Product Usage Commands

setup [**-e**] [**-k**] [*opts*] *product* [*version*]
Make a product (and its dependant products) available for use. The **-e** option requests "extended" setup actions, if any. The **-k** option says not to unsetup products already set up.

unsetup [*opts*] *product* [*version*]
Undo a previously performed **setup**.

ups list [**-a**] [**-l**] [**-K** *keywords*] [*opts*] *product* [*version*]
List available products on local system The **-l** option asks for a long listing.

ups depend [**-K** *keywords*] [*opts*] *product* [*version*]
List available products' dependencies on local system

upd list [**-h** *host*] [**-a**] [**-l**] [**-K** *keywords*] [*opts*] *product* [*version*]
List available products on distribution system *host*.

ups parent [*opts*] *product* [*version*]
List products which depend on the one specified.

upd depend [**-h** *host*] [**-K** *keywords*] [*opts*] *product* [*version*]
List available products' dependencies on distribution system *host*.

upd parent [**-h** *host*] [*opts*] *product* [*version*]
List products which depend on the one specified on *host*

The **-a** option means to list *all* products matching the other options. The **-h** option defaults to fnkits.fnal.gov.

opts

-f *flavor*[:*flavor*...]
specify allowable product flavor list

-q *qualifier*[:*qualifier*...]
specify qualifier list

-z *database*[:*database*...]
specify product databases to search (default $PRODUCTS)

-g *chain*
Specifies products associated with the given *chain*.

-c -n -o -t -d
Shorthands for "**-g** current", "**-g** new", "**-g** old", "**-g** test", and "**-g** development", respectively.

-H *flavor*
Specifies flavor list as if we were on a host of *flavor*.

-m *file* (**-M** *dir*)
Specify table file (and directory) to use for this product.

-j -R
No dependencies (-j) or only required dependencies (-R)

-v (-vv...-vvvv)
be verbose (more verbose...*really* verbose).

-O *options*
specify options to be passed in $UPS_OPTIONS to any associated actions.

-Z
time this command.

-0 -1 -2 -3
Equivalent to **-f** with zero, one, two, or three components of the local system flavor. (e.g. -0=-f NULL, -2=-f IRIX+5)

Product Maintenance Commands

ups declare [*declare-opts*] *product* [*version*]
Add a product definition to a ups database on the local system.

upd addproduct [**-h** host] [*declare-opts*] *product* [*version*]
Add a product definition and product files to a distribution node *host*

upd modproduct [**-h** host] [*declare-opts*] *product* [*version*]
Change a product to distribution node *host*
(note that **-T** is dissallowed here) **-f** can specify a flavor list here.

ups undeclare [**-C**] [**-y**] [**-Y**] [*opts*] *product* [*version*]
Remove a product definition from a local database. The **-C** option disables uncurrent/unconfigure actions The **-y** option deletes product related files. The **-Y** option deletes product related files, without asking

upd delproduct [*opts*] *product* [*version*]
Remove a product definition from a distribution system. Actions are not performed, and related files are deleted.

upd install [**-h** *host*] [**-i**] [**-I**] [*opts*] *product* [*version*] \
[**-G** "*declare-opts*"]
Install a product from *host*, *and its dependencies*, onto the local system. The **-G** option specifies extra declare options when declaring the product to the local system. **-f** can specify a flavor list here. **-i** means to ignore errors and continue. **-I** means to use an intermediate file for file transfers.

upd fetch [**-h** *host*] [**-I**] [*opts*] *product* [*version*] [**-J** filename]
Download an individual file from a given product, or (without **-J**) list files available for downloading. (Not all product files are available for separate download).

ups verify -a [**-z** *database*] [*product*] [*version*]
Check a ups database for consistency and errors.

declare-opts

All *opts* are also *declare-opts*, except they define the given property rather than searching for it.

-A *nodes*
specify authorized node list

-b *file*
specify file for compiling setups.

-C
Do not perform configure, current, etc. actions.

-D *origin*
Specify origin of product.

-r *directory*
Specify product root directory.

-s
Say what the command would do, but don't do it.

-T
Give the location of a tarfile or other distribution file of the product.

Keywords

These can be given to the **list** and **depend** subcommands of **ups** and **upd** to specify column-formatted output.

Per-product keywords:

product .... Product Name
version .... Product Version
chain .... Associated chain(s)
flavor .... Product Flavor
qualifiers .... Product Qualifiers
+ ..... short for "**product:version:flavor:qualifiers:chain**"
declarer .... User who declared
declared .... Date/time declared
modifier .... User who modified
modified .... Date/time modified
origin .... Product source location
prod_dir .... Product root directory
@prod_dir ... Expanded full path to product root directory
ups_dir .... Product ups directory
@ups_dir ..... Expanded full path to product ups directory
table_dir .... Directory containing following
table_file .... Product descripton table file
@table_file .... Expanded full path to table file
archive_file .... Tarfile or other archive file
authorized_nodes .... Hosts allowed to use product
description .... Description of product
compile_dir
compile_file .... Location of compiled setup info
catman_source_dir
html_source_dir
info_source_dir
man_source_dir
news_source_dir .... Location of documentation files

Per-database keywords:

prod_dir_prefix .... Prefix common to all products
man_target_dir
catman_target_dir
info_target_dir
html_target_dir
news_target_dir .... Destination for documentation files
upd_usercode_dir .... Upd config file information
setups_dir .... Location of bootstrap scripts

Table files

```
FILE=TABLE
PRODUCT=myproduct
GROUP:
  FLAVOR=ANY
  QUALIFIERS=""
  MAN_SOURCE_DIR=man
  DESCRIPTION="My demo product"
COMMON:
  ACTION=SETUP
    setupenv()
    proddir()
 ACTION=CURRENT
    proddir()
    Execute("${UPS_UPS_DIR}/current",UPS_ENV)
 ACTION=UNCURRENT
    proddir()
    Execute("${UPS_UPS_DIR}/uncurrent",UPS_ENV)
End:
```

Actions in a table file may call zero or more functions:

addAlias(NAME, VALUE)
Add an alias or function depending on the shell. A '%s' in the VALUE marks where the argument list should go.

doDefaults()
Perform the default UPS functions

envAppend(VARIABLE, VALUE [,DELIMITER])
Append to an environment variable

envPrepend(VARIABLE, VALUE [, DELIMITER])
Prepend to an environment variable

envRemove(VARIABLE, VALUE [,DELIMITER])
Remove a string from an environment variable

envSet(VARIABLE, VALUE)
Set an environment variable

envSetIfNotSet(VARIABLE, VALUE)
Set an environment variable if it is not already set

envUnset(VARIABLE)
Unset an environment variable

exeAccess(EXECUTABLE)
Check for access to an executable through the PATH

exeActionOptional("newaction")
do the functions associated with ACTION=newaction, do not fail if the newaction section does not exist.

exeActionRequired("newaction")
Process the functions associated with ACTION=newaction.

execute("COMMAND", [NO_]UPS_ENV, [, variable])
Execute a shell independent command [and assign the output to an environment variable]

fileTest(FILE, TEST [, ERRORMESSAGE])
Run a shell test on a file, fail if test is not true.

pathAppend(VARIABLE, VALUE [, DELIMITER])
Append to a PATH-like environment variable

pathPrepend(VARIABLE, VALUE [, DELIMITER])
Prepend to a PATH-like environment variable

pathRemove(VARIABLE, VALUE [, DELIMITER])
Remove a string from a PATH-like environment variable

pathSet(VARIABLE, VALUE)
Set a PATH-like (including PATH) environment variable

prodDir()
Set the *PROD_DIR* environment variable

setupEnv()
Allow "unsetup product" by setting SETUP_*PROD*.

setupOptional("[options] prod [version]")
Setup another UPS product, do not fail if the product doesn't exist

setupRequired("[options] prod [version]")
Setup another UPS product

sourceCompileOpt("fileName")
Check if the file exists and if it does source it and then exit, else continue with the rest of the functions.

sourceCompileReq("fileName")
Source the named script and ignore all subsequent functions

sourceOptCheck(SCRIPT, [NO_]UPS_ENV [, EXIT])
Check if a script exists and if so source it and check return status for error

sourceOptional(SCRIPT, [NO_]UPS_ENV [, EXIT])
Check if a script exists and if so source it

sourceReqCheck(SCRIPT, [NO_]UPS_ENV [, EXIT])
Source a script and check return status for error

sourceRequired(SCRIPT, [NO_]UPS_ENV [, EXIT])
Source a script

unAlias(NAME)
Remove an alias/function.

unprodDir()
Unset the *PROD_DIR* environment variable

unsetupEnv()
Unset the SETUP_*PROD* environment variable

unsetupOptional("[options] prod [version]")
Unsetup a product if it exists

unsetupRequired("[options] prod [version]")
Unsetup another UPS product, or fail.

writeCompileScript("fileName", "action" [, EXIT])
Write a file of compiled functions for the given action.

Variables

UPS_PROD_NAME	Product name.
UPS_PROD_VERSION	Product version.
UPS_PROD_FLAVOR	Product flavor.
UPS_OS_FLAVOR	Operating system.
UPS_PROD_QUALIFIERS	Product qualifiers.
UPS_PROD_DIR	Product instance home directory.
UPS_UPS_DIR	Product instance ups directory.
UPS_SHELL	The shell family (sh or csh).
UPS_OPTIONS	String passed with -O.
UPS_ORIGIN	The location of the master source.
UPS_VERBOSE	Set to 1 if -v specified.
UPS_EXTENDED	Set to 1 if -e specified.
UPS_THIS_DB	Containing database.
PRODUCTS	environment variable PRODUCTS.
UPS_COMPILE	COMPILE_DIR/COMPILE_FILE.
UPS_SOURCE	either "source" or "." per UPS_SHELL

UPS/UPD Reference Card

(for version v4_x)

Notation

Commands and options are listed as:

literal_text *fill_in* [optional_items]
Description

Common Command Examples

ups list fred
List instances of fred that have a current chain in the databases in $PRODUCTS

ups list -a -K+:prod_dir fred
List all instances of fred in the databases in $PRODUCTS in column format; columns are:
product:version:flavor:qualifiers:prod_dir

ups list -act fred
List all instances of fred that have a current or test chain in the databases in $PRODUCTS

setup fred
setup the current instance of the fred product in the databases in $PRODUCTS

setup -t fred **-z** /path/to/db
setup the test instance of the fred product in the database /path/to/db

setup -t -f IRIX+5 **-q** mips3 fred
setup the test instance of the fred product for IRIX+5 systems built for mips3

upd fetch fred v1_0
list files in the fred product directory on the distribution server

upd fetch fred v1_0 -J README
download the README file for fred v1_0

upd install fred v1_0 **-G** "**-c**"
install version v1_0 of fred on this system, as current.

ups declare -z /my/upsdb **-2 -t** fred vx_y \
-r /path/to/fred **-m** vX_Y.table
Declare fred vX_Y in database /my/upsdb as chain "test", with prod_dir of /path/to/fred, and table file named vX_Y.table which resides in either /my/upsdb/fred, or /path/to/fred. The flavor of the declaration is OS+major-release (i.e IRIX+5) due to the -2 option.

upd update table_file:ups_dir fred vY_Z
Update the table file and ups directory of fred vY_Z and its dependencies.

upd addproduct -f OSF1+V4 fred v2_0 \
-T mytarfile \
-m mytablefile \
-M /dir/containing/tablefile \
-U /path/to/upsdir
Add fred v2_0 for flavor OSF1+V4 to fnkits, using an already built tarfile and giving explicit paths for finding the table file and ups directory.

upd addproduct -f OSF1+V4 fred v2_0
Add fred v2_0 for flavor OSF1+V4 to fnkits, finding the product root, table file, and ups directory in the local ups database, and building the tarfile for the product automatically.