CATARINKA LUA LIBRARY

The information in this document applies to **version 6.8** of Syhunt Hybrid.

Catarinka is a multi-purpose set of Lua extensions developed to be used in Syhunt and the Sandcat Browser. Currently, this library extends Lua with over 60 functions and some useful classes. The project's goal is to make the development of Lua applications in Syhunt easier and to push the boundaries of the Lua language to do innovative things.

In Syhunt and Sandcat, Catarinka is available through the **ctk** global. For a list of available functions, see below. Each class has a "new" method that must be used for creating the object and a "release" method for freeing it.

FILE SYSTEM FUNCTIONS

FILE FUNCTIONS (FILE.*)

- **canopen** (filename): Returns true if a file can be opened. If the file is locked, returns false.
- **cleanname** (filename): Returns a filename with invalid characters stripped out.
- **copy** (source, dest): Copies a file to a new file.
- **delete** (filename): Deletes a file.
- **exec** (filename): Executes a file.
- **exechide** (filename): Executes a file in hidden state.
- **exists** (filename): Returns true if a file exists, and false otherwise.
- **getcontents** (filename): Returns the contents of a local file.
- **getdir** (filename): Gets the directory part of a filename.
- **getext** (filename): Gets the extension part of a filename.
- **getname** (filename): Gets the name and extension part of a filename.
- **getsize** (filename): Gets the size in bytes of a filename.
- **getver** (filename): Gets the version of a binary file.

DIRECTORY FUNCTIONS (DIR.*)

- **create** (dirname): Recursively creates a directory.
- **delete** (dirname): Deletes a directory and its subdirectories.
- **exists** (dirname): Returns true if a directory exists, and false otherwise.

- **getdirlist** (dirname): Returns the list of sub directories of a directory.
- **getfilelist** (dirname): Returns the list of files of a directory.
- **packtotar** (dirname, outfilename [, filemask]): Packs a directory to a TAR file.
- **unpackfromtar** (tarfilename, outdirname): Unpacks a TAR file to a directory.

STRING OPERATIONS (STRING.*)

- **after** (s, sub): Returns the portion of the string after a specific sub-string.
- **before** (s, sub): Returns the portion of the string before a specific sub-string.
- **between** (s, start, stop): Returns a string between 2 strings.
- **decrease** (s [,step]): Decreases the string characters.
- **gettoken** (s, delim, int): Returns what comes after a delimiter.
- **increase** (s [,step]): Increases the string characters.
- **lastchar** (s): Returns the last character of a string.
- **maxlen** (s, max [,addellipsis]): Cuts a string if it exceeds the max number of characters.
- **occur** (s, sub): Returns the count of the occurrence of a particular string or character.
- **random** (int): Returns a random string that is the length of your choosing.
- **replace** (s, find, rep): Replaces a string.
- **replacefirst** (s, find, rep): Replaces just the first occurrence of a sub string in a string.
- **stripquotes** (s): Returns a string with removed quotes.
- **stripblanklines** (s): Returns a multi-line string without any blank lines.
- **trim** (s): Returns a string without redundant whitespace.

STRING MATCHING FUNCTIONS

- **beginswith** (s, prefix): Checks if a string begins with a specific string.
- **endswith** (s, termination): Checks if a string ends with a string.
- **ishex** (s): Checks if a string is a hexadecimal representation.
- **isint** (s): Checks if a string is integer.
- **match** (s, pattern): Wildcard matching (* and ?).

These will return a boolean value.

STRING CLASSES

- **list**: Returns a stringlist object (see `classes.stringlist.md`).
- **loop**: Returns a stringloop object (see `classes.stringloop.md`).

REGULAR EXPRESSION FUNCTIONS (RE.*)

- **find** (s, regex): Regular expression finder. Returns a string.
- **match** (s, regex): Returns true if it matches a regular expression, false otherwise.
- **replace** (s, regex, rep): Finds a string using a regular expression and replaces it. Returns a new string.

WEB FUNCTIONS

HTML FUNCTIONS (HTML.*)

- **beautifycss** (css): Formats a CSS code.
- **beautifyjs** (js): Formats a JavaScript code.
- **escape** (s): Escapes HTML tag characters.
- **gettagcontents** (html, tag): Extracts the content of HTML tags.
- **parser**: Returns a HTML parser object (see `classes.htmlparser.md`).
- **striptags** (s): Removes tags from a string.
- **unescape** (s): Unescapes HTML tag characters.

URL FUNCTIONS (URL.*)

- **changepath** (url, newpath): Changes the path of an URL.
- **combine** (url, path): Combines a path to a URL.
- **crack** (url) : Returns the main components of an URL as a table.
  - fileext - filename extension (eg: .lp)
  - filename - filename (eg: index.lp)
  - host - host name (eg: www.lua.org)
  - path - location (eg: demo/index.lp)
  - port - port number (eg: 80)
  - proto - protocol (eg: https)

- **decode** (s): Decodes an URL.
- **encode** (s): Encodes an URL.
- **encodefull** (s): Full URL Encode.
- **fileurltofilename** (fileurl): Converts a file URL to a proper filename.
- **genfromhost** (hostname , port): Generates an URL from a hostname and a port.
- **getfileext** (url): Returns the extension from an URL filename.
- **getfilename** (url): Returns the URL filename.
- **gettiny** (url): Returns a tiny version of an URL (uses tinyurl.com).

JSON FUNCTIONS (JSON.*)

- **object**: Returns a JSON object (see `classes.jsonobject.md`).

HTTP FUNCTIONS (HTTP.*)

- **crackrequest** (headers): Returns the main components of the headers of a HTTP request as a table.
  - data - Request/POST data (if any)
  - method' - Request method (GET, POST, HEAD, etc...)
  - path - URL path

- **getheader** (headers, fieldname): Returns the value of a header field.

MISCELLANEOUS

NET FUNCTIONS (NET.*)

- **nametoip** (name): Converts host name to IP address.
- **iptoname** (ip): Converts IP address to host name.

BASE64 FUNCTIONS (BASE64.*)

- **encode** (s): Returns a string converted to a base64 string.
- **decode** (s): Converts a base64 string to a string.

CONVERSION FUNCTIONS (CONVERT.*)

- **commastrtostr** (s): Converts a comma string to a string.
- **strtoalphanum** (s): Converts a string to alphanumeric string.
- **strtocommastr** (s): Converts a string to a comma string.
- **strtohex** (s): Converts a string to a hexadecimal string.
- **hextoint** (s): Converts a hex string to integer.
- **hextostr** (s): Converts a hexadecimal string to string.

CRYPTO FUNCTIONS (CRYPTO.*)

- **md5** (s): Returns the MD5 hash of a given string.
- **sha1** (s): Returns the SHA-1 hash of a given string.

TASK FUNCTIONS (TASK.*)

- **isrunning** (exefilename): Returns true if a process is running, false otherwise.
- **kill** (exefilename): Closes a running process by its executable name.

UTILS (UTILS.*)

- **delay** (ms): Waits a specific number of milliseconds before proceeding.
- **getarg** (s or int , defaultvalue): Returns an argument passed to the executable as a string. Example: getarg('-pid') for returning the value of the argument provided as `-pid:somevalue` , getarg(1) for returning the first argument, getarg() for returning all arguments. If the argument value is empty, returns the default value.
- **hasarg** (s): Returns true if the argument has been provided, false otherwise.
- **clipboard_gettext** ( ): Returns the current Clipboard text (if any).
- **clipboard_settext** (s): Copies a text to the Clipboard.

HTML.PARSER

Parses a HTML document.

METHODS

- **clear** ( ): Clears the document text (deletes all lines).
- **getattrib** (attr): Returns the value of a tag attribute.
- **load** (html): Loads a document from a string.
- **parsing** ( ): Returns true while still parsing the document.
- **reset** ( ): Stops parsing the document, goes back to the beginning.
- **setattrib** (attr, s): Sets the value of a tag attribute.
- **stop** ( ): Stops parsing the document.

PROPERTIES

name	return type	description
pos	integer	Returns the current position.
tagcontent	string	Returns the content of the current tag.
tagline	integer	Returns the line of the current tag.
tagname	string	Returns the name of the current tag.
tagpos	integer	Returns the position of the current tag.

EXAMPLE - USING HTML.PARSER

```
local s = require "Catarinka"
local html = [[
<html>
<a href="http://www.lua.org">Lua</a>
</html>
]]
local p = s.html.parser:new()
p:load(html)
while p:parsing() do
 print(p.tagname)
 print(p:getattrib('href'))
end
p:release()
```

JSON.OBJECT

Stores and manipulates a JSON object.

METHODS

- **getjson** ( ): Returns the JSON object as a string. Alternatively, you can use the Lua tostring() function.
- **load** (s): Loads a JSON object from a string.
- **loadfromfile** (filename): Loads a JSON object from a file.
- **savetofile** (filename): Saves the JSON object to a file.

PROPERTIES

- **akey**: Gets or sets the value of a key.

USAGE EXAMPLE

```
local s = require "Catarinka"
local j = s.json.object:new()
j['name.first'] = 'Carla'
j['name.last'] = 'Coe'
j.year = 2013
print(tostring(j))
--[[
this will print:
{
 "year": 2013,
 "name": {
  "first": "Carla",
  "last": "Coe"
 }
}
]]
j:release()
```

STRING.LIST

Stores and manipulates a list of strings.

METHODS

- **add** (s): Adds a string.
- **clear** ( ): Clears the list, deletes all strings.
- **delete** (number): Deletes a line by its number.
- **get** (number): Gets a line by its number.
- **indexof** (s): Returns the line of a string. If not found, returns -1.
- **insert** (pos, s): Inserts a string to a given position.
- **loadfromfile** (filename): Loads the list from a file.
- **savetofile** (filename): Saves the list to a file.
- **sort** ( ): Sorts the list alphabetically

PROPERTIES

- **count**: Returns the number of strings in the list.
- **text**: Gets or sets the list of strings.

STRING.LOOP

Loops through a string list.

METHODS

- **add** (s): Adds a string.
- **clear** ( ): Clears the list (deletes all lines).
- **curdelete** ( ): Deletes the current line.
- **get** (number): Gets a line by its number.
- **indexof** (s): Returns the line of a string. If not found, returns -1.
- **load** (s): Loads a string list from a string.
- **loadfromfile** (filename): Loads the list from a file.
- **parsing** ( ): Returns true while still parsing the list.
- **reset** ( ): Stops parsing the list, goes back to the beginning.
- **reverse** ( ): Reverses the string list order.
- **savetofile** (filename): Saves the list to a file.
- **stop** ( ): Stops parsing the list.

PROPERTIES

name	return type	description
commatext	string	Gets (or sets) the list from a comma string.
count	integer	Returns the number of strings in the list.
curindex	integer	Returns the index of the current string.
current	string	Gets (or sets) the current string.
text	string	Gets or sets the list of strings.

USAGE EXAMPLE

```lua
local s = require "Catarinka"
local list = [[
Turkey
Russia
Azerbaijan
United Kingdom
Montenegro
]]
local p = s.string.loop:new()
p:load(list)
while p:parsing() do
 print(p.current)
end
p:release()
```