LearnIT Express is on its way!

- The demonstration will start at 12:30 pm.
- To use the Live Chat:
  - Maximize your browser window so you can see the postings.
  - If you refresh the browser page, you'll need to log in again.
  - After the demonstration, staff will remain available in the chat for a few minutes to answer additional questions.

Today's topic:

Mills HPC Cluster

LearnIT Express

in a Nutshell

University of Delaware

THE LINUX ENVIRONMENT

- Most programs have *preferences* that tailor their behavior
  - Word, Matlab, Mathematica: all rely on preference files
  - The BASH shell (and many other programs) use the *environment* to customize their behavior
    - A set of named values; *variable name* = *value*
    - e.g. PATH = ''/bin:/usr/bin''

Important Environment Variables

- PATH: Where does the shell look for commands?
- LD_LIBRARY_PATH: Where does the OS look for libraries?
- MANPATH: Where does *man* look for manual pages?
- INFOPATH: Where does *info* look for manual pages?

WHAT'S THE CONNECTION?

- HPC systems have many software packages, usually with multiple versions/variants
- Each software package has its own directories containing:
  - executable programs (**bin**) → PATH
  - shared libraries (**lib**) → LD_LIBRARY_PATH
  - manual pages (**man**, **share/info**) → MANPATH, INFOPATH

UNIVERSITY OF DELAWARE

What's the Connection?

- Example: Open MPI 1.4.4, GCC variant

```
$ cd /opt/shared/openmpi/1.4.4-gcc

$ ls -1
bin
etc
include
lib
share

$ ls -1d bin/mpicc
bin/mpicc

$ ls -1d lib/libmpi.so
lib/libmpi.so

$ ls -1 share/man
man1
man3
man7
```

University of Delaware

WHAT'S THE CONNECTION?

- Many software packages suggest you edit your .bashrc or .bash_profile to make them available in your environment
  - YOU must know *how* to make the changes
  - YOU must know *what* to add to PATH, etc.
  - YOU must re-edit whenever you use a different version of the package
  - YOU must debug any problems that arise

UNIVERSITY OF DELAWARE

Let the Computer Do It!

- That's a lot of work for YOU to be doing — none of which equates directly to "research"
- VALET Automates Linux Environment Tasks
  - On-demand alteration of PATH, LD_LIBRARY_PATH, etc.
  - Easily change which *version* of a package you're using
  - Add your own package definitions

University of Delaware

vpkg_list

- Displays a list of *packages*
- Shows the *package identifiers*
- Packages are defined in XML files
  - /opt/shared/valet/etc
  - your own directories

```
$ vpkg_list
Available packages:
  acml
  cmake
  fftw
  gaussian
  gcc
  gromacs
  gsl
  hdf4
  hdf5
  imsl
  intel
    :
```

UNIVERSITY OF DELAWARE

vpkg_versions

- For a given *package*, displays what *versions* of that package are available
  - An * indicates the *default version* of the package
  - Listed by *version identifier* and a description

```
$ vpkg_versions gaussian
gaussian  Gaussian - Quantum Chemistry
  g03e01  '03, Revision E01
* g09a02  '09, Revision A02
```

VALET IDENTIFIERS

versioned package identifier = package identifier [optional: / version identifier]

identifier	meaning
gaussian	The Gaussian package (default version)
gaussian/g09a02	Gaussian '09, revision A02
gaussian/g03e01	Gaussian '03, revision E01

vpkg_info

- Show what VALET knows about a *package* or *versioned package*
  - Using a *package identifier* alone shows info for all versions
  - Using a *versioned package identifier* shows info for that specific version

```
$ vpkg_info gaussian/g09a02
Versioned package information for `gaussian/
g09a02`:
    Gaussian - Quantum Chemistry
    http://www.gaussian.com/
    [g09a02] {
      '09, Revision A02
      prefix: /opt/shared/Gaussian/g09a02
      affect dev env: yes
      scripts: {
        sh = {
            g09.sh
        }
      }
      dependencies: {
        pgi/11
      }
      directories: {
        bin: {
        }
        lib: {
        }
```

What Info is Shown?

- The *prefix* is the directory in which the software package is installed
- The *dependencies* indicates what other packages are required by this one
- Under *directories*, the directories containing executables, libraries, manual pages are shown

vpkg_info

- For more "exotic" situations, the environment changes can be scripted
  - That script will be called by VALET

```
$ vpkg_info gaussian/g09a02
Versioned package information for `gaussian/
g09a02`:
    Gaussian - Quantum Chemistry
    http://www.gaussian.com/
    [g09a02] {
      '09, Revision A02
      prefix: /opt/shared/Gaussian/g09a02
      affect dev env: yes
      scripts: {
        sh = {
            g09.sh
        }
      }
      dependencies: {
        pgi/11
      }
      directories: {
        bin: {
        }
        lib: {
        }
```

vpkg_require

- Add a *versioned package* to your current environment
  - Any *dependencies* will be satisfied first (e.g. pgi/11)
- Re-adding the same package has no effect
- Adding one version on top of another is forbidden

```
$ vpkg_require gaussian/g09a02
Adding dependency `pgi/11` to your environment
Adding package `gaussian/g09a02` to your
environment

$ which g09
/opt/shared/Gaussian/g09a02/g09/g09

$ vpkg_require gaussian/g09a02

$ vpkg_require gaussian/g03e01
ERROR: unable to add versioned package: gaussian/
g03e01 conflicts with version: gaussian/g09a02
```

UNIVERSITY OF DELAWARE

vpkg_rollback

- Remove the environment changes introduced by the last vpkg_require command
  - Include the word **all** to remove the effects of every vpkg_require command issued in the current shell

```
$ vpkg_rollback

$ vpkg_rollback
WARNING: no snapshots defined

$ vpkg_rollback all
```

UNIVERSITY OF DELAWARE

Example: Compilers

- I want to use the Intel compiler; 32-bit version
  - Add to my environment
  - Compile hello.c
  - Use "file" to determine what kind of executable I produced

```
$ vpkg_versions intel
intel          Intel Compiler Suite
  2011-32bit   Version 2011 for 32-bit x86
* 2011-64bit   Version 2011 for x86-64

$ vpkg_require intel/2011-32bit
Adding package `intel/2011-32bit` to your
environment

$ icc -o hello hello.c
$ file hello
hello: ELF 32-bit LSB executable, Intel 80386,
version 1 (GNU/Linux), dynamically linked (uses
shared libs), for GNU/Linux 2.6.18, not stripped
```

Example: Compilers

- Now, switch to the 64-bit compiler and do the same process
  - Rollback to remove the 32-bit compiler first

```
$ vpkg_rollback

$ vpkg_require intel
Adding package `intel/2011-64bit` to your
environment

$ icc -o hello hello.c
$ file hello
hello: ELF 64-bit LSB executable, x86-64, version 1
(GNU/Linux), dynamically linked (uses shared libs),
for GNU/Linux 2.6.18, not stripped
```

Your Own Packages

- You can add your own package definitions on top of the ones that IT provides
  - First order of business: learn how to write a package definition file
    - http://beren.engr.udel.edu/doku.php/valet:packagefile

Your Own Packages

- You can add your own package definitions on top of the ones that IT provides
  - First order of business: learn how to write a package definition file
  - Next, create a directory to house your package definitions
    - VALET automatically looks for a directory named ".valet" in your home directory

Example

```
$ cd ~
$ mkdir .valet
$ cd .valet
   :
$ cat dummy.vpkg
<?xml version="1.0" encoding="UTF-8"?>
<package
  xmlns="http://www.udel.edu/xml/valet/1.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.udel.edu/xml/valet/1.0
                      http://www.udel.edu/xml/valet/1.0/schema.xsd"
  id="dummy">
  <description>Dummy:  A Fake Package</description>
  <prefix>/not/a/valid/path</prefix>
  <default-version>0.1a</default-version>
  <version id="0.1a">
    <description>Version 0.1a</description>
  </version>
</package>
```

Example

```
$ vpkg_list
Available packages:
  acml
  cmake
  dummy
  fftw
  gaussian
    :
```

Example

```
$vpkg_info dummy
[dummy] {
  Dummy:  A Fake Package
  prefix: /not/a/valid/path
  affect dev env: yes
  default version: 0.1a
  versions: {
    [0.1a] {
      Version 0.1a
      prefix: /not/a/valid/path/0.1a
      affect dev env: yes
      directories: {
        bin: {
        }
        lib: {
        }
        inc: {
        }
           :
```

SUMMARY

- VALET helps IT provide you with easy access to the many versions of software available on the cluster.
- VALET helps IT to deploy new software and new versions of software without users' having to change their scripts or login files.
- VALET helps YOU maintain your own software packages that are installed in your home directory or your group's LUSTRE work directory.

UNIVERSITY OF DELAWARE

FOR MORE INFORMATION…

- Use the **vpkg_help** command to summarize all of the commands.
- Use **man valet** to view the VALET manual page.

Official VALET documentation:
http://beren.engr.udel.edu/doku.php/valet:start

Contact the IT Support Center

- **Email**: consult@udel.edu

  If you make the first line of the e-mail message

  Type=Cluster-Mills

  your question will be routed more quickly.
- **Phone**: (302) 831-6000
- **Text**: (302) 722-6820