Installing Tool Dependencies

Occasionally additional software needs to be installed in a tool environment to support tool execution. Please follow these instructions in the given order. Keep in mind that an environment maybe shared by multiple tools.

Install a operating system package in the tool execution container

Tool execution containers are based on Docker images. At minimum Debian 7 (wheezy) and Debian 10 (buster) Docker images are provided. Operating system choice is not limited to Debian. We have used CentOS and Ubuntu images as the basis for images to meet tool requirements. Regardless of operating system choice there is a minimum set of scripts and configuration files that must be included to enable the middleware to manage container related processes.

The pairing of Docker image and tool is made according to the following convention:

1. Find an image tag matching tool name and revision.
2. Find an image tag matching the tool name.
3. A default image specified in the middleware.

Multiple Docker tags may be applied to the same image thus reducing the storage requirement for multiple images. Docker images may be modified or created using standard practices based on Dockerfiles. If all Docker images are not made available on all execution hosts, host requirements may be used to aid in execution host selection.

Manually install software in the 'use' infrastructure

Typically we create and run Hapi (HUBzero Apps Program Installers) scripts to download, configure, compile, and install software in the tool execution environment. This is especially important when multiple versions of the same software must be available to server different tools. For example, Tool A may require R version 3.6.3 while Tool B may require R version 4.2.1. These requirements are the result of an ever evolving software landscape. "Use" is very helpful in this case, among others, providing a way to load specific software versions as requirements demand.

All operations for manually installing dependent software for tools must be done by the apps user from a workspace terminal.

As the apps user clone the Hapi repo into the apps home directory:

```
git clone https://github.com/hubzero/hapi.git
```

In the hapi/scripts directory you will find a collection of shell scripts and csv files for software typically installed for use in tool environments. If a script exists for software that you need, just run it! Feel free to add new Hapi scripts of your creation to the GitHub repo by submitting pull requests. We occasionally add scripts as well.

If, after doing a git update on your repository, a script doesn't exist for the software you need, copy an existing Hapi script and modify it. Hapi scripts make your life much easier by downloading, configuring, compiling, installing and even adding the required 'use' environ.d file to the appropriate location.

All tool dependencies are installed by the apps user in an operating system specific directory such as /apps/share64/debian7 and should be owned by the apps user and group. All files must be readable by everyone and all directories must be searchable by everyone. No files or directories should be writable by everyone (seriously don't do this)

* For a full manual run "man use" from a tool session terminal.

USE(1) User Commands USE(1)

NAME

use, unuse - adjust the shell environment

SYNOPSIS

use [options]... [ENVIRONMENT]
unuse [options]... [ENVIRONMENT]

DESCRIPTION

The use command incorporates the specified ENVIRONMENT to the current shell. The unuse command removes it. It optionally records the selection persistently so that subsequent shells will use the ENVIRONMENT. These commands are independent of the shell being run.

An ENVIRONMENT is specified by a configuration file of the same name as found in one of the configuration directories. The ENVIRON_CONFIG_DIRS environment variable specifies a list of directories in which to search for configurations. Each configured ENVIRONMENT specifies a environment variables to set or prepend, shell variables to set, and shell aliases to set.

Some environments are configured to conflict with others. The use command will ask if conflicting ENVIRONMENT should be replaced.

With no arguments, the use and unuse commands will print a synopsys of options and lists all available environments.

-h print available help for a named ENVIRONMENT.

-e environment only. Do not ask about preserving the selection.

-p modify the environment and preserve selection. Do not ask about preserving the selection.

-k keep any conflicting environment. Do not ask about replacing it.

-r replace any conflicting environment without asking.

-x quietly ignore the command if the named ENVIRONMENT cannot be found.

MAKING IT WORK

The following command will describe an environment named xyz:

use -h xyz

The following command will incorporate the xyz environment preserving the environment for future shell invocations. It will also not override any conflicting environments:

use -p -k xyz

The following command will remove the xyz environment but retain its use for future sessions:

unuse -e xyz

INTERNAL OPERATION

use and unuse are actually implemented as shell functions (or as aliases in the case of csh derivatives). The functions pass their arguments to the /etc/environ script which determines the commands that the shell should execute to satisfy the new environment configuration. The script prints these commands, the shell function receives them and evals them.

ENVIRONMENT CONFIGURATION FILES

Configuration files are interpreted shell scripts. Several predefined functions are available to make the the process automatic.

alias NAME "Replacement"

Set a command alias in the shell.

conflict VARNAME
: Define an environment variable to indicate that a type of an ENVIRONMENT is in use. All conflicting ENVIRONMENT configurations should specify the same conflict. An ENVIRONMENT configuration may specify multiple conflicts.

desc "A short description..."
: A short description of the ENVIRONMENT.

help "A lengthy description..."
: A long description of the ENVIRONMENT and how to use it. This description will be formatted when printed.

prepend VARNAME ADDITION
: Prepend ADDITION to the environment variable VARNAME separated with a colon.

setenv VARNAME REPLACEMENT
: Set or replace the environment variable VARNAME with REPLACEMENT.

shellset VARNAME REPLACEMENT
: Set or replace the shell variable VARNAME with REPLACEMENT.