events etc. using your second copy of sbndcode. Go at your own pace, it doesn’t matter if you don’t get everything done.

Larsoft setup now we have compiled code

- We don’t need to compile each time we login.
- Close your terminal and start a fresh one.
- Navigate back to your 2nd larsoft working directory.
  Then, all you need to do is the following.

- Even better is if you can make these commands into a shell script.
- Then all you have to do is move to your working directory and source the setup script.

```
➔ source /cvmfs/sbnd.opensciencegrid.org/products/sbnd/setup_sbnd.sh
➔ source localProducts_larsoft_v09_32_00_prof_e20/setup
➔ mrbsetenv
➔ mrbslp
```

- “mrbslp” - Setup the products installed in the localProducts_xxx directory.
- “mrbsetenv” - Setup the development environment.

Generate some events

➔ lar -c prodsingle_sbnd.fcl -n 3 -o muons_gen.root

- -c specifies the file we want to run
- -n specifies the number of events (in this case 3)
- -o sets the name of the output file.

It doesn't matter in which directory you generate your events, but it's probably advisable to generate them outside the srcs/ directory i.e. generate them in your larsoft_workdir2 or if you want to be organised make another directory to hold all your events.

- If you're a root veteran you may be keen to open muons_gen.root in a TBrowser and have a look around.
- This of course is fine, but these types of files aren't really for "human consumption", so don't expect to see anything too meaningful in there.

Looking at the events.

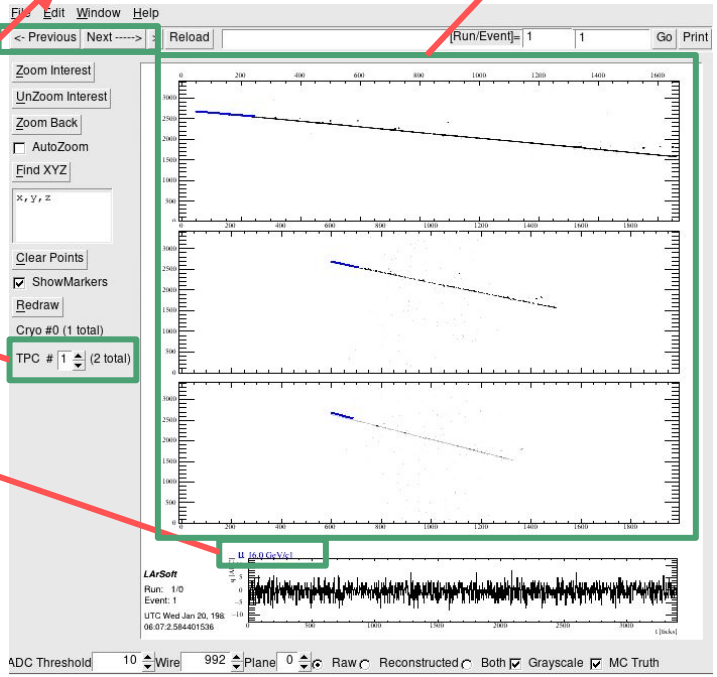
➔ lar -c evd_sbnd.fcl muons_gen.root

3 wire planes so 3 event displays. Top to bottom - collection plane, 2nd induction plane, 1st induction plane

Cycle through the events.

Might need to change the TPC to get a nice display.

This gives the momentum of the particle.



- We generated these events using a particle gun hence they're all identical - once we've seen one we've seen them all.
- Let's make this a bit more interesting.

Make some more events.

- Locate prodsingle_sbnd.fcl and make a personal copy we can tweak.
- It can be found in “/srcs/sbndcode/sbndcode/JobConfigurations/base”

➔ cp prodsingle_sbnd.fcl myversion_prodsingle_sbnd.fcl

- Look at the bottom of the file. You should see something like this;

```
129 #
130 # at the end of the configuration, we can override single parameters to reflect our needs:
131 #
132 physics.producers.generator.T0: [ 0 ]
133 physics.producers.generator.Theta0XZ: [ 10 ]
```

- Here we can override some previously set settings. For example Theta0XZ gives the angle in the XZ plane - lets change this to a different number, say 45, or whatever you want.
- Generate a few more events using your own fcl.

➔ lar -c srcs/sbndcode/sbndcode/JobConfigurations/base/myversion_prodsingle_sbnd.fcl -n3 -o muons_gen2.root

- Have a look at the events - do they look how you expect?

Make even more events

- We can do much more than just change the angle.
- How about changing the momentum or the location in the detector?
- We can also generate particles from a range of values which means they won't all be identical.

- Here are some more options you can try out.
- You can also take a look at some of the fcls in "srcs/sbndcode/sbndcode/JobConfigurations/standard/gen/single/" if you want some inspiration.

```
physics.producers.generator.X0: [ 0 ]
physics.producers.generator.SigmaX: [ 200 ]
physics.producers.generator.PDist: 1                    # Gaussian momentum dist.
physics.producers.generator.P0: [ 2 ]
physics.producers.generator.SigmaP: [ 2 ]
```

As an example P0: sets the base momentum (2 GeV in this case)
SigmaP: sets the momentum range - so we'll generate particles with P0 +- 2 GeV
PDist: 1 means we pick values in a Gaussian distribution around P0 (use Pdist: 0 if you want a uniform dist.)

Before we get too carried away..

- There will be more on generating and reconstructing events in the upcoming sessions plus an explanation of fcl files.

- You may also have noticed when we used the default event generation fcl (prodsingle_sbnd.fcl), we didn't bother specifying the path to it. Try running your own fcl without the path and you should get an error saying it can't find the file.

- Since we added a new file, we need to compile our code again so larsoft knows about it. We don't need to start from square one however.

➔ cd $MRB_BUILDDIR
➔ make install -j3
➔ cd $MRB_TOP

- This should hopefully be relatively quick and now we can run our own fcl without specifying the path.

How to know which version you should/want/need to use

- Now we've managed to make some events, let's go back and have a look at how we know what version of larsoft we should be using.
- In the setup slides, we used sbndcode version v09_32_00 - there are of course many other versions available.

➔ ups list -aK+ larsoft
➔ ups list -aK+ sbndcode

- Larsoft and sbndcode are usually kept in sync, but other packages such as larpandora are not (sbndcode depends on other packages, more on this later.)

➔ ups list -aK+ larpandora

- Note how the most recent larpandora version numbers are very different to sbndcode.
- So how do you know which version goes with which? There's some handy documentation here https://cdcvs.fnal.gov/redmine/projects/larsoft/wiki/LArSoft_release_list

Which version am I using?

- There’s a quick way to check which versions you’re running.

➔ ups active

- This will list all the active products and their version.

Adding additional products

- So we've seen that we actually have loads of different products setup, not just sbndcode. At the moment we only really have access to sbndcode. What if we want to edit some of these as well?
- Let's get ahold of larpandora as an example.

```
➔ cd $MRB_SOURCE
➔ mrb g larpandora
➔ cd larpandora;
➔ git checkout vxx_xx_xx -b vxx_xx_xx
```

- Go to the website mentioned on slide 22 and use that to decide which version you need.
  (Hint: The "uk_larsoft_workshop_2021" branch is based off version v09_32_00 of sbndcode.)

- Great, now our srcs/ directory has both sbndcode and larpandora. But, before we can do anything we need to do a full recompile so we link the local version of larpandora.

```
➔ cd $MRB_BUILDIR; mrb z
➔ mrbsetenv
➔ mrb install -j3
➔ mrbslp
```

- mrb z - removes everything in $MRB_BUILDDIR
- This allows us to cleanly compile from the start.

Finished with time to spare?

fcl (or fhicl = Fermilab **hierarchical** configuration language)

- Maybe have a think about how the fcl files are structured.
- If you had a look at some of the fcl files in “/srcs/sbndcode/sbndcode/JobConfigurations/standard/gen/single/”, on the surface most of them look simple since you only set a few parameters without all the boilerplate stuff you saw in “prodsingle_sbnd.fcl”.
- What happens when you run “fhicl-dump prodsingle_sbnd.fcl”?
- Instead of directly editing a copy of “prodsingle_sbnd.fcl” like on slide 19, try creating a fcl similar to the ones you see in “JobConfigurations/standard/gen/single/” and see if you can get the same results as from “myversion_prodsingle_sbnd.fcl”.

Some extras

Multi Repository Build (mrb) commands (mrb --help)

command	short hand	arguments	description
mrb newDev	mrb n	-v $version -q $qualifier	make a new development area
mrb gitCheckout	mrb g	sbndocde	clone a git repository
mrb install	mrb i	-j $numcores	run buildtool with install
mrb zapBuild	mrb z		remove everything from the build directory

More Resources

- Larsoft github repository - https://github.com/LArSoft
- SBN github repository - https://github.com/SBNsoftware
- Larsoft documentation - https://nusoft.fnal.gov/larsoft/doxsvn/html/index.html
- mrb reference guide - https://cdcvs.fnal.gov/redmine/projects/mrb/wiki/MrbRefereceGuide
- SBN software wiki - https://sbnsoftware.github.io/