NAME

yume-examples – Run examples for yume menu system

SYNOPSIS

yume-examples [examplesdir]

DESCRIPTION

yume-examples is a shell script that runs examples of yume menus and yume menu techniques. (yume is a flexible menu system for X Window System, based on Gtk or on xforms library. See yume(1).) Run yume-examples from within the yume examples directory, or provide it a path to that directory as the first parameter.

The body of this man page describes some of the examples found in the examples subdirectory of the yume distribution, which you can run from menus started by yume-examples.

There are several kinds of examples. Some examples illustrate how yume works, others illustrate how to avoid some shell-quoting and shell-environment restrictions or difficulties, and a few are supposed to be useful utilities. The menu that yume-examples starts (when you click "Show list of yume examples") includes buttons to display the shell code of each example. It also includes buttons to show explanatory notes for many of the examples.

The information shown in this man page, plus a few additional comments and numerous screen-shots, also appears in webpages <http://yume3.sourceforge.net/yume3/web/eg-overview.html> and <http://yume2.sourceforge.net/examples/eg-overview.html>. Also see <http://yume3.sourceforge.net/> and <http://yume2.sourceforge.net/>.

OPTIONS

Most of the examples have no option switches, but those that start with "la" accept one optional parameter, a file name. If the user doesn't provide a file name, the utility uses shell commands to select the most-recently-modified file with a given extension.

Note, the following shell variables affect the examples.

SHELL -- Name of shell that yume invokes to execute shell commands

YUME -- All of the scripts in yume3/examples will accept shell parameter YUME as the qualified filename to execute as yume. If, for example, one wishes to execute screen-save using yume2 or yume3 rather than yume, one can say:

YUME=‘which yume2‘ ./screen-save

or

YUME=‘which yume3‘ ./screen-save

in yume3/examples directory.

EXAMPLES - Useful Utilities

lac - Latest-C script

lac sets up a yume menu to use gcc to compile a C program, an editor to edit it, a command to execute it, and a command to "ls -l" it. For simple programs, just say "lac", click EE on the gcc command, and click Do: on the edit command. Each time you save your program and want to compile and test it, just roll the mouse cursor across the gcc command. Then click the execute button to run it. For more complex programs, say "export YMAKE=make" before using lac. Then lac will have a make command as well. Click EE on the make command after you say "lac".

If the program uses certain math functions or includes math.h, then lac adds -lm to gcc option list.

lac as it stands uses gcc as the compiler, with option list -O3 and -Wall. But if any of YGCC,

YOPTS, YEDIT, YMAKE, or YEXT are defined shell variables when lac starts, it will use those values instead. Scripts lacc (using g++ on .cc files) and lacx (linking with xforms libraries) are examples of tailoring lac by a wrapper to set YEDIT, YOPTS, YMAKE, etc. Or, you can edit lac directly if you prefer, or if you wish to add menu lines for gdb, etc.

Note that setup for the "ls -l" button in lac is via lines like

export BASE=‘basename $FILE $YEXT‘
-bu -la ’Ls it’ ’ls -l $BASE*’

where the first line exports BASE into the yume environment, and where the single-quotes in the second line prevent expansion of $BASE* when yume is invoked. That is, expansion is delayed until the ’Ls it’ button is clicked.

lacc - Latest-C++ script

lac sets up a yume menu to use g++ to compile a C++ program, an editor to edit it, a command to execute it, and a command to "ls -l" it. It creates the menu by exporting variables YGCC, YEXT, and YMAKE, then invoking lac.

lac - Latest-C script with xforms lib linking

lacx sets up a yume menu to use gcc to compile a C program, an editor to edit it, a command to make it and one to execute it, and a command to "ls -l" it. It creates the menu by exporting variables YEDIT, YOPTS, and YMAKE, then invoking lac.

lala - Latest-Latex script

lala locates the most-recently modified *.tex file in the current working directory, or locates a specified file. It prepares a yume menu with latex-2ps, xdvi, ps2pdf, edit, and lpr commands for that file. Some commands are -do items, so that EE can be enabled; others are -bu items. (v0.04 doesn’t support right-click button setups yet.) Change it as you like.

Invoke lala by "lala" or by "lala f" where f represents a filename. If f appears, rather than finding the most-recently-changed .tex file, lala finds the base name of f and sets up the yume menu accordingly.

latex-2ps latexs the specified tex file if it’s newer than its dvi file, or if a third parameter is given. latex-2ps runs latex invisibly unless an error occurs. It beeps when done if latex finishes without error; but if a latex error occurs, it uses script latex-2err to present selected latex error log messages in an xterm window.

The editor is selected via environment variable $YEDIT, or $EDITOR if $YEDIT is empty, or emacs if $EDITOR is empty too.

screen-save - Capture-a-screen-picture script

screen-save creates a menu with buttons labeled ls, ll, Show, exit, Import, Import window, and Import w/ frame, for saving screen-shots sequentially in a screen-saver directory.

It accepts 3 or 4 parameters, to set values of shell variables PIXDIR, PREFIX, and DELTA. If not specified otherwise, these default to $HOME/pix, "screen-", and 10. Under the defaults, screen shots are saved in $HOME/pix with names of form screen-nnnn.jpg, (eg, screen-1100.jpg, screen-1110.jpg, ...) where nnnn is a number that increases by 10 between pictures.

Click "Import" to save a screenshot as next-numbered .jpg file in $PIXDIR directory. This uses import from the ImageMagick suite. See import(1). After you click import, a fat plus sign will replace the usual mouse cursor. Move it to a corner of an area you wish to capture, press the left mouse button, move to an opposite corner, release mouse button. import will beep, will capture an image of your selected area, then will beep again.

Click "Import window" to save a screenshot of a particular window’s contents as next-numbered .jpg file in $PIXDIR. After you click "Import window" a small plus sign will replace the usual

mouse cursor. Move it onto some part within the window you wish to capture and press the left mouse button. import will beep and expose the window if it was buried, will capture an image of the window's contents, then will beep again. Clicking on screen background areas will capture the whole screen.

Click "Import w/ frame" to save a screenshot of a particular window's contents and its window-manager frame.

Click "Show" to run "kuickshow .", which will display pictures in $PIXDIR directory. If you use some other picture viewer, edit screen-save accordingly.

See file screen-save.txt in examples for more information about how screen-save sets import parameters.

EXAMPLES - Various shell techniques

find-ls-count - Find a file, ls files, Count files

This example makes a menu with five buttons, including two that are labeled "Find filename" and "Count Files".

You can "drop" text on "Find filename", after which find-ls-count will list all the file names in the current directory and its subtree that contain that text. This requires xclip, an X Window System clipboard utility. The value of ‘xclip -o‘ is current clipboard contents.

When you click "Count Files", find-ls-count tells how many files are in the current directory and in each of its children. This illustrates some of the necessary backslash-quoting of dollar signs in a script.

menu-at-var - placing a menu at different locations

This example makes a menu with one button, labeled with the hostname of the machine running the yume command. When you click the button, it opens an xterm from that machine. The example illustrates a simple technique for placing the buttons at different locations on your screen, depending on what host you are logged into. For the example, if you are logged into zeda when you run menu-at-var, the menu will appear at -121+1; if yuli, at -61+1; if some other, at -1+1. All these locations are close to the top right corner of the screen.

play-sound-delay - play a sound after a delay

This example makes a menu with a row of four buttons (labeled exit, 60, 180, and 240) and a do: item, containing the command "./play-sound-delay 3". When it starts, play-sound-delay tests if it has no parameters; if that is so, it uses yume to create a menu, and exits. If not, it uses the parameter as a length of time to sleep, in background, before playing a sound with "play pop.wav". play is a SoX (Sound eXchange) program, and file pop.wav appears in yume's examples directory. If you don't have play on your system, the script will just use "echo -ne '\a'" to beep.

Techniques illustrated: Invoking a script one way to start yume, and another to execute complicated actions ($0 represents script name); command grouping with braces, { ... ; }; redirecting stdout and stderr outputs to /dev/null.

Note that this script says "#!/bin/bash" in its first line, rather than "#!/bin/sh", to emphasize that some of the notation may be bash-specific.

example-starter1 - A menu to start some yume examples

This example has a hard-coded list of examples from the yume examples directory. Clicking a button runs the example corresponding to the button's label. Note that output from the started examples appears on the terminal that example-starter begins on.

This example has no button-width controls (-bw, buxMargin, or buQuanta), so button widths are

based on text labels of buttons and are rather varied. By contrast, example-starter3 uses buxMargin and buQuanta to obtain a smaller set of button widths, and example-starter4 uses -bw parameters, producing standard widths of buttons quite easily.

example-starter2 - Another menu to start some yume examples

This example creates a list of examples from the yume examples directory and displays a menu from which examples can be selected and run. Clicking a button runs the example corresponding to the button's label. example-starter2 is rather simplistic, producing a vertical line of buttons. All of the items listed by example-starter2 are executable, but several of them are just supporting files, like abc or hello, rather than yume examples per se.

example-starter3 - Another menu to start some yume examples

This example creates a list of examples from the yume examples directory and displays a menu from which examples can be selected and run. Clicking a button runs the example corresponding to the button's label. example-starter3 excludes some irrelevant programs via grep.

example-starter3 produces a vertical column of lines of 3 buttons, aside from a few top and bottom rows of fewer buttons. In the 3-button lines, the left button runs the example; the middle button displays its shell code; and the right button displays a note about the example, if one exists. That is, when a file x.txt corresponding to example file x exists, the right button will be labeled "N" rather than being a blank, no-op button. When you click an "N" button for example x, it runs "less x.txt" in a new xterm.

example-starter3 shows another variation on shell quoting, by using xargs. Either of these two commands

yume $L
yume "$L"

would treat the string 'xterm -e less x.txt' in $L as four parameters ("'xterm", "-e", "less", and "x.txt'"), but using xargs causes it to be properly treated as one parameter to yume.

example-starter4 - Another menu to start some yume examples

This example creates a list of examples from the yume examples directory and displays a menu from which examples can be selected and run. It is like example-starter3 but with two variations: (1) it has no -at geometry specification, and (2) it uses -bw aa and -bw daa codes to set column widths. It produces a better-looking menu than that of example-starter3, with simpler parameters.

url-and-misc - Sending URL's to browsers, xterm use, etc

This example with a bunch of buttons illustrates a yume command with a lengthy list of parameters. It is mostly a concatenation of the separate examples from early um1, um2, ... um6 files.

um1 um1 has 'NS paypal' and 'NS ebay-seller' buttons that open specific web pages in netscape. Also see um1.txt

um2 um2 has 'FF ebay-seller' and 'FF KLVM-weather' buttons that open specific web pages in firefox. Also see um2.txt

um3 um3 has 'Cal3', cdplay, and eject buttons that use right size of xterm for "cal -3" display, or do simple CD controls. Also see um3.txt

um4 um4 has 'Clip url -> FF' and 'Ebay# -> FF' buttons for dropping URL or Ebay# on a button to open corresponding pages in firefox browser. Also see um4.txt

um5 um5 has 'HMS' and 'Factor' buttons for dropping a number on a button to get computed results. Also see um5.txt

um6 um6 has a 'Host/Whois/Dig' button for dropping an IP number on button to get Host/Whois/Dig results. Also see um6.txt

wordexp-vals
Demonstrate values of shell special parameters that occur when including a file of yume parameters

xterms-w-sh
Illustrate several %x meta-mode command modifiers -- %:, %+, %%. Examples include running a few commands in an xterm and then starting a shell there, as well as doing the same thing while exiting from yume.

COMMENT - Meaning of yume

"yume" in Japanese = "dream" in English.

FILES

$HOME/.yumerc User configuration file for yume default values

ENVIRONMENT

yume uses a few environment variables as follows.

$SHELL -- Name of shell that yume invokes to execute shell commands

$YUME -- All of the scripts in yume3/examples will accept shell parameter YUME as the qualified filename to execute as yume. If, for example, one wishes to execute screen-save using yume2 or yume3 rather than yume, one can say:

YUME=‘which yume2‘ ./screen-save

or

YUME=‘which yume3‘ ./screen-save

in yume3/examples directory. For some scripts, one can use YUME=echo or YUME='printf <%s>' to see an approximation of the yume parameters line, that is, parameters with a layer of quoting stripped or with <,> substituted to distinguish parameters.

The environment that a yume menu runs in is a copy of the environment in place when yume started (except when menu contains used -iv items). In the usual UNIX scheme of things, changing the environment belonging to a parent or grandchild process has no effect on the environment belonging to a child process. Consider this sequence:
> export G=7
> yume 'echo $G; export G=8'
> echo $G; export G=9
Suppose the first two lines are entered at an xterm > prompt, and then do: is clicked several times on the menu that appears. Each time, "7" will be output to terminal. When the third line is entered, 7 will be output again, and when do: is clicked, 7 will be displayed. In short, the environment changes caused by export G=8 and export G=9 do not occur in the environment of the running yume process, hence do not affect it.

A yume script with -iv fields modifies yume's runtime environment (but not the environment of its parent); see yume(1) and also see calc-wrap.

DIAGNOSTICS

Some examples illustrate how to obtain diagnostic outputs for debugging scripts that use yume. See, for example, errors-demo, wordexp-vals, and xterms-w-sh.

BUGS

The examples hard-code numerous names of shell and application commands. These should instead be set by a configuration program.

AUTHOR

James Waldby <j-waldby at pat7 dot com> Please report bugs to: yume@pat7.com