



Canine-Table
March 31, 2025

Contents

Git
Python
Str
Algorithms
Int
Struct
Networking
Content
Dialog
Cmd
TtyXVI
Pkgmgr

Ι Git

I Git

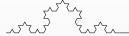
] The following functions enhance Git workflows by automating common social interactions and streamlining GitHub API integrations.

gh_social(): Automates GitHub social interactions. This function ensures proper authentication, manages "following" and "follower" relationships, and includes cleanup logic to remove unnecessary associations based on specified criteria.

III

ઌઙ૾ૢઌ

I GIT



II Python

II Python

] The following functions provide robust tools for managing Python virtual environments, focusing on seamless integration, modularity, and ease of use.

set_py_venv(): Manages the Python virtual environment. This function supports activating (-a), deactivating (-d), creating (-c), and utilizing (-s, -r) a Python virtual environment for a specified application.



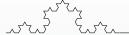
Ш Str

III Str

The following functions enable dynamic string manipulation, including random string generation, targeted searches and replacements, and case adjustments for versatile text handling.

- get_str_rand(): Generates a random string of specified length num, using character sets chars such as alphanumeric or others.
- get_str_locate(): Searches for occurrences of a string fnd within the input, optionally replacing it with **rpl**, separating content with **sep**, and supporting global or targeted searches.
- set str case(): Modifies the case of a string, converting it to uppercase, lowercase, or title case, based on the provided option (u, 1, or t).
- set_str_format(): Formats a string based on a specified format fmt, with optional separators sep, and alignment options like lft, rgt, or kp.
- add_str_append(): Appends a specified character char or string multiple times to reach a desired length **num**, optionally extending or modifying the input based on **ed**.
- **add_str_div()**: Creates a horizontal divider string of the length derived from terminal column size, using repeated characters like "-".
- get_str_parser(): Parses an input string D2 based on the format definition D1, extracting flags, key-value pairs, and unrecognized elements.
- get str print(): Formats arguments D for compatibility with AWK-based processing, wrapping non-options with quotes and preserving options as-is.

ೲಀೢಁೲ III STR V



IV Algorithms

IV Algorithms

] The following functions utilize efficient algorithms for sorting and processing data structures, with a focus on modularity and adaptability.

set_algor_qsort(): Implements a QuickSort algorithm to sort a list (**lst**) of elements, with options to reverse the sort order (**rvs**), apply a custom sorting mechanism (**meh**), and use specified delimiters (**sep** and **osep**).



Int

V Int

The following functions provide powerful computational tools for performing advanced numerical operations, including base-specific arithmetic, distribution, range adjustment, and mathematical constants handling.

- get_int_conv(): Converts a number num from its original base from to another base to, supporting optional signed number handling.
- get int bsubt(): Computes the difference of two numbers, minuend and subtrahend, in base from with a specified precision prec, supporting signed numbers.
- get_int_badd(): Computes the sum of two numbers, addend1 and addend2, in base from with a specified precision **prec**, supporting signed numbers.
- get_int_comp(): Computes the complement of a number num in the specified base, leveraging AWK utility functions for base-specific computations.
- get_int_abs(): Calculates the absolute value of num, ensuring the result is always a positive number, using AWK's utility functions.
- get int fact(): Computes the factorial of num, with an option to print intermediate steps if **prnt** is set to true.
- **get_int_fib()**: Computes the **num**-th Fibonacci number, optionally printing intermediate sums if **prnt** is set to true.
- get int round(): Rounds num according to the specified method rnd (e.g., ceiling or round), defaulting to truncation if no method is provided.
- get_int_gcd(): Computes the greatest common divisor (GCD) of two numbers, num1 and num2, using the Euclidean algorithm.
- **get_int_remainder()**: Computes the remainder of dividing **num1** by **num2**, ensuring both inputs are valid digits.
- get int lcd(): Calculates the least common denominator (LCD) of num1 and num2 using AWK's mathematical utilities.
- **get_int_tau()**: Returns the value of τ (the circle constant, $\tau = 2\pi$), optionally based on the input **num** for calculations or prints a default τ if no input is provided.
- \bigcirc get int pi(): Returns the value of π (pi constant), optionally using the input num for calculations or defaults to a general π value when no input is specified.

୶ୢୖୄ୶ଵ V INTVII



^ V Int

- **get_int_distribute()**: Distributes **num1** evenly across the range defined by **num2** and **num3**, ensuring all inputs are valid digits.
- **get_int_range()**: Adjusts **num1** to fit within the range defined by **num2** and **num3**, using modulus operations for precise computation.

ઌૺ૾ૢૺૺૺૺૺૺૺૺૺૺ

VIII V INT



VI Struct

VI Struct

The following functions provide a robust set of tools for managing structured data in shell scripts, covering retrieval, comparison, manipulation, and execution, with a focus on modularity and efficiency.

- **get_struct_ref()**: Retrieves the value of a variable by its name, allowing for dynamic access and reference in shell scripts.
- **get_struct_ref_append()**: Appends a value to the referenced variable, optionally inserting a separator before the new content, and returns the updated structure.
- **get_struct_compare()**: Compares two structures (input list and reference list), with options for case sensitivity, delimiters, and comparison modes (e.g., left, right, or intersection).
- **get_struct_list()**: Processes an input list with options for reversing, deduplication, or restructuring, while using specified separators for splitting and joining elements.
- new_struct_task(): Executes tasks iteratively on elements from a structured list, with configurable input, output, and error streams, as well as background execution control.
- **set_struct_noexpand()**: Prepares a variable for structured assignment by escaping special characters, ensuring its value is preserved in a non-expanded format.
- set_struct_opt(): Processes input and reference lists (inpt and reflst) using specified delimiters and options, matching input against reference values with configurable verbosity, case sensitivity, and length validation.

^ VI Struct

get_int_range(): Adjusts num1 to fit within the range defined by num2 and num3, using modulus operations for precise computation.



VII Networking

VII Networking

] The following functions provide utilities for working with Layer 2 and Layer 3 network addresses and types, including address generation, validation, and classification.

- __get_net_virt_types(): Provides a comma-separated list of supported virtual network device types for validation and reference.
- __chk_net_virt_type(): Validates the given virtual network type against the supported types returned by __get_net_virt_types().
- __get_net_dev_name(): Retrieves a comma-separated list of system network interface names using ip link show.
- __get_net_dev_alt(): Extracts alternative names (altnames) for network interfaces as a commaseparated list from ip link show.
- __get_net_dev_names(): Combines the outputs of __get_net_dev_alt() and __get_net_dev_name() into a unified comma-separated list of all interface names.
- __chk_net_dev_names(): Validates the given network interface name against the combined list of names produced by __get_net_dev_names().
- __get_net_dev_realname(): Retrieves the real name of a network device based on the given alias, using ip link show and validation via __chk_net_dev_names().
- <u>__is_net_dev()</u>: Checks whether the given device name corresponds to a valid network interface directory under /sys/class/net, using <u>__get_net_dev_realname()</u>.
- __get_net_obj(): Provides a comma-separated list of supported network object types, such as addresses, routes, and tunnels.
- __chk_net_obj(): Validates the given network object type against the list returned by __get_net_obj().
- __get_net_fam(): Provides a comma-separated list of supported network families, including inet, inet6, and others.
- __chk_net_fam(): Validates the given network family against the list produced by __get_net_fam().
- **get_net_dev_ipv6gen()**: Provides a comma-separated list of supported IPv6 address generation methods, such as eui64 and random.



^ VII Networking

- __chk_net_dev_ipv6gen(): Validates the given IPv6 address generation method against the list produced by __get_net_dev_ipv6gen().
- __get_net_dev_files(): Lists all files associated with a given network device by iterating over its directory in /sys/class/net.
- __get_net_dev_list(): Lists all network devices in /sys/class/net that are symbolic links, extracting their names.
- __get_net_dev_file(): Retrieves the content of a specific file for a given network device in /sys/class/net, returning 2 if the file does not exist.
- get_net_dev_alias(): Retrieves the alias of a specified network device by reading its ifalias file in /sys/class/net.
- **get_net_dev_index()**: Retrieves the index of a specified network device by reading its if index file.
- **get_net_dev_duplex()**: Retrieves the duplex mode of a specified network device by reading its duplex file.
- get_net_dev_state(): Retrieves the operational state of a specified network device by reading its operstate file.
- **get_net_dev_mtu()**: Retrieves the MTU (Maximum Transmission Unit) of a specified network device by reading its mtu file.
- get_net_dev_speed(): Retrieves the speed of a specified network device by reading its speed file
- **get_net_dev_l2()**: Determines the Layer 2 type of a network device (e.g., ether, loopback) using ip address show.
- **get_net_dev_qlen()**: Retrieves the transmit queue length (tx_queue_len) of a network device by reading its respective file.
- **get_net_dev_inet6()**: Extracts the IPv6 address details of a network device using ip address show.
- **get_net_dev_inet()**: Retrieves the first IPv4 or IPv6 address of a specified network device using ip address show.
- **get_net_dev_inet4()**: Extracts the first IPv4 address of a specified network device by filtering the output of ip address show.



^ VII Networking

- **get_net_dev_info()**: Constructs network device information options dynamically based on command-line arguments. Options include network family, object type, verbosity, and device type, enabling flexible data processing.
- get_net_l3_type(): Determines the Layer 3 type (inet or inet6) of a given address, using Awk to validate against IPv4 and IPv6 standards.
- **get_net_12_type()**: Identifies the Layer 2 type of a given address using Awk and returns its classification if valid.
- new_net_12_address(): Generates a new Layer 2 address, with options to customize universal/local, unicast/multicast, and formatting preferences. Uses Awk for validation and processing.
- get_net_eui64(): Generates an IPv6 EUI-64 address from a provided Layer 2 address. Offers customization for separators and casing (uppercase or lowercase), using Awk for validation and processing.

VIII Content

VIII Content

] The following functions streamline content operations, allowing for efficient path resolution, listing of files or directories, and modular scripting through dynamic file loading.

- get_content_trim(): Normalizes file or directory paths by removing redundant slashes, resolving relative paths (. /), and trimming trailing slashes.
- **get_content_leaf()**: Extracts the last component (leaf) of a file or directory path, such as the filename or the final directory in a hierarchy. Ensures accurate results by resolving the container path first.
- **get_content_container()**: Resolves the parent directory (container) of a given file or directory. Validates the path and returns the absolute directory path after normalization.
- **get_content_path()**: Resolves the full absolute path for a given file or directory, normalizing the input and accounting for symbolic links or relative paths.
- **get_content_list()**: Lists details of files or directories specified in the input. Differentiates between containers (directories) and leaf elements (files) for accurate display.
- **add_content_modules()**: Dynamically loads modular shell scripts from a specified source directory, excluding the **content-mod.sh** file itself. Ensures readability and prevents redundant loading.



IX Dialog

IX Dialog

] The following functions enable dynamic, customizable dialog interactions for time management, user confirmations, and various structured inputs.

- __get_dialog_factory(): Constructs dialog window definitions dynamically, based on specified options (-v, -m, -b, -p, -e). Supports multiple dialog types, argument parsing, and output formatting.
- __get_dialog_output(opt, fld, dft): Processes dialog options opt and fields fld to generate a formatted output string. Uses auxiliary Awk scripts for operations like trimming, splitting, and retrieving default values dft when specified. Escapes non-alphanumeric characters in field names.
- __get_dialog_selected(opt, fld): Identifies selected fields from dialog options opt, based on the mapping provided in fld. Uses auxiliary Awk scripts for splitting parameters and retrieving structured outputs. Outputs formatted variables with prefix indicators like GDF_SL_.
- __get_dialog_size(): Retrieves the current terminal dimensions, including the number of rows and columns, for adaptive dialog layouts.
- **get_dialog_explorer()**: Opens an interactive dialog window for file or directory exploration. Supports multiple dialog types (fselect, dselect, textbox, editbox, tailboxbg, tailbox) based on the provided input and conditions.
- **get_dialog_yn()**: Displays a yes/no or message box dialog. Dynamically determines the type (yesno or msgbox) based on the provided options and user interaction requirements.
- **get_dialog_cal()**: Launches a calendar dialog for date selection. Automatically formats the output to display the current date or allows custom configurations.
- **get_dialog_time()**: Launches a dialog window to manage time-based interactions. Supports **pause** for countdowns and **timebox** for specific time selection, with configurable defaults and user-defined inputs.
- **get_dialog_form()**: Generates a form-based dialog using dynamically formatted options and parameters. Processes user input via **__get_dialog_factory** and extracts output using **__get_dialog_output**. Returns the dialog's exit status.
- get_dialog_menu(): Constructs a menu-based dialog with options for various styles (e.g., radiolist, checklist). Utilizes __get_dialog_factory for string formatting and __get_dialog_selected for processing selections. Returns the dialog's exit status.



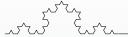
X Cmd

X Cmd

The following functions offer essential utilities for discovering and verifying the availability of commands, enhancing the portability and adaptability of shell scripts across different environments.

- get_cmd(): Iterates through a list of commands provided as arguments, checks their availability using command -v, and returns the first found command or exits if none are found.
- **get_cmd_pager()**: Searches for commonly used pager commands (less, more, and tee) by leveraging the **get_cmd()** function.
- get_cmd_awk(): Searches for AWK implementations (mawk, nawk, awk, gawk) using the get_cmd() function.
- **get_cmd_shell()**: Searches for available shell interpreters (dash, sh, bash, zsh, fish, and others) in the current environment.
- **get_cmd_editor()**: Locates command-line text editors (nvim, vim, gvim, vi) for editing files.
- get_cmd_tex_compiler(): Searches for LaTeX compilation utilities (latexmk, pdflatex, lualatex, xelatex).
- get_cmd_pdf_viewer(): Finds installed PDF viewers (zathura, mupdf, evince).
- get_cmd_pkgmgr(): Searches for package management tools (pacman, apt, dnf, brew, and others) in the system.

X CMD XV



XI Tty

XI Tty

The following functions provide utilities for managing and querying TTY properties, including property retrieval, structured formatting, and signal handling for enhanced user interaction.

- get_tty_prop_list(): Lists all TTY properties in key-value pairs, processing the output of stty -a for structured formatting.
- **get_tty_prop()**: Retrieves specific TTY properties based on provided keys (-k) or values (-v), enabling focused property queries.
- **set_tty_hault()**: Temporarily disables the cursor using setterm and traps signals to reenable it upon script exit or interruption.

ୢ୶ୄୖଵୄୢୄଊ

XVI XI TTY



XII Pkgmgr

XII Pkgmgr

] The following functions serve as wrappers for various package managers, offering a unified interface for common operations like updating, searching, installing, and managing software packages across different environments.

- get_pkgmgr(): A wrapper function for interacting with the defined package manager, supporting operations like updating (-u), querying (-q), searching (-s), installing (-i), removing (-r), and cleaning caches (-c).
- __set_pkgmgr(): Manages the execution of package manager commands by mapping user-specified options to the corresponding commands for the chosen package manager.
- __get_pkgmgr_*(): Defines package manager-specific command mappings for each supported package manager, such as pacman, apt, apk, brew, and others.