luaharfbuzz Documentation

Module `harfbuzz`

Lua bindings to Harfbuzz.

- Wiki
- Source on Github
- API Coverage Status

Info:

- **Copyright**: 2016
- **License**: MIT
- **Author**: Deepak Jois <deepak.jois@gmail.com>

Functions

version ()	Wraps `hb_version`.
shape (font, buffer[, options])	Wraps `hb_shape`.

Class Blob

Blob.new (data)	Wraps `hb_blob_create`.
Blob.new_from_file (filename)	Wraps `hb_blob_create_from_file`.
Blob:get_length ()	Wraps `hb_blob_get_length`.
Blob:get_data ()	Wraps `hb_blob_get_data`.

Class Face

Face.new_from_blob (blob[, font_index=0])	Wraps `hb_face_create`.
Face.new (file[, font_index=0])	Create a new `Face` from a file.
Face:collect_unicodes ()	Wraps `hb_face_collect_unicodes`.
Face:get_glyph_count ()	Wraps `hb_face_get_glyph_count`.
Face:get_table (tag)	Wraps `hb_face_reference_table`.
Face:get_table_tags ()	Wraps `hb_face_get_table_tags`.
Face:get_upem ()	Wraps `hb_face_get_upem`.
Face:ot_color_has_palettes ()	Wraps `hb_ot_color_has_palettes`.
Face:ot_color_palette_get_count ()	Wraps `hb_ot_color_palette_get_count`.
Face:ot_color_palette_get_colors ()	Wraps `hb_ot_color_palette_get_colors`.
Face:ot_color_has_layers ()	Wraps `hb_ot_color_has_layers`.
Face:ot_color_glyph_get_layers ()	Wraps `hb_ot_color_glyph_get_layers`.
Face:ot_color_has_png ()	Wraps `hb_ot_color_has_png`.
Face:ot_layout_get_script_tags ()	Wraps `hb_ot_layout_table_get_script_tags`.
Face:ot_layout_get_language_tags ()	Wraps `hb_ot_layout_script_get_language_tags`.
Face:ot_layout_get_feature_tags ()	Wraps `hb_ot_layout_language_get_feature_tags`.

Face:ot_layout_find_script ()	Wraps `hb_ot_layout_table_find_script`.
Face:ot_layout_find_language ()	Wraps `hb_ot_layout_script_find_language`.
Face:ot_layout_find_feature ()	Wraps `hb_ot_layout_language_find_feature`.

Class Font

Font.new (face)	Wraps `hb_font_create`, and sets up some defaults for scale and shaping functions.
Font:get_scale ()	Wraps `hb_font_get_scale`.
Font:set_scale (x_scale, y_scale)	Wraps `hb_font_set_scale`.
Font:get_h_extents ()	Wraps `hb_font_get_h_extents`.
Font:get_v_extents ()	Wraps `hb_font_get_v_extents`.
Font:get_glyph_extents (glyph)	Wraps `hb_font_get_glyph_extents`.
Font:get_glyph_name (glyph)	Wraps `hb_font_get_glyph_name`.
Font:get_glyph_from_name (name)	Wraps `hb_font_get_glyph_from_name`.
Font:get_glyph_h_advance (glyph)	Wraps `hb_font_get_glyph_h_advance`.
Font:get_glyph_v_advance (glyph)	Wraps `hb_font_get_glyph_v_advance`.
Font:get_nominal_glyph (codepoint.)	Wraps `hb_font_get_nominal_glyph`.
Font:ot_color_glyph_get_png ()	Wraps `hb_ot_color_glyph_get_png`.

Class Buffer

Buffer.new ()	Wraps `hb_buffer_create`.
Buffer:add_utf8 (text[, item_offset=0[, item_length=-1]])	Wraps `hb_buffer_add_utf8`.
Buffer:add_codepoints (text[, item_offset=0[, item_length=-1]])	Wraps `hb_buffer_add_codepoints`.
Buffer:set_direction (dir)	Wraps `hb_buffer_set_direction`.
Buffer:get_direction ()	Wraps `hb_buffer_get_direction`.
Buffer:set_script (script)	Wraps `hb_buffer_set_script`.
Buffer:get_script ()	Wraps `hb_buffer_get_script`.
Buffer:set_language (lang)	Wraps `hb_buffer_set_language`.
Buffer:get_language ()	Wraps `hb_buffer_get_language`.
Buffer:reverse ()	Wraps `hb_buffer_reverse`.
Buffer:get_length ()	Wraps `hb_buffer_get_length`.
Buffer:get_cluster_level ()	Wraps `hb_buffer_get_cluster_level`.
Buffer:set_cluster_level (level)	Wraps `hb_buffer_set_cluster_level`.
Buffer:guess_segment_properties ()	Wraps `hb_buffer_guess_segment_properties`.
Buffer:get_glyphs ()	Helper method to get shaped glyph data.

Cluster Levels

Buffer.CLUSTER_LEVEL_MONOTONE_GRAPHEMES	Wraps `HB_BUFFER_CLUSTER_LEVEL_MONOTONE_GRAPHEMES`.
Buffer.CLUSTER_LEVEL_MONOTONE_CHARACTERS	Wraps `HB_BUFFER_CLUSTER_LEVEL_MONOTONE_CHARACTERS`.
Buffer.CLUSTER_LEVEL_CHARACTERS	Wraps `HB_BUFFER_CLUSTER_LEVEL_CHARACTERS`.
Buffer.CLUSTER_LEVEL_DEFAULT	Wraps `HB_BUFFER_CLUSTER_LEVEL_DEFAULT`.

Buffer.GLYPH_FLAG_UNSAFE_TO_BREAK	Wraps `HB_GLYPH_FLAG_UNSAFE_TO_BREAK`.
Buffer.GLYPH_FLAG_DEFINED	Wraps `HB_GLYPH_FLAG_DEFINED`.

Class Feature

Feature.new (feature_string)	Wraps `hb_feature_from_string`
Feature:__tostring ()	Wraps `hb_feature_to_string`.

Class Tag

Tag.new (string)	Wraps `hb_tag_from_string`.
Tag:__tostring ()	Wraps `hb_tag_to_string`.
Tag:__eq ()	Enables equality comparisions with == between two tags.

Class Script

Script.new (script)	Wraps `hb_script_from_string`.
Script.from_iso15924_tag (tag)	Wraps `hb_script_from_iso15924_tag`
Script:to_iso15924_tag ()	Wraps `hb_script_to_iso15924_tag`.
Script:__tostring ()	Enable nice output with `tostring(…)`
Script:__eq ()	Enables equality comparisions with == between two scripts.

Predefined Script Codes

Script.COMMON	Wraps `HB_SCRIPT_COMMON`.
Script.INHERITED	Wraps `HB_SCRIPT_INHERITED`.
Script.UNKNOWN	Wraps `HB_SCRIPT_UNKNOWN`.
Script.INVALID	Wraps `HB_SCRIPT_INVALID`.

Class Direction

Direction.new (dir)	Wraps `hb_direction_from_string`.
Direction:__tostring ()	Wraps `hb_direction_to_string`.
Direction:__eq ()	Enables equality comparisions with == between two directions.
Direction:is_valid ()	Wraps `HB_DIRECTION_IS_VALID`.
Direction:is_horizontal ()	Wraps `HB_DIRECTION_IS_HORIZONTAL`.
Direction:is_vertical ()	Wraps `HB_DIRECTION_IS_VERTICAL`.
Direction:is_forward ()	Wraps `HB_DIRECTION_IS_FORWARD`.
Direction:is_backward ()	Wraps `HB_DIRECTION_IS_BACKWARD`.

Predefined directions

Direction.LTR	Wraps `HB_DIRECTION_LTR`.
Direction.RTL	Wraps `HB_DIRECTION_RTL`.
Direction.TTB	Wraps `HB_DIRECTION_TTB`.
Direction.BTT	Wraps `HB_DIRECTION_LTR`.

Class Language

Language.new (lang)	Wraps `hb_language_from_string`.
Language:__tostring ()	Wraps `hb_language_to_string`.
Language:__eq ()	Enables equality comparisions with == between two languages.

Predefined languages

Language.INVALID	Wraps `HB_LANGUAGE_INVALID`.

Class Set

Set:add (codepoint)	Wraps `hb_set_add`.

Unicode functions

unicode.script (char)	Wraps `hb_unicode_script`

Predefined Name IDs

ot.NAME_ID_COPYRIGHT	Wraps `HB_OT_NAME_ID_COPYRIGHT`
ot.NAME_ID_FONT_FAMILY	Wraps `HB_OT_NAME_ID_FONT_FAMILY`
ot.NAME_ID_FONT_SUBFAMILY	Wraps `HB_OT_NAME_ID_FONT_SUBFAMILY`
ot.NAME_ID_UNIQUE_ID	Wraps `HB_OT_NAME_ID_UNIQUE_ID`
ot.NAME_ID_FULL_NAME	Wraps `HB_OT_NAME_ID_FULL_NAME`
ot.NAME_ID_VERSION_STRING	Wraps `HB_OT_NAME_ID_VERSION_STRING`
ot.NAME_ID_POSTSCRIPT_NAME	Wraps `HB_OT_NAME_ID_POSTSCRIPT_NAME`
ot.NAME_ID_TRADEMARK	Wraps `HB_OT_NAME_ID_TRADEMARK`
ot.NAME_ID_MANUFACTURER	Wraps `HB_OT_NAME_ID_MANUFACTURER`
ot.NAME_ID_DESIGNER	Wraps `HB_OT_NAME_ID_DESIGNER`
ot.NAME_ID_DESCRIPTION	Wraps `HB_OT_NAME_ID_DESCRIPTION`
ot.NAME_ID_VENDOR_URL	Wraps `HB_OT_NAME_ID_VENDOR_URL`
ot.NAME_ID_DESIGNER_URL	Wraps `HB_OT_NAME_ID_DESIGNER_URL`
ot.NAME_ID_LICENSE	Wraps `HB_OT_NAME_ID_LICENSE`
ot.NAME_ID_LICENSE_URL	Wraps `HB_OT_NAME_ID_LICENSE_URL`
ot.NAME_ID_TYPOGRAPHIC_FAMILY	Wraps `HB_OT_NAME_ID_TYPOGRAPHIC_FAMILY`
ot.NAME_ID_TYPOGRAPHIC_SUBFAMILY	Wraps `HB_OT_NAME_ID_TYPOGRAPHIC_SUBFAMILY`
ot.NAME_ID_MAC_FULL_NAME	Wraps `HB_OT_NAME_ID_MAC_FULL_NAME`
ot.NAME_ID_SAMPLE_TEXT	Wraps `HB_OT_NAME_ID_SAMPLE_TEXT`
ot.NAME_ID_CID_FINDFONT_NAME	Wraps `HB_OT_NAME_ID_CID_FINDFONT_NAME`
ot.NAME_ID_WWS_FAMILY	Wraps `HB_OT_NAME_ID_WWS_FAMILY`
ot.NAME_ID_WWS_SUBFAMILY	Wraps `HB_OT_NAME_ID_WWS_SUBFAMILY`
ot.NAME_ID_LIGHT_BACKGROUND	Wraps `HB_OT_NAME_ID_LIGHT_BACKGROUND`
ot.NAME_ID_DARK_BACKGROUND	Wraps `HB_OT_NAME_ID_DARK_BACKGROUND`
ot.NAME_ID_VARIATIONS_PS_PREFIX	Wraps `HB_OT_NAME_ID_VARIATIONS_PS_PREFIX`
ot.NAME_ID_INVALID	Wraps `HB_OT_NAME_ID_INVALID`

ot.LAYOUT_NO_SCRIPT_INDEX	Wraps `HB_OT_LAYOUT_NO_SCRIPT_INDEX`
ot.LAYOUT_NO_FEATURE_INDEX	Wraps `HB_OT_LAYOUT_NO_FEATURE_INDEX`
ot.LAYOUT_DEFAULT_LANGUAGE_INDEX	Wraps `HB_OT_LAYOUT_DEFAULT_LANGUAGE_INDEX`
ot.LAYOUT_NO_VARIATIONS_INDEX	Wraps `HB_OT_LAYOUT_NO_VARIATIONS_INDEX`

version ()

Wraps `hb_version`.

shape (font, buffer[, options])

Wraps `hb_shape`.

Parameters:

- font `Font` to use for shaping
- buffer `Buffer` to shape
- options

  table containing one or more supported options:

  - `direction`: A `Direction` object representing the object.
  - `script`: A `Script` object representing the script.
  - `language`: A `Language` object representing the language.
  - `features`: features to enable, specified as either of the following.
    - comma-separated list of features. See feature string syntax reference
    - table of `Feature` objects

  (*optional*)

Lua wrapper for `hb_blob_t` type

Blob.new (data)

Wraps `hb_blob_create`. Initializes a new `hb_blob_t`.

Parameters:

- data lua string containing binary or character data.

Blob.new_from_file (filename)

Wraps `hb_blob_create_from_file`. Initializes a new `hb_blob_t`.

Parameters:

- filename lua string.

Blob:get_length ()

Wraps `hb_blob_get_length`.

Blob:get_data ()

Wraps `hb_blob_get_data`.

Lua wrapper for `hb_face_t` type

Face.new_from_blob (blob[, font_index=0])

Wraps `hb_face_create`. Initializes a new `hb_face_t` from a `Blob` object.

Parameters:

- blob `Blob` to read the font from.
- font_index index of font to read. (*default* 0)

Face.new (file[, font_index=0])

Create a new `Face` from a file. Makes a call to `Face:new_from_blob` after creating a `Blob` from the file contents.

Parameters:

- file path to font file.
- font_index index of font to read. (*default* 0)

Face:collect_unicodes ()

Wraps `hb_face_collect_unicodes`.

Returns:

table of codepoints supported by the face.

Face:get_glyph_count ()

Wraps `hb_face_get_glyph_count`.

Face:get_table (tag)

Wraps `hb_face_reference_table`.

Parameters:

- tag `Tag` object of the table.

Returns:

`Blob` object for the face table of `tag`.

Face:get_table_tags ()

Wraps `hb_face_get_table_tags`.

Returns:

table of `Tag`s representing face table tags.

Face:get_upem ()

Wraps `hb_face_get_upem`.

Face:ot_color_has_palettes ()

Wraps `hb_ot_color_has_palettes`.

Face:ot_color_palette_get_count ()

Wraps `hb_ot_color_palette_get_count`.

Face:ot_color_palette_get_colors ()

Wraps `hb_ot_color_palette_get_colors`.

Face:ot_color_has_layers ()

Wraps `hb_ot_color_has_layers`.

Face:ot_color_glyph_get_layers ()

Wraps `hb_ot_color_glyph_get_layers`.

Face:ot_color_has_png ()

Wraps `hb_ot_color_has_png`.

Face:ot_layout_get_script_tags ()

Wraps `hb_ot_layout_table_get_script_tags`.

Face:ot_layout_get_language_tags ()

Wraps `hb_ot_layout_script_get_language_tags`.

Face:ot_layout_get_feature_tags ()

Wraps `hb_ot_layout_language_get_feature_tags`.

Face:ot_layout_find_script ()

Wraps `hb_ot_layout_table_find_script`.

Face:ot_layout_find_language ()

Wraps `hb_ot_layout_script_find_language`.

Face:ot_layout_find_feature ()

Wraps `hb_ot_layout_language_find_feature`.

Lua wrapper for `hb_font_t` type

Font.new (face)

Wraps `hb_font_create`, and sets up some defaults for scale and shaping functions. Initializes a new `hb_font_t` from a `Face` object. Sets the default scale to the face's upem value, and sets the font shaping functions by calling `hb_ot_font_set_funcs` on it.

Parameters:

- face `Face` object.

Font:get_scale ()

Wraps `hb_font_get_scale`.

Returns:

two values for the x-scale and y-scale of the font.

Font:set_scale (x_scale, y_scale)

Wraps `hb_font_set_scale`.

Parameters:

- x_scale desired x-scale of font.
- y_scale desired y-scale of font.

Font:get_h_extents ()

Wraps `hb_font_get_h_extents`.

Returns:

font extents table for horizontal direction, contains the following or `nil` if HarfBuzz fails to load font extents:

- ○ `ascender`: typographic ascender.
- ○ `descender`: typographic descender.
- ○ `line_gap`: line spacing gap.

Font:get_v_extents ()

Wraps `hb_font_get_v_extents`.

Returns:

font extents table for vertical direction, similar to `Font:get_h_extents`, or `nil` if HarfBuzz fails to load font extents:

Font:get_glyph_extents (glyph)

Wraps `hb_font_get_glyph_extents`.

Parameters:

- glyph index inside the font.

Returns:

extents table contains the following or `nil` if HarfBuzz fails to load glyph extents:

- ○ `x_bearing`: left side of glyph from origin.
- ○ `y_bearing`: top side of glyph from origin.
- ○ `width`: distance from left to right side.
- ○ `height`: distance from top to bottom side.

Font:get_glyph_name (glyph)

Wraps `hb_font_get_glyph_name`.

Parameters:

- glyph index inside the font.

Returns:

name of the glyph or nil.

Font:get_glyph_from_name (name)

Wraps `hb_font_get_glyph_from_name`.

Parameters:

- name of the glyph.

Returns:

glyph index inside the font or nil.

Font:get_glyph_h_advance (glyph)

Wraps `hb_font_get_glyph_h_advance`.

Parameters:

- glyph index inside the font.

Returns:

advance glyph advance of the glyph in horizontal direction.

Font:get_glyph_v_advance (glyph)

Wraps `hb_font_get_glyph_v_advance`.

Parameters:

- glyph index inside the font.

Returns:

advance glyph advance of the glyph in vertical direction.

Font:get_nominal_glyph (codepoint.)

Wraps `hb_font_get_nominal_glyph`.

Parameters:

- codepoint.

Returns:

glyph index or `nil` if `codepoint` is not supported by the font.

Font:ot_color_glyph_get_png ()

Wraps `hb_ot_color_glyph_get_png`.

Lua wrapper for `hb_buffer_t` type.

Buffer.new ()

Wraps `hb_buffer_create`.

Buffer:add_utf8 (text[, item_offset=0[, item_length=-1]])

Wraps `hb_buffer_add_utf8`.

Parameters:

- text UTF8 encoded string.
- item_offset 0-indexed offset in `text`, from where to start adding. (*default* 0)
- item_length length to add from `item_offset`. `-1` adds till end of `text`. (*default* -1)

Buffer:add_codepoints (text[, item_offset=0[, item_length=-1]])

Wraps `hb_buffer_add_codepoints`.

Parameters:

- text table with codepoints as lua numbers.
- item_offset 0-indexed offset in `text`, from where to start adding. (*default* 0)
- item_length length to add from `item_offset`. `-1` adds till end of `text`. (*default* -1)

Buffer:set_direction (dir)

Wraps `hb_buffer_set_direction`.

Parameters:

- dir A `Direction` object.

Buffer:get_direction ()

Wraps `hb_buffer_get_direction`.

Returns:

A `Direction` object.

Buffer:set_script (script)

Wraps `hb_buffer_set_script`.

Parameters:

- script A `Script` object.

Buffer:get_script ()

Wraps `hb_buffer_get_script`.

Returns:

A `Script` object.

Buffer:set_language (lang)

Wraps `hb_buffer_set_language`.

Parameters:

- lang A `Language` object

Buffer:get_language ()

Wraps `hb_buffer_get_language`.

Returns:

A `Language` object

Buffer:reverse ()

Wraps `hb_buffer_reverse`.

Buffer:get_length ()

Wraps `hb_buffer_get_length`.

Buffer:get_cluster_level ()

Wraps `hb_buffer_get_cluster_level`.

Returns:

see Cluster Levels

Buffer:set_cluster_level (level)

Wraps `hb_buffer_set_cluster_level`.

Parameters:

- level see Cluster Levels

Buffer:guess_segment_properties ()

Wraps `hb_buffer_guess_segment_properties`.

Buffer:get_glyphs ()

Helper method to get shaped glyph data. Calls `hb_buffer_get_glyph_infos`, `hb_buffer_get_glyph_positions` and `hb_glyph_info_get_glyph_flags`, and assembles the data into a Lua table.

Returns:

table containing data for each glyph, in a nested table. Each nested table contains the following:

- `x_advance`: horizontal advance.
- `y_advance`: vertical advance.
- `x_offset`: horizontal displacement.
- `y_offset`: vertical displacement.
- `cluster`: glyph cluster index within input.
- `codepoint`: glyph index inside the font *(this field name is a bit misleading, but that's what Harfbuzz uses)*.
- `flags`: glyph flags

See Harfbuzz docs for more details about what each of these levels mean.

Buffer.CLUSTER_LEVEL_MONOTONE_GRAPHEMES

Wraps `HB_BUFFER_CLUSTER_LEVEL_MONOTONE_GRAPHEMES`.

Buffer.CLUSTER_LEVEL_MONOTONE_CHARACTERS

Wraps `HB_BUFFER_CLUSTER_LEVEL_MONOTONE_CHARACTERS`.

Buffer.CLUSTER_LEVEL_CHARACTERS

Wraps `HB_BUFFER_CLUSTER_LEVEL_CHARACTERS`.

Buffer.CLUSTER_LEVEL_DEFAULT

Wraps `HB_BUFFER_CLUSTER_LEVEL_DEFAULT`.

Buffer.GLYPH_FLAG_UNSAFE_TO_BREAK

Wraps `HB_GLYPH_FLAG_UNSAFE_TO_BREAK`.

Buffer.GLYPH_FLAG_DEFINED

Wraps `HB_GLYPH_FLAG_DEFINED`.

Lua wrapper for `hb_feature_t` type

Feature.new (feature_string)

Wraps `hb_feature_from_string`

Parameters:

- feature_string See feature string syntax reference

Feature:__tostring ()

Wraps `hb_feature_to_string`. Enables nice output with `tostring(…)`.

Lua wrapper for `hb_tag_t` type.

Tag.new (string)

Wraps `hb_tag_from_string`.

Parameters:

- string to be converted to a `Tag` object.

Returns:

a `Tag` object.

Tag:__tostring ()

Wraps `hb_tag_to_string`. Enable nice output with `tostring(…)`.

Returns:

Returns a string representation for the tag object.

Tag:__eq ()

Enables equality comparisions with `==` between two tags.

Returns:

`true` or `false` depending on whether the two tags are equal.

Lua wrapper for `hb_script_t` type.

Script.new (script)

Wraps `hb_script_from_string`.

Parameters:

- script 4-letter script code according to the ISO 15924 standard.

Returns:

a `Script` object.

Script.from_iso15924_tag (tag)

Wraps `hb_script_from_iso15924_tag`

Parameters:

- tag a `Tag` object representing a ISO 15924 script.

Script:to_iso15924_tag ()

Wraps `hb_script_to_iso15924_tag`.

Returns:

a `Tag` object representing the script.

Script:__tostring ()

Enable nice output with `tostring(…)`

Returns:

Returns a 4-letter ISO 15924 script code for the script object.

Script:__eq ()

Enables equality comparisions with == between two scripts.

Returns:

`true` or `false` depending on whether the two scripts are equal.

Predefined directions that correspond to their original definitions in Harfbuzz.

Script.COMMON

Wraps `HB_SCRIPT_COMMON`.

Script.INHERITED

Wraps `HB_SCRIPT_INHERITED`.

Script.UNKNOWN

Wraps `HB_SCRIPT_UNKNOWN`.

Script.INVALID

Wraps `HB_SCRIPT_INVALID`.

Lua wrapper for `hb_direction_t` type.

Direction.new (dir)

Wraps `hb_direction_from_string`.

Parameters:

- dir can be one of `ltr`, `rtl`, `ttb`, `btt` or `invalid`.

Returns:

a `Direction` object.

Direction:__tostring ()

Wraps `hb_direction_to_string`. Enable nice output with `tostring(…)`.

Returns:

Returns a string representation for direction.

Direction:__eq ()

Enables equality comparisions with == between two directions.

Returns:

`true` or `false` depending on whether the two tags are equal.

Direction:is_valid ()

Wraps `HB_DIRECTION_IS_VALID`.

Returns:

a boolean value

Direction:is_horizontal ()

Wraps `HB_DIRECTION_IS_HORIZONTAL`.

Returns:

a boolean value

Direction:is_vertical ()

Wraps `HB_DIRECTION_IS_VERTICAL`.

Returns:

a boolean value

Direction:is_forward ()

Wraps `HB_DIRECTION_IS_FORWARD`.

Returns:

a boolean value

Direction:is_backward ()

Wraps `HB_DIRECTION_IS_BACKWARD`.

Returns:

a boolean value

Predefined directions that correspond to their original definitions in Harfbuzz.

Direction.LTR

Wraps `HB_DIRECTION_LTR`.

Direction.RTL

Wraps `HB_DIRECTION_RTL`.

Direction.TTB

Wraps `HB_DIRECTION_TTB`.

Direction.BTT

Wraps `HB_DIRECTION_LTR`.

Lua wrapper for `hb_language_t` type.

Language.new (lang)

Wraps `hb_language_from_string`.

Parameters:

- lang three-letter language tag to be converted to a `Language` object.

Returns:

a `Language` object.

Language:__tostring ()

Wraps `hb_language_to_string`. Enable nice output with `tostring(…)`.

Returns:

Returns a string representation for the language object.

Language:__eq ()

Enables equality comparisions with `==` between two languages.

Returns:

`true` or `false` depending on whether the two languages are equal.

Predefined languages that correspond to their original definitions in Harfbuzz.

Language.INVALID
Wraps `HB_LANGUAGE_INVALID`.

Lua wrapper for `hb_set_t` type.

Set objects represent a mathematical set of integer values.

Set:add (codepoint)
Wraps `hb_set_add`.

Adds a codepoint to the set.

Parameters:

- codepoint The element to add.

unicode.script (char)
Wraps `hb_unicode_script`

Parameters:

- char Unicode codepoint

Returns:

a `Script` object.

Predefined OpenType 'name' table name identifier.

ot.NAME_ID_COPYRIGHT
Wraps `HB_OT_NAME_ID_COPYRIGHT`
ot.NAME_ID_FONT_FAMILY
Wraps `HB_OT_NAME_ID_FONT_FAMILY`
ot.NAME_ID_FONT_SUBFAMILY
Wraps `HB_OT_NAME_ID_FONT_SUBFAMILY`
ot.NAME_ID_UNIQUE_ID
Wraps `HB_OT_NAME_ID_UNIQUE_ID`
ot.NAME_ID_FULL_NAME
Wraps `HB_OT_NAME_ID_FULL_NAME`
ot.NAME_ID_VERSION_STRING
Wraps `HB_OT_NAME_ID_VERSION_STRING`
ot.NAME_ID_POSTSCRIPT_NAME
Wraps `HB_OT_NAME_ID_POSTSCRIPT_NAME`
ot.NAME_ID_TRADEMARK
Wraps `HB_OT_NAME_ID_TRADEMARK`
ot.NAME_ID_MANUFACTURER
Wraps `HB_OT_NAME_ID_MANUFACTURER`
ot.NAME_ID_DESIGNER
Wraps `HB_OT_NAME_ID_DESIGNER`
ot.NAME_ID_DESCRIPTION
Wraps `HB_OT_NAME_ID_DESCRIPTION`
ot.NAME_ID_VENDOR_URL
Wraps `HB_OT_NAME_ID_VENDOR_URL`
ot.NAME_ID_DESIGNER_URL
Wraps `HB_OT_NAME_ID_DESIGNER_URL`

ot.NAME_ID_LICENSE

Wraps `HB_OT_NAME_ID_LICENSE`

ot.NAME_ID_LICENSE_URL

Wraps `HB_OT_NAME_ID_LICENSE_URL`

ot.NAME_ID_TYPOGRAPHIC_FAMILY

Wraps `HB_OT_NAME_ID_TYPOGRAPHIC_FAMILY`

ot.NAME_ID_TYPOGRAPHIC_SUBFAMILY

Wraps `HB_OT_NAME_ID_TYPOGRAPHIC_SUBFAMILY`

ot.NAME_ID_MAC_FULL_NAME

Wraps `HB_OT_NAME_ID_MAC_FULL_NAME`

ot.NAME_ID_SAMPLE_TEXT

Wraps `HB_OT_NAME_ID_SAMPLE_TEXT`

ot.NAME_ID_CID_FINDFONT_NAME

Wraps `HB_OT_NAME_ID_CID_FINDFONT_NAME`

ot.NAME_ID_WWS_FAMILY

Wraps `HB_OT_NAME_ID_WWS_FAMILY`

ot.NAME_ID_WWS_SUBFAMILY

Wraps `HB_OT_NAME_ID_WWS_SUBFAMILY`

ot.NAME_ID_LIGHT_BACKGROUND

Wraps `HB_OT_NAME_ID_LIGHT_BACKGROUND`

ot.NAME_ID_DARK_BACKGROUND

Wraps `HB_OT_NAME_ID_DARK_BACKGROUND`

ot.NAME_ID_VARIATIONS_PS_PREFIX

Wraps `HB_OT_NAME_ID_VARIATIONS_PS_PREFIX`

ot.NAME_ID_INVALID

Wraps `HB_OT_NAME_ID_INVALID`

ot.LAYOUT_NO_SCRIPT_INDEX

Wraps `HB_OT_LAYOUT_NO_SCRIPT_INDEX`

ot.LAYOUT_NO_FEATURE_INDEX

Wraps `HB_OT_LAYOUT_NO_FEATURE_INDEX`

ot.LAYOUT_DEFAULT_LANGUAGE_INDEX

Wraps `HB_OT_LAYOUT_DEFAULT_LANGUAGE_INDEX`

ot.LAYOUT_NO_VARIATIONS_INDEX

Wraps `HB_OT_LAYOUT_NO_VARIATIONS_INDEX`