The QDL Installer

QDL has an installer that manages the distribution. It allows for a fresh install, un-install and upgrades. It will install QDL, and it can also manage the syntax highlighting for vim and nano.

Getting it

The installer is always available as qdl-installer.jar from the QDL releases page.

Using it

Once downloaded, it is a standalone, executable jar, so to show the current help issue

```
java -jar qdlinstaller.jar
QDL installer version …
This will install QDL to your system. Options are:
(none) = same as help
install = install
upgrade = upgrade
… more
```

starting it with no arguments, displays help. The basic operation is

operation arguments options

where the supported operations are install, upgrade, remove, list and help.

The arguments at this point is a single pair,

-dir root_directory

which specifies the root directory of the install. It is possible to have multiple installs of QDL (though relatively rare). If you have installed it and set your QDL_HOME variable, you should use that.

options are

- -all = all components
- -qdl = QDL only
- -vim = syntax highlighting for vim
- -nano = syntax highlighting for the nano editor, version 4+.

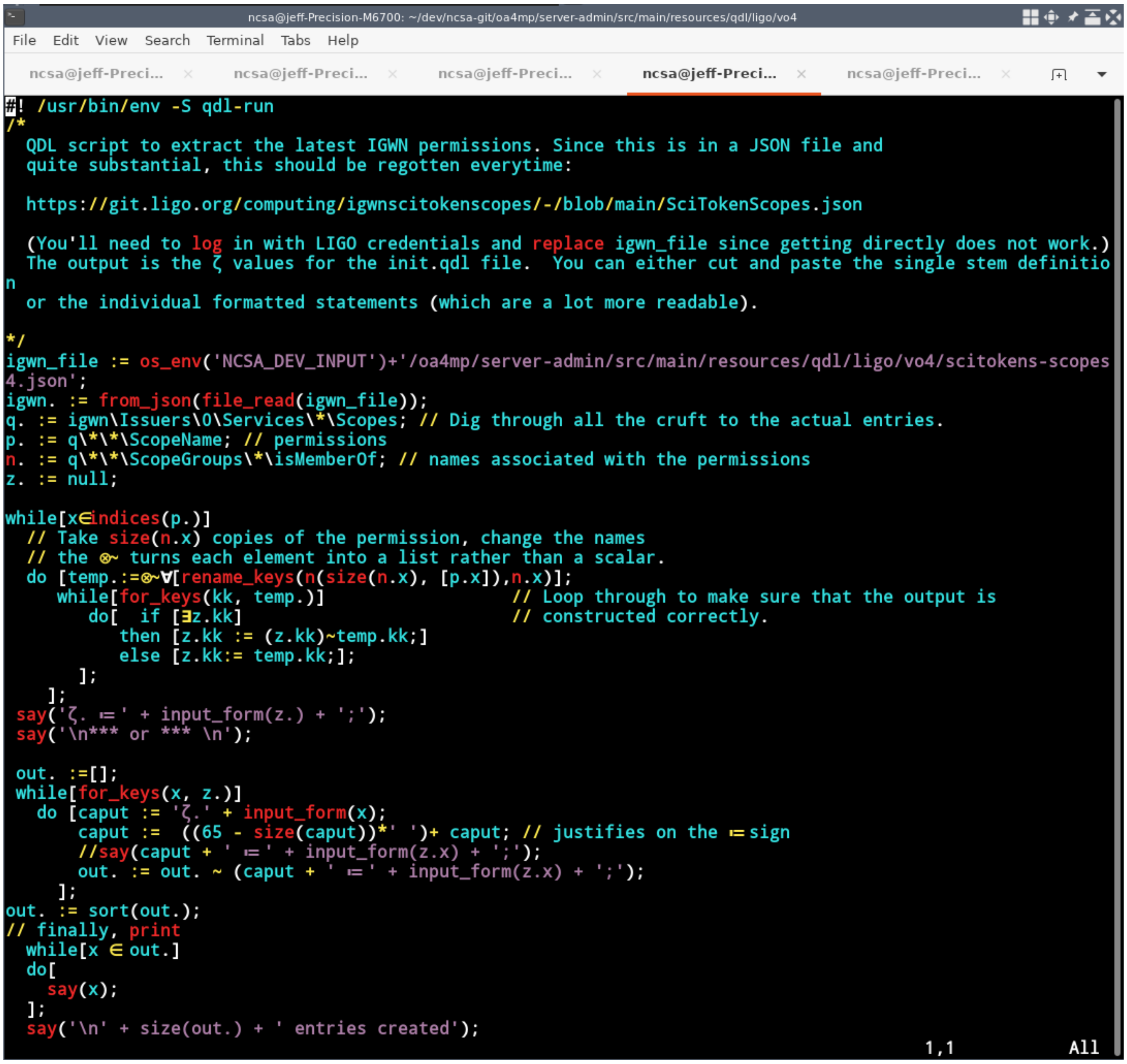
Editors

If you need an older version of nano, there is a qdl.nanorc-2.3.1 file in $QDL_HOME/etc. You should link to that in you ~/.nanorc file.

Note that both nano and vim support simply link to your existing QDL install,, so that upgrades take place automatically once installed.   These set the type to *.qdl (qdl script) and *.mdl (module) and invoking either of those editors on one of those file types will turn on syntax highlighting for QDL.

Also, your workspace default editor for QDL buffers will be set to whatever you have chosen. If you have specified both, then the default is set to vim. If you have neither, then the built in line editor will be used. These editor choices only apply to running QDL in a terminal. If you start it in swing mode, then it has its own editor.

Example of editing session in VIM with syntax highlighting enabled.


ncsa@jeff-Precision-M6700: ~/dev/ncsa-git/oa4mp/server-admin/src/main/resources/qdl/ligo/vo4
File Edit View Search Terminal Tabs Help
ncsa@jeff-Preci...
ncsa@jeff-Preci...
ncsa@jeff-Preci...
ncsa@jeff-Preci...
ncsa@jeff-Preci...

```
#! /usr/bin/env -S qdl-run
/*
  QDL script to extract the latest IGWN permissions. Since this is in a JSON file and
  quite substantial, this should be regotten everytime:

  https://git.ligo.org/computing/igwnscitokenscopes/-/blob/main/SciTokenScopes.json

  (You'll need to log in with LIGO credentials and replace igwn_file since getting directly does not work.)
  The output is the ζ values for the init.qdl file.  You can either cut and paste the single stem definitio
n
  or the individual formatted statements (which are a lot more readable).

*/
igwn_file := os_env('NCSA_DEV_INPUT')+'/oa4mp/server-admin/src/main/resources/qdl/ligo/vo4/scitokens-scopes
4.json';
igwn. := from_json(file_read(igwn_file));
q. := igwn\Issuers\0\Services\*\Scopes; // Dig through all the cruft to the actual entries.
p. := q\*\*\ScopeName; // permissions
n. := q\*\*\ScopeGroups\*\isMemberOf; // names associated with the permissions
z. := null;

while[x∈indices(p.)]
  // Take size(n.x) copies of the permission, change the names
  // the ⊗~ turns each element into a list rather than a scalar.
  do [temp.:=⊗~∀[rename_keys(n(size(n.x), [p.x]),n.x)];
     while[for_keys(kk, temp.)]                  // Loop through to make sure that the output is
        do[  if [∃z.kk]                          // constructed correctly.
           then [z.kk := (z.kk)~temp.kk;]
           else [z.kk:= temp.kk;];
        ];
    ];
 say('ζ. ≔' + input_form(z.) + ';');
 say('\n*** or *** \n');

 out. :=[];
 while[for_keys(x, z.)]
   do [caput := 'ζ.' + input_form(x);
       caput :=  ((65 - size(caput))*' ')+ caput; // justifies on the ≔sign
       //say(caput + ' ≔' + input_form(z.x) + ';');
       out. := out. ~ (caput + ' ≔' + input_form(z.x) + ';');
     ];
out. := sort(out.);
// finally, print
  while[x ∈ out.]
  do[
    say(x);
  ];
  say('\n' + size(out.) + ' entries created');
                                                                                     1,1           All
```

Install

You can install any of the main components, the QDL language and syntax highlighting for vim or nano. The QDL distribution includes a great deal of documentation, various modules and examples.

A typical installation

```
java -jar qdl-installer.jar install -dir /path/to/qdl -qdl -nano
```

In which case qdl and support for nano are installed and QDL will be put into the directory /path/to/qdl. Note that if you did not specify the directory, the installer would prompt you.

Upgrade

You may also use this tool to upgrade your current install. So if you have an existing version with everything installed, you might want to issue

```
java -jar qdl-installer.jar upgrade -dir $QDL_HOME -all
```

Most files are overwritten except for the cfg-min.xml (default minimal configuration) since it is assumed you may have tinkered with that. Modules, documentation and examples will be replaced.

Remove

You may also remove any component. Note that the **`-qdl`** flag will remove ***everything*** in the specified directory.

Other operations

You may also pass in the **`list`** operation which will list all of the files in the installer's distribution. In the same way, the operation **`help`** will print out the help. Since the installer is just a jar file, if you want to unjar it and pick and choose the distribution, feel free to.

Sample installation

Here is a sample session. It downloads the installer and then sets the actual QDL directory to be a subdirectory (**qdl**) the current directory (**$PWD**). It installs all components

```
>mkdir my_install
>cd my_install
>wget https://github.com/ncsa/qdl/releases/download/v1.6/qdl-installer.jar
. . . bunch of stuff
> ls
qdl-installer.jar (maybe  others)
>java -jar qdl_installer.jar install -dir $PWD/qdl -all
QDL installer version 1.6
(. . . bunch of messages)
Done! You should add
   export QDL_HOME="/path/to/your/qdl"
to your environment and
   $QDL_HOME/bin"
to your PATH
```

(N.B. /path/to/your/qdl will be the actual install directory for QDL.) So this installs it and vim support. You ought to add **$QDL_HOME** as a variable to your environment and add **$QDL_HOME/bin** to your path. The installer will not updated your system configuration (like **~/.bashrc**) since it is too easy to make a mistake if it is complex, and trash your system. You have to do this change manually.

Testing QDL

Starting from where the above left off, the simplest way is to go to the QDL directory, henceforth **$QDL_HOME**, and issue

```
>cd $QDL_HOME
>bin/qdl
-- -- @ --  / -- @ @  / @ -- @ @
  .g8""8q. `7MM"""Yb. `7MMF'
.dP'    `YM. MM    `Yb. MM
dM'      `MM MM     `Mb MM
MM        MM MM      MM MM
MM.      ,MP MM     ,MP MM      ,
`Mb.    ,dP' MM    ,dP' MM     ,M
  `"bmmd"' .JMMmmmdP' .JMMmmmmMMM
      MMb
       `bood'
-- -- @ --  / -- @ @  / @ -- @ @
Welcome to the QDL Workspace
Version 1.6
Type )help for help.
*****************************************
ISO 6429 terminal
```

If you have set your **PATH** and **QDL_HOME**, then you can run scripts from the command line. Here is a test running one of the example scripts.

```
>cd $QDL_HOME/examples
>./hello_world.qdl
Hello world!
```

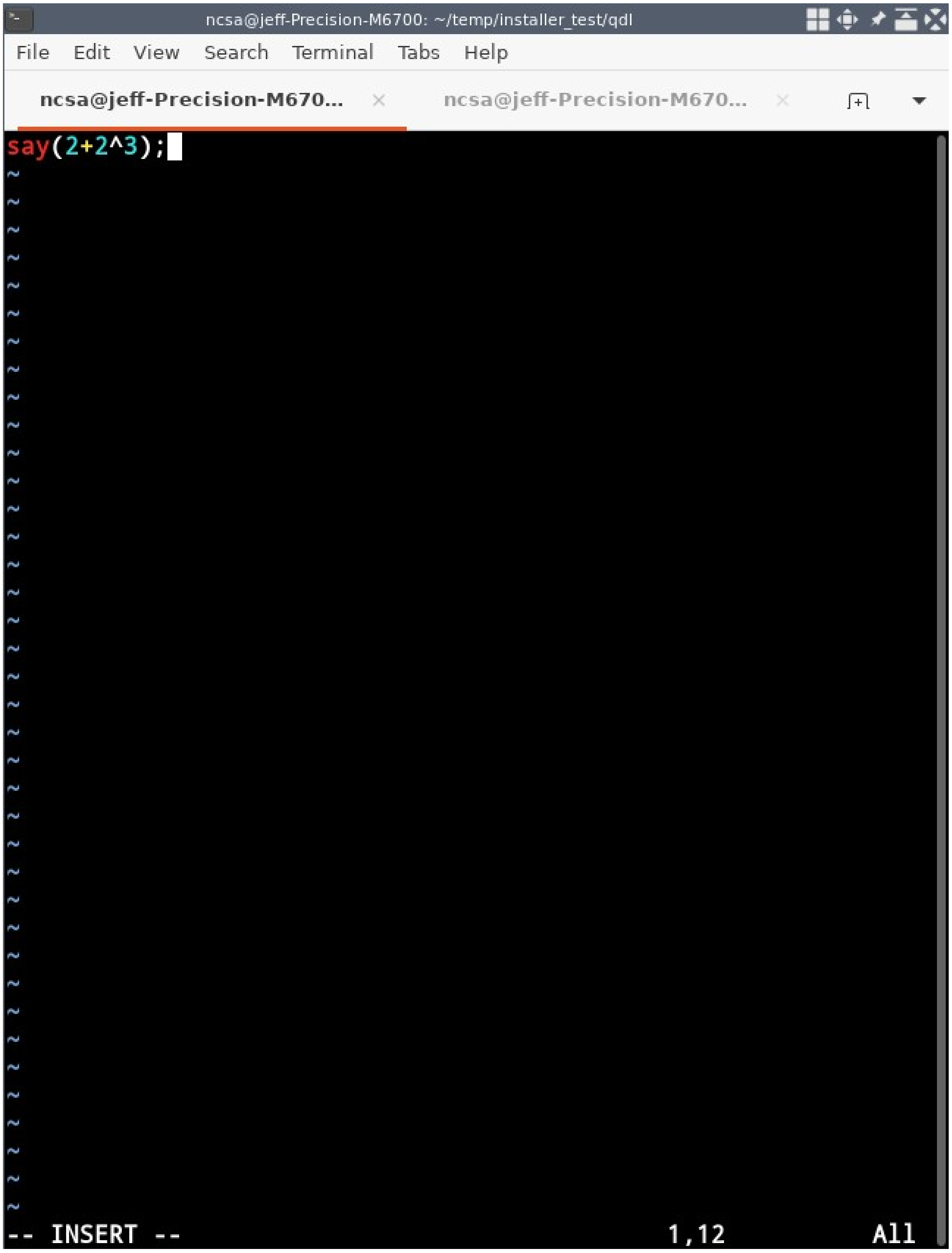
Not surprisingly, this script is the ubiquitous and gratuitous hello world program. If it runs, you can now run QDL scripts like any other scripting language on your system, You should look at the script reference (also qdl_scripting.pdf is included in the **$QDL_HOME/docs** directory).

Testing your external editor

If you installed, e.g. vim, then you should be able to create a memory buffer and edit it exactly using vim. Once QDL comes up issue

```
  )b create t -m
0|m|t: t
  )edit 0
```

and you should shell out to vim. You can enter QDL commands and they should highlight.


ncsa@jeff-Precision-M6700: ~/temp/installer_test/qdl
File Edit View Search Terminal Tabs Help
ncsa@jeff-Precision-M670...
ncsa@jeff-Precision-M670...

```
say(2+2^3);
~
~
~
-- INSERT --                                          1,12          All
```

When you save then exit, you will end up back in the workspace and can execute the buffer by issuing

```
   ) 0
10
```

Note that 10 = 2 + 2^3.

Documentation

A standard QDL install includes **$QDL_HOME/docs** which is an extensive collection of PDFs including the reference manual and pretty much anything else you need. If you are on a reasonably modern unix system, then the command **less** is usually pdf aware and you can therefore either read the manual at the command line or just bring up the formatted text in a PDF reader.