Hull

A collection of all kinds of shell scripts

Organisation: Copyright (C) 2008-2021 Olivier Boudeville

Contact: about (dash) hull (at) esperide (dot) com

Creation date: Sunday, August 17, 2008

Lastly updated: Tuesday, June 29, 2021

Version: 1.0.10

Status: Stable

Dedication: Users and maintainers of `Ceylan-Hull`.

Abstract: The role of `Hull` is to concentrate various, generic-purpose convenience shell scripts, on behalf of the Ceylan project.
The latest version of this documentation is to be found at the official Ceylan-Hull website (`http://hull.esperide.org`).
The documentation is also mirrored here.

Overview

Here are **a few scripts, sorted by themes, that may be convenient for at least some users**.
Many of them can display their intended usage on the console by specifying them the `-h` / `--help` command-line option.
In each category, scripts are roughly sorted by decreasing interest/update status. The scripts that we deem the most useful are described more precisely at the end of this document.
As much as possible, we try to name these scripts functionally rather than in terms of tools, so that the implementation they rely upon can be updated as transparently as possible (i.e. with as little change as possible in the user's habits).
Each script entry is a link pointing directly to the script itself. They may be directly read from the Hull repository.
Most of them are certainly not rocket science.

Note

Some of these scripts may be a bit outdated, as only a small subset of them are routinely used; rely on them with caution!
And tell us if you would like some of them to be updated.

Table of Contents

Listing of the recommended shell scripts per usage

Related to Filesystems

To search for files and content

- wh ("where") : a more convenient "find"; see the wh full usage

- regrep: recursive grep for the ones that lack it; see the regrep full usage

To list filesystem elements

- list-filesystem-entries-by-size.sh: lists, from any specified directory otherwise from the current one, the direct filesystem entries (local files and directories), sorted by decreasing size of their content

- list-files-in-tree-by-size.sh: lists, from any specified directory otherwise from the current one, all files in tree, sorted by decreasing size of their content

- list-files-in-tree-by-most-recent-modification-time.sh: lists, from any specified directory otherwise from the current one, all files in tree, sorted from the most recently modified to the least

- list-local-directories.sh:: lists all local, direct directories of any specified directory, or from the current one. Useful for overcrowded base directories

To fix names, paths, permissions, content

- fix-filename.sh: "corrects" the name of the specified file (or directory), to remove spaces and quotes (replaced by '-'), accentuated characters in it, etc.

- fix-paths-in-tree.sh: does the same as `fix-filename.sh`, yet in a tree

- fix-file-permissions.sh: corrects, for all files in the current directory, the UNIX permissions for the most common file extensions

- set-files-unexecutable-in-tree.sh: ensures that all files found recursively from the current directory are not executable

- fix-unbreakable-spaces.sh: removes any unbreakable space in specified file

- fix-unbreakable-spaces-in-source-tree.sh: removes any unbreakable space in specified tree

- fix-whitespaces.sh: fixes whitespace problems into specified file; useful to properly format files that shall committed when not using Emacs as text editor

- fix-whitespaces-in-tree.sh: fixes whitespace problems in all elligible files found from specified root directory tree; useful, when operating on a fork, to properly format files once for all and commit the result so that these formatting changes remain cleanly and clearly separated from the others

- rename-files-in-tree.sh (just an example of pattern substitution in filenames)

To inspect file and directory content

- compute-checksum-in-tree.sh: computes the checksum of all files in specified tree and stores them in the specified text output file

- display-tree-stats.sh: displays simple, key stats about the specified tree (typically in order to compare merged trees)

- list-broken-symlinks.sh: lists, from any specified directory, all the symbolic links that are broken

To compare files and trees

- diff-dir.sh: performs a (single-level, non-recursive) comparison of the content of the two specified directories

- diff-tree.sh: compares all files that are present both in first and second trees, and warns if they are not identical; warns too if some files are in one directory but not in the other

See also: Myriad's merge.sh script, a considerably more powerful tool for merging trees.

To copy/transfer

- transfer-tree.sh: copies (possibly through the network) a tree existing in one location to another one, in a merge-friendly manner

To remove filesystem elements

- srm (for "secure rm"): stores deleted files in a trash directory, instead of deleting them directly; see the srm full usage

- empty-trash.sh: empties the trash directory that can be filled thanks to our srm script

- remove-broken-symlinks.sh: removes from the specified directory all the symbolic links that are broken

Related to (UNIX) Processes

- top.sh: triggers the best "top" available, to monitor processes and system resources

- watch.sh: tracks (over time) processes that may be transient

- monitor-demanding-processes.sh: monitors endlessly the most CPU-demanding processes. Typically useful to catch an otherwise idle process that uses a full core as soon as the system gets not actively used anymore

- benchmark-command.sh: returns a mean resource consumption for the specified shell command

- list-processes-by-size.sh: lists processes by decreasing size in RAM

- list-processes-by-cpu-use.sh: lists processes by decreasing use of CPU

- kill-every.sh: kills all processes that match specified name pattern

- kill-always.sh: as long as this script is kept running, kills any process matching the specified name

- kill-trace-supervision-tool.sh: kills all instances, run by the current user on the local host, of the tool that is used to perform trace (log) supervision

Related to Network Management

- test-network.sh: diagnoses whether the various network basic facilities are functional (IP connectivity, DNS, on the LAN or on the WAN)

- manage-wifi.sh: starts/gets status/scans/stops the wifi support

- ip-scan.sh: scans all IPs with any specified prefix, searching for ICMP ping answers (useful to locate some devices in a local network)

For Development

- update-copyright-notices.sh: updates the copyright notices of code of specified type found from specified root directory (to run at each new year)

- update-all-copyright-notices.sh: updates the copyright notices of code of specified type found from specified root directory, based on the specified year range

- add-header-to-files.sh: adds specified header to specified files

- remake.sh: forces a remake of specified generated file (ex: `.o` or `.beam`)

- list-core-dumps-in-tree.sh: locates all core dump files in current tree

- test-with-valgrind.sh: uses Valgrind to perform quality test on the specified executable

- reformat-source-style.sh: applies some style change to specified file

- reformat-source-style-in-tree.sh: applies some style change to C/C++ files in specified tree

See also the Erlang-related Myriad scripts.

For Version Control System (VCS)

- dci: assists efficiently and conveniently the commit of specified file(s)

- dci-all: selects recursively from current directory the files that should be committed (either added or modified), and commits them; for each of the modified files, shows the diff with previous version before requesting a commit message

- dif: shows on the console the differences between the current versions of the (possibly specified) files on the filesystem and the staged ones (i.e. the changes that might be added)

- difg: graphical version of `dif`

- dif-prev.sh: compares the current (committed) version of specified file(s) with their previous one

- difs: shows the differences between the staged files and their committed version

- st: shows the current VCS status of the specified files

- up: updates the current local version of the VCS repository

- show-branch-hierarchy.sh: shows the hierarchy of the branches in the current VCS repository

- list-tags-by-date.sh: lists, for the current VCS repository, all (annotated) tags, from the oldest one to the latest one

- list-largest-vcs-blobs.sh: lists the largest blob objects stored in the current VCS repository, sorted by decreasing size

Replacement-related

- replace-in-file.sh: replaces in specified file the specified target pattern with the replacement one

- replace-in-tree.sh: replaces, in files matching the specified pattern found from the current directory, the specified target pattern with the replacement one

- substitute-pattern-in-file.sh (possible duplicate): replaces in specified file every source pattern by specified target one

- substitute-pattern-in-tree.sh (possible duplicate): substitutes every source pattern by specified target one in all files, starting recursively from current directory

- replace-lines-starting-by.sh: replaces in specified file every line starting with the specified pattern by the specified full line

Multimedia-related

For Audio

- play-audio.sh: performs an audio-only playback of specified content files (including video ones) and directories

- extract-audio-from-video.sh: strips the video information from specified MP4 file to generate a pure audio file (.ogg) out of it (original MP4 file not modified); useful, as the resulting file is smaller and less resource-demanding to playback

- resample.sh: resamples the target audio file to the specified frequency, keeping the same bitdepth

- ogg-encode.sh: encodes specified sound file in OggVorbis after having removed any leading and ending silences, adjusting volume

- convert-vorbis-to-mp3.sh: converts a Vorbis-encoded Ogg file to MP3 (sometimes it is useful to use older players)

- convert-vorbis-tree-to-mp3.sh: converts all Vorbis-encoded Ogg files to MP3 in specified tree

- trim-silence-in.sh: removes any silence at begin and end of specified file, which is updated (initial content is thus not kept)

For Voice Synthesis / Text-To-Speech

- say.sh: says specified text, based on text to speech

- record-speech.sh: records the specified speech with specified voice in the specified prefixed filename

- install-speech-syntheses.sh: installs all listed voices for speech synthesis

- get-and-install-MBROLA-and-voices.sh: installs MBROLA and corresponding voices

- test-espeak-voices.sh: tests the voices supported by espeak

- test-voices.sh: tests all supported voices

For Video

- convert-mov-to-x264-mp4.sh: converts specified MOV file to MP4

- fix-video-mode.sh: forces a specified video resolution

For Snapshots (Camera Pictures)

- rename-snapshot.sh: renames the specified picture file, based on its embedded date (used as a prefix, if appropriate), and with a proper extension; new assigned names are typically '20160703-foo-bar.jpeg'; if no filename is specified, operates on all files of the current directory

- rename-snapshots.sh: renames *all* snapshots found from current directory, so that they respect better naming conventions

- remove-snapshot-metadata.sh: removes any metadata (typically EXIF) stored in the specified snapshot(s)

- generate-lighter-image.sh: generates a lighter (smaller and of decreased quality) version of the specified image

- generate-lighter-images.sh: reduces the size of *all* image files found in current directory

CCTV-related

- monitor-cctv.sh: performs online, direct monitoring from a networked security camera (CCTV), with an average quality and no audio

- fetch-cctv-monitorings.sh: fetches locally (and leaves on remote host) the set of CCTV recordings dating back from yesterday and the three days before; designed to be called typically from the crontab of your usual reviewing user

- review-cctv-monitorings.sh: allows to (possibly) fetch from server, and review conveniently / efficiently any set of CCTV recordings dating back from yesterday and the three days before

Document-related

- check-rst-includes.sh: checks that all RST files found from current directory are included once and only once in the sources found

- convert-rst-to-mediawiki.sh: converts specified RST source file in a mediawiki counterpart file

- convert-mediawiki-to-gitlab-markdown.sh: converts specified Mediawiki source file in a GitLab Markdown counterpart file

- generate-mermaid-diagram.sh: generates a PNG file corresponding to the specified file describing a Mermaid diagram

- generate-pdf-from-latex.sh: generates a PDF file from a LaTeX one, and displays it

- regenerate-rst-files.sh: updates generated files from more recent docutils files

- track-rst-updates.sh: tracks changes in the specified RST source file in order to regenerate the target file accordingly

- spell-check-rst-tree.sh: spellchecks all RST files found from current directory

- switch-text-encoding.sh: reencodes the specified text file according to the specified encoding (by default Unicode, precisely `utf-8`)

One may also rely on the Ceylan-Myriad's scripts for documentation, notably generate-docutils.sh and generate-pdf-from-rst.sh.

System-related

Admin-related

- check-filesystem.sh: checks for errors, and repairs if needed, the specified filesystem

- check-ntp.sh: reports the current, local NTP status

- set-time-and-date-by-ntp.sh: sets time and date by NTP thanks to specified or default server

- display-ups-status.sh: displays the status of specified UPS

- report-raid-disk-status.sh: reports the status of the specified RAID array (script for automation)

- report-disk-smart-monitoring.sh: reports a state change of the specified SMART-compliant disk (script for automation)

- report-ups-status.sh: reports a state change of the specified UPS (script for automation)

- record-system-settings.sh: records in-file the main system settings of the local host (typically for a durable storage in VCS to help later troubleshootings)

- get-host-information.sh (possible duplicate): returns the main system settings of the local host, and stores them in-file

- shutdown-local-host.sh: shutdowns current, local host after having performed any relevant system update

- update-locate-database.sh: updates the 'locate' database, for faster lookups in filesystems

- mount-encrypted-usb-device.sh: mounts specified LUKS-encrypted device (ex: a USB key, or a disk), as root or (preferably) as a normal user

To install software

- install-rebar3.sh: installs the rebar3 Erlang build tool

- install-godot.sh: installs the Godot 3D engine

- install-unity3d.sh: install the Unity3D engine

System information-related

- display-opengl-information.sh: displays information regarding the local OpenGL support

- display-raid-status.sh: displays information regarding a local RAID array.

Convenience-related

- activate-keyboard-backlighting.sh: (des)activates (per-level) the keyboard backlighting

- disable-touchpad-if-mouse-available.sh: ensures that the touchpad (if any) is enabled iff there is no mouse connected

- toggle-touchpad-enabling.sh: toggles the touchpad activation state

- display-to-video-projector.sh: displays a screen to a video projector (various examples thereof)

Distribution-related

- update-distro.sh: updates the current distribution, and traces it

- update-distro-mirrors.sh: updates the mirrors used for packages by the current distribution

- update-aur-installer.sh: updates the local AUR (Arch User Repository) installer

For encryption

- crypt.sh: encrypts as strongly as reasonably possible the specified file(s), and removes their unencrypted sources

- decrypt.sh: decrypts specified file(s) (does not remove their encrypted version)

For security

- for the management of credentials (i.e. sets of login/password pairs):

  - open-credentials.sh: unlocks (decrypts) the credential file whose path is read from the user environment, and opens it; once closed, re-locks it (with the same passphrase)
  - lock-credentials.sh: locks (encrypts) the credential file whose path is read from the user environment
  - unlock-credentials.sh: unlocks (decrypts) the credential file whose path is read from the user environment

- lock-screen.sh: locks immediately the screen

- inspect-opened-ports.sh: lists the local TCP/UDP ports that are currently opened

One may also rely on Myriad's generate-password.sh script in order to generate safe, strong passwords (typically one per account listed based the `*-credentials.sh` scripts just above).

On a side note, we also recommend registering a domain name of one's own (ex: `myfoobar.org`) and settting up a catch-all regarding the recipient email addresses (so that all emails sent to `*@myfoobar.org` are routed to your own personal email address).

Then, to avoid messing with your wildcard naming in terms of email addresses, you may choose first any conventional, meaningless email address prefix (ex: `deneb`) to be dedicated to per-service communications. Next step is, in order to register to any online service (let's say it is named `http://someacme.com`), to declare to this service that your email address is `deneb-someacme@myfoobar.org` (or perhaps `deneb-someacmedotcom@myfoobar.org` if some ambiguity could remain). You would then generate a corresponding password with `generate-password.sh` and store the email/password pair among the credentials managed by the scripts above.

The advantages of this procedure as a whole are: strong and unique password, stronger pair of credentials as the login is not constant (hence less predictable[1]), and if ever you start receiving spam targeted to `deneb-someacme@myfoobar.org` then the corresponding website (ex: `http://someacme.com`) may have some explanations to share...

Firewall configuration

- iptables.rules-Gateway.sh: manages a well-configured firewall suitable for a gateway host with masquerading and various services

- iptables.rules-Minimal-gateway.sh: sets up a minimal yet functional firewall suitable for a gateway host

- iptables.rules-LANBox.sh: manages a well-configured firewall suitable for a LAN host

- iptables.rules-inspect.sh: lists the currently-used firewall rules

- iptables.rules-FullDisabling.sh: disables all firewall rules

- iptables.rules-TemporaryDisabling.sh: disables temporarily all firewall rules

For smartphones

- adb-pull.sh: uploads specified local files, possibly based on expressions to the already connected and authorizing Android device

- adb-push.sh: downloads in the current directory, from the already connected and authorizing Android device, files and directories (recursively)

- set-usb-tethering.sh: sets (or stops) USB tethering on local host, typically so that a smartphone connected through USB and with such tethering enabled shares its Internet connectivity with this host

For archive management

- archive-emails.sh: archives properly and reliably (compressed, cyphered, possibly transferred) the user emails

- Manages reference version of files, by storing them in a "vault":

  - catch.sh: stores a file in a vault directory and makes a symbolic link to it, so that even if current tree is removed, this file will not be lost

  - retrieve.sh: retrieves at least one file already stored in vault by creating link towards it, from current directory

  - update-directory-from-vault.sh: updates all files in specified directory from their vault counterparts

---

[1]Depending on the use, the suffix for a site may be different from the site name, provided of course a translation table is kept on your side (typically in said credentials file). For example, one just has to record that for `http://someacme.com` he chose `deneb-bluetiger@myfoobar.org` (stealthier/less obvious - but then harder to incriminate a rogue website).

- make-git-archive.sh: makes a backup (as an archived GIT bundle) of specified project directory, stored in specified archive directory

- snapshot.sh: performs a snapshot (tar.xz.gpg archive) of specified directory

- list-for-backup.sh: enumerates in current directory all files, specifies their name, size and MD5 sum, and stores the result in a relevant file

Web-related facilities

- generate-html-map.sh: generates a simple HTML map with links from the available pages in specified web root

- backup-directory.sh: backups specified directory to specified backup directory on the specified server, using specified SSH port

- fetch-website.sh: downloads correctly, recursively (fully but slowly) web content accessible from the specified URL

- generate-awstats-report.sh: to trigger the generation of an Awstats report (prefer using US-Web instead)

- make-markup-shortcut-links.sh: creates shortcuts (symlinks) for the `put-*-markup.sh` micro-scripts, in order to assist a bit the user of following languages:

  - for HTML: `bold`, `box`, `cent`, `code`, `def`, `defel`, `em`, `img`, `linked`, `lnk`, `ordered`, `para`, `sni`, `strong`, `table`, `tit`, `toc`
  - for RST: `imgr`, `linkr`
  - in general: `fuda`, `newda`

For user notifications

- notify.sh: notifies the user about specified message, possibly with a title and a category

- timer-at.sh: requests to trigger a timer notification at specified (absolute) timestamp

- timer-in.sh: requests to trigger a timer notification in specified duration

- timer-every.sh: requests to trigger (indefinitely, just use CTRL-C to stop) a timer notification every specified duration

- start-jam-session.sh: starts a jam session interrupted by a notification every period, to avoid remaining still for too long

- bong.sh: plays the specified number of bong sound(s)

- beep.sh: plays a beep to notify the user of an event

Shell Helpers

To facilitate shell sessions:

- mo: shorthand for a relevant version of `more`

- hide.sh: hides specified file or directory (simply by adding a `-hidden` suffix to its filename), while unhide.sh does the reverse operation

- set-display.sh: sets the X display to specified host; if none is specified, sets it to the local one

- get-date.sh: returns the current date in our standard short format (ex: `20210219`)

About Configuration

For keyboards

- reset-keyboard-mode.sh: resets the keyboard mode, typically should it have been modified by a misbehaving program

- set-keyboard-layout.sh: sets the X keyboard layout

- set-french-keyboard.sh: sets the keyboard layout to the French (for all X)

- set-auto-repeat.sh: enables the keyboard auto-repeat mode (to issue multiple characters in case of longer keypresses)

Of Environment and related

- unset-all-environment-variables.sh: unsets all environment variables, typically to rule out stranger account-related side-effects

- update-emacs-modules.sh: updates the basic Emacs modules that we use

- set-local-environment.sh: sets up a few installation conventions

Script-related

- protect-special-characters.sh: prevents special characters in specified expression from being interpreted by tools like sed

- encode-to-rot13.sh: returns a ROT13-encoded version of specified parameters

- default-locations.sh: detects the path of the most common UNIX-related tools (this script shall be sourced, not executed)

- term-utils.sh: defines facilities to make a better use of terminals, as a very basic, limited text user interface (this script shall be sourced, not executed)

Launch-related

- e: to edit (i.e. open potentially for updating) all kinds of files

- v: to view (i.e. open for reading only) all kinds of files and directories

- email.sh / courriels.sh: to launch a suitable e-mail client

- launch-irc.sh: to launch a suitable IRC client

Miscellaneous

- eat-CPU-cycles.sh: generates some CPU load

Detailed description of some frequently-used scripts

wh wh (for "where"): searches (as a more user-friendly 'find') all files and directories matching <filePattern>, from <starting_directory> if specified, otherwise from current directory.

Usage: wh [-h|--help] [--verbose] [-q|--quiet] [--no-path] [--exclude-path <a directory>] <filePattern> [<startingDirectory>]

Options:

> [-q|--quiet]: only returns file entries (no extra display); suitable for scripts (ex: for f in $(wh -q 'foo*'); do...)
>
> --no-path: returns the filenames without any leading path
>
> --exclude-path DIR: excludes specified directory DIR from search

regrep regrep: recursive grep for the ones that lack it.

Usage: regrep [-v|--verbose] [-q|--quiet] [-i|--insensitive] [-r|--restructured] <Expression to be found in files> [<Filter on files>]

Options:

> -v or --verbose: be specifically verbose
>
> -q or --quiet: be specifically quiet, just listing matches
>
> -f or --filenames-only: display only filenames, not also the matched patterns, and if there are multiple matches in the same file, its filename will be output only once (implies quiet)
>
> -i or --insensitive: perform case-insensitive searches in the content of files, and also when filtering any filenames
>
> -r or --restructured: use ReStructured text mode (skip tmp-rst directories, search only *.rst files)

Example: regrep -i 'little red rooster' '*.txt'

See also: ergrep (for searching specifically in Erlang files), pygrep (Python version) and cgrep (C version)

srm srm (for "secure rm"): stores deleted files in a trash instead of deleting them directly, in order to give one more chance of retrieving them if necessary. Ensures that no two filenames can collide in trash so that all contents are preserved.

Usage: srm <files to delete securely>

See also: emptyTrash.sh

See also

- the `tests` subdirectory, for a few tests of specific facilities provided here

- the `mostly-obsolete` subdirectory, for the scripts we deprecated

Support

Bugs, questions, remarks, patches, requests for enhancements, etc. are to be reported to the project interface (typically issues) or directly at the email address mentioned at the beginning of this longer document.

Please React!

If you have information more detailed or more recent than those presented in this document, if you noticed errors, neglects or points insufficiently discussed, drop us a line! (for that, follow the Support guidelines).

Ending Word

Have fun with Ceylan-Hull!

Hull