freetypy Documentation

Release 0+untagged.103.gab59162

Michael Droettboom

Sep 20, 2017

Contents

`freetypy` is a general-purpose Python wrapper to FreeType with a goal to be fast and flexible as possible.

It is open source under the BSD license.

It is developed as part of the matplotlib project, though it is an independent library that intends to be of general use outside of matplotlib as well.

CHAPTER 1

Introduction to freetypy

Since freetypy closely follows FreeType's native C API, the FreeType documentation serves as a good introduction to freetypy as well.

Basic usage

The convention for importing `freetypy` is:

```
>>> import freetypy as ft
```

First, load a font from a file. `freetypy` also supports loading a font from a Python file-like object:

```
>>> face = ft.Face("Vera.ttf")
```

Then the size (in points) must be selected:

```
>>> face.set_char_size(48)
```

Once that setup is performed, a glyph may be loaded. We pass `load_flags=ft.LOAD.RENDER` to tell FreeType to load the glyph outline and then subsequently render it to a bitmap:

```
>>> glyph = face.load_char_unicode(u"S", load_flags=ft.LOAD.RENDER)
<freetypy.Glyph at 0x17afb00>
>>> bitmap = glyph.bitmap
```

The bitmap data can be converted to a list:

```
>>> bitmap.to_list()
[[0L, 0L, 0L, 0L, 0L, 0L, 10L, 85L, 142L, 195L, 222L, 238L, 252L,
 243L, 230L, 207L, 170L, 132L, 76L, 17L, 0L, 0L, 0L, 0L, 0L, 0L] ...
```

However, to display it at the console, it's usually nicer to use the `freetypy.util.bitmap_to_ascii` helper function:

```
>>> ft.util.bitmap_to_ascii(bitmap)
       .+*******++.
     +*############**.
   .*##################
  .*###################
  *####*+.     .++*####
 *###*.            .*##
.###*.               .*
+###+
*###
*###
*###
*###
*###+
+####+
 #####*+.
 +#######**+.
  *###########**+.
   +##############*+
    .*##############*.
      .+*############*.
          .++*########*
               .+*#####*
                  +*####.
                   .*###+
                    +###*
                     ###*
                     ###*
                     ###*
                     ###*
                    +###+
*.                 .####.
##*+              +*###*
#####*+..      .+*#####.
#####################*.
####################*.
 .+*##############*+
     .++*******+.
```

This *Bitmap* is a Python buffer object. If you have Numpy, you can easily convert this data to an array:

```
>>> import numpy as np
>>> array = np.array(bitmap)
```

Getting the outline from a glyph

To get the outline data from a glyph, for example to convert it to another format, the easiest way is to use *freetypy.Outline.decompose*:

```
class Decomposer(object):
    def __init__(self):
        self.entries = []

    def move_to(self, point):
        self.entries.append(('move_to', point))
    def line_to(self, point):
        self.entries.append(('line_to', point))

    def conic_to(self, a, b):
        self.entries.append(('conic_to', a, b))

    def cubic_to(self, a, b, c):
        self.entries.append(('cubic_to', a, b, c))

face = ft.Face("Vera.ttf")
face.set_char_size(12, 12, 300, 300)
glyph = face.load_char(ord('B'))

d = Decomposer()
glyph.outline.decompose(d)
print(d.entries)
```

This outputs:

```
[('move_to', (640.0, 1088.0)),
 ('line_to', (640.0, 256.0)),
 ('line_to', (1130.0, 256.0)),
 ('conic_to', (1402.0, 256.0), (1533.0, 358.0)),
 ('conic_to', (1664.0, 461.0), (1664.0, 673.0)),
 ('conic_to', (1664.0, 886.0), (1533.0, 987.0)),
 ('conic_to', (1402.0, 1088.0), (1130.0, 1088.0)),
 ('line_to', (640.0, 1088.0)),
 ('move_to', (640.0, 2048.0)),
 ('line_to', (640.0, 1344.0)),
 ('line_to', (1093.0, 1344.0)),
 ('conic_to', (1317.0, 1344.0), (1426.0, 1431.0)),
 ('conic_to', (1536.0, 1518.0), (1536.0, 1696.0)),
 ('conic_to', (1536.0, 1873.0), (1426.0, 1960.0)),
 ('conic_to', (1317.0, 2048.0), (1093.0, 2048.0)),
 ('line_to', (640.0, 2048.0)),
 ('move_to', (320.0, 2304.0)),
 ('line_to', (1115.0, 2304.0)),
 ('conic_to', (1471.0, 2304.0), (1663.0, 2154.0)),
 ('conic_to', (1856.0, 2005.0), (1856.0, 1729.0)),
 ('conic_to', (1856.0, 1516.0), (1756.0, 1389.0)),
 ('conic_to', (1656.0, 1263.0), (1462.0, 1232.0)),
 ('conic_to', (1710.0, 1183.0), (1847.0, 1027.0)),
 ('conic_to', (1984.0, 872.0), (1984.0, 640.0)),
 ('conic_to', (1984.0, 334.0), (1766.0, 167.0)),
 ('conic_to', (1548.0, 0.0), (1146.0, 0.0)),
 ('line_to', (320.0, 0.0)),
 ('line_to', (320.0, 2304.0))]
```

CHAPTER 2

Freetypy API documentation

The API closely follows the FreeType API. It is not 100% complete, but it does contain the functionality that most would need.

Names have been modified to follow Python's PEP8 style guidelines.

Where appropriate, extra API functions have been added to make working with FreeType more Pythonic. These methods are marked with the in the documentation.

Face

Face	Models a given typeface, in a given style.
Size	A *Face* object at a given size.
Size_Metrics	The metrics of a size object.
FACE_FLAG	Bit flags for important metadata about a *Face*.
STYLE_FLAG	Bit flags indicating the style of a *Face*.
LOAD	Indicates what to load for a glyph.
SIZE_REQUEST_TYPE	Modes for selecting the size of a bitmap *Face*.
FSTYPE	Bit flags indicating the embedding and subsetting restrictions of a *Face*.
KERNING	Kerning modes.

freetypy.Face

class `freetypy.`**`Face`**

Models a given typeface, in a given style.

Parameters

- **`file`** (*`path, or readable file-like object`*) – The file containing the face data.

- **face_index** (*int, optional*) – The index of the face within the font. The first face has index 0.

Returns face

Return type Face instance

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Methods

attach	'Attach' data to a face object.
get_char_index	Get the glyph index of a given char code.
get_char_index_unicode	Get the glyph index of a given Unicode character.
get_char_name	Get the glyph name of the given unicode code point.
get_char_variant_index	Get the glyph id of a char code as modified by the variant selector.
get_chars	Iterate over all of the char codes in the current charmap.
get_fstype_flags	Get the *FSTYPE* flags for a font.
get_glyph_name	Get the ASCII name of the given glyph in a face.
get_kerning	Get the kerning vector between two glyphs of a same face.
get_name_index	Get the glyph index of a given glyph name.
get_postscript_name	Get the ASCII PostScript name of a given face, if available.
get_track_kerning	Get the track kerning for a given face object at a given size.
has_ps_glyph_names	Return `True` if a given face provides reliable PostScript glyph names.
load_char	Load a single glyph, according to its char code.
load_char_unicode	Load a single glyph, from a given unicode character.
load_glyph	Load a single glyph.
request_size	Resize the scale of the active *Size* object in a face.
select_charmap	Select a given charmap by its encoding tag.
select_size	Select a bitmap strike.
set_char_size	Set the nominal font size (in points).
set_charmap	Select a charmap for char code to glyph index mapping.
set_transform	Set the transformation applied to glyph images.

freetypy.Face.attach

Face.**attach**()

'Attach' data to a face object.

Normally, this is used to read additional information for the face object. For example, you can attach an AFM file that comes with a Type 1 font to get the kerning values and other metrics.

Parameters file (*path, or readable file-like object*) – The file containing the data to attach.

freetypy.Face.get_char_index

`Face.get_char_index()`

Get the glyph index of a given char code.

This function uses a charmap object to do the mapping.

Parameters `charcode` (*int*) – The char code.

Returns **glyph_index** – The glyph index. 0 means ‘undefined char code’.

Return type int

Notes

If you use FreeType to manipulate the contents of font files directly, be aware that the glyph index returned by this function doesn’t always correspond to the internal indices used within the file. This is done to ensure that value 0 always corresponds to the ‘missing glyph’. If the first glyph is not named ‘.notdef’, then for Type 1 and Type 42 fonts, ‘.notdef’ will be moved into the glyph ID 0 position, and whatever was there will be moved to the position ‘.notdef’ had. For Type 1 fonts, if there is no ‘.notdef’ glyph at all, then one will be created at index 0 and whatever was there will be moved to the last index – Type 42 fonts are considered invalid under this condition.

freetypy.Face.get_char_index_unicode

`Face.get_char_index_unicode()`

Get the glyph index of a given Unicode character.

Unlike *get_char_index*, if the selected *CharMap* is not Unicode, the given unicode character will first be encoded to the correct character map, if possible.

This is a freetypy-specific function.

Parameters `charcode` (*length-1 unicode string or int*) – The Unicode character.

Returns **glyph_index** – The glyph index. 0 means ‘undefined char code’.

Return type int

freetypy.Face.get_char_name

`Face.get_char_name()`

Get the glyph name of the given unicode code point.

Unlike *get_glyph_name*, this function will always return something, even if the font contains no glyph names.

Parameters `charcode` (*int*) – The char code.

Returns **glyph_name** – The name of the associated glyph.

Return type unicode

freetypy.Face.get_char_variant_index

`Face.`**`get_char_variant_index`**`()`

Get the glyph id of a char code as modified by the variant selector.

Parameters

- **`charcode`** (*`int`*) – The char code point in Unicode.
- **`variantSelector`** (*`int`*) – The Unicode code point of the variation selector.

Returns **glyph_index** – The glyph index. 0 means either 'undefined char code', or 'undefined selector code', or 'no variation selector cmap subtable', or 'current CharMap is not Unicode'.

Return type int

Notes

If you use FreeType to manipulate the contents of font files directly, be aware that the glyph index returned by this function doesn't always correspond to the internal indices used within the file. This is done to ensure that value 0 always corresponds to the 'missing glyph'.

This function is only meaningful if a) the font has a variation selector cmap sub table, and b) the current charmap has a Unicode encoding.

freetypy.Face.get_chars

`Face.`**`get_chars`**`()`

Iterate over all of the char codes in the current charmap.

Each result of the iterator is a tuple of the form `(charcode, glyph_index)`.

freetypy.Face.get_fstype_flags

`Face.`**`get_fstype_flags`**`()`

Get the *`FSTYPE`* flags for a font.

These indicate the licensing for embedding and subsetting a font.

Returns **flags** – A bitfield made up of the *`FSTYPE`* flags.

Return type int

freetypy.Face.get_glyph_name

`Face.`**`get_glyph_name`**`()`

Get the ASCII name of the given glyph in a face.

This only works for those faces where *`has_ps_glyph_names`* returns `True`.

Parameters **`glyph_index`** (*`int`*) – The glyph index.

Returns **glyph_name** – The glyph name.

Return type str

Notes

An exception is raised if the face doesn't provide glyph names or if the glyph index is invalid.

Be aware that FreeType reorders glyph indices internally so that glyph index 0 always corresponds to the 'missing glyph' (called '.notdef').

freetypy.Face.get_kerning

`Face.`**`get_kerning`**`()`

Get the kerning vector between two glyphs of a same face.

Parameters

- **`left_glyph`** (*`int`*) – The index of the left glyph in the kern pair.
- **`right_glyph`** (*`int`*) – The index of the right glyph in the kern pair.
- **`kern_mode`** (*`int, optional`*) – A *`KERNING`* mode.

Returns **vector** – The kerning between the two glyphs.

Return type *Vector*

Notes

Only horizontal layouts (left-to-right & right-to-left) are supported by this method. Other layouts, or more sophisticated kernings, are out of the scope of this API function – they can be implemented through format-specific interfaces.

freetypy.Face.get_name_index

`Face.`**`get_name_index`**`()`

Get the glyph index of a given glyph name.

This function uses driver specific objects to do the translation.

Parameters **`glyph_name`** (*`str`*) – The glyph name.

Returns **glyph_index** – The glyph index. 0 means 'undefined char code'.

Return type int

freetypy.Face.get_postscript_name

`Face.`**`get_postscript_name`**`()`

Get the ASCII PostScript name of a given face, if available.

This only works with PostScript and TrueType fonts.

freetypy.Face.get_track_kerning

`Face.`**`get_track_kerning`**`()`

Get the track kerning for a given face object at a given size.

Parameters

- **point_size** (*float*) – The point size in fractional points.
- **degree** (*int*) – The degree of tightness. Increasingly negative values represent tighter track kerning, while increasingly positive values represent looser track kerning. Value zero means no track kerning.

Returns **kerning** – The track kerning between the glyphs.

Return type float

Notes

Currently, only the Type 1 font driver supports track kerning, using data from AFM files (if attached with *attach*).

Only very few AFM files come with track kerning data; please refer to the Adobe's AFM specification for more details.

freetypy.Face.has_ps_glyph_names

Face.**has_ps_glyph_names**()

Return `True` if a given face provides reliable PostScript glyph names.

When this function returns `True`, the caller is sure that the glyph names returned by *Face.get_glyph_name* are reliable.

freetypy.Face.load_char

Face.**load_char**()

Load a single glyph, according to its char code.

Parameters

- **char_code** (*int*) – The glyph's char code, according to the current charmap used in the face.
- **load_flags** (*int, optional*) – A flag indicating what to load for this glyph. The *LOAD* constants can be used to control the glyph loading process (e.g., whether the outline should be scaled, whether to load bitmaps or not, whether to hint the outline, etc).

Returns **glyph** – The loaded *Glyph*.

Return type *Glyph*

Notes

See *load_glyph* for more information.

freetypy.Face.load_char_unicode

Face.**load_char_unicode**()

Load a single glyph, from a given unicode character.

Unlike *load_char*, if the selected *CharMap* is not Unicode, the given unicode character will first be encoded to the correct character map, if possible.

This is a freetypy-specific function.

Parameters

- **char_code** (*unicode character (1-element unicode string) or int*) – The unicode character.
- **load_flags** (*int, optional*) – A flag indicating what to load for this glyph. The *LOAD* constants can be used to control the glyph loading process (e.g., whether the outline should be scaled, whether to load bitmaps or not, whether to hint the outline, etc).

Returns **glyph** – The loaded *Glyph*.

Return type *Glyph*

Notes

See *load_glyph* for more information.

freetypy.Face.load_glyph

Face.**load_glyph**()

Load a single glyph.

Parameters

- **glyph_index** (*int*) – The index of the glyph in the font file. For CID-keyed fonts (either in PS or in CFF format) this argument specifies the CID value.
- **load_flags** (*int, optional*) – A flag indicating what to load for this glyph. The *LOAD* constants can be used to control the glyph loading process (e.g., whether the outline should be scaled, whether to load bitmaps or not, whether to hint the outline, etc).

Returns **glyph** – The loaded *Glyph*.

Return type *Glyph*

Notes

By default, hinting is enabled and the font's native hinter (see *FACE_FLAG.HINTER*) is preferred over the auto-hinter. You can disable hinting by setting *LOAD.NO_HINTING* or change the precedence by setting *LOAD.FORCE_AUTOHINT*. You can also set *LOAD.NO_AUTOHINT* in case you don't want the auto-hinter to be used at all.

See the description of *FACE_FLAG.TRICKY* for a special exception (affecting only a handful of Asian fonts).

Besides deciding which hinter to use, you can also decide which hinting algorithm to use. See *LOAD*'s TARGET_XXX constants for details.

Note that the auto-hinter needs a valid Unicode cmap (either a native one or synthesized by FreeType) for producing correct results. If a font provides an incorrect mapping (for example, assigning the char code U+005A, LATIN CAPITAL LETTER Z, to a glyph depicting a mathematical integral sign), the auto-hinter might produce useless results.

The loaded glyph may be transformed. See *set_transform* for the details.

For subsetted CID-keyed fonts, a `ValueError` exception is thrown for invalid CID values (this is, for CID values which don't have a corresponding glyph in the font). See the discussion of the *FACE_FLAG.CID_KEYED* flag for more details.

You should use only one of the *LOAD* `TARGET_XXX` values in your `load_flags`. They can't be OR'ed.

If *LOAD.RENDER* is also set, the glyph is rendered in the corresponding mode (i.e., the mode which matches the used algorithm best). An exception is *LOAD.TARGET_MONO* since it implies *LOAD.MONOCHROME*.

freetypy.Face.request_size

`Face.`**`request_size`**`()`

Resize the scale of the active *Size* object in a face.

Parameters

- **`type`** (*size request constant*) – The size request type. See *SIZE_REQUEST_TYPE* for the available constants.
- **`width`** (*float, optional*) – The desired width.
- **`height`** (*float, optional*) – The desired height.
- **`horiResolution`** (*int, optional*) – The horizontal resolution. If set to zero, `width` is treated as a fractional pixel value.
- **`vertResolution`** (*int, optional*) – The vertical resolution. If set to zero, *height* is treated as a fractional pixel value.

Notes

If `width` is zero, then the horizontal scaling value is set equal to the vertical scaling value, and vice versa.

Although drivers may select the bitmap strike matching the request, you should not rely on this if you intend to select a particular bitmap strike. Use *select_size* instead in that case.

The relation between the requested size and the resulting glyph size is dependent entirely on how the size is defined in the source face. The font designer chooses the final size of each glyph relative to this size. For more information refer to http://www.freetype.org/freetype2/docs/glyphs/glyphs-2.html.

See *Size_Metrics* for how size requesting relates to scaling values.

freetypy.Face.select_charmap

`Face.`**`select_charmap`**`()`

Select a given charmap by its encoding tag.

Parameters **`encoding`** (*int*) – An encoding constant. See *ENCODING*.

Notes

This function raises an exception if no charmap in the face corresponds to the encoding queried here.

Because many fonts contain more than a single cmap for Unicode encoding, this function has some special code to select the one which covers Unicode best ('best' in the sense that a UCS-4 cmap is preferred to a UCS-2 cmap). It is thus preferable to use *set_charmap* in this case.

freetypy.Face.select_size

`Face.`**`select_size`**`()`

Select a bitmap strike.

Parameters **`strike_index`** (*`int`*) – The index of the bitmap strike in the *`available_sizes`* field.

freetypy.Face.set_char_size

`Face.`**`set_char_size`**`()`

Set the nominal font size (in points).

Parameters

- **`char_width`** (*`float, optional`*) – The nominal width, in fractional points.
- **`char_height`** (*`float, optional`*) – The nominal height, in fractional points.
- **`horz_resolution`** (*`int, optional`*) – The horizontal resolution, in dpi.
- **`vert_resolution`** (*`int, optional`*) – The vertical resolution, in dpi.

Notes

If either the character width or height is zero (or not provided), it is set equal to the other value. : If both width and height are not provided, it defaults to 12.0.

If either the horizontal or vertical resolution is zero, it is set equal to the other value. If both resolution values are zero, they are set to 72dpi.

freetypy.Face.set_charmap

`Face.`**`set_charmap`**`()`

Select a charmap for char code to glyph index mapping.

Parameters **`charmap_index`** (*`int`*) – The index in *`charmaps`* to use.

Notes

This fails if a type 14 charmap is selected.

freetypy.Face.set_transform

`Face.`**`set_transform`**`()`

Set the transformation applied to glyph images.

Parameters

- **`matrix`** (*`2x2 sequence of floats, optional`*) – A matrix of the form:

```
.  [[xx, xy]
.   [yx, yy]]
```

- **`delta`** (*`2-sequence of floats, optional`*) –

Notes

The transformation is only applied to scalable image formats after the glyph has been loaded. It means that hinting is unaltered by the transformation and is performed on the character size given in the last call to `set_char_size` or `set_pixel_sizes`.

Note that this also transforms the `Face.glyph.advance` field, but not the values in `Face.glyph.metrics`.

Attributes

`ascender`	The typographic ascender of the face, expressed in font units.
`available_sizes`	An list of `Bitmap_Size` objects for all bitmap strikes in the face.
`bbox`	The font bounding box.
`charmap`	The currently active `CharMap`.
`charmaps`	A list of `CharMap` objects.
`descender`	The typographic descender of the face, expressed in font units.
`face_flags`	A set of `FACE_FLAG` flags that give important information about the face.
`face_index`	The index of the face in the font file.
`family_name`	The face's family name.
`filename`	The argument used to initialize the font.
`glyph`	The face's currently loaded glyph slot.
`has_horizontal`	Horizontal metrics are present.
`has_kerning`	Kerning data is present.
`has_vertical`	Vertical metrics are present.
`height`	The vertical distance between two consecutive baselines, expressed in font units.
`is_fixed_width`	Contains fixed-width (or 'monospace') glyphs.
`is_scalable`	Is a scalable `Face`.
`is_sfnt`	Based on the SFNT storage scheme.
`max_advance_height`	The maximum advance height, in font units, for all glyphs in this face.
`max_advance_width`	The maximum advance width, in font units, for all glyphs in this face.
`num_faces`	The number of faces in the font file.
`num_glyphs`	The number of glyphs in the face.
`sfnt_names`	Get a `SfntNames` list of `SfntName` entries for the Face.
`size`	The currently active size.
`style_flags`	A set of `STYLE_FLAG` flags indicating the style of the face.
`style_name`	The face's style name.
`tt_header`	The TrueType header (`TT_Header`).
`tt_horiheader`	The TrueType horizontal header (`TT_HoriHeader`).
`tt_os2`	The OS/2 TrueType header (`TT_OS2`).
`tt_pclt`	The PCLT header (`TT_Pclt`).

Continued on next page

Table 2.3 – continued from previous page

tt_postscript	The Postscript header (*TT_Postscript*).
tt_vertheader	The TrueType vertical header (*TT_VertHeader*).
underline_position	The position of the underline line, in font units.
underline_thickness	The thickness of the underline, in font units.
units_per_em	The number of font units per EM square.

freetypy.Face.ascender

Face.**ascender**

The typographic ascender of the face, expressed in font units. For font formats not having this information, it is set to `BBox.yMax`. Only relevant for scalable formats.

freetypy.Face.available_sizes

Face.**available_sizes**

An list of *Bitmap_Size* objects for all bitmap strikes in the face.

freetypy.Face.bbox

Face.**bbox**

The font bounding box. Coordinates are expressed in font units (see *units_per_em*). The box is large enough to contain any glyph from the font. Thus, *BBox.y_max* can be seen as the 'maximum ascender', and *BBox.y_min* as the 'minimum descender'. Only relevant for scalable formats.

Note that the bounding box might be off by (at least) one pixel for hinted fonts. See *Size_Metrics* for further discussion.

freetypy.Face.charmap

Face.**charmap**

The currently active *CharMap*.

freetypy.Face.charmaps

Face.**charmaps**

A list of *CharMap* objects.

freetypy.Face.descender

Face.**descender**

The typographic descender of the face, expressed in font units. For font formats not having this information, it is set to *BBox.y_min*. Note that this field is usually negative. Only relevant for scalable formats.

freetypy.Face.face_flags

Face.**face_flags**

A set of *FACE_FLAG* flags that give important information about the face.

freetypy.Face.face_index

`Face.face_index`

The index of the face in the font file. It is set to 0 if there is only one face in the font file.

freetypy.Face.family_name

`Face.family_name`

The face's family name. This is an ASCII string, usually in English, which describes the typeface's family (like 'Times New Roman', 'Bodoni', 'Garamond', etc). This is a least common denominator used to list fonts. Some formats (TrueType & OpenType) provide localized and Unicode versions of this string. Applications should use the format specific interface to access them. Can be `None` (e.g., in fonts embedded in a PDF file).

freetypy.Face.filename

`Face.filename`

The argument used to initialize the font.

It is usually a filename, but may be a file object.

freetypy.Face.glyph

`Face.glyph`

The face's currently loaded glyph slot.

freetypy.Face.has_horizontal

`Face.has_horizontal`

Horizontal metrics are present.

This is `True` for all font formats though.

freetypy.Face.has_kerning

`Face.has_kerning`

Kerning data is present.

The kerning data can be accessed with *`Face.get_kerning`*.

freetypy.Face.has_vertical

`Face.has_vertical`

Vertical metrics are present.

`False` when the face contains only synthesized vertical metrics.

freetypy.Face.height

`Face.height`

The vertical distance between two consecutive baselines, expressed in font units.

It is always positive. Only relevant for scalable formats.

If you want the global glyph height, use `ascender` - `descender`.

freetypy.Face.is_fixed_width

`Face.is_fixed_width`

Contains fixed-width (or ‘monospace’) glyphs.

freetypy.Face.is_scalable

`Face.is_scalable`

Is a scalable `Face`.

This includes TrueType, Type 1, Type 42, CID, OpenType/CFF, and PFR font formats.

freetypy.Face.is_sfnt

`Face.is_sfnt`

Based on the SFNT storage scheme.

This usually means: TrueType fonts, OpenType fonts, as well as SFNT-based embedded bitmap fonts.

freetypy.Face.max_advance_height

`Face.max_advance_height`

The maximum advance height, in font units, for all glyphs in this face. This is only relevant for vertical layouts, and is set to `height` for fonts that do not provide vertical metrics. Only relevant for scalable formats.

freetypy.Face.max_advance_width

`Face.max_advance_width`

The maximum advance width, in font units, for all glyphs in this face. This can be used to make word wrapping computations faster. Only relevant for scalable formats.

freetypy.Face.num_faces

`Face.num_faces`

The number of faces in the font file. Some font formats can have multiple faces in a font file.

freetypy.Face.num_glyphs

Face.**num_glyphs**

The number of glyphs in the face. If the face is scalable and has sbits (see `num_fixed_sizes`), it is set to the number of outline glyphs.

For CID-keyed fonts, this value gives the highest CID used in the font.

freetypy.Face.sfnt_names

Face.**sfnt_names**

Get a *SfntNames* list of *SfntName* entries for the Face.

freetypy.Face.size

Face.**size**

The currently active size.

freetypy.Face.style_flags

Face.**style_flags**

A set of *STYLE_FLAG* flags indicating the style of the face.

freetypy.Face.style_name

Face.**style_name**

The face's style name. This is an ASCII string, usually in English, which describes the typeface's style (like 'Italic', 'Bold', 'Condensed', etc). Not all font formats provide a style name, so this field is optional, and can be set to `None`. As for *family_name*, some formats provide localized and Unicode versions of this string. Applications should use the format specific interface to access them.

freetypy.Face.tt_header

Face.**tt_header**

The TrueType header (*TT_Header*).

freetypy.Face.tt_horiheader

Face.**tt_horiheader**

The TrueType horizontal header (*TT_HoriHeader*).

Isn't present in all font files. If missing, `AttributeError` is raised.

freetypy.Face.tt_os2

Face.**tt_os2**

The OS/2 TrueType header (*TT_OS2*).

Isn't present in all font files. If missing, `AttributeError` is raised.

freetypy.Face.tt_pclt

Face.**tt_pclt**

The PCLT header (*TT_Pclt*).

Isn't present in all font files. If missing, AttributeError is raised.

freetypy.Face.tt_postscript

Face.**tt_postscript**

The Postscript header (*TT_Postscript*).

Isn't present in all font files. If missing, AttributeError is raised.

freetypy.Face.tt_vertheader

Face.**tt_vertheader**

The TrueType vertical header (*TT_VertHeader*).

Isn't present in all font files. If missing, AttributeError is raised.

freetypy.Face.underline_position

Face.**underline_position**

The position of the underline line, in font units.

It is the center of the underlining stem. Only relevant for scalable formats.

freetypy.Face.underline_thickness

Face.**underline_thickness**

The thickness of the underline, in font units.

Only relevant for scalable formats.

freetypy.Face.units_per_em

Face.**units_per_em**

The number of font units per EM square.

This is typically 2048 for TrueType fonts, and 1000 for Type 1 fonts. Only relevant for scalable formats.

freetypy.Size

class freetypy.**Size**

A *Face* object at a given size.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

face	The parent *Face* object.
metrics	Metrics (*Size_Metrics*) for this size object.

freetypy.Size.face

Size.**face**

The parent *Face* object.

freetypy.Size.metrics

Size.**metrics**

Metrics (*Size_Metrics*) for this size object.

freetypy.Size_Metrics

class freetypy.**Size_Metrics**

The metrics of a size object.

The scaling values, if relevant, are determined first during a size changing operation. The remaining fields are then set by the driver. For scalable formats, they are usually set to scaled values of the corresponding fields in *Face*.

Note that due to glyph hinting, these values might not be exact for certain fonts. Thus they must be treated as unreliable with an error margin of at least one pixel!

Indeed, the only way to get the exact metrics is to render all glyphs. As this would be a definite performance hit, it is up to client applications to perform such computations.

The *Size_Metrics* object is valid for bitmap fonts also.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

ascender	The ascender in fractional pixels.
descender	The descender in fractional pixels.
height	The height in fractional pixels.
max_advance	The maximum advance width in fractional pixels.
x_ppem	The width of the scaled EM square in pixels.
x_scale	A fractional scaling value.
y_ppem	The height of the scaled EM square in pixels.
y_scale	A fractional scaling value.

freetypy.Size_Metrics.ascender

Size_Metrics.**ascender**

The ascender in fractional pixels.

freetypy.Size_Metrics.descender

`Size_Metrics.descender`

The descender in fractional pixels.

freetypy.Size_Metrics.height

`Size_Metrics.height`

The height in fractional pixels.

freetypy.Size_Metrics.max_advance

`Size_Metrics.max_advance`

The maximum advance width in fractional pixels.

freetypy.Size_Metrics.x_ppem

`Size_Metrics.x_ppem`

The width of the scaled EM square in pixels.

It is also referred to as ‘nominal width’.

freetypy.Size_Metrics.x_scale

`Size_Metrics.x_scale`

A fractional scaling value.

Used to convert horizontal metrics from font units to 26.6 fractional pixels.

Only relevant for scalable font formats.

freetypy.Size_Metrics.y_ppem

`Size_Metrics.y_ppem`

The height of the scaled EM square in pixels.

It is also referred to as ‘nominal height’.

freetypy.Size_Metrics.y_scale

`Size_Metrics.y_scale`

A fractional scaling value.

Used to convert horizontal metrics from font units to 26.6 fractional pixels.

Only relevant for scalable font formats.

freetypy.FACE_FLAG

class `freetypy.`**`FACE_FLAG`**

Bit flags for important metadata about a *`Face`*.

- *`SCALABLE`*: Indicates that the face contains outline glyphs. This doesn't prevent bitmap strikes, i.e., a face can have both this and and *`FACE_FLAG.FIXED_SIZES`* set.
- *`FIXED_SIZES`*: Indicates that the face contains bitmap strikes. See also *`Face.available_sizes`*.
- *`FIXED_WIDTH`*: Indicates that the face contains fixed-width characters (like Courier, Lucido, MonoType, etc.).
- *`SFNT`*: Indicates that the face uses the 'sfnt' storage scheme. For now, this means TrueType and OpenType.
- *`HORIZONTAL`*: Indicates that the face contains horizontal glyph metrics. This should be set for all common formats.
- *`VERTICAL`*: Indicates that the face contains vertical glyph metrics. This is only available in some formats, not all of them.
- *`KERNING`*: Indicates that the face contains kerning information. If set, the kerning distance can be retrieved through *`Face.get_kerning`*. Otherwise the function always returns the vector (0, 0). Note that FreeType doesn't handle kerning data from the 'GPOS' table (as present in some OpenType fonts).
- *`MULTIPLE_MASTERS`*: Indicates that the font contains multiple masters and is capable of interpolating between them. See the multiple-masters specific API for details.
- *`GLYPH_NAMES`*: Indicates that the font contains glyph names that can be retrieved through *`Face.get_glyph_name`*. Note that some TrueType fonts contain broken glyph name tables. Use the function *`Face.has_ps_glyph_names`* when needed.
- *`EXTERNAL_STREAM`*: Used internally by FreeType to indicate that a face's stream was provided by the client application and should not be destroyed when the face is destructed. Don't read or test this flag.
- *`HINTER`*: Set if the font driver has a hinting machine of its own. For example, with TrueType fonts, it makes sense to use data from the SFNT 'gasp' table only if the native TrueType hinting engine (with the bytecode interpreter) is available and active.
- *`CID_KEYED`*: Set if the font is CID-keyed. In that case, the font is not accessed by glyph indices but by CID values. For subsetted CID-keyed fonts this has the consequence that not all index values are a valid argument to *`Face.load_glyph`*. Only the CID values for which corresponding glyphs in the subsetted font exist make *`Face.load_glyph`* return successfully; in all other cases you get a `ValueError` exception.

  Note that CID-keyed fonts which are in an SFNT wrapper don't have this flag set since the glyphs are accessed in the normal way (using contiguous indices); the 'CID-ness' isn't visible to the application.
- *`TRICKY`*: Set if the font is 'tricky', this is, it always needs the font format's native hinting engine to get a reasonable result. A typical example is the Chinese font 'mingli.ttf' which uses TrueType bytecode instructions to move and scale all of its subglyphs.

  It is not possible to autohint such fonts using *`LOAD.FORCE_AUTOHINT`*; it will also ignore *`LOAD.NO_HINTING`*. You have to set both *`LOAD.NO_HINTING`* and *`LOAD.NO_AUTOHINT`* to really disable hinting; however, you probably never want this except for demonstration purposes.

  Currently, there are about a dozen TrueType fonts in the list of tricky fonts; they are hard-coded.

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

CID_KEYED
EXTERNAL_STREAM
FIXED_SIZES
FIXED_WIDTH
GLYPH_NAMES
HINTER
HORIZONTAL
KERNING
MULTIPLE_MASTERS
SCALABLE
SFNT
TRICKY
VERTICAL

freetypy.FACE_FLAG.CID_KEYED

`FACE_FLAG.`**`CID_KEYED`** = **freetypy.FACE_FLAG.CID_KEYED**

freetypy.FACE_FLAG.EXTERNAL_STREAM

`FACE_FLAG.`**`EXTERNAL_STREAM`** = **freetypy.FACE_FLAG.EXTERNAL_STREAM**

freetypy.FACE_FLAG.FIXED_SIZES

`FACE_FLAG.`**`FIXED_SIZES`** = **freetypy.FACE_FLAG.FIXED_SIZES**

freetypy.FACE_FLAG.FIXED_WIDTH

`FACE_FLAG.`**`FIXED_WIDTH`** = **freetypy.FACE_FLAG.FIXED_WIDTH**

freetypy.FACE_FLAG.GLYPH_NAMES

`FACE_FLAG.`**`GLYPH_NAMES`** = **freetypy.FACE_FLAG.GLYPH_NAMES**

freetypy.FACE_FLAG.HINTER

`FACE_FLAG.`**`HINTER`** = **freetypy.FACE_FLAG.HINTER**

freetypy.FACE_FLAG.HORIZONTAL

`FACE_FLAG.`**`HORIZONTAL`** = **freetypy.FACE_FLAG.HORIZONTAL**

freetypy.FACE_FLAG.KERNING

`FACE_FLAG.`**`KERNING`** = **freetypy.FACE_FLAG.KERNING**

freetypy.FACE_FLAG.MULTIPLE_MASTERS

FACE_FLAG.**MULTIPLE_MASTERS** = **freetypy.FACE_FLAG.MULTIPLE_MASTERS**

freetypy.FACE_FLAG.SCALABLE

FACE_FLAG.**SCALABLE** = **freetypy.FACE_FLAG.SCALABLE**

freetypy.FACE_FLAG.SFNT

FACE_FLAG.**SFNT** = **freetypy.FACE_FLAG.SFNT**

freetypy.FACE_FLAG.TRICKY

FACE_FLAG.**TRICKY** = **freetypy.FACE_FLAG.TRICKY**

freetypy.FACE_FLAG.VERTICAL

FACE_FLAG.**VERTICAL** = **freetypy.FACE_FLAG.VERTICAL**

freetypy.STYLE_FLAG

class freetypy.**STYLE_FLAG**

Bit flags indicating the style of a *Face*.

- *ITALIC*: Indicates that a given face style is italic or oblique.
- *BOLD*: Indicates that a given face is bold.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

BOLD
ITALIC

freetypy.STYLE_FLAG.BOLD

STYLE_FLAG.**BOLD** = **freetypy.STYLE_FLAG.BOLD**

freetypy.STYLE_FLAG.ITALIC

STYLE_FLAG.**ITALIC** = **freetypy.STYLE_FLAG.ITALIC**

freetypy.LOAD

class `freetypy.`**`LOAD`**

Indicates what to load for a glyph.

The *`LOAD`* constants can be used to control the glyph loading process (e.g., whether the outline should be scaled, whether to load bitmaps or not, whether to hint the outline, etc).

- *`DEFAULT`*: Corresponding to 0, this value is used as the default glyph load operation. In this case, the following happens:

  1. FreeType looks for a bitmap for the glyph corresponding to the face's current size. If one is found, the function returns. The bitmap data can be accessed from the glyph slot (see note below).
  2. If no embedded bitmap is searched or found, FreeType looks for a scalable outline. If one is found, it is loaded from the font file, scaled to device pixels, then 'hinted' to the pixel grid in order to optimize it. The outline data can be accessed from the glyph slot (see note below).

  Note that by default, the glyph loader doesn't render outlines into bitmaps. The following flags are used to modify this default behaviour to more specific and useful cases.

- *`NO_SCALE`*: Don't scale the loaded outline glyph but keep it in font units.

  This flag implies *`LOAD.NO_HINTING`* and *`LOAD.NO_BITMAP`*, and unsets *`LOAD.RENDER`*.

  If the font is 'tricky' (see *`FACE_FLAG.TRICKY`* for more), using *`LOAD.NO_SCALE`* usually yields meaningless outlines because the subglyphs must be scaled and positioned with hinting instructions. This can be solved by loading the font without *`LOAD.NO_SCALE`* and setting the character size to *`units_per_em`*.

- *`NO_HINTING`*: Disable hinting. This generally generates 'blurrier' bitmap glyphs when the glyph are rendered in any of the anti-aliased modes. See also the note below.

  This flag is implied by *`LOAD.NO_SCALE`*.

- *`RENDER`*: Render after the glyph is loaded. By default, the glyph is rendered in *`RENDER_MODE.NORMAL`* mode. This can be overridden by `LOAD.TARGET_XXX` or *`LOAD.MONOCHROME`*.

  This flag is unset by *`LOAD.NO_SCALE`*.

- *`NO_BITMAP`*: Ignore bitmap strikes when loading. Bitmap-only fonts ignore this flag.

  `LOAD.NO_SCALE` always sets this flag.

- *`VERTICAL_LAYOUT`*: Load the glyph for vertical text layout. In particular, the `advance` value in the *`Glyph`* object is set to the `vert_advance` value of the `metrics` field.

  In case *`Face.has_vertical`* doesn't return `True`, you shouldn't use this flag currently. Reason is that in this case vertical metrics get synthesized, and those values are not always consistent across various font formats.

- *`FORCE_AUTOHINT`*: Indicates that the auto-hinter is preferred over the font's native hinter. See also the note below.

- *`CROP_BITMAP`*: Indicates that the font driver should crop the loaded bitmap glyph (i.e., remove all space around its black bits). Not all drivers implement this.

- *`PEDANTIC`*: Indicates that the font driver should perform pedantic verifications during glyph loading. This is mostly used to detect broken glyphs in fonts. By default, FreeType tries to handle broken fonts also.

  In particular, errors from the TrueType bytecode engine are not passed to the application if this flag is not set; this might result in partially hinted or distorted glyphs in case a glyph's bytecode is buggy.

- *`NO_RECURSE`*: This flag is only used internally. It merely indicates that the font driver should not load composite glyphs recursively. Instead, it should set the `num_subglyph` and `subglyphs` values of the *`Glyph`*, and set *`Glyph.format`* to *`GLYPH_FORMAT.COMPOSITE`*.

  The description of sub-glyphs is not available to client applications for now.

  This flag implies *`LOAD.NO_SCALE`* and *`LOAD.IGNORE_TRANSFORM`*.

- *`IGNORE_TRANSFORM`*: Indicates that the transform matrix set by *`Face.set_transform`* should be ignored.

- *`MONOCHROME`*: This flag is used with *`LOAD.RENDER`* to indicate that you want to render an outline glyph to a 1-bit monochrome bitmap glyph, with 8 pixels packed into each byte of the bitmap data.

  Note that this has no effect on the hinting algorithm used. You should rather use *`LOAD.TARGET_MONO`* so that the monochrome-optimized hinting algorithm is used.

- *`LINEAR_DESIGN`*: Indicates that the `linear_hori_advance` and `linear_vert_advance` fields of *`Glyph`* should be kept in font units. See *`Glyph`* for details.

- *`NO_AUTOHINT`*: Disable auto-hinter. See also the note below.

The following flags select a specific hinting algorithm to use by the hinter.

- *`TARGET_NORMAL`*: This corresponds to the default hinting algorithm, optimized for standard gray-level rendering. For monochrome output, use *`LOAD.TARGET_MONO`* instead.

- *`TARGET_LIGHT`*: A lighter hinting algorithm for non-monochrome modes. Many generated glyphs are more fuzzy but better resemble its original shape. A bit like rendering on Mac OS X.

  As a special exception, this target implies *`LOAD.FORCE_AUTOHINT`*.

- *`TARGET_MONO`*: Strong hinting algorithm that should only be used for monochrome output. The result is probably unpleasant if the glyph is rendered in non-monochrome modes.

- *`TARGET_LCD`*: A variant of *`LOAD.TARGET_NORMAL`* optimized for horizontally decimated LCD displays. On many FreeType builds, this functionality will be disabled due to patent restrictions, in which case the resulting bitmap will be grayscale.

- *`TARGET_LCD_V`*: Strong hinting algorithm that should only be used for monochrome output. The result is probably unpleasant if the glyph is rendered in non-monochrome modes. On many freetype builds, this functionality will be disabled due to patent restrictions, in which case the resulting bitmap will be grayscale.

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

`CROP_BITMAP`
`DEFAULT`
`FORCE_AUTOHINT`
`IGNORE_GLOBAL_ADVANCE_WIDTH`
`IGNORE_TRANSFORM`
`LINEAR_DESIGN`
`MONOCHROME`
`NO_AUTOHINT`
`NO_BITMAP`

Continued on next page

Table 2.8 – continued from previous page

NO_HINTING
NO_RECURSE
NO_SCALE
PEDANTIC
RENDER
TARGET_LCD
TARGET_LCD_V
TARGET_LIGHT
TARGET_MONO
TARGET_NORMAL
VERTICAL_LAYOUT

freetypy.LOAD.CROP_BITMAP

LOAD.**CROP_BITMAP** = **freetypy.LOAD.CROP_BITMAP**

freetypy.LOAD.DEFAULT

LOAD.**DEFAULT** = **0**

freetypy.LOAD.FORCE_AUTOHINT

LOAD.**FORCE_AUTOHINT** = **freetypy.LOAD.FORCE_AUTOHINT**

freetypy.LOAD.IGNORE_GLOBAL_ADVANCE_WIDTH

LOAD.**IGNORE_GLOBAL_ADVANCE_WIDTH** = **freetypy.LOAD.IGNORE_GLOBAL_ADVANCE_WIDTH**

freetypy.LOAD.IGNORE_TRANSFORM

LOAD.**IGNORE_TRANSFORM** = **freetypy.LOAD.IGNORE_TRANSFORM**

freetypy.LOAD.LINEAR_DESIGN

LOAD.**LINEAR_DESIGN** = **freetypy.LOAD.LINEAR_DESIGN**

freetypy.LOAD.MONOCHROME

LOAD.**MONOCHROME** = **freetypy.LOAD.MONOCHROME**

freetypy.LOAD.NO_AUTOHINT

LOAD.**NO_AUTOHINT** = **freetypy.LOAD.NO_AUTOHINT**

freetypy.LOAD.NO_BITMAP

LOAD.**NO_BITMAP** = **freetypy.LOAD.NO_BITMAP**

freetypy.LOAD.NO_HINTING

`LOAD.`**`NO_HINTING`** = **freetypy.LOAD.NO_HINTING**

freetypy.LOAD.NO_RECURSE

`LOAD.`**`NO_RECURSE`** = **freetypy.LOAD.NO_RECURSE**

freetypy.LOAD.NO_SCALE

`LOAD.`**`NO_SCALE`** = **freetypy.LOAD.NO_SCALE**

freetypy.LOAD.PEDANTIC

`LOAD.`**`PEDANTIC`** = **freetypy.LOAD.PEDANTIC**

freetypy.LOAD.RENDER

`LOAD.`**`RENDER`** = **freetypy.LOAD.RENDER**

freetypy.LOAD.TARGET_LCD

`LOAD.`**`TARGET_LCD`** = **freetypy.LOAD.TARGET_LCD | freetypy.LOAD.TARGET_LIGHT | freetypy.LOAD.TARGET_M**

freetypy.LOAD.TARGET_LCD_V

`LOAD.`**`TARGET_LCD_V`** = **freetypy.LOAD.TARGET_LCD_V**

freetypy.LOAD.TARGET_LIGHT

`LOAD.`**`TARGET_LIGHT`** = **freetypy.LOAD.TARGET_LCD | freetypy.LOAD.TARGET_LIGHT**

freetypy.LOAD.TARGET_MONO

`LOAD.`**`TARGET_MONO`** = **freetypy.LOAD.TARGET_LCD | freetypy.LOAD.TARGET_MONO**

freetypy.LOAD.TARGET_NORMAL

`LOAD.`**`TARGET_NORMAL`** = **0**

freetypy.LOAD.VERTICAL_LAYOUT

`LOAD.`**`VERTICAL_LAYOUT`** = **freetypy.LOAD.VERTICAL_LAYOUT**

freetypy.SIZE_REQUEST_TYPE

class freetypy.**SIZE_REQUEST_TYPE**

Modes for selecting the size of a bitmap *Face*.

- *NOMINAL*: The nominal size. The *Face.units_per_em* is used to determine both scaling values.
- *REAL_DIM*: The real dimension. The sum of the the *ascender* and (minus of) the *descender* are used to determine both scaling values.
- *BBOX*: The font bounding box. The width and height of the *bbox* are used to determine the horizontal and vertical scaling value, respectively.
- *CELL*: The *max_advance_width* is used to determine the horizontal scaling value; the vertical scaling value is determined the same way as *SIZE_REQUEST_TYPE.REAL_DIM* does. Finally, both scaling values are set to the smaller one. This type is useful if you want to specify the font size for, say, a window of a given dimension and 80x24 cells.
- *SCALES*: Specify the scaling values directly.

The above descriptions only apply to scalable formats. For bitmap formats, the behaviour is up to the driver.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

BBOX
CELL
NOMINAL
REAL_DIM
SCALES

freetypy.SIZE_REQUEST_TYPE.BBOX

SIZE_REQUEST_TYPE.**BBOX** = **freetypy.SIZE_REQUEST_TYPE.BBOX**

freetypy.SIZE_REQUEST_TYPE.CELL

SIZE_REQUEST_TYPE.**CELL** = **freetypy.SIZE_REQUEST_TYPE.CELL**

freetypy.SIZE_REQUEST_TYPE.NOMINAL

SIZE_REQUEST_TYPE.**NOMINAL** = **freetypy.SIZE_REQUEST_TYPE.NOMINAL**

freetypy.SIZE_REQUEST_TYPE.REAL_DIM

SIZE_REQUEST_TYPE.**REAL_DIM** = **freetypy.SIZE_REQUEST_TYPE.REAL_DIM**

freetypy.SIZE_REQUEST_TYPE.SCALES

SIZE_REQUEST_TYPE.**SCALES** = **freetypy.SIZE_REQUEST_TYPE.SCALES**

freetypy.FSTYPE

class freetypy.**FSTYPE**

Bit flags indicating the embedding and subsetting restrictions of a *Face*.

- *INSTALLABLE_EMBEDDING*: Fonts with no fsType bit set may be embedded and permanently installed on the remote system by an application.
- *RESTRICTED_LICENSE_EMBEDDING*: Fonts that have only this bit set must not be modified, embedded or exchanged in any manner without first obtaining permission of the font software copyright owner.
- *PREVIEW_AND_PRINT_EMBEDDING*: If this bit is set, the font may be embedded and temporarily loaded on the remote system. Documents containing Preview & Print fonts must be opened 'read-only'; no edits can be applied to the document.
- *EDITABLE_EMBEDDING*: If this bit is set, the font may be embedded but must only be installed temporarily on other systems. In contrast to Preview & Print fonts, documents containing editable fonts may be opened for reading, editing is permitted, and changes may be saved.
- *NO_SUBSETTING*: If this bit is set, the font may not be subsetted prior to embedding.
- *BITMAP_EMBEDDING_ONLY*: If this bit is set, only bitmaps contained in the font may be embedded; no outline data may be embedded. If there are no bitmaps available in the font, then the font is unembeddable.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

BITMAP_EMBEDDING_ONLY
EDITABLE_EMBEDDING
INSTALLABLE_EMBEDDING
NO_SUBSETTING
PREVIEW_AND_PRINT_EMBEDDING
RESTRICTED_LICENSE_EMBEDDING

freetypy.FSTYPE.BITMAP_EMBEDDING_ONLY

FSTYPE.**BITMAP_EMBEDDING_ONLY = freetypy.FSTYPE.BITMAP_EMBEDDING_ONLY**

freetypy.FSTYPE.EDITABLE_EMBEDDING

FSTYPE.**EDITABLE_EMBEDDING = freetypy.FSTYPE.EDITABLE_EMBEDDING**

freetypy.FSTYPE.INSTALLABLE_EMBEDDING

FSTYPE.**INSTALLABLE_EMBEDDING = 0**

freetypy.FSTYPE.NO_SUBSETTING

FSTYPE.**NO_SUBSETTING = freetypy.FSTYPE.NO_SUBSETTING**

freetypy.FSTYPE.PREVIEW_AND_PRINT_EMBEDDING

FSTYPE.**PREVIEW_AND_PRINT_EMBEDDING = freetypy.FSTYPE.PREVIEW_AND_PRINT_EMBEDDING**

freetypy.FSTYPE.RESTRICTED_LICENSE_EMBEDDING

FSTYPE.**RESTRICTED_LICENSE_EMBEDDING = freetypy.FSTYPE.RESTRICTED_LICENSE_EMBEDDING**

freetypy.KERNING

class freetypy.**KERNING**

Kerning modes.

- *DEFAULT*: Return scaled and grid-fitted kerning distances (value is 0).
- *UNFITTED*: Return scaled but un-grid-fitted kerning distances.
- *UNSCALED*: Return the kerning vector in original font units.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

DEFAULT
UNFITTED
UNSCALED

freetypy.KERNING.DEFAULT

KERNING.**DEFAULT = freetypy.KERNING.DEFAULT**

freetypy.KERNING.UNFITTED

KERNING.**UNFITTED = freetypy.KERNING.UNFITTED**

freetypy.KERNING.UNSCALED

KERNING.**UNSCALED = freetypy.KERNING.UNSCALED**

Layout

Freetypy includes a very basic layout algorithm for left-to-right text. For more serious usage, one should use a real layout engine, such as Pango.

Layout	Manages very simple layout of left-to-right text.

freetypy.Layout

class `freetypy.`**`Layout`**

Manages very simple layout of left-to-right text.

Parameters

- **`face`** (*`Face`*) – The face for the layout.
- **`text`** (*`unicode`*) – The text to display in the layout.
- **`load_flags`** (*`LOAD`* flags, optional) – Any glyph load flags

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

`ink_bbox`	The tight bounding box (*`BBox`*) of the physical characters in the layout.
`layout`	Returns a list of tuples describing the layout.
`layout_bbox`	The logical bounding box (*`BBox`*) of the layout.

freetypy.Layout.ink_bbox

`Layout.`**`ink_bbox`**

The tight bounding box (*`BBox`*) of the physical characters in the layout. The origin is at (0, 0). The result is in pixels.

freetypy.Layout.layout

`Layout.`**`layout`**

Returns a list of tuples describing the layout.

Each tuple is of the form:

- *`Face`*: The *`Face`* object containing the glyph
- `glyph_index`: The glyph index within the *`Face`*
- `(x, y)`: The x, y position of the glyph

freetypy.Layout.layout_bbox

`Layout.`**`layout_bbox`**

The logical bounding box (*`BBox`*) of the layout. This should be used to manage the layout of the text against other text. The result is in pixels.

Subset

Freetypy includes support for subsetting TrueType and OpenType fonts.

subset.subset_font(input_fd, output_fd, ...)	Subset a SFNT-style (TrueType or OpenType) font.

freetypy.subset.subset_font

freetypy.subset.**subset_font**(*input_fd*, *output_fd*, *charcodes*, *tables_to_remove=None*)

Subset a SFNT-style (TrueType or OpenType) font.

If the font is not one of these types, a ValueError is raised.

Parameters

- **input_fd** (*readable file-like object, for bytes*) – The font file to read.
- **output_fd** (*writable file-like object, for bytes*) – The file to write a subsetted font file to.
- **charcodes** (*list of int or unicode string*) – The character codes to include in the output font file.
- **tables_to_remove** (*list of bytes, optional*) – The tags of tables to remove completely. If not provided, this defaults to:

  [b'GPOS', b'GSUB']

Glyph

Glyph	Information pertaining to a single glyph.
Glyph_Metrics	Models the metrics of a single *Glyph*.
GLYPH_BBOX	Indicates how the values of *Glyph.get_cbox* are returned.
GLYPH_FORMAT	Describes the format of a given glyph image.
SubGlyph	A description of a given subglyph.
SUBGLYPH_FLAG	Describes subglyphs.

freetypy.Glyph

class freetypy.**Glyph**

Information pertaining to a single glyph.

Notes

If *Face.load_glyph* is called with default flags (see *LOAD.DEFAULT*) the glyph image is loaded in the glyph slot in its native format (e.g., an outline glyph for TrueType and Type 1 formats).

This image can later be converted into a bitmap by calling *Glyph.render*.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Methods

get_cbox	Get the glyph's 'control box'.
render	Convert a given glyph image to a *Bitmap*.

freetypy.Glyph.get_cbox

Glyph.**get_cbox**()

Get the glyph's 'control box'.

The control box encloses all the outline's points, including Bézier control points. Though it coincides with the exact bounding box for most glyphs, it can be slightly larger in some situations (like when rotating an outline which contains Bézier outside arcs).

Computing the control box is very fast, while getting the bounding box can take much more time as it needs to walk over all segments and arcs in the outline.

Parameters **mode** (*int, optional*) – The mode which indicates how to interpret the returned bounding box values. For the available options, see *GLYPH_BBOX*.

Returns **bbox** – The glyph coordinate bounding box.

Return type *BBox*

Notes

Coordinates are relative to the glyph origin, using the *y* upwards convention.

If the glyph has been loaded with *LOAD.NO_SCALE*, mode must be set to *GLYPH_BBOX.UNSCALED* to get unscaled font units in 26.6 fixed-point pixel format.

If the font is tricky and the glyph has been loaded with *LOAD.NO_SCALE*, the resulting *BBox* is meaningless. To get reasonable values for the *BBox* it is necessary to load the glyph at a large ppem value (so that the hinting instructions can properly shift and scale the subglyphs), then extracting the *BBox* which can be eventually converted back to font units.

Note that the maximum coordinates are exclusive, which means that one can compute the width and height of the glyph image as:

```
width  = bbox.xMax - bbox.xMin
height = bbox.yMax - bbox.yMin
```

Note also that for 26.6 coordinates, if mode is set to *GLYPH_BBOX.GRIDFIT*, the coordinates will also be grid-fitted, which corresponds to:

```
bbox.xMin = floor(bbox.xMin)
bbox.yMin = floor(bbox.yMin)
bbox.xMax = ceil(bbox.xMax)
bbox.yMax = ceil(bbox.yMax)
```

To get the bbox in pixel coordinates, set mode to *GLYPH_BBOX.TRUNCATE*.

To get the bbox in grid-fitted pixel coordinates, set mode to *GLYPH_BBOX.PIXELS*.

freetypy.Glyph.render

Glyph.**render**()

Convert a given glyph image to a *Bitmap*. It does so by inspecting the glyph image format, finding the relevant renderer, and invoking it.

Parameters

- **render_mode** (*int, optional*) – This is the render mode used to render the glyph image into a bitmap.

  See *RENDER_MODE* for the available options.

- **origin** (*2-sequence of floats*) – The (x, y) origin to translate the glyph image before rendering.

Returns **bitmap** – The generated bitmap.

Return type *Bitmap*

Notes

The selected render mode only affects vector glyphs of a font. Embedded bitmaps often have a different pixel mode like *PIXEL_MODE.MONO*. You can use *Bitmap.convert* to transform them into 8-bit pixmaps.

Attributes

advance	The transformed advance width for the glyph (in fractional pixels).
bitmap	Get the *Bitmap* from this *Glyph* if it has been rendered.
bitmap_left	The bitmap's left bearing expressed in integer pixels.
bitmap_top	The bitmap's top bearing expressed in integer pixels.
face	The parent face object.
format	The format of the image in the glyph slot.
linear_hori_advance	The advance width of the unhinted glyph.
linear_vert_advance	The advance height of the unhinted glyph.
lsb_delta	The difference between hinted and unhinted left side bearing.
metrics	The *Glyph_Metrics* of the glyph.
outline	The outline descriptor for the current glyph image if its format is *GLYPH_FORMAT.OUTLINE*.
rsb_delta	The difference between hinted and unhinted right side bearing
subglyphs	A sequence of all of the subglyphs that make up the glyph.

freetypy.Glyph.advance

`Glyph.`**`advance`**

The transformed advance width for the glyph (in fractional pixels). May not be transformed, depending on the value of *LOAD.IGNORE_TRANSFORM*. As specified with *LOAD.VERTICAL_LAYOUT*, it uses either the `horiAdvance` or the `vertAdvance` value of *metrics* field.

freetypy.Glyph.bitmap

Glyph.**bitmap**

Get the *Bitmap* from this *Glyph* if it has been rendered.

freetypy.Glyph.bitmap_left

Glyph.**bitmap_left**

The bitmap's left bearing expressed in integer pixels. Of course, this is only valid if the format is *GLYPH_FORMAT.BITMAP*.

freetypy.Glyph.bitmap_top

Glyph.**bitmap_top**

The bitmap's top bearing expressed in integer pixels. Remember that this is the distance from the baseline to the top-most glyph scanline, upwards *y* coordinates being positive. Of course, this is only valid if the format is *GLYPH_FORMAT.BITMAP*.

freetypy.Glyph.face

Glyph.**face**

The parent face object.

freetypy.Glyph.format

Glyph.**format**

The format of the image in the glyph slot. Typically *GLYPH_FORMAT.BITMAP*, *GLYPH_FORMAT.OUTLINE*, or *GLYPH_FORMAT.COMPOSITE*, but others are possible.

See *GLYPH_FORMAT* for the set of available formats.

freetypy.Glyph.linear_hori_advance

Glyph.**linear_hori_advance**

The advance width of the unhinted glyph. Its value is expressed in fractional pixels, unless *LOAD.LINEAR_DESIGN* is set when loading the glyph. This field can be important to perform correct WYSIWYG layout. Only relevant for outline glyphs.

freetypy.Glyph.linear_vert_advance

Glyph.**linear_vert_advance**

The advance height of the unhinted glyph. Its value is expressed in fractional pixels, unless *LOAD.LINEAR_DESIGN* is set when loading the glyph. This field can be important to perform correct WYSIWYG layout. Only relevant for outline glyphs.

freetypy.Glyph.lsb_delta

`Glyph.`**`lsb_delta`**

The difference between hinted and unhinted left side bearing.

Valid when autohinting is active. Zero otherwise.

freetypy.Glyph.metrics

`Glyph.`**`metrics`**

The *`Glyph_Metrics`* of the glyph. The returned values depend on the last load flags (see the *`Face.load_glyph`* method) and can be expressed either in 26.6 fractional pixels or font units.

Note that even when the glyph image is transformed, the metrics are not.

freetypy.Glyph.outline

`Glyph.`**`outline`**

The outline descriptor for the current glyph image if its format is *`GLYPH_FORMAT.OUTLINE`*. Once a glyph is loaded, *`outline`* can be transformed, distorted, embolded, etc.

freetypy.Glyph.rsb_delta

`Glyph.`**`rsb_delta`**

The difference between hinted and unhinted right side bearing

Valid when autohinting is active. Zero otherwise.

freetypy.Glyph.subglyphs

`Glyph.`**`subglyphs`**

A sequence of all of the subglyphs that make up the glyph.

Empty unless *`format`* is *`GLYPH_FORMAT.COMPOSITE`*.

freetypy.Glyph_Metrics

class `freetypy.`**`Glyph_Metrics`**

Models the metrics of a single *`Glyph`*.

Notes

If not disabled with *`LOAD.NO_HINTING`*, the values represent dimensions of the hinted glyph (in case hinting is applicable).

Stroking a glyph with an outside border does not increase `horiAdvance` or `vertAdvance`; you have to manually adjust these values to account for the added width and height.

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

height	The glyph's height.
hori_advance	Advance width for horizontal layout.
hori_bearing_x	Left side bearing for horizontal layout.
hori_bearing_y	Top side bearing for horizontal layout.
vert_advance	Advance height for vertical layout.
vert_bearing_x	Left side bearing for vertical layout.
vert_bearing_y	Top side bearing for vertical layout.
width	The glyph's width.

freetypy.Glyph_Metrics.height

Glyph_Metrics.**height**

The glyph's height.

freetypy.Glyph_Metrics.hori_advance

Glyph_Metrics.**hori_advance**

Advance width for horizontal layout.

freetypy.Glyph_Metrics.hori_bearing_x

Glyph_Metrics.**hori_bearing_x**

Left side bearing for horizontal layout.

freetypy.Glyph_Metrics.hori_bearing_y

Glyph_Metrics.**hori_bearing_y**

Top side bearing for horizontal layout.

freetypy.Glyph_Metrics.vert_advance

Glyph_Metrics.**vert_advance**

Advance height for vertical layout. Positive values mean the glyph has a positive advance downward.

freetypy.Glyph_Metrics.vert_bearing_x

Glyph_Metrics.**vert_bearing_x**

Left side bearing for vertical layout.

freetypy.Glyph_Metrics.vert_bearing_y

Glyph_Metrics.**vert_bearing_y**

Top side bearing for vertical layout. Larger positive values mean further below the vertical glyph origin.

freetypy.Glyph_Metrics.width

Glyph_Metrics.**width**
The glyph's width.

freetypy.GLYPH_BBOX

class freetypy.**GLYPH_BBOX**
Indicates how the values of *Glyph.get_cbox* are returned.

- *UNSCALED*: Return unscaled font units.
- *SUBPIXELS*: Return unfitted fractional coordinates.
- *GRIDFIT*: Return grid-fitted factional coordinates.
- *TRUNCATE*: Return coordinates in integer pixels.
- *PIXELS*: Return grid-fitted pixel coordinates.

__init__()
x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

GRIDFIT
PIXELS
SUBPIXELS
TRUNCATE
UNSCALED

freetypy.GLYPH_BBOX.GRIDFIT

GLYPH_BBOX.**GRIDFIT** = **freetypy.GLYPH_BBOX.GRIDFIT**

freetypy.GLYPH_BBOX.PIXELS

GLYPH_BBOX.**PIXELS** = **freetypy.GLYPH_BBOX.PIXELS**

freetypy.GLYPH_BBOX.SUBPIXELS

GLYPH_BBOX.**SUBPIXELS** = **freetypy.GLYPH_BBOX.SUBPIXELS**

freetypy.GLYPH_BBOX.TRUNCATE

GLYPH_BBOX.**TRUNCATE** = **freetypy.GLYPH_BBOX.TRUNCATE**

freetypy.GLYPH_BBOX.UNSCALED

GLYPH_BBOX.**UNSCALED** = **freetypy.GLYPH_BBOX.SUBPIXELS**

freetypy.GLYPH_FORMAT

class `freetypy.`**`GLYPH_FORMAT`**

Describes the format of a given glyph image.

Note that this version of FreeType only supports two image formats, even though future font drivers will be able to register their own format.

- *`NONE`*: The value 0 is reserved.
- *`COMPOSITE`*: The glyph image is a composite of several other images. This format is only used with *`LOAD.NO_RECURSE`*, and is used to report compound glyphs (like accented characters).
- *`BITMAP`*: The glyph image is a bitmap, and can be described as an *`Bitmap`*.
- *`OUTLINE`*: The glyph image is a vectorial outline made of line segments and Bézier arcs; it can be described as an *`Outline`*.
- *`PLOTTER`*: The glyph image is a vectorial path with no inside and outside contours. Some Type 1 fonts, like those in the Hershey family, contain glyphs in this format. These are described as *`Outline`*, but FreeType isn't currently capable of rendering them correctly.

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

`BITMAP`
`COMPOSITE`
`NONE`
`OUTLINE`
`PLOTTER`

freetypy.GLYPH_FORMAT.BITMAP

`GLYPH_FORMAT.`**`BITMAP`** = **freetypy.GLYPH_FORMAT.BITMAP**

freetypy.GLYPH_FORMAT.COMPOSITE

`GLYPH_FORMAT.`**`COMPOSITE`** = **freetypy.GLYPH_FORMAT.COMPOSITE**

freetypy.GLYPH_FORMAT.NONE

`GLYPH_FORMAT.`**`NONE`** = **freetypy.GLYPH_FORMAT.NONE**

freetypy.GLYPH_FORMAT.OUTLINE

`GLYPH_FORMAT.`**`OUTLINE`** = **freetypy.GLYPH_FORMAT.OUTLINE**

freetypy.GLYPH_FORMAT.PLOTTER

`GLYPH_FORMAT.`**`PLOTTER`** = **freetypy.GLYPH_FORMAT.PLOTTER**

freetypy.SubGlyph

class `freetypy.SubGlyph`

A description of a given subglyph.

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

arg1	The subglyph's first argument (if any).
arg2	The subglyph's second argument (if any).
flags	The *SUBGLYPH_FLAG* bitfield.
index	The glyph index of the subglyph.
transform	The subglyph transformation (if any).

freetypy.SubGlyph.arg1

`SubGlyph.arg1`

The subglyph's first argument (if any).

freetypy.SubGlyph.arg2

`SubGlyph.arg2`

The subglyph's second argument (if any).

freetypy.SubGlyph.flags

`SubGlyph.flags`

The *SUBGLYPH_FLAG* bitfield.

freetypy.SubGlyph.index

`SubGlyph.index`

The glyph index of the subglyph.

freetypy.SubGlyph.transform

`SubGlyph.transform`

The subglyph transformation (if any).

freetypy.SUBGLYPH_FLAG

class `freetypy.SUBGLYPH_FLAG`

Describes subglyphs.

- *ARGS_ARE_WORDS*: If set, *SubGlyph.arg1* and *SubGlyph.arg2* are words, else bytes.
- *ARGS_ARE_XY_VALUES*: If set, *SubGlyph.arg1* and *SubGlyph.arg2* are xy values.

- *ROUND_XY_TO_GRID*: If set, round the xy values to grid.
- *SCALE*: If set, there is a simple scale for the component.
- *XY_SCALE*: If set, the x direction will use a different scale than the y direction.
- *TWOXTWO*: If set, there is a 2-by-2 transformation that will be used to scale the component. (This is named `2X2` in the underlying FreeType library, but was renamed here so it doesn't begin with a digit.)
- *USE_MY_METRICS*: Use metrics from this component for the compound glyph.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

ARGS_ARE_WORDS
ARGS_ARE_XY_VALUES
ROUND_XY_TO_GRID
SCALE
TWOXTWO
USE_MY_METRICS
XY_SCALE

freetypy.SUBGLYPH_FLAG.ARGS_ARE_WORDS

SUBGLYPH_FLAG.**ARGS_ARE_WORDS** = **freetypy.SUBGLYPH_FLAG.ARGS_ARE_WORDS**

freetypy.SUBGLYPH_FLAG.ARGS_ARE_XY_VALUES

SUBGLYPH_FLAG.**ARGS_ARE_XY_VALUES** = **freetypy.SUBGLYPH_FLAG.ARGS_ARE_XY_VALUES**

freetypy.SUBGLYPH_FLAG.ROUND_XY_TO_GRID

SUBGLYPH_FLAG.**ROUND_XY_TO_GRID** = **freetypy.SUBGLYPH_FLAG.ROUND_XY_TO_GRID**

freetypy.SUBGLYPH_FLAG.SCALE

SUBGLYPH_FLAG.**SCALE** = **freetypy.SUBGLYPH_FLAG.SCALE**

freetypy.SUBGLYPH_FLAG.TWOXTWO

SUBGLYPH_FLAG.**TWOXTWO** = **freetypy.SUBGLYPH_FLAG.TWOXTWO**

freetypy.SUBGLYPH_FLAG.USE_MY_METRICS

SUBGLYPH_FLAG.**USE_MY_METRICS** = **freetypy.SUBGLYPH_FLAG.USE_MY_METRICS**

freetypy.SUBGLYPH_FLAG.XY_SCALE

SUBGLYPH_FLAG.**XY_SCALE = freetypy.SUBGLYPH_FLAG.XY_SCALE**

Bitmap

Bitmap	Describes a bitmap raster.
Bitmap_Size	The metrics of a bitmap strike in a bitmap *Face*.
PIXEL_MODE	Constants related to the pixel mode of *Bitmap*.
RENDER_MODE	Selects a *Bitmap* renderer.

freetypy.Bitmap

class freetypy.**Bitmap**

Describes a bitmap raster.

Bitmap supports the Python buffer interface, so it is easy to convert it to a Numpy array. For example:

```
>>> import numpy as np
>>> a = np.asarray(bitmap)
```

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Methods

convert	Convert a *Bitmap* to 8 bits per pixel.
to_list	Convert the bitmap to a nested list.

freetypy.Bitmap.convert

Bitmap.**convert**()

Convert a *Bitmap* to 8 bits per pixel. Given a *Bitmap* with depth 1bpp, 2bpp, 4bpp, or 8bpp converts it to one with depth 8bpp, making the number of used bytes per line (a.k.a. the 'pitch') a multiple of alignment.

Parameters **alignment** (*int, optional*) – The pitch of the bitmap is a multiple of this parameter. Common values are 1, 2, or 4.

Returns **target** – The bitmap, converted to 8bpp.

Return type *Bitmap*

freetypy.Bitmap.to_list

Bitmap.**to_list**()

Convert the bitmap to a nested list.

Attributes

`buffer`	Get the bitmap's contents as a buffer.
`num_grays`	The number of gray levels used in the bitmap.
`pitch`	The number of bytes taken by one bitmap row.
`pixel_mode`	The `PIXEL_MODE`, i.e., how pixel bits are stored.
`rows`	The number of bitmap rows.
`width`	The number of pixels in bitmap row.

freetypy.Bitmap.buffer

Bitmap.**buffer**

Get the bitmap's contents as a buffer.

In most cases, the preferred method to get the data is to cast the `Bitmap` object to a memoryview, since that will also have size and type information.

freetypy.Bitmap.num_grays

Bitmap.**num_grays**

The number of gray levels used in the bitmap. This field is only used with `PIXEL_MODE.GRAY`.

freetypy.Bitmap.pitch

Bitmap.**pitch**

The number of bytes taken by one bitmap row.

Includes padding.

The pitch is positive when the bitmap has a 'down' flow, and negative when it has an 'up' flow. In all cases, the pitch is an offset to add to a bitmap pointer in order to go down one row.

Note that 'padding' means the alignment of a bitmap to a byte border, and FreeType functions normally align to the smallest possible integer value.

For the B/W rasterizer, `pitch` is always an even number.

To change the pitch of a bitmap (say, to make it a multiple of 4), use `Bitmap.convert`. Alternatively, you might use callback functions to directly render to the application's surface.

freetypy.Bitmap.pixel_mode

Bitmap.**pixel_mode**

The `PIXEL_MODE`, i.e., how pixel bits are stored.

freetypy.Bitmap.rows

Bitmap.**rows**

The number of bitmap rows.

freetypy.Bitmap.width

`Bitmap.width`

The number of pixels in bitmap row.

freetypy.Bitmap_Size

class `freetypy.Bitmap_Size`

The metrics of a bitmap strike in a bitmap *Face*.

It is used for *Face.available_sizes*.

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

height	The vertical distance, in pixels, between two consecutive baselines.
size	The nominal size of the strike in fractional points.
width	The average width, in pixels, of all glyphs in the strike.
x_ppem	The horizontal ppem (nominal width) in fractional pixels.
y_ppem	The vertical ppem (nominal height) in fractional pixels.

freetypy.Bitmap_Size.height

`Bitmap_Size.height`

The vertical distance, in pixels, between two consecutive baselines. It is always positive.

freetypy.Bitmap_Size.size

`Bitmap_Size.size`

The nominal size of the strike in fractional points. This field is not very useful.

freetypy.Bitmap_Size.width

`Bitmap_Size.width`

The average width, in pixels, of all glyphs in the strike.

freetypy.Bitmap_Size.x_ppem

`Bitmap_Size.x_ppem`

The horizontal ppem (nominal width) in fractional pixels.

freetypy.Bitmap_Size.y_ppem

`Bitmap_Size.y_ppem`

The vertical ppem (nominal height) in fractional pixels.

freetypy.PIXEL_MODE

class `freetypy.`**`PIXEL_MODE`**

Constants related to the pixel mode of *`Bitmap`*.

- *`MONO`*: A monochrome bitmap, using 1 bit per pixel. Note that pixels are stored in most-significant order (MSB), which means that the left-most pixel in a byte has value 128.
- *`GRAY`*: An 8-bit bitmap, generally used to represent anti-aliased glyph images. Each pixel is stored in one byte. Note that the number of 'gray' levels is stored in the 'num_grays' field of the Bitmap structure (it generally is 256).
- *`GRAY2`*: A 2-bit per pixel bitmap, used to represent embedded anti-aliased bitmaps in font files according to the OpenType specification. We haven't found a single font using this format, however.
- *`GRAY4`*: A 4-bit per pixel bitmap, representing embedded anti-aliased bitmaps in font files according to the OpenType specification. We haven't found a single font using this format, however.
- *`LCD`*: An 8-bit bitmap, representing RGB or BGR decimated glyph images used for display on LCD displays; the bitmap is three times wider than the original glyph image. See also *`RENDER_MODE.LCD`*. On many FreeType builds, this functionality will be disabled due to patent restrictions, in which case the resulting bitmap will be grayscale.
- *`LCD_V`*: An 8-bit bitmap, representing RGB or BGR decimated glyph images used for display on rotated LCD displays; the bitmap is three times taller than the original glyph image. See also *`RENDER_MODE.LCD_V`*. On many FreeType builds, this functionality will be disabled due to patent restrictions, in which case the resulting bitmap will be grayscale.

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

`GRAY`
`GRAY2`
`GRAY4`
`LCD`
`LCD_V`
`MONO`

freetypy.PIXEL_MODE.GRAY

`PIXEL_MODE.`**`GRAY`** **= freetypy.PIXEL_MODE.GRAY**

freetypy.PIXEL_MODE.GRAY2

`PIXEL_MODE.`**`GRAY2`** **= freetypy.PIXEL_MODE.GRAY2**

freetypy.PIXEL_MODE.GRAY4

`PIXEL_MODE.`**`GRAY4`** **= freetypy.PIXEL_MODE.GRAY4**

freetypy.PIXEL_MODE.LCD

`PIXEL_MODE.`**LCD** = **freetypy.PIXEL_MODE.LCD**

freetypy.PIXEL_MODE.LCD_V

`PIXEL_MODE.`**LCD_V** = **freetypy.PIXEL_MODE.LCD_V**

freetypy.PIXEL_MODE.MONO

`PIXEL_MODE.`**MONO** = **freetypy.PIXEL_MODE.MONO**

freetypy.RENDER_MODE

class `freetypy.`**RENDER_MODE**

Selects a *Bitmap* renderer.

- *NORMAL*: This is the default render mode; it corresponds to 8-bit anti-aliased bitmaps.
- *LIGHT*: This is equivalent to *RENDER_MODE.NORMAL*. It is only defined as a separate value because render modes are also used indirectly to define hinting algorithm selectors. See `LOAD.TARGET_XXX` for details.
- *MONO*: This mode corresponds to 1-bit bitmaps (with 2 levels of opacity).
- *LCD*: This mode corresponds to horizontal RGB and BGR sub-pixel displays like LCD screens. It produces 8-bit bitmaps that are 3 times the width of the original glyph outline in pixels, and which use the *PIXEL_MODE.LCD* mode. On many freetype builds, this functionality will be disabled due to patent restrictions, in which case the resulting bitmap will be grayscale.
- *LCD_V*: This mode corresponds to vertical RGB and BGR sub-pixel displays (like PDA screens, rotated LCD displays, etc.). It produces 8-bit bitmaps that are 3 times the height of the original glyph outline in pixels and use the *PIXEL_MODE.LCD_V* mode. On many freetype builds, this functionality will be disabled due to patent restrictions, in which case the resulting bitmap will be grayscale.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

LCD
LCD_V
LIGHT
MONO
NORMAL

freetypy.RENDER_MODE.LCD

`RENDER_MODE.`**LCD** = **freetypy.RENDER_MODE.LCD**

freetypy.RENDER_MODE.LCD_V

RENDER_MODE.**LCD_V** = **freetypy.RENDER_MODE.LCD_V**

freetypy.RENDER_MODE.LIGHT

RENDER_MODE.**LIGHT** = **freetypy.RENDER_MODE.LIGHT**

freetypy.RENDER_MODE.MONO

RENDER_MODE.**MONO** = **freetypy.RENDER_MODE.MONO**

freetypy.RENDER_MODE.NORMAL

RENDER_MODE.**NORMAL** = **freetypy.RENDER_MODE.NORMAL**

Outline

Outline	Represents an outline.
OUTLINE	Bit flags to characterize an *Outline*.
ORIENTATION	Specifies how an *Outline* is oriented.
CODES	Codes returned by *Outline.to_points_and_codes*:

freetypy.Outline

class freetypy.**Outline**

Represents an outline.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Methods

check	Check the contents of an outline descriptor.
decompose	Decompose the outline into individual segments and Bézier arcs.
embolden	Embolden an outline.
get_bbox	Compute the exact bounding box of an outline.
get_cbox	Get the control box.
get_orientation	This function analyzes a glyph outline and tries to compute its fill orientation.
reverse	Reverse the drawing direction of an outline.
to_points_and_codes	Convert the outline to a pair of arrays (points, codes).
to_string	Convert the outline to a text format string of commands.
transform	Apply a simple 2x2 matrix to all of an outline's points.

Continued on next page

Table 2.30 – continued from previous page

translate	Apply a simple translation to the points of an outline.

freetypy.Outline.check

`Outline.check()`

Check the contents of an outline descriptor.

Raises an exception if the outline has problems.

freetypy.Outline.decompose

`Outline.decompose()`

Decompose the outline into individual segments and Bézier arcs. This function also emits ‘move to’ operations to indicate the start of new contours in the outline.

Depending on the context, the values passed to the callback object will either be in 26.6, 16.16 or native font units.

Parameters

- **callback_object** (*object*) – A Python object containing methods that will be called for each element of the outline. The methods are:

  - `move_to(point)`
  - `line_to(point)`
  - `conic_to(control, endpoint)`
  - `cubic_to(control1, control2, endpoint)`

  If `callback_object` does not have a `conic_to` method, conic curves will be converted to cubic ones and `cubic_to` will be called.

- **shift** (*int, optional*) – The number of bits to shift coordinates before they are sent to the emitter.

- **delta** (*int, optional*) – The delta that is applied to coordinates before they are sent to the emitter, but after the shift.

  The transformation applied by shift and delta is:

  ```
  x' = (x << shift) - delta
  y' = (y << shift) - delta
  ```

Examples

```python
class Decomposer(object):
    def __init__(self):
        self.entries = []

    def move_to(self, point):
        self.entries.append(('move_to', point))

    def line_to(self, point):
        self.entries.append(('line_to', point))
```

```
    def conic_to(self, a, b):
        self.entries.append(('conic_to', a, b))

    def cubic_to(self, a, b, c):
        self.entries.append(('cubic_to', a, b, c))

face = ft.Face("Vera.ttf")
face.set_char_size(12, 12, 300, 300)
glyph = face.load_char(ord('B'))

d = Decomposer()
glyph.outline.decompose(d)
print(d.entries)
```

Notes

The point coordinates sent to the emitters are the transformed version of the original coordinates (this is important for high accuracy during scan-conversion). The transformation is simple:

```
x = (x << shift) - delta
y = (y << shift) - delta
```

freetypy.Outline.embolden

Outline.**embolden**()

Embolden an outline. The new outline will be at most 4 times `strength` pixels wider and higher. You may think of the left and bottom borders as unchanged.

Negative `strength` values to reduce the outline thickness are possible also.

Parameters **strength** (*float*) – How strong the glyph is emboldened.

Notes

The used algorithm to increase or decrease the thickness of the glyph doesn't change the number of points; this means that certain situations like acute angles or intersections are sometimes handled incorrectly.

If you need 'better' metrics values you should call *Outline.get_cbox* or *Outline.get_bbox*.

freetypy.Outline.get_bbox

Outline.**get_bbox**()

Compute the exact bounding box of an outline. This is slower than computing the control box. However, it uses an advanced algorithm which returns very quickly with the two boxes coincide. Otherwise, the outline Bézier arcs are traversed to extract their extrema.

Returns **bbox** – The outline's exact bounding box.

Return type *BBox*

Notes

If the font is tricky and the glyph has been loaded with *LOAD.NO_SCALE*, the resulting *BBox* is meaningless. To get reasonable values for the *BBox* it is necessary to load the glyph at a large ppem value (so that the hinting instructions can properly shift and scale the subglyphs), then extracting the *BBox* which can be eventually converted back to font units.

freetypy.Outline.get_cbox

Outline.**get_cbox**()

Get the control box. The control box encloses all the outline's points, including Bézier control points. Though it coincides with the exact bounding box for most glyphs, it can be slightly larger in some situations (like when rotating an outline which contains Bézier outside arcs).

Computing the control box is very fast, while getting the bounding box can take much more time as it needs to walk over all segments and arcs in the outline.

See *Glyph.get_cbox* for a discussion of tricky fonts.

freetypy.Outline.get_orientation

Outline.**get_orientation**()

This function analyzes a glyph outline and tries to compute its fill orientation. This is done by computing the direction of each global horizontal and/or vertical extrema within the outline.

See *ORIENTATION* for a list of the result values.

Note that this will return *ORIENTATION.TRUETYPE* for empty outlines.

freetypy.Outline.reverse

Outline.**reverse**()

Reverse the drawing direction of an outline. This is used to ensure consistent fill conventions for mirrored glyphs.

Notes

The function toggles the *OUTLINE.REVERSE_FILL* bit flag in *Outline.flags*.

It shouldn't be used by a normal client application, unless it knows what it is doing.

freetypy.Outline.to_points_and_codes

Outline.**to_points_and_codes**()

Convert the outline to a pair of arrays (points, codes).

- points is an Nx2 array of floats for each point
- codes in a length N array of *CODES* constants:
  - 0: `MOVETO` (1 point)
  - 1: `LINETO` (1 point)
  - 2: `CUBIC` (2 points)

–3: `CONIC` (3 points)

Returns arrays – A (points, codes) pair

Return type tuple

freetypy.Outline.to_string

`Outline.`**`to_string`**`()`

Convert the outline to a text format string of commands. This function is flexible enough to create path commands for PDF, Postcript and SVG.

Parameters

- **`move_command`** (*`bytes`*) – The character or command to use for "move to" commands.
- **`line_command`** (*`bytes`*) – The character or command to use for "line to" commands.
- **`cubic_command`** (*`bytes`*) – The character or command to use for "cubic curve" commands.
- **`conic_command`** (*`bytes, optional`*) – The character or command to use for "conic curve" commands. If one is not provided, conic curves will be implicitly converted to cubic curves.
- **`relative`** (*`bool, optional`*) –

prefix [bool, optional] If `True`, the command will appear before the points it refers to. Otherwise, the default is for them to appear after.

Returns string – A text-based string of commands to render the character.

Return type bytes

Examples

To generate a PDF-compatible path:

```
outline.to_string(" m ", " l ", " c ")
```

freetypy.Outline.transform

`Outline.`**`transform`**`()`

Apply a simple 2x2 matrix to all of an outline's points. Useful for applying rotations, slanting, flipping, etc.

Parameters matrix (*`2x2 sequence of floats, optional`*) – A matrix of the form:

```
. [[xx, xy]
.  [yx, yy]]
```

freetypy.Outline.translate

`Outline.translate()`

Apply a simple translation to the points of an outline.

Parameters

- **xOffset** (*int*) – The horizontal offset.
- **yOffset** (*int*) – The vertical offset.

Attributes

contours	A `memoryview` giving the end point of each contour within the outline.
flags	A set of *OUTLINE* flags used to characterize the outline.
n_contours	The number of contours in the outline.
n_points	The number of points in the outline.
points	A `memoryview` of points in the outline.
tags	A `memoryview` of *n_points* values, giving each outline point's type.

freetypy.Outline.contours

`Outline.contours`

A `memoryview` giving the end point of each contour within the outline. For example, the first contour is defined by the points `0` to `contours[0]`, the second one is defined by the points `contours[0] + 1` to `contours[1]`, etc.

If using numpy, you may want to use this as a Numpy array as follows:

```
>>> import numpy as np
>>> a = np.asarray(outline.contours)
```

freetypy.Outline.flags

`Outline.flags`

A set of *OUTLINE* flags used to characterize the outline. Gives hints to the scan-converter and hinter on how to convert/grid-fit it.

freetypy.Outline.n_contours

`Outline.n_contours`

The number of contours in the outline.

freetypy.Outline.n_points

`Outline.n_points`

The number of points in the outline.

freetypy.Outline.points

`Outline.points`

A `memoryview` of points in the outline. Each entry is an (x, y) pair.

freetypy.Outline.tags

`Outline.tags`

A `memoryview` of *n_points* values, giving each outline point's type.

- If bit 0 is unset, the point is 'off' the curve, i.e., a Bézier control point, while it is 'on' if set.
- Bit 1 is meaningful for 'off' points only. If set, it indicates a third-order Bézier arc control point; and a second-order control point if unset.
- If bit 2 is set, bits 5-7 contain the drop-out mode (as defined in the OpenType specification; the value is the same as the argument to the `SCANMODE` instruction).
- Bits 3 and 4 are reserved for internal purposes.

freetypy.OUTLINE

class `freetypy.OUTLINE`

Bit flags to characterize an *Outline*.

- *NONE*: Value 0 is reserved.
- *OWNER*: If set, this flag indicates that the outline's field arrays (i.e., *points*, *flags*, and *contours*) are 'owned' by the outline object, and should thus be freed when it is destroyed.
- *EVEN_ODD_FILL*: By default, outlines are filled using the non-zero winding rule. If set to 1, the outline will be filled using the even-odd fill rule (only works with the smooth rasterizer).
- *REVERSE_FILL*: By default, outside contours of an outline are oriented in clock-wise direction, as defined in the TrueType specification. This flag is set if the outline uses the opposite direction (typically for Type 1 fonts). This flag is ignored by the scan converter.
- *IGNORE_DROPOUTS*: By default, the scan converter will try to detect drop-outs in an outline and correct the glyph bitmap to ensure consistent shape continuity. If set, this flag hints the scan-line converter to ignore such cases. See below for more information.
- *SMART_DROPOUTS*: Select smart dropout control. If unset, use simple dropout control. Ignored if *OUTLINE.IGNORE_DROPOUTS* is set. See below for more information.
- *INCLUDE_STUBS*: If set, turn pixels on for 'stubs', otherwise exclude them. Ignored if *OUTLINE.IGNORE_DROPOUTS* is set. See below for more information.
- *HIGH_PRECISION*: This flag indicates that the scan-line converter should try to convert this outline to bitmaps with the highest possible quality. It is typically set for small character sizes. Note that this is only a hint that might be completely ignored by a given scan-converter.
- *SINGLE_PASS*: This flag is set to force a given scan-converter to only use a single pass over the outline to render a bitmap glyph image. Normally, it is set for very large character sizes. It is only a hint that might be completely ignored by a given scan-converter.

Notes

The flags *OUTLINE.IGNORE_DROPOUTS*, *OUTLINE.SMART_DROPOUTS*, and *OUTLINE.INCLUDE_STUBS* are ignored by the smooth rasterizer.

There exists a second mechanism to pass the drop-out mode to the B/W rasterizer; see the *Outline.tags*.

Please refer to the description of the 'SCANTYPE' instruction in the OpenType specification (in file 'ttinst1.doc') how simple drop-outs, smart drop-outs, and stubs are defined.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

EVEN_ODD_FILL
HIGH_PRECISION
IGNORE_DROPOUTS
INCLUDE_STUBS
NONE
OWNER
REVERSE_FILL
SINGLE_PASS
SMART_DROPOUTS

freetypy.OUTLINE.EVEN_ODD_FILL

OUTLINE.**EVEN_ODD_FILL = freetypy.OUTLINE.EVEN_ODD_FILL**

freetypy.OUTLINE.HIGH_PRECISION

OUTLINE.**HIGH_PRECISION = freetypy.OUTLINE.HIGH_PRECISION**

freetypy.OUTLINE.IGNORE_DROPOUTS

OUTLINE.**IGNORE_DROPOUTS = freetypy.OUTLINE.IGNORE_DROPOUTS**

freetypy.OUTLINE.INCLUDE_STUBS

OUTLINE.**INCLUDE_STUBS = freetypy.OUTLINE.INCLUDE_STUBS**

freetypy.OUTLINE.NONE

OUTLINE.**NONE = freetypy.OUTLINE.NONE**

freetypy.OUTLINE.OWNER

OUTLINE.**OWNER = freetypy.OUTLINE.OWNER**

freetypy.OUTLINE.REVERSE_FILL

`OUTLINE.`**`REVERSE_FILL`** = **freetypy.OUTLINE.REVERSE_FILL**

freetypy.OUTLINE.SINGLE_PASS

`OUTLINE.`**`SINGLE_PASS`** = **freetypy.OUTLINE.SINGLE_PASS**

freetypy.OUTLINE.SMART_DROPOUTS

`OUTLINE.`**`SMART_DROPOUTS`** = **freetypy.OUTLINE.SMART_DROPOUTS**

freetypy.ORIENTATION

class `freetypy.`**`ORIENTATION`**

Specifies how an *`Outline`* is oriented.

- *`TRUETYPE`*: According to the TrueType specification, clockwise contours must be filled, and counter-clockwise ones must be unfilled.
- *`POSTSCRIPT`*: According to the PostScript specification, counter-clockwise contours must be filled, and clockwise ones must be unfilled.
- *`FILL_RIGHT`*: This is identical to *`ORIENTATION.TRUETYPE`*, but is used to remember that in TrueType, everything that is to the right of the drawing direction of a contour must be filled.
- *`FILL_LEFT`*: This is identical to *`ORIENTATION.POSTSCRIPT`*, but is used to remember that in PostScript, everything that is to the left of the drawing direction of a contour must be filled.
- *`NONE`*: The orientation cannot be determined. That is, different parts of the glyph have different orientation.

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

`FILL_LEFT`
`FILL_RIGHT`
`NONE`
`POSTSCRIPT`
`TRUETYPE`

freetypy.ORIENTATION.FILL_LEFT

`ORIENTATION.`**`FILL_LEFT`** = **freetypy.ORIENTATION.FILL_LEFT**

freetypy.ORIENTATION.FILL_RIGHT

`ORIENTATION.`**`FILL_RIGHT`** = **freetypy.ORIENTATION.FILL_RIGHT**

freetypy.ORIENTATION.NONE

ORIENTATION.**NONE** = **freetypy.ORIENTATION.NONE**

freetypy.ORIENTATION.POSTSCRIPT

ORIENTATION.**POSTSCRIPT** = **freetypy.ORIENTATION.FILL_LEFT**

freetypy.ORIENTATION.TRUETYPE

ORIENTATION.**TRUETYPE** = **freetypy.ORIENTATION.FILL_RIGHT**

freetypy.CODES

class freetypy.**CODES**

Codes returned by *Outline.to_points_and_codes*:

- 1: *MOVETO* (1 point)
- 2: *LINETO* (1 point)
- 3: *CONIC* (2 points)
- 4: *CUBIC* (3 points)

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

CONIC
CUBIC
LINETO
MOVETO
STOP

freetypy.CODES.CONIC

CODES.**CONIC** = **freetypy.CODES.CONIC**

freetypy.CODES.CUBIC

CODES.**CUBIC** = **freetypy.CODES.CUBIC**

freetypy.CODES.LINETO

CODES.**LINETO** = **freetypy.CODES.LINETO**

freetypy.CODES.MOVETO

CODES.**MOVETO** = **freetypy.CODES.MOVETO**

freetypy.CODES.STOP

CODES.**STOP** = **freetypy.CODES.STOP**

CharMap

CharMap	A mapping from code points to glyph identifiers.
ENCODING	A tag identifying a *CharMap* type.

freetypy.CharMap

class freetypy.**CharMap**

A mapping from code points to glyph identifiers.

Some font formats may provide several *CharMap* instances per font.

Each face object owns zero or more charmaps, but only one of them can be 'active' and used by *Face.get_char_index* or *Face.load_char*.

The list of available charmaps in a face is available through *Face.charmaps*.

The currently active charmap is available as *Face.charmap*.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Methods

get_format	Get the TrueType/sfnt specific cmap format.
get_language_id	Get the TrueType/sfnt specific cmap language ID.

freetypy.CharMap.get_format

CharMap.**get_format**()

Get the TrueType/sfnt specific cmap format.

Returns format – The format of 'charmap'. If 'charmap' doesn't belong to a TrueType/sfnt face, raises a ValueError.

Return type int

freetypy.CharMap.get_language_id

CharMap.**get_language_id**()

Get the TrueType/sfnt specific cmap language ID.

It will be one of the constants in *TT_MAC_LANGID* or *TT_MS_LANGID*.

Returns

language_id – The language ID of the *CharMap*. If it doesn't belong to a TrueType/sfnt face, raises a `ValueError`.

For a format 14 cmap (to access Unicode IVS), the return value is 0xFFFFFFFF.

Return type int

Attributes

encoding	An *ENCODING* tag identifying the charmap.
encoding_id	A platform-specific encoding number.
face	The parent *Face* object.
platform_id	An *TT_PLATFORM* id describing the platform for the following encoding ID.

freetypy.CharMap.encoding

`CharMap.`**`encoding`**

An *ENCODING* tag identifying the charmap.

Despite the name, this value specifies a specific character repertory (i.e., charset), and not a text encoding method (e.g., UTF-8, UTF-16, etc.).

freetypy.CharMap.encoding_id

`CharMap.`**`encoding_id`**

A platform-specific encoding number. This also comes from the TrueType specification and should be emulated similarly.

Depending on *platform_id*, this will be from *TT_APPLE_ID*, *TT_MAC_ID*, *TT_MS_ID* or *TT_ADOBE_ID*.

freetypy.CharMap.face

`CharMap.`**`face`**

The parent *Face* object.

freetypy.CharMap.platform_id

`CharMap.`**`platform_id`**

An *TT_PLATFORM* id describing the platform for the following encoding ID. This comes directly from the TrueType specification and should be emulated for other formats.

freetypy.ENCODING

class `freetypy.`**`ENCODING`**

A tag identifying a *CharMap* type.

- *NONE*: The encoding value 0 is reserved.

- *UNICODE*: Corresponds to the Unicode character set. This value covers all versions of the Unicode repertoire, including ASCII and Latin-1. Most fonts include a Unicode charmap, but not all of them.

  For example, if you want to access Unicode value U+1F028 (and the font contains it), use value 0x1F028 as the input value for *Face.get_char_index*.

- *MS_SYMBOL*: Corresponds to the Microsoft Symbol encoding, used to encode mathematical symbols in the 32..255 character code range. For more information, see http://www.ceviz.net/symbol.htm.
- *SJIS*: Corresponds to Japanese SJIS encoding. More info at at http://langsupport.japanreference.com/encoding.shtml. See note on multi-byte encodings below.
- *GB2312*: Corresponds to an encoding system for Simplified Chinese as used used in mainland China.
- *BIG5*: Corresponds to an encoding system for Traditional Chinese as used in Taiwan and Hong Kong.
- *WANSUNG*: Corresponds to the Korean encoding system known as Wansung. For more information see http://www.microsoft.com/typography/unicode/949.txt.
- *JOHAB*: The Korean standard character set (KS C 5601-1992), which corresponds to MS Windows code page 1361. This character set includes all possible Hangeul character combinations.
- *ADOBE_LATIN_1*: Corresponds to a Latin-1 encoding as defined in a Type 1 PostScript font. It is limited to 256 character codes.
- *ADOBE_STANDARD*: Corresponds to the Adobe Standard encoding, as found in Type 1, CFF, and OpenType/CFF fonts. It is limited to 256 character codes.
- *ADOBE_EXPERT*: Corresponds to the Adobe Expert encoding, as found in Type 1, CFF, and OpenType/CFF fonts. It is limited to 256 character codes.
- *ADOBE_CUSTOM*: Corresponds to a custom encoding, as found in Type 1, CFF, and OpenType/CFF fonts. It is limited to 256 character codes.
- *APPLE_ROMAN*: Corresponds to the 8-bit Apple roman encoding. Many TrueType and OpenType fonts contain a charmap for this encoding, since older versions of Mac OS are able to use it.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

ADOBE_CUSTOM
ADOBE_EXPERT
ADOBE_LATIN_1
ADOBE_STANDARD
APPLE_ROMAN
BIG5
GB2312
JOHAB
MS_SYMBOL
NONE
SJIS
UNICODE
WANSUNG

freetypy.ENCODING.ADOBE_CUSTOM

`ENCODING.`**`ADOBE_CUSTOM`** = freetypy.ENCODING.ADOBE_CUSTOM

freetypy.ENCODING.ADOBE_EXPERT

`ENCODING.`**`ADOBE_EXPERT`** = freetypy.ENCODING.ADOBE_EXPERT

freetypy.ENCODING.ADOBE_LATIN_1

`ENCODING.`**`ADOBE_LATIN_1`** = freetypy.ENCODING.ADOBE_LATIN_1

freetypy.ENCODING.ADOBE_STANDARD

`ENCODING.`**`ADOBE_STANDARD`** = freetypy.ENCODING.ADOBE_STANDARD

freetypy.ENCODING.APPLE_ROMAN

`ENCODING.`**`APPLE_ROMAN`** = freetypy.ENCODING.APPLE_ROMAN

freetypy.ENCODING.BIG5

`ENCODING.`**`BIG5`** = freetypy.ENCODING.BIG5

freetypy.ENCODING.GB2312

`ENCODING.`**`GB2312`** = freetypy.ENCODING.GB2312

freetypy.ENCODING.JOHAB

`ENCODING.`**`JOHAB`** = freetypy.ENCODING.JOHAB

freetypy.ENCODING.MS_SYMBOL

`ENCODING.`**`MS_SYMBOL`** = freetypy.ENCODING.MS_SYMBOL

freetypy.ENCODING.NONE

`ENCODING.`**`NONE`** = freetypy.ENCODING.NONE

freetypy.ENCODING.SJIS

`ENCODING.`**`SJIS`** = freetypy.ENCODING.SJIS

freetypy.ENCODING.UNICODE

ENCODING.**UNICODE** = **freetypy.ENCODING.UNICODE**

freetypy.ENCODING.WANSUNG

ENCODING.**WANSUNG** = **freetypy.ENCODING.WANSUNG**

TrueType information

SfntName	An SFNT 'name' table entry.
SfntNames	A sequence of SFNT names in a *Face*.
TT_Header	TrueType header.
TT_HoriHeader	TrueType horizontal header ('hhea').
TT_VertHeader	TrueType vertical header ('vhea').
TT_OS2	Inf about the TrueType font, used on OS/2 and Microsoft Windows.
TT_Pclt	TrueType PCLT table.
TT_Postscript	TrueType PostScript table.
TT_PLATFORM	Platform identifier codes.
TT_APPLE_ID	Apple-specific encoding values.
TT_ISO_ID	Standard ISO encodings.
TT_MAC_ID	Macintosh-specific encoding values.
TT_MAC_LANGID	Language identifier.
TT_MS_ID	Microsoft-specific encoding values.
TT_MS_LANGID	Language identifier.
TT_ADOBE_ID	Adobe-specific encoding values.
TT_NAME_ID	The type of value stored in a *SfntName* record.
TT_MAC_STYLE	Bit flags for the style of the face.
TT_HEADER_FLAGS	Bit flags for global information about a *Face*.
TT_WIDTH_CLASS	Width values for the *TT_OS2.width_class* property.
TT_WEIGHT_CLASS	Weight values for the *TT_OS2.weight_class* property.
TT_FS_SELECTION	Bit flag for font style.

freetypy.SfntName

class freetypy.**SfntName**

An SFNT 'name' table entry.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

encoding_id	The encoding ID.
language_id	The language ID.

Continued on next page

Table 2.40 – continued from previous page

name_id	The type of value stored in *string*.
platform_id	The platform ID.
string	The value of the *SfntName* entry.
string_bytes	The raw undecoded bytes of the value of the *SfntName* entry.

freetypy.SfntName.encoding_id

SfntName.**encoding_id**

The encoding ID.

Depending on the value of *platform_id*, this will be one of the constants in *TT_APPLE_ID*, *TT_MAC_ID*, *TT_ISO_ID*, *TT_MS_ID* or *TT_ADOBE_ID*.

freetypy.SfntName.language_id

SfntName.**language_id**

The language ID.

Depending on the value of *platform_id*, this will be one of the constants in *TT_MAC_LANGID* or *TT_MS_LANGID*.

freetypy.SfntName.name_id

SfntName.**name_id**

The type of value stored in *string*. See *TT_NAME_ID* for the possible values.

freetypy.SfntName.platform_id

SfntName.**platform_id**

The platform ID. One of *TT_PLATFORM*.

freetypy.SfntName.string

SfntName.**string**

The value of the *SfntName* entry.

: Unlike the raw C interface, this value is always decoded into a Unicode string, so users do not need to worry about its encoding. To get the undecoded bytes, use *string_bytes*.

freetypy.SfntName.string_bytes

SfntName.**string_bytes**

The raw undecoded bytes of the value of the *SfntName* entry.

: Most of the time, you will want to use *string* instead, which automatically handles the decoding of the bytes.

freetypy.SfntNames

class `freetypy.SfntNames`

A sequence of SFNT names in a *Face*.

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Methods

get_name	Get the best available SFNT name of a given type.

freetypy.SfntNames.get_name

`SfntNames.get_name()`

Get the best available SFNT name of a given type.

This searches the SFNT names to first find a Unicode version of the name, if available, otherwise returning the first found name of the given type. Raises `KeyError` if no name of the given type was found.

Parameters **name** (*int*) – A *TT_NAME_ID* constant indicating the type of SFNT name to return.

Returns **sfnt_name**

Return type *SfntName* object

freetypy.TT_Header

class `freetypy.TT_Header`

TrueType header.

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

checksum_adjust	A checksum for the header.
created	The date the font was created.
flags	*TT_HEADER_FLAGS* bitflags that apply to the face as a whole.
font_direction	The direction of the font.
font_revision	The version of the font.
lowest_rec_ppem	The smallest readable size in pixels.
mac_style	*TT_MAC_STYLE* bitflags.
magic_number	Always 0x5F0F3CF5.
modified	The date the font was modified.
table_version	The version of the TrueType header.
units_per_em	Units per em.
x_max	The maximum x for all bounding boxes.

Continued on next page

Table 2.42 – continued from previous page

x_min	The minimum x for all bounding boxes.
y_max	The maximum y for all bounding boxes.
y_min	The minimum y for all bounding boxes.

freetypy.TT_Header.checksum_adjust

`TT_Header.`**`checksum_adjust`**

A checksum for the header.

freetypy.TT_Header.created

`TT_Header.`**`created`**

The date the font was created.

freetypy.TT_Header.flags

`TT_Header.`**`flags`**

TT_HEADER_FLAGS bitflags that apply to the face as a whole.

freetypy.TT_Header.font_direction

`TT_Header.`**`font_direction`**

The direction of the font.

- 0: Mixed directional glyphs
- 1: Only strong left to right glyphs
- 2: Like 1, but also contains neutrals
- -1: Only strong right to left glyphs
- -2: Like -1, but also contains neutrals

freetypy.TT_Header.font_revision

`TT_Header.`**`font_revision`**

The version of the font. Set by the font manufacturer.

freetypy.TT_Header.lowest_rec_ppem

`TT_Header.`**`lowest_rec_ppem`**

The smallest readable size in pixels.

freetypy.TT_Header.mac_style

`TT_Header.`**`mac_style`**

TT_MAC_STYLE bitflags.

freetypy.TT_Header.magic_number

`TT_Header.magic_number`

Always 0x5F0F3CF5.

freetypy.TT_Header.modified

`TT_Header.modified`

The date the font was modified.

freetypy.TT_Header.table_version

`TT_Header.table_version`

The version of the TrueType header.

freetypy.TT_Header.units_per_em

`TT_Header.units_per_em`

Units per em. In the range 64 - 16384.

freetypy.TT_Header.x_max

`TT_Header.x_max`

The maximum x for all bounding boxes.

freetypy.TT_Header.x_min

`TT_Header.x_min`

The minimum x for all bounding boxes.

freetypy.TT_Header.y_max

`TT_Header.y_max`

The maximum y for all bounding boxes.

freetypy.TT_Header.y_min

`TT_Header.y_min`

The minimum y for all bounding boxes.

freetypy.TT_HoriHeader

class `freetypy.TT_HoriHeader`

TrueType horizontal header (‘hhea’).

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

advance_width_max	This field is the maximum of all advance widths found in the font.
ascender	The font's ascender, i.e., the distance from the baseline to the top-most of all glyph points found in the font.
caret_offset	
caret_slope_rise	The rise coefficient of the cursor's slope of the cursor (slope=rise/run).
caret_slope_run	The run coefficient of the cursor's slope.
descender	The font's descender, i.e., the distance from the baseline to the bottom-most of all glyph points found in the font.
line_gap	The font's line gap, i.e., the distance to add to the ascender and descender to get the BTB, i.e., the baseline-to-baseline distance for the font.
metric_data_format	Always 0.
min_left_side_bearing	The minimum left side bearing of all glyphs within the font.
min_right_side_bearing	The minimum right side bearing of all glyphs within the font.
number_of_hmetrics	Number of HMetrics entries in the 'hmtx' table – this value can be smaller than the total number of glyphs in the font.
version	The table version.
xmax_extent	The maximum horizontal extent (i.e., the 'width' of a glyph's bounding box) for all glyphs in the font.

freetypy.TT_HoriHeader.advance_width_max

`TT_HoriHeader.`**`advance_width_max`**

This field is the maximum of all advance widths found in the font. It can be used to compute the maximum width of an arbitrary string of text.

freetypy.TT_HoriHeader.ascender

`TT_HoriHeader.`**`ascender`**

The font's ascender, i.e., the distance from the baseline to the top-most of all glyph points found in the font.

This value is invalid in many fonts, as it is usually set by the font designer, and often reflects only a portion of the glyphs found in the font (maybe ASCII).

You should use the `Face.get_os2_table().s_typo_ascender` instead if you want the correct one.

freetypy.TT_HoriHeader.caret_offset

`TT_HoriHeader.`**`caret_offset`**

freetypy.TT_HoriHeader.caret_slope_rise

`TT_HoriHeader.`**`caret_slope_rise`**

The rise coefficient of the cursor's slope of the cursor (slope=rise/run).

freetypy.TT_HoriHeader.caret_slope_run

`TT_HoriHeader.`**`caret_slope_run`**

The run coefficient of the cursor's slope.

freetypy.TT_HoriHeader.descender

`TT_HoriHeader.`**`descender`**

The font's descender, i.e., the distance from the baseline to the bottom-most of all glyph points found in the font. It is negative.

This value is invalid in many fonts, as it is usually set by the font designer, and often reflects only a portion of the glyphs found in the font (maybe ASCII).

You should use the `Face.get_os2_table().s_typo_descender` if you want the correct one.

freetypy.TT_HoriHeader.line_gap

`TT_HoriHeader.`**`line_gap`**

The font's line gap, i.e., the distance to add to the ascender and descender to get the BTB, i.e., the baseline-to-baseline distance for the font.

freetypy.TT_HoriHeader.metric_data_format

`TT_HoriHeader.`**`metric_data_format`**

Always 0.

freetypy.TT_HoriHeader.min_left_side_bearing

`TT_HoriHeader.`**`min_left_side_bearing`**

The minimum left side bearing of all glyphs within the font.

freetypy.TT_HoriHeader.min_right_side_bearing

`TT_HoriHeader.`**`min_right_side_bearing`**

The minimum right side bearing of all glyphs within the font.

freetypy.TT_HoriHeader.number_of_hmetrics

`TT_HoriHeader.`**`number_of_hmetrics`**

Number of HMetrics entries in the 'hmtx' table – this value can be smaller than the total number of glyphs in the font.

freetypy.TT_HoriHeader.version

`TT_HoriHeader.version`

The table version.

freetypy.TT_HoriHeader.xmax_extent

`TT_HoriHeader.xmax_extent`

The maximum horizontal extent (i.e., the ‘width’ of a glyph’s bounding box) for all glyphs in the font.

freetypy.TT_VertHeader

class `freetypy.TT_VertHeader`

TrueType vertical header (‘vhea’).

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

advance_height_max	This field is the maximum of all advance heights found in the font.
ascender	The font’s ascender, i.e., the distance from the baseline to the top-most of all glyph points found in the font.
caret_offset	
caret_slope_rise	The rise coefficient of the cursor’s slope of the cursor (slope=rise/run).
caret_slope_run	The run coefficient of the cursor’s slope.
descender	The font’s descender, i.e., the distance from the baseline to the bottom-most of all glyph points found in the font.
line_gap	The font’s line gap, i.e., the distance to add to the ascender and descender to get the BTB, i.e., the baseline-to-baseline distance for the font.
metric_data_format	Always 0.
min_bottom_side_bearing	The minimum bottom side bearing of all glyphs within the font.
min_top_side_bearing	The minimum top side bearing of all glyphs within the font.
number_of_vmetrics	Number of VMetrics entries in the ‘hmtx’ table – this value can be smaller than the total number of glyphs in the font.
version	The table version.
ymax_extent	The maximum vertical extent (i.e., the ‘height’ of a glyph’s bounding box) for all glyphs in the font.

freetypy.TT_VertHeader.advance_height_max

`TT_VertHeader.advance_height_max`

This field is the maximum of all advance heights found in the font. It can be used to compute the maximum height of an arbitrary string of text.

freetypy.TT_VertHeader.ascender

`TT_VertHeader.`**`ascender`**

The font's ascender, i.e., the distance from the baseline to the top-most of all glyph points found in the font.

This value is invalid in many fonts, as it is usually set by the font designer, and often reflects only a portion of the glyphs found in the font (maybe ASCII).

You should use the `Face.get_os2_table().sTypoAscender` instead if you want the correct one.

freetypy.TT_VertHeader.caret_offset

`TT_VertHeader.`**`caret_offset`**

freetypy.TT_VertHeader.caret_slope_rise

`TT_VertHeader.`**`caret_slope_rise`**

The rise coefficient of the cursor's slope of the cursor (slope=rise/run).

freetypy.TT_VertHeader.caret_slope_run

`TT_VertHeader.`**`caret_slope_run`**

The run coefficient of the cursor's slope.

freetypy.TT_VertHeader.descender

`TT_VertHeader.`**`descender`**

The font's descender, i.e., the distance from the baseline to the bottom-most of all glyph points found in the font. It is negative.

This value is invalid in many fonts, as it is usually set by the font designer, and often reflects only a portion of the glyphs found in the font (maybe ASCII).

You should use the `Face.get_os2_table().sTypoDescender` if you want the correct one.

freetypy.TT_VertHeader.line_gap

`TT_VertHeader.`**`line_gap`**

The font's line gap, i.e., the distance to add to the ascender and descender to get the BTB, i.e., the baseline-to-baseline distance for the font.

freetypy.TT_VertHeader.metric_data_format

`TT_VertHeader.`**`metric_data_format`**

Always 0.

freetypy.TT_VertHeader.min_bottom_side_bearing

`TT_VertHeader.`**`min_bottom_side_bearing`**

The minimum bottom side bearing of all glyphs within the font.

freetypy.TT_VertHeader.min_top_side_bearing

TT_VertHeader.**min_top_side_bearing**
The minimum top side bearing of all glyphs within the font.

freetypy.TT_VertHeader.number_of_vmetrics

TT_VertHeader.**number_of_vmetrics**
Number of VMetrics entries in the 'hmtx' table – this value can be smaller than the total number of glyphs in the font.

freetypy.TT_VertHeader.version

TT_VertHeader.**version**
The table version.

freetypy.TT_VertHeader.ymax_extent

TT_VertHeader.**ymax_extent**
The maximum vertical extent (i.e., the 'height' of a glyph's bounding box) for all glyphs in the font.

freetypy.TT_OS2

class freetypy.**TT_OS2**
Inf about the TrueType font, used on OS/2 and Microsoft Windows.

Note that we now support old Mac fonts which do not include an OS/2 table. In this case, the *version* field is always set to 0xFFFF.

__init__()
x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

break_char	This is the Unicode encoding of the glyph that Windows uses as the break character.
cap_height	This metric specifies the distance between the baseline and the approximate height of uppercase letters measured in font units.
default_char	Whenever a request is made for a character that is not in the font, Windows provides this default character.
family_class	Classifies a font design as to its appearance.
family_subclass	Classifies a font design as to its appearance.
first_char_index	The minimum Unicode index in this font.
last_char_index	The maximum Unicode index in this font.
max_context	The maximum length of a target glyph context for any feature in this font.
panose	The PANOSE classification number.
selection	A *TT_FS_SELECTION* bitflag.

Continued on next page

Table 2.45 – continued from previous page

type	*FSTYPE* bitflags indicating the licensing restrictions on a font.
typo_ascender	The typographic ascender for this font.
typo_descender	The typographic descender for this font.
typo_line_gap	The typographic line gap for this font.
vend_id	Indentifies the font vendor.
version	The version of the *TT_OS2* table.
weight_class	A *TT_WEIGHT_CLASS* value.
width_class	A *TT_WIDTH_CLASS* value.
win_ascent	The ascender metric for Windows.
win_descent	The descender metric for Windows.
x_avg_char_width	Specifies the arithmetic average of the advance width of all of the 26 lowercase letters of the Latin alphabet and the space character.
x_height	This metric specifies the distance between the baseline and the approximate height of non-ascending lowercase letters measured in font units.
y_strikeout_position	The width of the strikeout line.
y_strikeout_size	The size of the strikeout line.
y_subscript_x_offset	Specifies a font designer's recommended horizontal offset for subscript characters.
y_subscript_x_size	Maps to the em square size of the font being used for a subscript.
y_subscript_y_offset	Specifies a font designer's recommended vertical offset from the character baseline to the character baseline for subscript characters associated with this font.
y_subscript_y_size	Maps to the emHeight of the font being used for a subscript.
y_superscript_x_offset	Specifies a font designer's recommended horizontal offset for the superscript characters associated with this font.
y_superscript_x_size	Maps to the em square size of the font being used for a subscript.
y_superscript_y_offset	Specifies a font designer's recommended vertical offset for superscript characeters.
y_superscript_y_size	Maps to the emHeight of the font being used for a subscript.

freetypy.TT_OS2.break_char

TT_OS2.**break_char**

This is the Unicode encoding of the glyph that Windows uses as the break character. The break character is used to separate words and justify text. Most fonts specify 'space' as the break character. This field cannot represent supplementary character values (codepoints greater than `0xFFFF`), and so applications are strongly discouraged from using this field.

freetypy.TT_OS2.cap_height

TT_OS2.**cap_height**

This metric specifies the distance between the baseline and the approximate height of uppercase letters measured in font units. This value would normally be specified by a type designer but in situations where that is not possible, for example when a legacy font is being converted, the value may be set equal to the top of the unscaled and unhinted glyph bounding box of the glyph encoded at `U+0048` (`LATIN CAPITAL LETTER H`). If no glyph is encoded in this position the field should be set to 0.

This metric, if specified, can be used in systems that specify type size by capital height measured in millimeters. It can also be used as an alignment metric; the top of a drop capital, for instance, can be aligned to the *cap_height* metric of the first line of text.

freetypy.TT_OS2.default_char

`TT_OS2.`**`default_char`**

Whenever a request is made for a character that is not in the font, Windows provides this default character. If the value of this field is zero, glyph ID 0 is to be used for the default character otherwise this is the Unicode encoding of the glyph that Windows uses as the default character. This field cannot represent supplementary character values (codepoints greater than `0xFFFF`), and so applications are strongly discouraged from using this field.

freetypy.TT_OS2.family_class

`TT_OS2.`**`family_class`**

Classifies a font design as to its appearance. It does not identify the specific font family, typeface variation, designer, supplier, size, or metric table differences. This is the high byte of the `sFamilyClass` field.

The use of this is limited, so the values are not provided as enumerations. See the TrueType specification for more information.

freetypy.TT_OS2.family_subclass

`TT_OS2.`**`family_subclass`**

Classifies a font design as to its appearance. It does not identify the specific font family, typeface variation, designer, supplier, size, or metric table differences. This is the low byte of the `sFamilyClass` field.

The use of this is limited, so the values are not provided as enumerations. See the TrueType specification for more information.

freetypy.TT_OS2.first_char_index

`TT_OS2.`**`first_char_index`**

The minimum Unicode index in this font.

freetypy.TT_OS2.last_char_index

`TT_OS2.`**`last_char_index`**

The maximum Unicode index in this font.

freetypy.TT_OS2.max_context

`TT_OS2.`**`max_context`**

The maximum length of a target glyph context for any feature in this font. For example, a font which has only a pair kerning feature should set this field to 2. If the font also has a ligature feature in which the glyph sequence "f f i" is substituted by the ligature "", then this field should be set to 3. This field could be useful to sophisticated line-breaking engines in determining how far they should look ahead to test whether something could change that effects the line breaking. For chaining contextual lookups, the length of the string (covered glyph) + (input sequence) + (lookahead sequence) should be considered.

freetypy.TT_OS2.panose

`TT_OS2.`**`panose`**

The PANOSE classification number.

The use of this is limited, so the values are not provided as enumerations. See the TrueType specification for more information. A more Pythonic interface may be provided in the future if needed.

It is a 10-byte string where each byte is as follows:

- 0: Family type
- 1: Serif style
- 2: Weight
- 3: Proportion
- 4: Contrast
- 5: Stroke Variation
- 6: Arm Style
- 7: Letterform
- 8: Midline
- 9: x height

freetypy.TT_OS2.selection

`TT_OS2.`**`selection`**

A *`TT_FS_SELECTION`* bitflag.

freetypy.TT_OS2.type

`TT_OS2.`**`type`**

`FSTYPE` bitflags indicating the licensing restrictions on a font.

freetypy.TT_OS2.typo_ascender

`TT_OS2.`**`typo_ascender`**

The typographic ascender for this font. Remember that this is not the same as *`TT_HoriHeader.ascender`*, which Apple defines in a far different manner.

The suggested usage for *`typo_ascender`* is that it be used in conjunction with *`TT_Header.units_per_em`* to compute a typographically correct default line spacing. The goal is to free applications from Macintosh or Windows-specific metrics which are constrained by backward compatibility requirements. These new metrics, when combined with the character design widths, will allow applications to lay out documents in a typographically correct and portable fashion.

For CJK (Chinese, Japanese, and Korean) fonts that are intended to be used for vertical writing (in addition to horizontal writing), the required value for *`typo_ascender`* is that which describes the top of the of the ideographic em-box. For example, if the ideographic em-box of the font extends from coordinates 0,-120 to 1000,880 (that is, a 1000x1000 box set 120 design units below the Latin baseline), then the value of `typo_ascender` must be set to 880. Failing to adhere to these requirements will result in incorrect vertical layout.

freetypy.TT_OS2.typo_descender

`TT_OS2.`**`typo_descender`**

The typographic descender for this font. Remember that this is not the same as the `TT_HoriHeader.descender`, which Apple defines in a far different manner.

The suggested usage for `typo_descender` is that it be used in conjunction with `TT_Header.units_per_em` to compute a typographically correct default line spacing. The goal is to free applications from Macintosh or Windows-specific metrics which are constrained by backward compatability requirements. These new metrics, when combined with the character design widths, will allow applications to lay out documents in a typographically correct and portable fashion.

For CJK (Chinese, Japanese, and Korean) fonts that are intended to be used for vertical writing (in addition to horizontal writing), the required value for `typo_descender` is that which describes the bottom of the of the ideographic em-box. For example, if the ideographic em-box of the font extends from coordinates 0,-120 to 1000,880 (that is, a 1000x1000 box set 120 design units below the Latin baseline), then the value of `typo_descender` must be set to -120. Failing to adhere to these requirements will result in incorrect vertical layout.

freetypy.TT_OS2.typo_line_gap

`TT_OS2.`**`typo_line_gap`**

The typographic line gap for this font. Remember that this is not the same as `TT_HoriHeader.line_gap`, which Apple defines in a far different manner.

The suggested usage for `typo_line_gap` is that it be used in conjunction with `TTHeader.units_per_em` to compute a typographically correct default line spacing. Typical values average 7-10% of units per em. The goal is to free applications from Macintosh or Windows-specific metrics which are constrained by backward compatability requirements. These new metrics, when combined with the character design widths, will allow applications to lay out documents in a typographically correct and portable fashion.

freetypy.TT_OS2.vend_id

`TT_OS2.`**`vend_id`**

Indentifies the font vendor.

It is not the royalty owner of the original artwork but the company responsible for the marketing and distribution of the typeface that is being classified. It is reasonable to assume that there will be 6 vendors of ITC Zapf Dingbats for use on desktop platforms in the near future (if not already). It is also likely that the vendors will have other inherent benefits in their fonts (more kern pairs, unregularized data, hand hinted, etc.). This identifier will allow for the correct vendors type to be used over another, possibly inferior, font file. These id's are assigned by Microsoft.

freetypy.TT_OS2.version

`TT_OS2.`**`version`**

The version of the `TT_OS2` table.

If this table was synthesized from a font that has no OS/2 table, the version is set to 0xFFFF.

freetypy.TT_OS2.weight_class

`TT_OS2.weight_class`

A *TT_WEIGHT_CLASS* value. Indicates the visual weight (degree of blackness or thickness of strokes) of the characters in the font.

freetypy.TT_OS2.width_class

`TT_OS2.width_class`

A *TT_WIDTH_CLASS* value. Indicates a relative change from the normal aspect ratio (width to height ratio) as specified by a font designer for the glyphs in a font.

Although every character in a font may have a different numeric aspect ratio, each character in a font of normal width has a relative aspect ratio of one. When a new type style is created of a different width class (either by a font designer or by some automated means) the relative aspect ratio of the characters in the new font is some percentage greater or less than those same characters in the normal font it is this difference that this parameter specifies.

freetypy.TT_OS2.win_ascent

`TT_OS2.win_ascent`

The ascender metric for Windows. This, too, is distinct from Apple's *TT_HoriHeader.ascender* value and from the *typo_ascender* values. *win_ascent* is computed as the `y_max` for all characters in the Windows ANSI character set. *win_ascent* is used to compute the Windows font height and default line spacing. For platform 3 encoding 0 fonts, it is the same as *TT_Header.y_max*. Windows will clip the bitmap of any portion of a glyph that appears above this value. Some applications use this value to determine default line spacing. This is strongly discouraged. The typographic ascender, descender and line gap fields in conjunction with *TT_Header.units_per_em* should be used for this purpose. Developers should set this field keeping the above factors in mind.

If any clipping is unacceptable, then the value should be set to *TT_Header.y_max*.

However, if a developer desires to provide appropriate default line spacing using this field, for those applications that continue to use this field for doing so (against OpenType recommendations), then the value should be set appropriately. In such a case, it may result in some glyph bitmaps being clipped.

freetypy.TT_OS2.win_descent

`TT_OS2.win_descent`

The descender metric for Windows. This, too, is distinct from Apple's *TT_HoriHeader.descender* value and from the *typo_descender*. *win_descent* is computed as the `-y_min` for all characters in the Windows ANSI character set. *win_descent* is used to compute the Windows font height and default line spacing. For platform 3 encoding 0 fonts, it is the same as `-TTHeader.y_min`. Windows will clip the bitmap of any portion of a glyph that appears below this value. Some applications use this value to determine default line spacing. This is strongly discouraged. The typographic ascender, descender and line gap fields in conjunction with unitsPerEm should be used for this purpose. Developers should set this field keeping the above factors in mind.

If any clipping is unacceptable, then the value should be set to *TT_Header.y_min*.

However, if a developer desires to provide appropriate default line spacing using this field, for those applications that continue to use this field for doing so (against OpenType recommendations), then the value should be set appropriately. In such a case, it may result in some glyph bitmaps being clipped.

freetypy.TT_OS2.x_avg_char_width

`TT_OS2.`**`x_avg_char_width`**

Specifies the arithmetic average of the advance width of all of the 26 lowercase letters of the Latin alphabet and the space character. If any of the 26 lowercase letters are not present, this parameter should equal zero.

This parameter is a descriptive attribute of the font that specifies the spacing of characters used for comparing one font to another for selection or substitution. For proportionally spaced fonts, this value is useful in estimating the length for lines of text.

freetypy.TT_OS2.x_height

`TT_OS2.`**`x_height`**

This metric specifies the distance between the baseline and the approximate height of non-ascending lowercase letters measured in font units. This value would normally be specified by a type designer but in situations where that is not possible, for example when a legacy font is being converted, the value may be set equal to the top of the unscaled and unhinted glyph bounding box of the glyph encoded at `U+0078` (`LATIN SMALL LETTER X`). If no glyph is encoded in this position the field should be set to 0.

This metric, if specified, can be used in font substitution: the *`x_height`* value of one font can be scaled to approximate the apparent size of another.

freetypy.TT_OS2.y_strikeout_position

`TT_OS2.`**`y_strikeout_position`**

The width of the strikeout line. Positive values represent distances above the baseline, while negative values represent distances below the baseline. A value of zero falls directly on the baseline, while a value of one falls one pixel above the baseline. The value of strikeout position should not interfere with the recognition of standard characters, and therefore should not line up with crossbars in the font.

freetypy.TT_OS2.y_strikeout_size

`TT_OS2.`**`y_strikeout_size`**

The size of the strikeout line. This field should normally be the width of the em-dash for the current font. If the size is one, the strikeout line will be the line represented by the strikeout position field. If the value is two, the strikeout line will be the line represented by the strikeout position and the line immediately above the strikeout position.

freetypy.TT_OS2.y_subscript_x_offset

`TT_OS2.`**`y_subscript_x_offset`**

Specifies a font designer's recommended horizontal offset for subscript characters. It is from the character origin of the font to the character origin of the subscript's character. If a font does not include all of the required subscript characters for an application, and the application can substitute characters, this parameter specifies the recommended horizontal position from the character escapement point of the last character before the first subscript character. For upright characters, this value is usually zero; however, if the characters of a font have an incline (italic characters) the reference point for subscript characters is usually adjusted to compensate for the angle of incline.

freetypy.TT_OS2.y_subscript_x_size

TT_OS2.**y_subscript_x_size**

Maps to the em square size of the font being used for a subscript. If a font has two recommended sizes for subscripts, e.g., numerics and other, the numeric sizes should be stressed. The horizontal font size specifies a font designer's recommended horizontal font size for subscript characters associated with this font. If a font does not include all of the required subscript characters for an application, and the application can substitute characters by scaling the character of a font or by substituting characters from another font, this parameter specifies the recommended em square for those subscript characters. For example, if the em square for a font is 2048 and *y_subscript_x_size* is set to 205, then the horizontal size for a simulated subscript character would be 1/10th the size of the normal character.

freetypy.TT_OS2.y_subscript_y_offset

TT_OS2.**y_subscript_y_offset**

Specifies a font designer's recommended vertical offset from the character baseline to the character baseline for subscript characters associated with this font. Values are expressed as a positive offset below the character baseline. If a font does not include all of the required subscript for an application, this parameter specifies the recommended vertical distance below the character baseline for those subscript characters.

freetypy.TT_OS2.y_subscript_y_size

TT_OS2.**y_subscript_y_size**

Maps to the emHeight of the font being used for a subscript. If a font has two recommended sizes for subscripts, e.g. numerics and other, the numeric sizes should be stressed. The horizontal font size specifies a font designer's recommendation for horizontal font size of subscript characters associated with this font. If a font does not include all of the required subscript characters for an application, and the application can substitute characters by scaling the characters in a font or by substituting characters from another font, this parameter specifies the recommended horizontal EmInc for those subscript characters. For example, if the em square for a font is 2048 and y_subScript_y_size is set to 205, then the vertical size for a simulated subscript character would be 1/10th the size of the normal character.

freetypy.TT_OS2.y_superscript_x_offset

TT_OS2.**y_superscript_x_offset**

Specifies a font designer's recommended horizontal offset for the superscript characters associated with this font. It is from the character origin to the superscript character's origin. If a font does not include all of the required superscript characters for an application, this parameter specifies the recommended horizontal position from the escapement point of the character before the first superscript character. For upright characters, this value is usually zero; however, if the characters of a font have an incline (italic characters) the reference point for superscript characters is usually adjusted to compensate for the angle of incline.

freetypy.TT_OS2.y_superscript_x_size

TT_OS2.**y_superscript_x_size**

Maps to the em square size of the font being used for a subscript. If a font has two recommended sizes for subscripts, e.g., numerics and other, the numeric sizes should be stressed. The horizontal font size specifies a font designer's recommended horizontal font size for superscript characters associated with this font. If a font does not include all of the required superscript characters for an application, and the application can substitute characters by scaling the character of a font or by substituting characters from another font, this parameter specifies the recommended em square for those superscript characters. For example, if the em square for a font is 2048 and `ySuperScriptXSize` is set to 205, then the horizontal size for a simulated superscript character would be 1/10th the size of the normal character.

freetypy.TT_OS2.y_superscript_y_offset

`TT_OS2.`**`y_superscript_y_offset`**

Specifies a font designer's recommended vertical offset for superscript characeters. It is from the character baseline to the superscript character's baseline associated with this font. Values for this parameter are expressed as a positive offset above the character baseline. If a font does not include all of the required superscript characters for an application, this parameter specifies the recommended vertical distance above the character baseline for those superscript characters.

freetypy.TT_OS2.y_superscript_y_size

`TT_OS2.`**`y_superscript_y_size`**

Maps to the emHeight of the font being used for a subscript. If a font has two recommended sizes for subscripts, e.g., numerics and other, the numeric sizes should be stressed. The vertical font size specifies a font designer's recommended vertical font size for superscript characters associated with this font. If a font does not include all of the required superscript characters for an application, and the application can substitute characters by scaling the character of a font or by substituting characters from another font, this parameter specifies the recommended EmHeight for those superscript characters. For example, if the em square for a font is 2048 and `ySuperScriptYSize` is set to 205, then the vertical size for a simulated superscript character would be 1/10th the size of the normal character.

freetypy.TT_Pclt

class `freetypy.`**`TT_Pclt`**

TrueType PCLT table.

All fields comply to the TrueType specification.

This field is discouraged in OpenType and modern TrueType fonts, so it should be used sparingly and not assumed to be available.

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

cap_height
character_complement
file_name
font_number
pitch
serif_style
stroke_weight
style
symbol_set
type_face

Continued on next page

Table 2.46 – continued from previous page

type_family
version
width_type
x_height

freetypy.TT_Pclt.cap_height

TT_Pclt.**cap_height**

freetypy.TT_Pclt.character_complement

TT_Pclt.**character_complement**

freetypy.TT_Pclt.file_name

TT_Pclt.**file_name**

freetypy.TT_Pclt.font_number

TT_Pclt.**font_number**

freetypy.TT_Pclt.pitch

TT_Pclt.**pitch**

freetypy.TT_Pclt.serif_style

TT_Pclt.**serif_style**

freetypy.TT_Pclt.stroke_weight

TT_Pclt.**stroke_weight**

freetypy.TT_Pclt.style

TT_Pclt.**style**

freetypy.TT_Pclt.symbol_set

TT_Pclt.**symbol_set**

freetypy.TT_Pclt.type_face

TT_Pclt.**type_face**

freetypy.TT_Pclt.type_family

`TT_Pclt.type_family`

freetypy.TT_Pclt.version

`TT_Pclt.version`

freetypy.TT_Pclt.width_type

`TT_Pclt.width_type`

freetypy.TT_Pclt.x_height

`TT_Pclt.x_height`

freetypy.TT_Postscript

class `freetypy.TT_Postscript`

TrueType PostScript table.

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

format_type	Format of this table.
is_fixed_pitch	If `True`, the font is monospaced.
italic_angle	Italic angle in degrees.
max_mem_type1	Maximum memory usage when the font is downloaded as a Type 1 font.
max_mem_type42	Maximum memory usage when the font is downloaded as a Type 42 font.
min_mem_type1	Minimum memory usage when the font is downloaded as a Type 1 font.
min_mem_type42	Minimum memory usage when the font is downloaded as a Type 42 font.
underline_position	Underline position.
underline_thickness	Underline thickness.

freetypy.TT_Postscript.format_type

`TT_Postscript.format_type`

Format of this table.

freetypy.TT_Postscript.is_fixed_pitch

`TT_Postscript.`**`is_fixed_pitch`**

If `True`, the font is monospaced.

freetypy.TT_Postscript.italic_angle

`TT_Postscript.`**`italic_angle`**

Italic angle in degrees.

freetypy.TT_Postscript.max_mem_type1

`TT_Postscript.`**`max_mem_type1`**

Maximum memory usage when the font is downloaded as a Type 1 font.

freetypy.TT_Postscript.max_mem_type42

`TT_Postscript.`**`max_mem_type42`**

Maximum memory usage when the font is downloaded as a Type 42 font.

freetypy.TT_Postscript.min_mem_type1

`TT_Postscript.`**`min_mem_type1`**

Minimum memory usage when the font is downloaded as a Type 1 font.

freetypy.TT_Postscript.min_mem_type42

`TT_Postscript.`**`min_mem_type42`**

Minimum memory usage when the font is downloaded as a Type 42 font.

freetypy.TT_Postscript.underline_position

`TT_Postscript.`**`underline_position`**

Underline position.

freetypy.TT_Postscript.underline_thickness

`TT_Postscript.`**`underline_thickness`**

Underline thickness.

freetypy.TT_PLATFORM

class `freetypy.`**`TT_PLATFORM`**

Platform identifier codes.

- *`APPLE_UNICODE`*: Used by Apple to indicate a Unicode character map and/or name entry. See *`TT_APPLE_ID`* for corresponding 'encoding_id' values. Note that name entries in this format are coded as big-endian UCS-2 character codes only.

- *MACINTOSH*: Used by Apple to indicate a MacOS-specific charmap and/or name entry. See *TT_MAC_ID* for corresponding 'encoding_id' values. Note that most TrueType fonts contain an Apple roman charmap to be usable on MacOS systems (even if they contain a Microsoft charmap as well).
- *ISO*: This value was used to specify ISO/IEC 10646 charmaps. It is however now deprecated. See *TT_ISO_ID* for a list of corresponding 'encoding_id' values.
- *MICROSOFT*: Used by Microsoft to indicate Windows-specific charmaps. See *TT_MS_ID* for a list of corresponding 'encoding_id' values. Note that most fonts contain a Unicode charmap using (*TT_PLATFORM.MICROSOFT*, *TT_MS_ID.UNICODE_CS*).
- *CUSTOM*: Used to indicate application-specific charmaps.
- *ADOBE*: This value isn't part of any font format specification, but is used by FreeType to report Adobe-specific charmaps in an *CharMap* object. See *TT_ADOBE_ID*.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

ADOBE
APPLE_UNICODE
CUSTOM
ISO
MACINTOSH
MICROSOFT

freetypy.TT_PLATFORM.ADOBE

TT_PLATFORM.**ADOBE** = **freetypy.TT_PLATFORM.ADOBE**

freetypy.TT_PLATFORM.APPLE_UNICODE

TT_PLATFORM.**APPLE_UNICODE** = **freetypy.TT_PLATFORM.APPLE_UNICODE**

freetypy.TT_PLATFORM.CUSTOM

TT_PLATFORM.**CUSTOM** = **freetypy.TT_PLATFORM.CUSTOM**

freetypy.TT_PLATFORM.ISO

TT_PLATFORM.**ISO** = **freetypy.TT_PLATFORM.ISO**

freetypy.TT_PLATFORM.MACINTOSH

TT_PLATFORM.**MACINTOSH** = **freetypy.TT_PLATFORM.MACINTOSH**

freetypy.TT_PLATFORM.MICROSOFT

TT_PLATFORM.**MICROSOFT** = **freetypy.TT_PLATFORM.MICROSOFT**

freetypy.TT_APPLE_ID

class `freetypy.`**`TT_APPLE_ID`**

Apple-specific encoding values.

- *`DEFAULT`*: Unicode version 1.0.
- *`UNICODE_1_1`*: Unicode 1.1; specifies Hangul characters starting at U+34xx.
- *`ISO_10646`*: Deprecated (identical to preceding).
- *`UNICODE_2_0`*: Unicode 2.0 and beyond (UTF-16 BMP only).
- *`UNICODE_32`*: Unicode 3.1 and beyond, using UTF-32.
- *`VARIANT_SELECTOR`*: From Adobe, not Apple. Not a normal cmap. Specifies variations on a real cmap.

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

`DEFAULT`
`ISO_10646`
`UNICODE_1_1`
`UNICODE_2_0`
`UNICODE_32`
`VARIANT_SELECTOR`

freetypy.TT_APPLE_ID.DEFAULT

`TT_APPLE_ID.`**`DEFAULT`** **= freetypy.TT_APPLE_ID.DEFAULT**

freetypy.TT_APPLE_ID.ISO_10646

`TT_APPLE_ID.`**`ISO_10646`** **= freetypy.TT_APPLE_ID.ISO_10646**

freetypy.TT_APPLE_ID.UNICODE_1_1

`TT_APPLE_ID.`**`UNICODE_1_1`** **= freetypy.TT_APPLE_ID.UNICODE_1_1**

freetypy.TT_APPLE_ID.UNICODE_2_0

`TT_APPLE_ID.`**`UNICODE_2_0`** **= freetypy.TT_APPLE_ID.UNICODE_2_0**

freetypy.TT_APPLE_ID.UNICODE_32

`TT_APPLE_ID.`**`UNICODE_32`** **= freetypy.TT_APPLE_ID.UNICODE_32**

freetypy.TT_APPLE_ID.VARIANT_SELECTOR

`TT_APPLE_ID.`**`VARIANT_SELECTOR`** **= freetypy.TT_APPLE_ID.VARIANT_SELECTOR**

freetypy.TT_ISO_ID

class `freetypy.TT_ISO_ID`

Standard ISO encodings.

- *ISO_7BIT_ASCII*: ASCII.
- *ISO_10646*: ISO/10646.
- *ISO_8859_1*: Also known as Latin-1.

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

ISO_10646
ISO_7BIT_ASCII
ISO_8859_1

freetypy.TT_ISO_ID.ISO_10646

`TT_ISO_ID.ISO_10646` = **freetypy.TT_ISO_ID.ISO_10646**

freetypy.TT_ISO_ID.ISO_7BIT_ASCII

`TT_ISO_ID.ISO_7BIT_ASCII` = **freetypy.TT_ISO_ID.ISO_7BIT_ASCII**

freetypy.TT_ISO_ID.ISO_8859_1

`TT_ISO_ID.ISO_8859_1` = **freetypy.TT_ISO_ID.ISO_8859_1**

freetypy.TT_MAC_ID

class `freetypy.TT_MAC_ID`

Macintosh-specific encoding values.

- *ROMAN*
- *JAPANESE*
- *TRADITIONAL_CHINESE*
- *KOREAN*
- *ARABIC*
- *HEBREW*
- *GREEK*
- *RUSSIAN*
- *RSYMBOL*
- *DEVANAGARI*

- *GURMUKHI*
- *GUJARATI*
- *ORIYA*
- *BENGALI*
- *TAMIL*
- *TELUGU*
- *KANNADA*
- *MALAYALAM*
- *SINHALESE*
- *BURMESE*
- *KHMER*
- *THAI*
- *LAOTIAN*
- *GEORGIAN*
- *ARMENIAN*
- *MALDIVIAN*
- *SIMPLIFIED_CHINESE*
- *TIBETAN*
- *MONGOLIAN*
- *GEEZ*
- *SLAVIC*
- *VIETNAMESE*
- *SINDHI*
- *UNINTERP*

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

ARABIC
ARMENIAN
BENGALI
BURMESE
DEVANAGARI
GEEZ
GEORGIAN
GREEK
GUJARATI
GURMUKHI

Continued on next page

Table 2.51 – continued from previous page

HEBREW
JAPANESE
KANNADA
KHMER
KOREAN
LAOTIAN
MALAYALAM
MALDIVIAN
MONGOLIAN
ORIYA
ROMAN
RSYMBOL
RUSSIAN
SIMPLIFIED_CHINESE
SINDHI
SINHALESE
SLAVIC
TAMIL
TELUGU
THAI
TIBETAN
TRADITIONAL_CHINESE
UNINTERP
VIETNAMESE

freetypy.TT_MAC_ID.ARABIC

`TT_MAC_ID.`**`ARABIC`** **= freetypy.TT_MAC_ID.ARABIC**

freetypy.TT_MAC_ID.ARMENIAN

`TT_MAC_ID.`**`ARMENIAN`** **= freetypy.TT_MAC_ID.ARMENIAN**

freetypy.TT_MAC_ID.BENGALI

`TT_MAC_ID.`**`BENGALI`** **= freetypy.TT_MAC_ID.BENGALI**

freetypy.TT_MAC_ID.BURMESE

`TT_MAC_ID.`**`BURMESE`** **= freetypy.TT_MAC_ID.BURMESE**

freetypy.TT_MAC_ID.DEVANAGARI

`TT_MAC_ID.`**`DEVANAGARI`** **= freetypy.TT_MAC_ID.DEVANAGARI**

freetypy.TT_MAC_ID.GEEZ

`TT_MAC_ID.`**`GEEZ`** **= freetypy.TT_MAC_ID.GEEZ**

freetypy.TT_MAC_ID.GEORGIAN

TT_MAC_ID.**GEORGIAN** = **freetypy.TT_MAC_ID.GEORGIAN**

freetypy.TT_MAC_ID.GREEK

TT_MAC_ID.**GREEK** = **freetypy.TT_MAC_ID.GREEK**

freetypy.TT_MAC_ID.GUJARATI

TT_MAC_ID.**GUJARATI** = **freetypy.TT_MAC_ID.GUJARATI**

freetypy.TT_MAC_ID.GURMUKHI

TT_MAC_ID.**GURMUKHI** = **freetypy.TT_MAC_ID.GURMUKHI**

freetypy.TT_MAC_ID.HEBREW

TT_MAC_ID.**HEBREW** = **freetypy.TT_MAC_ID.HEBREW**

freetypy.TT_MAC_ID.JAPANESE

TT_MAC_ID.**JAPANESE** = **freetypy.TT_MAC_ID.JAPANESE**

freetypy.TT_MAC_ID.KANNADA

TT_MAC_ID.**KANNADA** = **freetypy.TT_MAC_ID.KANNADA**

freetypy.TT_MAC_ID.KHMER

TT_MAC_ID.**KHMER** = **freetypy.TT_MAC_ID.KHMER**

freetypy.TT_MAC_ID.KOREAN

TT_MAC_ID.**KOREAN** = **freetypy.TT_MAC_ID.KOREAN**

freetypy.TT_MAC_ID.LAOTIAN

TT_MAC_ID.**LAOTIAN** = **freetypy.TT_MAC_ID.LAOTIAN**

freetypy.TT_MAC_ID.MALAYALAM

TT_MAC_ID.**MALAYALAM** = **freetypy.TT_MAC_ID.MALAYALAM**

freetypy.TT_MAC_ID.MALDIVIAN

`TT_MAC_ID.`**`MALDIVIAN`** = **freetypy.TT_MAC_ID.SIMPLIFIED_CHINESE**

freetypy.TT_MAC_ID.MONGOLIAN

`TT_MAC_ID.`**`MONGOLIAN`** = **freetypy.TT_MAC_ID.MONGOLIAN**

freetypy.TT_MAC_ID.ORIYA

`TT_MAC_ID.`**`ORIYA`** = **freetypy.TT_MAC_ID.ORIYA**

freetypy.TT_MAC_ID.ROMAN

`TT_MAC_ID.`**`ROMAN`** = **freetypy.TT_MAC_ID.ROMAN**

freetypy.TT_MAC_ID.RSYMBOL

`TT_MAC_ID.`**`RSYMBOL`** = **freetypy.TT_MAC_ID.RSYMBOL**

freetypy.TT_MAC_ID.RUSSIAN

`TT_MAC_ID.`**`RUSSIAN`** = **freetypy.TT_MAC_ID.RUSSIAN**

freetypy.TT_MAC_ID.SIMPLIFIED_CHINESE

`TT_MAC_ID.`**`SIMPLIFIED_CHINESE`** = **freetypy.TT_MAC_ID.SIMPLIFIED_CHINESE**

freetypy.TT_MAC_ID.SINDHI

`TT_MAC_ID.`**`SINDHI`** = **freetypy.TT_MAC_ID.SINDHI**

freetypy.TT_MAC_ID.SINHALESE

`TT_MAC_ID.`**`SINHALESE`** = **freetypy.TT_MAC_ID.SINHALESE**

freetypy.TT_MAC_ID.SLAVIC

`TT_MAC_ID.`**`SLAVIC`** = **freetypy.TT_MAC_ID.SLAVIC**

freetypy.TT_MAC_ID.TAMIL

`TT_MAC_ID.`**`TAMIL`** = **freetypy.TT_MAC_ID.TAMIL**

freetypy.TT_MAC_ID.TELUGU

TT_MAC_ID.**TELUGU** = **freetypy.TT_MAC_ID.TELUGU**

freetypy.TT_MAC_ID.THAI

TT_MAC_ID.**THAI** = **freetypy.TT_MAC_ID.THAI**

freetypy.TT_MAC_ID.TIBETAN

TT_MAC_ID.**TIBETAN** = **freetypy.TT_MAC_ID.TIBETAN**

freetypy.TT_MAC_ID.TRADITIONAL_CHINESE

TT_MAC_ID.**TRADITIONAL_CHINESE** = **freetypy.TT_MAC_ID.TRADITIONAL_CHINESE**

freetypy.TT_MAC_ID.UNINTERP

TT_MAC_ID.**UNINTERP** = **freetypy.TT_MAC_ID.UNINTERP**

freetypy.TT_MAC_ID.VIETNAMESE

TT_MAC_ID.**VIETNAMESE** = **freetypy.TT_MAC_ID.VIETNAMESE**

freetypy.TT_MAC_LANGID

class freetypy.**TT_MAC_LANGID**

Language identifier.

Used in the name records of the TTF "name" table if the "platform" identifier code is *TT_PLATFORM.MACINTOSH*.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

AFRIKAANS
ALBANIAN
AMHARIC
ARABIC
ARMENIAN
ASSAMESE
AYMARA
AZERBAIJANI
AZERBAIJANI_ARABIC_SCRIPT
AZERBAIJANI_CYRILLIC_SCRIPT

Continued on next page

Table 2.52 – continued from previous page

AZERBAIJANI_ROMAN_SCRIPT
BASQUE
BENGALI
BRETON
BULGARIAN
BURMESE
BYELORUSSIAN
CATALAN
CHEWA
CHINESE_SIMPLIFIED
CHINESE_TRADITIONAL
CROATIAN
CZECH
DANISH
DUTCH
DZONGKHA
ENGLISH
ESPERANTO
ESTONIAN
FAEROESE
FARSI
FINNISH
FLEMISH
FRENCH
GALICIAN
GALLA
GEORGIAN
GERMAN
GREEK
GREEK_POLYTONIC
GREELANDIC
GUARANI
GUJARATI
HEBREW
HINDI
HUNGARIAN
ICELANDIC
INDONESIAN
INUKTITUT
IRISH
IRISH_GAELIC
ITALIAN
JAPANESE
JAVANESE
KANNADA
KASHMIRI
KAZAKH
KHMER
KIRGHIZ
KOREAN

Continued on next page

Table 2.52 – continued from previous page

KURDISH
LAO
LATIN
LETTISH
LITHUANIAN
MACEDONIAN
MALAGASY
MALAYALAM
MALAY_ARABIC_SCRIPT
MALAY_ROMAN_SCRIPT
MALTESE
MANX_GAELIC
MARATHI
MOLDAVIAN
MONGOLIAN
MONGOLIAN_CYRILLIC_SCRIPT
MONGOLIAN_MONGOLIAN_SCRIPT
NEPALI
NORWEGIAN
ORIYA
PASHTO
POLISH
PORTUGUESE
PUNJABI
QUECHUA
ROMANIAN
RUANDA
RUNDI
RUSSIAN
SAAMISK
SANSKRIT
SCOTTISH_GAELIC
SERBIAN
SINDHI
SINHALESE
SLOVAK
SLOVENIAN
SOMALI
SPANISH
SUNDANESE
SWAHILI
SWEDISH
TAGALOG
TAJIKI
TAMIL
TATAR
TELUGU
THAI
TIBETAN
TIGRINYA

Continued on next page

Table 2.52 – continued from previous page

TONGAN
TURKISH
TURKMEN
UIGHUR
UKRAINIAN
URDU
UZBEK
VIETNAMESE
WELSH
YIDDISH

freetypy.TT_MAC_LANGID.AFRIKAANS

TT_MAC_LANGID.**AFRIKAANS** = **freetypy.TT_MAC_LANGID.AFRIKAANS**

freetypy.TT_MAC_LANGID.ALBANIAN

TT_MAC_LANGID.**ALBANIAN** = **freetypy.TT_MAC_LANGID.ALBANIAN**

freetypy.TT_MAC_LANGID.AMHARIC

TT_MAC_LANGID.**AMHARIC** = **freetypy.TT_MAC_LANGID.AMHARIC**

freetypy.TT_MAC_LANGID.ARABIC

TT_MAC_LANGID.**ARABIC** = **freetypy.TT_MAC_LANGID.ARABIC**

freetypy.TT_MAC_LANGID.ARMENIAN

TT_MAC_LANGID.**ARMENIAN** = **freetypy.TT_MAC_LANGID.ARMENIAN**

freetypy.TT_MAC_LANGID.ASSAMESE

TT_MAC_LANGID.**ASSAMESE** = **freetypy.TT_MAC_LANGID.ASSAMESE**

freetypy.TT_MAC_LANGID.AYMARA

TT_MAC_LANGID.**AYMARA** = **freetypy.TT_MAC_LANGID.AYMARA**

freetypy.TT_MAC_LANGID.AZERBAIJANI

TT_MAC_LANGID.**AZERBAIJANI** = **freetypy.TT_MAC_LANGID.AZERBAIJANI_CYRILLIC_SCRIPT**

freetypy.TT_MAC_LANGID.AZERBAIJANI_ARABIC_SCRIPT

TT_MAC_LANGID.**AZERBAIJANI_ARABIC_SCRIPT** = **freetypy.TT_MAC_LANGID.AZERBAIJANI_ARABIC_SCRIP**

freetypy.TT_MAC_LANGID.AZERBAIJANI_CYRILLIC_SCRIPT

`TT_MAC_LANGID.`**`AZERBAIJANI_CYRILLIC_SCRIPT`** = **freetypy.TT_MAC_LANGID.AZERBAIJANI_CYRILLIC_S**

freetypy.TT_MAC_LANGID.AZERBAIJANI_ROMAN_SCRIPT

`TT_MAC_LANGID.`**`AZERBAIJANI_ROMAN_SCRIPT`** = **freetypy.TT_MAC_LANGID.AZERBAIJANI_ROMAN_SCRIPT**

freetypy.TT_MAC_LANGID.BASQUE

`TT_MAC_LANGID.`**`BASQUE`** = **freetypy.TT_MAC_LANGID.BASQUE**

freetypy.TT_MAC_LANGID.BENGALI

`TT_MAC_LANGID.`**`BENGALI`** = **freetypy.TT_MAC_LANGID.BENGALI**

freetypy.TT_MAC_LANGID.BRETON

`TT_MAC_LANGID.`**`BRETON`** = **freetypy.TT_MAC_LANGID.BRETON**

freetypy.TT_MAC_LANGID.BULGARIAN

`TT_MAC_LANGID.`**`BULGARIAN`** = **freetypy.TT_MAC_LANGID.BULGARIAN**

freetypy.TT_MAC_LANGID.BURMESE

`TT_MAC_LANGID.`**`BURMESE`** = **freetypy.TT_MAC_LANGID.BURMESE**

freetypy.TT_MAC_LANGID.BYELORUSSIAN

`TT_MAC_LANGID.`**`BYELORUSSIAN`** = **freetypy.TT_MAC_LANGID.BYELORUSSIAN**

freetypy.TT_MAC_LANGID.CATALAN

`TT_MAC_LANGID.`**`CATALAN`** = **freetypy.TT_MAC_LANGID.CATALAN**

freetypy.TT_MAC_LANGID.CHEWA

`TT_MAC_LANGID.`**`CHEWA`** = **freetypy.TT_MAC_LANGID.CHEWA**

freetypy.TT_MAC_LANGID.CHINESE_SIMPLIFIED

`TT_MAC_LANGID.`**`CHINESE_SIMPLIFIED`** = **freetypy.TT_MAC_LANGID.CHINESE_SIMPLIFIED**

freetypy.TT_MAC_LANGID.CHINESE_TRADITIONAL

`TT_MAC_LANGID.`**`CHINESE_TRADITIONAL`** = **freetypy.TT_MAC_LANGID.CHINESE_TRADITIONAL**

freetypy.TT_MAC_LANGID.CROATIAN

`TT_MAC_LANGID.`**`CROATIAN`** = **freetypy.TT_MAC_LANGID.CROATIAN**

freetypy.TT_MAC_LANGID.CZECH

`TT_MAC_LANGID.`**`CZECH`** = **freetypy.TT_MAC_LANGID.CZECH**

freetypy.TT_MAC_LANGID.DANISH

`TT_MAC_LANGID.`**`DANISH`** = **freetypy.TT_MAC_LANGID.DANISH**

freetypy.TT_MAC_LANGID.DUTCH

`TT_MAC_LANGID.`**`DUTCH`** = **freetypy.TT_MAC_LANGID.DUTCH**

freetypy.TT_MAC_LANGID.DZONGKHA

`TT_MAC_LANGID.`**`DZONGKHA`** = **freetypy.TT_MAC_LANGID.DZONGKHA**

freetypy.TT_MAC_LANGID.ENGLISH

`TT_MAC_LANGID.`**`ENGLISH`** = **freetypy.TT_MAC_LANGID.ENGLISH**

freetypy.TT_MAC_LANGID.ESPERANTO

`TT_MAC_LANGID.`**`ESPERANTO`** = **freetypy.TT_MAC_LANGID.ESPERANTO**

freetypy.TT_MAC_LANGID.ESTONIAN

`TT_MAC_LANGID.`**`ESTONIAN`** = **freetypy.TT_MAC_LANGID.ESTONIAN**

freetypy.TT_MAC_LANGID.FAEROESE

`TT_MAC_LANGID.`**`FAEROESE`** = **freetypy.TT_MAC_LANGID.FAEROESE**

freetypy.TT_MAC_LANGID.FARSI

`TT_MAC_LANGID.`**`FARSI`** = **freetypy.TT_MAC_LANGID.FARSI**

freetypy.TT_MAC_LANGID.FINNISH

TT_MAC_LANGID.**FINNISH** = **freetypy.TT_MAC_LANGID.FINNISH**

freetypy.TT_MAC_LANGID.FLEMISH

TT_MAC_LANGID.**FLEMISH** = **freetypy.TT_MAC_LANGID.FLEMISH**

freetypy.TT_MAC_LANGID.FRENCH

TT_MAC_LANGID.**FRENCH** = **freetypy.TT_MAC_LANGID.FRENCH**

freetypy.TT_MAC_LANGID.GALICIAN

TT_MAC_LANGID.**GALICIAN** = **freetypy.TT_MAC_LANGID.GALICIAN**

freetypy.TT_MAC_LANGID.GALLA

TT_MAC_LANGID.**GALLA** = **freetypy.TT_MAC_LANGID.GALLA**

freetypy.TT_MAC_LANGID.GEORGIAN

TT_MAC_LANGID.**GEORGIAN** = **freetypy.TT_MAC_LANGID.GEORGIAN**

freetypy.TT_MAC_LANGID.GERMAN

TT_MAC_LANGID.**GERMAN** = **freetypy.TT_MAC_LANGID.GERMAN**

freetypy.TT_MAC_LANGID.GREEK

TT_MAC_LANGID.**GREEK** = **freetypy.TT_MAC_LANGID.GREEK**

freetypy.TT_MAC_LANGID.GREEK_POLYTONIC

TT_MAC_LANGID.**GREEK_POLYTONIC** = **freetypy.TT_MAC_LANGID.GREEK_POLYTONIC**

freetypy.TT_MAC_LANGID.GREELANDIC

TT_MAC_LANGID.**GREELANDIC** = **freetypy.TT_MAC_LANGID.GREELANDIC**

freetypy.TT_MAC_LANGID.GUARANI

TT_MAC_LANGID.**GUARANI** = **freetypy.TT_MAC_LANGID.GUARANI**

freetypy.TT_MAC_LANGID.GUJARATI

TT_MAC_LANGID.**GUJARATI** = freetypy.TT_MAC_LANGID.GUJARATI

freetypy.TT_MAC_LANGID.HEBREW

TT_MAC_LANGID.**HEBREW** = freetypy.TT_MAC_LANGID.HEBREW

freetypy.TT_MAC_LANGID.HINDI

TT_MAC_LANGID.**HINDI** = freetypy.TT_MAC_LANGID.HINDI

freetypy.TT_MAC_LANGID.HUNGARIAN

TT_MAC_LANGID.**HUNGARIAN** = freetypy.TT_MAC_LANGID.HUNGARIAN

freetypy.TT_MAC_LANGID.ICELANDIC

TT_MAC_LANGID.**ICELANDIC** = freetypy.TT_MAC_LANGID.ICELANDIC

freetypy.TT_MAC_LANGID.INDONESIAN

TT_MAC_LANGID.**INDONESIAN** = freetypy.TT_MAC_LANGID.INDONESIAN

freetypy.TT_MAC_LANGID.INUKTITUT

TT_MAC_LANGID.**INUKTITUT** = freetypy.TT_MAC_LANGID.INUKTITUT

freetypy.TT_MAC_LANGID.IRISH

TT_MAC_LANGID.**IRISH** = freetypy.TT_MAC_LANGID.IRISH

freetypy.TT_MAC_LANGID.IRISH_GAELIC

TT_MAC_LANGID.**IRISH_GAELIC** = freetypy.TT_MAC_LANGID.IRISH_GAELIC

freetypy.TT_MAC_LANGID.ITALIAN

TT_MAC_LANGID.**ITALIAN** = freetypy.TT_MAC_LANGID.ITALIAN

freetypy.TT_MAC_LANGID.JAPANESE

TT_MAC_LANGID.**JAPANESE** = freetypy.TT_MAC_LANGID.JAPANESE

freetypy.TT_MAC_LANGID.JAVANESE

`TT_MAC_LANGID.`**`JAVANESE`** = **freetypy.TT_MAC_LANGID.JAVANESE**

freetypy.TT_MAC_LANGID.KANNADA

`TT_MAC_LANGID.`**`KANNADA`** = **freetypy.TT_MAC_LANGID.KANNADA**

freetypy.TT_MAC_LANGID.KASHMIRI

`TT_MAC_LANGID.`**`KASHMIRI`** = **freetypy.TT_MAC_LANGID.KASHMIRI**

freetypy.TT_MAC_LANGID.KAZAKH

`TT_MAC_LANGID.`**`KAZAKH`** = **freetypy.TT_MAC_LANGID.KAZAKH**

freetypy.TT_MAC_LANGID.KHMER

`TT_MAC_LANGID.`**`KHMER`** = **freetypy.TT_MAC_LANGID.KHMER**

freetypy.TT_MAC_LANGID.KIRGHIZ

`TT_MAC_LANGID.`**`KIRGHIZ`** = **freetypy.TT_MAC_LANGID.KIRGHIZ**

freetypy.TT_MAC_LANGID.KOREAN

`TT_MAC_LANGID.`**`KOREAN`** = **freetypy.TT_MAC_LANGID.KOREAN**

freetypy.TT_MAC_LANGID.KURDISH

`TT_MAC_LANGID.`**`KURDISH`** = **freetypy.TT_MAC_LANGID.KURDISH**

freetypy.TT_MAC_LANGID.LAO

`TT_MAC_LANGID.`**`LAO`** = **freetypy.TT_MAC_LANGID.LAO**

freetypy.TT_MAC_LANGID.LATIN

`TT_MAC_LANGID.`**`LATIN`** = **freetypy.TT_MAC_LANGID.LATIN**

freetypy.TT_MAC_LANGID.LETTISH

`TT_MAC_LANGID.`**`LETTISH`** = **freetypy.TT_MAC_LANGID.LETTISH**

freetypy.TT_MAC_LANGID.LITHUANIAN

TT_MAC_LANGID.**LITHUANIAN** = freetypy.TT_MAC_LANGID.LITHUANIAN

freetypy.TT_MAC_LANGID.MACEDONIAN

TT_MAC_LANGID.**MACEDONIAN** = freetypy.TT_MAC_LANGID.MACEDONIAN

freetypy.TT_MAC_LANGID.MALAGASY

TT_MAC_LANGID.**MALAGASY** = freetypy.TT_MAC_LANGID.MALAGASY

freetypy.TT_MAC_LANGID.MALAYALAM

TT_MAC_LANGID.**MALAYALAM** = freetypy.TT_MAC_LANGID.MALAYALAM

freetypy.TT_MAC_LANGID.MALAY_ARABIC_SCRIPT

TT_MAC_LANGID.**MALAY_ARABIC_SCRIPT** = freetypy.TT_MAC_LANGID.MALAY_ARABIC_SCRIPT

freetypy.TT_MAC_LANGID.MALAY_ROMAN_SCRIPT

TT_MAC_LANGID.**MALAY_ROMAN_SCRIPT** = freetypy.TT_MAC_LANGID.MALAY_ROMAN_SCRIPT

freetypy.TT_MAC_LANGID.MALTESE

TT_MAC_LANGID.**MALTESE** = freetypy.TT_MAC_LANGID.MALTESE

freetypy.TT_MAC_LANGID.MANX_GAELIC

TT_MAC_LANGID.**MANX_GAELIC** = freetypy.TT_MAC_LANGID.MANX_GAELIC

freetypy.TT_MAC_LANGID.MARATHI

TT_MAC_LANGID.**MARATHI** = freetypy.TT_MAC_LANGID.MARATHI

freetypy.TT_MAC_LANGID.MOLDAVIAN

TT_MAC_LANGID.**MOLDAVIAN** = freetypy.TT_MAC_LANGID.MOLDAVIAN

freetypy.TT_MAC_LANGID.MONGOLIAN

TT_MAC_LANGID.**MONGOLIAN** = freetypy.TT_MAC_LANGID.MONGOLIAN_MONGOLIAN_SCRIPT

freetypy.TT_MAC_LANGID.MONGOLIAN_CYRILLIC_SCRIPT

TT_MAC_LANGID.**MONGOLIAN_CYRILLIC_SCRIPT** = freetypy.TT_MAC_LANGID.MONGOLIAN_CYRILLIC_SCR

freetypy.TT_MAC_LANGID.MONGOLIAN_MONGOLIAN_SCRIPT

TT_MAC_LANGID.**MONGOLIAN_MONGOLIAN_SCRIPT** = freetypy.TT_MAC_LANGID.MONGOLIAN_MONGOLIAN_

freetypy.TT_MAC_LANGID.NEPALI

TT_MAC_LANGID.**NEPALI** = freetypy.TT_MAC_LANGID.NEPALI

freetypy.TT_MAC_LANGID.NORWEGIAN

TT_MAC_LANGID.**NORWEGIAN** = freetypy.TT_MAC_LANGID.NORWEGIAN

freetypy.TT_MAC_LANGID.ORIYA

TT_MAC_LANGID.**ORIYA** = freetypy.TT_MAC_LANGID.ORIYA

freetypy.TT_MAC_LANGID.PASHTO

TT_MAC_LANGID.**PASHTO** = freetypy.TT_MAC_LANGID.PASHTO

freetypy.TT_MAC_LANGID.POLISH

TT_MAC_LANGID.**POLISH** = freetypy.TT_MAC_LANGID.POLISH

freetypy.TT_MAC_LANGID.PORTUGUESE

TT_MAC_LANGID.**PORTUGUESE** = freetypy.TT_MAC_LANGID.PORTUGUESE

freetypy.TT_MAC_LANGID.PUNJABI

TT_MAC_LANGID.**PUNJABI** = freetypy.TT_MAC_LANGID.PUNJABI

freetypy.TT_MAC_LANGID.QUECHUA

TT_MAC_LANGID.**QUECHUA** = freetypy.TT_MAC_LANGID.QUECHUA

freetypy.TT_MAC_LANGID.ROMANIAN

TT_MAC_LANGID.**ROMANIAN** = freetypy.TT_MAC_LANGID.ROMANIAN

freetypy.TT_MAC_LANGID.RUANDA

TT_MAC_LANGID.**RUANDA** = freetypy.TT_MAC_LANGID.RUANDA

freetypy.TT_MAC_LANGID.RUNDI

TT_MAC_LANGID.**RUNDI** = freetypy.TT_MAC_LANGID.RUNDI

freetypy.TT_MAC_LANGID.RUSSIAN

TT_MAC_LANGID.**RUSSIAN** = freetypy.TT_MAC_LANGID.RUSSIAN

freetypy.TT_MAC_LANGID.SAAMISK

TT_MAC_LANGID.**SAAMISK** = freetypy.TT_MAC_LANGID.SAAMISK

freetypy.TT_MAC_LANGID.SANSKRIT

TT_MAC_LANGID.**SANSKRIT** = freetypy.TT_MAC_LANGID.SANSKRIT

freetypy.TT_MAC_LANGID.SCOTTISH_GAELIC

TT_MAC_LANGID.**SCOTTISH_GAELIC** = freetypy.TT_MAC_LANGID.SCOTTISH_GAELIC

freetypy.TT_MAC_LANGID.SERBIAN

TT_MAC_LANGID.**SERBIAN** = freetypy.TT_MAC_LANGID.SERBIAN

freetypy.TT_MAC_LANGID.SINDHI

TT_MAC_LANGID.**SINDHI** = freetypy.TT_MAC_LANGID.SINDHI

freetypy.TT_MAC_LANGID.SINHALESE

TT_MAC_LANGID.**SINHALESE** = freetypy.TT_MAC_LANGID.SINHALESE

freetypy.TT_MAC_LANGID.SLOVAK

TT_MAC_LANGID.**SLOVAK** = freetypy.TT_MAC_LANGID.SLOVAK

freetypy.TT_MAC_LANGID.SLOVENIAN

TT_MAC_LANGID.**SLOVENIAN** = freetypy.TT_MAC_LANGID.SLOVENIAN

freetypy.TT_MAC_LANGID.SOMALI

`TT_MAC_LANGID.SOMALI` = freetypy.TT_MAC_LANGID.SOMALI

freetypy.TT_MAC_LANGID.SPANISH

`TT_MAC_LANGID.SPANISH` = freetypy.TT_MAC_LANGID.SPANISH

freetypy.TT_MAC_LANGID.SUNDANESE

`TT_MAC_LANGID.SUNDANESE` = freetypy.TT_MAC_LANGID.SUNDANESE

freetypy.TT_MAC_LANGID.SWAHILI

`TT_MAC_LANGID.SWAHILI` = freetypy.TT_MAC_LANGID.SWAHILI

freetypy.TT_MAC_LANGID.SWEDISH

`TT_MAC_LANGID.SWEDISH` = freetypy.TT_MAC_LANGID.SWEDISH

freetypy.TT_MAC_LANGID.TAGALOG

`TT_MAC_LANGID.TAGALOG` = freetypy.TT_MAC_LANGID.TAGALOG

freetypy.TT_MAC_LANGID.TAJIKI

`TT_MAC_LANGID.TAJIKI` = freetypy.TT_MAC_LANGID.TAJIKI

freetypy.TT_MAC_LANGID.TAMIL

`TT_MAC_LANGID.TAMIL` = freetypy.TT_MAC_LANGID.TAMIL

freetypy.TT_MAC_LANGID.TATAR

`TT_MAC_LANGID.TATAR` = freetypy.TT_MAC_LANGID.TATAR

freetypy.TT_MAC_LANGID.TELUGU

`TT_MAC_LANGID.TELUGU` = freetypy.TT_MAC_LANGID.TELUGU

freetypy.TT_MAC_LANGID.THAI

`TT_MAC_LANGID.THAI` = freetypy.TT_MAC_LANGID.THAI

freetypy.TT_MAC_LANGID.TIBETAN

TT_MAC_LANGID.**TIBETAN** = freetypy.TT_MAC_LANGID.TIBETAN

freetypy.TT_MAC_LANGID.TIGRINYA

TT_MAC_LANGID.**TIGRINYA** = freetypy.TT_MAC_LANGID.TIGRINYA

freetypy.TT_MAC_LANGID.TONGAN

TT_MAC_LANGID.**TONGAN** = freetypy.TT_MAC_LANGID.TONGAN

freetypy.TT_MAC_LANGID.TURKISH

TT_MAC_LANGID.**TURKISH** = freetypy.TT_MAC_LANGID.TURKISH

freetypy.TT_MAC_LANGID.TURKMEN

TT_MAC_LANGID.**TURKMEN** = freetypy.TT_MAC_LANGID.TURKMEN

freetypy.TT_MAC_LANGID.UIGHUR

TT_MAC_LANGID.**UIGHUR** = freetypy.TT_MAC_LANGID.UIGHUR

freetypy.TT_MAC_LANGID.UKRAINIAN

TT_MAC_LANGID.**UKRAINIAN** = freetypy.TT_MAC_LANGID.UKRAINIAN

freetypy.TT_MAC_LANGID.URDU

TT_MAC_LANGID.**URDU** = freetypy.TT_MAC_LANGID.URDU

freetypy.TT_MAC_LANGID.UZBEK

TT_MAC_LANGID.**UZBEK** = freetypy.TT_MAC_LANGID.UZBEK

freetypy.TT_MAC_LANGID.VIETNAMESE

TT_MAC_LANGID.**VIETNAMESE** = freetypy.TT_MAC_LANGID.VIETNAMESE

freetypy.TT_MAC_LANGID.WELSH

TT_MAC_LANGID.**WELSH** = freetypy.TT_MAC_LANGID.WELSH

freetypy.TT_MAC_LANGID.YIDDISH

`TT_MAC_LANGID.`**`YIDDISH`** **= freetypy.TT_MAC_LANGID.YIDDISH**

freetypy.TT_MS_ID

class `freetypy.`**`TT_MS_ID`**

Microsoft-specific encoding values.

- *SYMBOL_CS*: Corresponds to Microsoft symbol encoding. See *ENCODING.MS_SYMBOL*.
- *UNICODE_CS*: Corresponds to a Microsoft WGL4 charmap, matching Unicode. See *ENCODING.UNICODE*.
- *SJIS*: Corresponds to SJIS Japanese encoding. See *ENCODING.SJIS*.
- *GB2312*: Corresponds to Simplified Chinese as used in Mainland China. See *ENCODING.GB2312*.
- *BIG_5*: Corresponds to Traditional Chinese as used in Taiwan and Hong Kong. See *ENCODING.BIG5*.
- *WANSUNG*: Corresponds to Korean Wansung encoding. See *ENCODING.WANSUNG*.
- *JOHAB*: Corresponds to Johab encoding. See *ENCODING.JOHAB*.
- *UCS_4*: Corresponds to UCS-4 or UTF-32 charmaps. This has been added to the OpenType specification version 1.4 (mid-2001.)

`__init__``()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

BIG_5
GB2312
JOHAB
SJIS
SYMBOL_CS
UCS_4
UNICODE_CS
WANSUNG

freetypy.TT_MS_ID.BIG_5

`TT_MS_ID.`**`BIG_5`** **= freetypy.TT_MS_ID.BIG_5**

freetypy.TT_MS_ID.GB2312

`TT_MS_ID.`**`GB2312`** **= freetypy.TT_MS_ID.GB2312**

freetypy.TT_MS_ID.JOHAB

`TT_MS_ID.`**`JOHAB`** **= freetypy.TT_MS_ID.JOHAB**

freetypy.TT_MS_ID.SJIS

TT_MS_ID.**SJIS** = freetypy.TT_MS_ID.SJIS

freetypy.TT_MS_ID.SYMBOL_CS

TT_MS_ID.**SYMBOL_CS** = freetypy.TT_MS_ID.SYMBOL_CS

freetypy.TT_MS_ID.UCS_4

TT_MS_ID.**UCS_4** = freetypy.TT_MS_ID.UCS_4

freetypy.TT_MS_ID.UNICODE_CS

TT_MS_ID.**UNICODE_CS** = freetypy.TT_MS_ID.UNICODE_CS

freetypy.TT_MS_ID.WANSUNG

TT_MS_ID.**WANSUNG** = freetypy.TT_MS_ID.WANSUNG

freetypy.TT_MS_LANGID

class freetypy.**TT_MS_LANGID**

Language identifier.

Used in in the name records of the TTF "name" table if the "platform" identifier code is *TT_PLATFORM.MICROSOFT*.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

AFRIKAANS_SOUTH_AFRICA
ALBANIAN_ALBANIA
AMHARIC_ETHIOPIA
ARABIC_ALGERIA
ARABIC_BAHRAIN
ARABIC_EGYPT
ARABIC_GENERAL
ARABIC_IRAQ
ARABIC_JORDAN
ARABIC_KUWAIT
ARABIC_LEBANON
ARABIC_LIBYA
ARABIC_MOROCCO
ARABIC_OMAN
ARABIC_QATAR

Continued on next page

Table 2.54 – continued from previous page

ARABIC_SAUDI_ARABIA
ARABIC_SYRIA
ARABIC_TUNISIA
ARABIC_UAE
ARABIC_YEMEN
ARMENIAN_ARMENIA
ASSAMESE_INDIA
AZERI_AZERBAIJAN_CYRILLIC
AZERI_AZERBAIJAN_LATIN
BASQUE_SPAIN
BELARUSIAN_BELARUS
BENGALI_BANGLADESH
BENGALI_INDIA
BOSNIAN_BOSNIA_HERZEGOVINA
BULGARIAN_BULGARIA
BURMESE_MYANMAR
CATALAN_SPAIN
CHEROKEE_UNITED_STATES
CHINESE_GENERAL
CHINESE_HONG_KONG
CHINESE_MACAU
CHINESE_PRC
CHINESE_SINGAPORE
CHINESE_TAIWAN
CLASSIC_LITHUANIAN_LITHUANIA
CROATIAN_BOSNIA_HERZEGOVINA
CROATIAN_CROATIA
CZECH_CZECH_REPUBLIC
DANISH_DENMARK
DHIVEHI_MALDIVES
DIVEHI_MALDIVES
DUTCH_BELGIUM
DUTCH_NETHERLANDS
EDO_NIGERIA
ENGLISH_AUSTRALIA
ENGLISH_BELIZE
ENGLISH_CANADA
ENGLISH_CARIBBEAN
ENGLISH_GENERAL
ENGLISH_HONG_KONG
ENGLISH_INDIA
ENGLISH_INDONESIA
ENGLISH_IRELAND
ENGLISH_JAMAICA
ENGLISH_MALAYSIA
ENGLISH_NEW_ZEALAND
ENGLISH_PHILIPPINES
ENGLISH_SINGAPORE
ENGLISH_SOUTH_AFRICA
ENGLISH_TRINIDAD

Continued on next page

Table 2.54 – continued from previous page

ENGLISH_UNITED_KINGDOM
ENGLISH_UNITED_STATES
ENGLISH_ZIMBABWE
ESTONIAN_ESTONIA
FAEROESE_FAEROE_ISLANDS
FARSI_IRAN
FILIPINO_PHILIPPINES
FINNISH_FINLAND
FRENCH_BELGIUM
FRENCH_CAMEROON
FRENCH_CANADA
FRENCH_CONGO
FRENCH_COTE_D_IVOIRE
FRENCH_FRANCE
FRENCH_HAITI
FRENCH_LUXEMBOURG
FRENCH_MALI
FRENCH_MONACO
FRENCH_MOROCCO
FRENCH_NORTH_AFRICA
FRENCH_REUNION
FRENCH_SENEGAL
FRENCH_SWITZERLAND
FRENCH_WEST_INDIES
FRENCH_ZAIRE
FRISIAN_NETHERLANDS
FULFULDE_NIGERIA
GALICIAN_SPAIN
GEORGIAN_GEORGIA
GERMAN_AUSTRIA
GERMAN_GERMANY
GERMAN_LIECHTENSTEI
GERMAN_LUXEMBOURG
GERMAN_SWITZERLAND
GREEK_GREECE
GUARANI_PARAGUAY
GUJARATI_INDIA
HAUSA_NIGERIA
HAWAIIAN_UNITED_STATES
HEBREW_ISRAEL
HINDI_INDIA
HUNGARIAN_HUNGARY
IBIBIO_NIGERIA
ICELANDIC_ICELAND
IGBO_NIGERIA
INDONESIAN_INDONESIA
INUKTITUT_CANADA
IRISH_GAELIC_IRELAND
ITALIAN_ITALY
ITALIAN_SWITZERLAND

Continued on next page

Table 2.54 – continued from previous page

JAPANESE_JAPAN
KANNADA_INDIA
KANURI_NIGERIA
KASHMIRI_INDIA
KASHMIRI_PAKISTAN
KASHMIRI_SASIA
KAZAK_KAZAKSTAN
KHMER_CAMBODIA
KIRGHIZ_KIRGHIZSTAN
KIRGHIZ_KIRGHIZ_REPUBLIC
KONKANI_INDIA
KOREAN_EXTENDED_WANSUNG_KOREA
KOREAN_JOHAB_KOREA
LAO_LAOS
LATIN
LATVIAN_LATVIA
LITHUANIAN_LITHUANIA
MACEDONIAN_MACEDONIA
MALAYALAM_INDIA
MALAY_BRUNEI_DARUSSALAM
MALAY_MALAYSIA
MALTESE_MALTA
MANIPURI_INDIA
MAORI_NEW_ZEALAND
MARATHI_INDIA
MOLDAVIAN_MOLDAVIA
MONGOLIAN_MONGOLIA
MONGOLIAN_MONGOLIA_MONGOLIAN
NEPALI_INDIA
NEPALI_NEPAL
NORWEGIAN_NORWAY_BOKMAL
NORWEGIAN_NORWAY_NYNORSK
ORIYA_INDIA
OROMO_ETHIOPIA
PAPIAMENTU_NETHERLANDS_ANTILLES
PASHTO_AFGHANISTAN
POLISH_POLAND
PORTUGUESE_BRAZIL
PORTUGUESE_PORTUGAL
PUNJABI_ARABIC_PAKISTAN
PUNJABI_INDIA
QUECHUA_BOLIVIA
QUECHUA_ECUADOR
QUECHUA_PERU
RHAETO_ROMANIC_SWITZERLAND
ROMANIAN_ROMANIA
RUSSIAN_MOLDAVIA
RUSSIAN_RUSSIA
SAAMI_LAPONIA
SAMI_INARI_FINLAND

Continued on next page

Table 2.54 – continued from previous page

SAMI_LULE_NORWAY
SAMI_LULE_SWEDEN
SAMI_NORTHERN_FINLAND
SAMI_NORTHERN_NORWAY
SAMI_NORTHERN_SWEDEN
SAMI_SKOLT_FINLAND
SAMI_SOUTHERN_NORWAY
SAMI_SOUTHERN_SWEDEN
SANSKRIT_INDIA
SCOTTISH_GAELIC_UNITED_KINGDOM
SEPEDI_SOUTH_AFRICA
SERBIAN_BOSNIA_HERZ_CYRILLIC
SERBIAN_BOSNIA_HERZ_LATIN
SERBIAN_SERBIA_CYRILLIC
SERBIAN_SERBIA_LATIN
SINDHI_INDIA
SINDHI_PAKISTAN
SINHALESE_SRI_LANKA
SLOVAK_SLOVAKIA
SLOVENE_SLOVENIA
SOMALI_SOMALIA
SORBIAN_GERMANY
SOTHO_SOUTHERN_SOUTH_AFRICA
SPANISH_ARGENTINA
SPANISH_BOLIVIA
SPANISH_CHILE
SPANISH_COLOMBIA
SPANISH_COSTA_RICA
SPANISH_DOMINICAN_REPUBLIC
SPANISH_ECUADOR
SPANISH_EL_SALVADOR
SPANISH_GUATEMALA
SPANISH_HONDURAS
SPANISH_LATIN_AMERICA
SPANISH_MEXICO
SPANISH_NICARAGUA
SPANISH_PANAMA
SPANISH_PARAGUAY
SPANISH_PERU
SPANISH_PUERTO_RICO
SPANISH_SPAIN_INTERNATIONAL_SORT
SPANISH_SPAIN_TRADITIONAL_SORT
SPANISH_UNITED_STATES
SPANISH_URUGUAY
SPANISH_VENEZUELA
SUTU_SOUTH_AFRICA
SWAHILI_KENYA
SWEDISH_FINLAND
SWEDISH_SWEDEN
SYRIAC_SYRIA

Continued on next page

Table 2.54 – continued from previous page

TAJIK_TAJIKISTAN
TAMAZIGHT_MOROCCO
TAMAZIGHT_MOROCCO_LATIN
TAMIL_INDIA
TATAR_TATARSTAN
TELUGU_INDIA
THAI_THAILAND
TIBETAN_BHUTAN
TIBETAN_CHINA
TIGRIGNA_ERYTHREA
TIGRIGNA_ERYTREA
TIGRIGNA_ETHIOPIA
TSONGA_SOUTH_AFRICA
TSWANA_SOUTH_AFRICA
TURKISH_TURKEY
TURKMEN_TURKMENISTAN
UIGHUR_CHINA
UKRAINIAN_UKRAINE
URDU_INDIA
URDU_PAKISTAN
UZBEK_UZBEKISTAN_CYRILLIC
UZBEK_UZBEKISTAN_LATIN
VENDA_SOUTH_AFRICA
VIETNAMESE_VIET_NAM
WELSH_WALES
XHOSA_SOUTH_AFRICA
YIDDISH_GERMANY
YI_CHINA
YORUBA_NIGERIA
ZULU_SOUTH_AFRICA

freetypy.TT_MS_LANGID.AFRIKAANS_SOUTH_AFRICA

TT_MS_LANGID.**AFRIKAANS_SOUTH_AFRICA = freetypy.TT_MS_LANGID.AFRIKAANS_SOUTH_AFRICA**

freetypy.TT_MS_LANGID.ALBANIAN_ALBANIA

TT_MS_LANGID.**ALBANIAN_ALBANIA = freetypy.TT_MS_LANGID.ALBANIAN_ALBANIA**

freetypy.TT_MS_LANGID.AMHARIC_ETHIOPIA

TT_MS_LANGID.**AMHARIC_ETHIOPIA = freetypy.TT_MS_LANGID.AMHARIC_ETHIOPIA**

freetypy.TT_MS_LANGID.ARABIC_ALGERIA

TT_MS_LANGID.**ARABIC_ALGERIA = freetypy.TT_MS_LANGID.ARABIC_ALGERIA**

freetypy.TT_MS_LANGID.ARABIC_BAHRAIN

TT_MS_LANGID.**ARABIC_BAHRAIN** = freetypy.TT_MS_LANGID.ARABIC_BAHRAIN

freetypy.TT_MS_LANGID.ARABIC_EGYPT

TT_MS_LANGID.**ARABIC_EGYPT** = freetypy.TT_MS_LANGID.ARABIC_EGYPT

freetypy.TT_MS_LANGID.ARABIC_GENERAL

TT_MS_LANGID.**ARABIC_GENERAL** = freetypy.TT_MS_LANGID.ARABIC_GENERAL

freetypy.TT_MS_LANGID.ARABIC_IRAQ

TT_MS_LANGID.**ARABIC_IRAQ** = freetypy.TT_MS_LANGID.ARABIC_IRAQ

freetypy.TT_MS_LANGID.ARABIC_JORDAN

TT_MS_LANGID.**ARABIC_JORDAN** = freetypy.TT_MS_LANGID.ARABIC_JORDAN

freetypy.TT_MS_LANGID.ARABIC_KUWAIT

TT_MS_LANGID.**ARABIC_KUWAIT** = freetypy.TT_MS_LANGID.ARABIC_KUWAIT

freetypy.TT_MS_LANGID.ARABIC_LEBANON

TT_MS_LANGID.**ARABIC_LEBANON** = freetypy.TT_MS_LANGID.ARABIC_LEBANON

freetypy.TT_MS_LANGID.ARABIC_LIBYA

TT_MS_LANGID.**ARABIC_LIBYA** = freetypy.TT_MS_LANGID.ARABIC_LIBYA

freetypy.TT_MS_LANGID.ARABIC_MOROCCO

TT_MS_LANGID.**ARABIC_MOROCCO** = freetypy.TT_MS_LANGID.ARABIC_MOROCCO

freetypy.TT_MS_LANGID.ARABIC_OMAN

TT_MS_LANGID.**ARABIC_OMAN** = freetypy.TT_MS_LANGID.ARABIC_OMAN

freetypy.TT_MS_LANGID.ARABIC_QATAR

TT_MS_LANGID.**ARABIC_QATAR** = freetypy.TT_MS_LANGID.ARABIC_QATAR

freetypy.TT_MS_LANGID.ARABIC_SAUDI_ARABIA

TT_MS_LANGID.**ARABIC_SAUDI_ARABIA** = freetypy.TT_MS_LANGID.ARABIC_SAUDI_ARABIA

freetypy.TT_MS_LANGID.ARABIC_SYRIA

TT_MS_LANGID.**ARABIC_SYRIA** = freetypy.TT_MS_LANGID.ARABIC_SYRIA

freetypy.TT_MS_LANGID.ARABIC_TUNISIA

TT_MS_LANGID.**ARABIC_TUNISIA** = freetypy.TT_MS_LANGID.ARABIC_TUNISIA

freetypy.TT_MS_LANGID.ARABIC_UAE

TT_MS_LANGID.**ARABIC_UAE** = freetypy.TT_MS_LANGID.ARABIC_UAE

freetypy.TT_MS_LANGID.ARABIC_YEMEN

TT_MS_LANGID.**ARABIC_YEMEN** = freetypy.TT_MS_LANGID.ARABIC_YEMEN

freetypy.TT_MS_LANGID.ARMENIAN_ARMENIA

TT_MS_LANGID.**ARMENIAN_ARMENIA** = freetypy.TT_MS_LANGID.ARMENIAN_ARMENIA

freetypy.TT_MS_LANGID.ASSAMESE_INDIA

TT_MS_LANGID.**ASSAMESE_INDIA** = freetypy.TT_MS_LANGID.ASSAMESE_INDIA

freetypy.TT_MS_LANGID.AZERI_AZERBAIJAN_CYRILLIC

TT_MS_LANGID.**AZERI_AZERBAIJAN_CYRILLIC** = freetypy.TT_MS_LANGID.AZERI_AZERBAIJAN_CYRILLIC

freetypy.TT_MS_LANGID.AZERI_AZERBAIJAN_LATIN

TT_MS_LANGID.**AZERI_AZERBAIJAN_LATIN** = freetypy.TT_MS_LANGID.AZERI_AZERBAIJAN_LATIN

freetypy.TT_MS_LANGID.BASQUE_SPAIN

TT_MS_LANGID.**BASQUE_SPAIN** = freetypy.TT_MS_LANGID.BASQUE_SPAIN

freetypy.TT_MS_LANGID.BELARUSIAN_BELARUS

TT_MS_LANGID.**BELARUSIAN_BELARUS** = freetypy.TT_MS_LANGID.BELARUSIAN_BELARUS

freetypy.TT_MS_LANGID.BENGALI_BANGLADESH

TT_MS_LANGID.**BENGALI_BANGLADESH = freetypy.TT_MS_LANGID.BENGALI_BANGLADESH**

freetypy.TT_MS_LANGID.BENGALI_INDIA

TT_MS_LANGID.**BENGALI_INDIA = freetypy.TT_MS_LANGID.BENGALI_INDIA**

freetypy.TT_MS_LANGID.BOSNIAN_BOSNIA_HERZEGOVINA

TT_MS_LANGID.**BOSNIAN_BOSNIA_HERZEGOVINA = freetypy.TT_MS_LANGID.BOSNIAN_BOSNIA_HERZEGOVI**

freetypy.TT_MS_LANGID.BULGARIAN_BULGARIA

TT_MS_LANGID.**BULGARIAN_BULGARIA = freetypy.TT_MS_LANGID.BULGARIAN_BULGARIA**

freetypy.TT_MS_LANGID.BURMESE_MYANMAR

TT_MS_LANGID.**BURMESE_MYANMAR = freetypy.TT_MS_LANGID.BURMESE_MYANMAR**

freetypy.TT_MS_LANGID.CATALAN_SPAIN

TT_MS_LANGID.**CATALAN_SPAIN = freetypy.TT_MS_LANGID.CATALAN_SPAIN**

freetypy.TT_MS_LANGID.CHEROKEE_UNITED_STATES

TT_MS_LANGID.**CHEROKEE_UNITED_STATES = freetypy.TT_MS_LANGID.CHEROKEE_UNITED_STATES**

freetypy.TT_MS_LANGID.CHINESE_GENERAL

TT_MS_LANGID.**CHINESE_GENERAL = freetypy.TT_MS_LANGID.CHINESE_GENERAL**

freetypy.TT_MS_LANGID.CHINESE_HONG_KONG

TT_MS_LANGID.**CHINESE_HONG_KONG = freetypy.TT_MS_LANGID.CHINESE_HONG_KONG**

freetypy.TT_MS_LANGID.CHINESE_MACAU

TT_MS_LANGID.**CHINESE_MACAU = freetypy.TT_MS_LANGID.CHINESE_MACAU**

freetypy.TT_MS_LANGID.CHINESE_PRC

TT_MS_LANGID.**CHINESE_PRC = freetypy.TT_MS_LANGID.CHINESE_PRC**

freetypy.TT_MS_LANGID.CHINESE_SINGAPORE

TT_MS_LANGID.**CHINESE_SINGAPORE** = freetypy.TT_MS_LANGID.CHINESE_SINGAPORE

freetypy.TT_MS_LANGID.CHINESE_TAIWAN

TT_MS_LANGID.**CHINESE_TAIWAN** = freetypy.TT_MS_LANGID.CHINESE_TAIWAN

freetypy.TT_MS_LANGID.CLASSIC_LITHUANIAN_LITHUANIA

TT_MS_LANGID.**CLASSIC_LITHUANIAN_LITHUANIA** = freetypy.TT_MS_LANGID.CLASSIC_LITHUANIAN_LITH

freetypy.TT_MS_LANGID.CROATIAN_BOSNIA_HERZEGOVINA

TT_MS_LANGID.**CROATIAN_BOSNIA_HERZEGOVINA** = freetypy.TT_MS_LANGID.CROATIAN_BOSNIA_HERZEGO

freetypy.TT_MS_LANGID.CROATIAN_CROATIA

TT_MS_LANGID.**CROATIAN_CROATIA** = freetypy.TT_MS_LANGID.CROATIAN_CROATIA

freetypy.TT_MS_LANGID.CZECH_CZECH_REPUBLIC

TT_MS_LANGID.**CZECH_CZECH_REPUBLIC** = freetypy.TT_MS_LANGID.CZECH_CZECH_REPUBLIC

freetypy.TT_MS_LANGID.DANISH_DENMARK

TT_MS_LANGID.**DANISH_DENMARK** = freetypy.TT_MS_LANGID.DANISH_DENMARK

freetypy.TT_MS_LANGID.DHIVEHI_MALDIVES

TT_MS_LANGID.**DHIVEHI_MALDIVES** = freetypy.TT_MS_LANGID.DIVEHI_MALDIVES

freetypy.TT_MS_LANGID.DIVEHI_MALDIVES

TT_MS_LANGID.**DIVEHI_MALDIVES** = freetypy.TT_MS_LANGID.DIVEHI_MALDIVES

freetypy.TT_MS_LANGID.DUTCH_BELGIUM

TT_MS_LANGID.**DUTCH_BELGIUM** = freetypy.TT_MS_LANGID.DUTCH_BELGIUM

freetypy.TT_MS_LANGID.DUTCH_NETHERLANDS

TT_MS_LANGID.**DUTCH_NETHERLANDS** = freetypy.TT_MS_LANGID.DUTCH_NETHERLANDS

freetypy.TT_MS_LANGID.EDO_NIGERIA

`TT_MS_LANGID.`**`EDO_NIGERIA`** = **freetypy.TT_MS_LANGID.EDO_NIGERIA**

freetypy.TT_MS_LANGID.ENGLISH_AUSTRALIA

`TT_MS_LANGID.`**`ENGLISH_AUSTRALIA`** = **freetypy.TT_MS_LANGID.ENGLISH_AUSTRALIA**

freetypy.TT_MS_LANGID.ENGLISH_BELIZE

`TT_MS_LANGID.`**`ENGLISH_BELIZE`** = **freetypy.TT_MS_LANGID.ENGLISH_BELIZE**

freetypy.TT_MS_LANGID.ENGLISH_CANADA

`TT_MS_LANGID.`**`ENGLISH_CANADA`** = **freetypy.TT_MS_LANGID.ENGLISH_CANADA**

freetypy.TT_MS_LANGID.ENGLISH_CARIBBEAN

`TT_MS_LANGID.`**`ENGLISH_CARIBBEAN`** = **freetypy.TT_MS_LANGID.ENGLISH_CARIBBEAN**

freetypy.TT_MS_LANGID.ENGLISH_GENERAL

`TT_MS_LANGID.`**`ENGLISH_GENERAL`** = **freetypy.TT_MS_LANGID.ENGLISH_GENERAL**

freetypy.TT_MS_LANGID.ENGLISH_HONG_KONG

`TT_MS_LANGID.`**`ENGLISH_HONG_KONG`** = **freetypy.TT_MS_LANGID.ENGLISH_HONG_KONG**

freetypy.TT_MS_LANGID.ENGLISH_INDIA

`TT_MS_LANGID.`**`ENGLISH_INDIA`** = **freetypy.TT_MS_LANGID.ENGLISH_INDIA**

freetypy.TT_MS_LANGID.ENGLISH_INDONESIA

`TT_MS_LANGID.`**`ENGLISH_INDONESIA`** = **freetypy.TT_MS_LANGID.ENGLISH_INDONESIA**

freetypy.TT_MS_LANGID.ENGLISH_IRELAND

`TT_MS_LANGID.`**`ENGLISH_IRELAND`** = **freetypy.TT_MS_LANGID.ENGLISH_IRELAND**

freetypy.TT_MS_LANGID.ENGLISH_JAMAICA

`TT_MS_LANGID.`**`ENGLISH_JAMAICA`** = **freetypy.TT_MS_LANGID.ENGLISH_JAMAICA**

freetypy.TT_MS_LANGID.ENGLISH_MALAYSIA

`TT_MS_LANGID.`**`ENGLISH_MALAYSIA`** = **freetypy.TT_MS_LANGID.ENGLISH_MALAYSIA**

freetypy.TT_MS_LANGID.ENGLISH_NEW_ZEALAND

`TT_MS_LANGID.`**`ENGLISH_NEW_ZEALAND`** = **freetypy.TT_MS_LANGID.ENGLISH_NEW_ZEALAND**

freetypy.TT_MS_LANGID.ENGLISH_PHILIPPINES

`TT_MS_LANGID.`**`ENGLISH_PHILIPPINES`** = **freetypy.TT_MS_LANGID.ENGLISH_PHILIPPINES**

freetypy.TT_MS_LANGID.ENGLISH_SINGAPORE

`TT_MS_LANGID.`**`ENGLISH_SINGAPORE`** = **freetypy.TT_MS_LANGID.ENGLISH_SINGAPORE**

freetypy.TT_MS_LANGID.ENGLISH_SOUTH_AFRICA

`TT_MS_LANGID.`**`ENGLISH_SOUTH_AFRICA`** = **freetypy.TT_MS_LANGID.ENGLISH_SOUTH_AFRICA**

freetypy.TT_MS_LANGID.ENGLISH_TRINIDAD

`TT_MS_LANGID.`**`ENGLISH_TRINIDAD`** = **freetypy.TT_MS_LANGID.ENGLISH_TRINIDAD**

freetypy.TT_MS_LANGID.ENGLISH_UNITED_KINGDOM

`TT_MS_LANGID.`**`ENGLISH_UNITED_KINGDOM`** = **freetypy.TT_MS_LANGID.ENGLISH_UNITED_KINGDOM**

freetypy.TT_MS_LANGID.ENGLISH_UNITED_STATES

`TT_MS_LANGID.`**`ENGLISH_UNITED_STATES`** = **freetypy.TT_MS_LANGID.ENGLISH_UNITED_STATES**

freetypy.TT_MS_LANGID.ENGLISH_ZIMBABWE

`TT_MS_LANGID.`**`ENGLISH_ZIMBABWE`** = **freetypy.TT_MS_LANGID.ENGLISH_ZIMBABWE**

freetypy.TT_MS_LANGID.ESTONIAN_ESTONIA

`TT_MS_LANGID.`**`ESTONIAN_ESTONIA`** = **freetypy.TT_MS_LANGID.ESTONIAN_ESTONIA**

freetypy.TT_MS_LANGID.FAEROESE_FAEROE_ISLANDS

`TT_MS_LANGID.`**`FAEROESE_FAEROE_ISLANDS`** = **freetypy.TT_MS_LANGID.FAEROESE_FAEROE_ISLANDS**

freetypy.TT_MS_LANGID.FARSI_IRAN

TT_MS_LANGID.**FARSI_IRAN** = freetypy.TT_MS_LANGID.FARSI_IRAN

freetypy.TT_MS_LANGID.FILIPINO_PHILIPPINES

TT_MS_LANGID.**FILIPINO_PHILIPPINES** = freetypy.TT_MS_LANGID.FILIPINO_PHILIPPINES

freetypy.TT_MS_LANGID.FINNISH_FINLAND

TT_MS_LANGID.**FINNISH_FINLAND** = freetypy.TT_MS_LANGID.FINNISH_FINLAND

freetypy.TT_MS_LANGID.FRENCH_BELGIUM

TT_MS_LANGID.**FRENCH_BELGIUM** = freetypy.TT_MS_LANGID.FRENCH_BELGIUM

freetypy.TT_MS_LANGID.FRENCH_CAMEROON

TT_MS_LANGID.**FRENCH_CAMEROON** = freetypy.TT_MS_LANGID.FRENCH_CAMEROON

freetypy.TT_MS_LANGID.FRENCH_CANADA

TT_MS_LANGID.**FRENCH_CANADA** = freetypy.TT_MS_LANGID.FRENCH_CANADA

freetypy.TT_MS_LANGID.FRENCH_CONGO

TT_MS_LANGID.**FRENCH_CONGO** = freetypy.TT_MS_LANGID.FRENCH_ZAIRE

freetypy.TT_MS_LANGID.FRENCH_COTE_D_IVOIRE

TT_MS_LANGID.**FRENCH_COTE_D_IVOIRE** = freetypy.TT_MS_LANGID.FRENCH_COTE_D_IVOIRE

freetypy.TT_MS_LANGID.FRENCH_FRANCE

TT_MS_LANGID.**FRENCH_FRANCE** = freetypy.TT_MS_LANGID.FRENCH_FRANCE

freetypy.TT_MS_LANGID.FRENCH_HAITI

TT_MS_LANGID.**FRENCH_HAITI** = freetypy.TT_MS_LANGID.FRENCH_HAITI

freetypy.TT_MS_LANGID.FRENCH_LUXEMBOURG

TT_MS_LANGID.**FRENCH_LUXEMBOURG** = freetypy.TT_MS_LANGID.FRENCH_LUXEMBOURG

freetypy.TT_MS_LANGID.FRENCH_MALI

TT_MS_LANGID.**FRENCH_MALI** = freetypy.TT_MS_LANGID.FRENCH_MALI

freetypy.TT_MS_LANGID.FRENCH_MONACO

TT_MS_LANGID.**FRENCH_MONACO** = freetypy.TT_MS_LANGID.FRENCH_MONACO

freetypy.TT_MS_LANGID.FRENCH_MOROCCO

TT_MS_LANGID.**FRENCH_MOROCCO** = freetypy.TT_MS_LANGID.FRENCH_MOROCCO

freetypy.TT_MS_LANGID.FRENCH_NORTH_AFRICA

TT_MS_LANGID.**FRENCH_NORTH_AFRICA** = freetypy.TT_MS_LANGID.FRENCH_NORTH_AFRICA

freetypy.TT_MS_LANGID.FRENCH_REUNION

TT_MS_LANGID.**FRENCH_REUNION** = freetypy.TT_MS_LANGID.FRENCH_REUNION

freetypy.TT_MS_LANGID.FRENCH_SENEGAL

TT_MS_LANGID.**FRENCH_SENEGAL** = freetypy.TT_MS_LANGID.FRENCH_SENEGAL

freetypy.TT_MS_LANGID.FRENCH_SWITZERLAND

TT_MS_LANGID.**FRENCH_SWITZERLAND** = freetypy.TT_MS_LANGID.FRENCH_SWITZERLAND

freetypy.TT_MS_LANGID.FRENCH_WEST_INDIES

TT_MS_LANGID.**FRENCH_WEST_INDIES** = freetypy.TT_MS_LANGID.FRENCH_WEST_INDIES

freetypy.TT_MS_LANGID.FRENCH_ZAIRE

TT_MS_LANGID.**FRENCH_ZAIRE** = freetypy.TT_MS_LANGID.FRENCH_ZAIRE

freetypy.TT_MS_LANGID.FRISIAN_NETHERLANDS

TT_MS_LANGID.**FRISIAN_NETHERLANDS** = freetypy.TT_MS_LANGID.FRISIAN_NETHERLANDS

freetypy.TT_MS_LANGID.FULFULDE_NIGERIA

TT_MS_LANGID.**FULFULDE_NIGERIA** = freetypy.TT_MS_LANGID.FULFULDE_NIGERIA

freetypy.TT_MS_LANGID.GALICIAN_SPAIN

TT_MS_LANGID.**GALICIAN_SPAIN** = freetypy.TT_MS_LANGID.GALICIAN_SPAIN

freetypy.TT_MS_LANGID.GEORGIAN_GEORGIA

TT_MS_LANGID.**GEORGIAN_GEORGIA** = freetypy.TT_MS_LANGID.GEORGIAN_GEORGIA

freetypy.TT_MS_LANGID.GERMAN_AUSTRIA

TT_MS_LANGID.**GERMAN_AUSTRIA** = freetypy.TT_MS_LANGID.GERMAN_AUSTRIA

freetypy.TT_MS_LANGID.GERMAN_GERMANY

TT_MS_LANGID.**GERMAN_GERMANY** = freetypy.TT_MS_LANGID.GERMAN_GERMANY

freetypy.TT_MS_LANGID.GERMAN_LIECHTENSTEI

TT_MS_LANGID.**GERMAN_LIECHTENSTEI** = freetypy.TT_MS_LANGID.GERMAN_LIECHTENSTEI

freetypy.TT_MS_LANGID.GERMAN_LUXEMBOURG

TT_MS_LANGID.**GERMAN_LUXEMBOURG** = freetypy.TT_MS_LANGID.GERMAN_LUXEMBOURG

freetypy.TT_MS_LANGID.GERMAN_SWITZERLAND

TT_MS_LANGID.**GERMAN_SWITZERLAND** = freetypy.TT_MS_LANGID.GERMAN_SWITZERLAND

freetypy.TT_MS_LANGID.GREEK_GREECE

TT_MS_LANGID.**GREEK_GREECE** = freetypy.TT_MS_LANGID.GREEK_GREECE

freetypy.TT_MS_LANGID.GUARANI_PARAGUAY

TT_MS_LANGID.**GUARANI_PARAGUAY** = freetypy.TT_MS_LANGID.GUARANI_PARAGUAY

freetypy.TT_MS_LANGID.GUJARATI_INDIA

TT_MS_LANGID.**GUJARATI_INDIA** = freetypy.TT_MS_LANGID.GUJARATI_INDIA

freetypy.TT_MS_LANGID.HAUSA_NIGERIA

TT_MS_LANGID.**HAUSA_NIGERIA** = freetypy.TT_MS_LANGID.HAUSA_NIGERIA

freetypy.TT_MS_LANGID.HAWAIIAN_UNITED_STATES

TT_MS_LANGID.**HAWAIIAN_UNITED_STATES** = freetypy.TT_MS_LANGID.HAWAIIAN_UNITED_STATES

freetypy.TT_MS_LANGID.HEBREW_ISRAEL

TT_MS_LANGID.**HEBREW_ISRAEL** = freetypy.TT_MS_LANGID.HEBREW_ISRAEL

freetypy.TT_MS_LANGID.HINDI_INDIA

TT_MS_LANGID.**HINDI_INDIA** = freetypy.TT_MS_LANGID.HINDI_INDIA

freetypy.TT_MS_LANGID.HUNGARIAN_HUNGARY

TT_MS_LANGID.**HUNGARIAN_HUNGARY** = freetypy.TT_MS_LANGID.HUNGARIAN_HUNGARY

freetypy.TT_MS_LANGID.IBIBIO_NIGERIA

TT_MS_LANGID.**IBIBIO_NIGERIA** = freetypy.TT_MS_LANGID.IBIBIO_NIGERIA

freetypy.TT_MS_LANGID.ICELANDIC_ICELAND

TT_MS_LANGID.**ICELANDIC_ICELAND** = freetypy.TT_MS_LANGID.ICELANDIC_ICELAND

freetypy.TT_MS_LANGID.IGBO_NIGERIA

TT_MS_LANGID.**IGBO_NIGERIA** = freetypy.TT_MS_LANGID.IGBO_NIGERIA

freetypy.TT_MS_LANGID.INDONESIAN_INDONESIA

TT_MS_LANGID.**INDONESIAN_INDONESIA** = freetypy.TT_MS_LANGID.INDONESIAN_INDONESIA

freetypy.TT_MS_LANGID.INUKTITUT_CANADA

TT_MS_LANGID.**INUKTITUT_CANADA** = freetypy.TT_MS_LANGID.INUKTITUT_CANADA

freetypy.TT_MS_LANGID.IRISH_GAELIC_IRELAND

TT_MS_LANGID.**IRISH_GAELIC_IRELAND** = freetypy.TT_MS_LANGID.IRISH_GAELIC_IRELAND

freetypy.TT_MS_LANGID.ITALIAN_ITALY

TT_MS_LANGID.**ITALIAN_ITALY** = freetypy.TT_MS_LANGID.ITALIAN_ITALY

freetypy.TT_MS_LANGID.ITALIAN_SWITZERLAND

TT_MS_LANGID.**ITALIAN_SWITZERLAND** = freetypy.TT_MS_LANGID.ITALIAN_SWITZERLAND

freetypy.TT_MS_LANGID.JAPANESE_JAPAN

TT_MS_LANGID.**JAPANESE_JAPAN** = freetypy.TT_MS_LANGID.JAPANESE_JAPAN

freetypy.TT_MS_LANGID.KANNADA_INDIA

TT_MS_LANGID.**KANNADA_INDIA** = freetypy.TT_MS_LANGID.KANNADA_INDIA

freetypy.TT_MS_LANGID.KANURI_NIGERIA

TT_MS_LANGID.**KANURI_NIGERIA** = freetypy.TT_MS_LANGID.KANURI_NIGERIA

freetypy.TT_MS_LANGID.KASHMIRI_INDIA

TT_MS_LANGID.**KASHMIRI_INDIA** = freetypy.TT_MS_LANGID.KASHMIRI_INDIA

freetypy.TT_MS_LANGID.KASHMIRI_PAKISTAN

TT_MS_LANGID.**KASHMIRI_PAKISTAN** = freetypy.TT_MS_LANGID.KASHMIRI_PAKISTAN

freetypy.TT_MS_LANGID.KASHMIRI_SASIA

TT_MS_LANGID.**KASHMIRI_SASIA** = freetypy.TT_MS_LANGID.KASHMIRI_INDIA

freetypy.TT_MS_LANGID.KAZAK_KAZAKSTAN

TT_MS_LANGID.**KAZAK_KAZAKSTAN** = freetypy.TT_MS_LANGID.KAZAK_KAZAKSTAN

freetypy.TT_MS_LANGID.KHMER_CAMBODIA

TT_MS_LANGID.**KHMER_CAMBODIA** = freetypy.TT_MS_LANGID.KHMER_CAMBODIA

freetypy.TT_MS_LANGID.KIRGHIZ_KIRGHIZSTAN

TT_MS_LANGID.**KIRGHIZ_KIRGHIZSTAN** = freetypy.TT_MS_LANGID.KIRGHIZ_KIRGHIZ_REPUBLIC

freetypy.TT_MS_LANGID.KIRGHIZ_KIRGHIZ_REPUBLIC

TT_MS_LANGID.**KIRGHIZ_KIRGHIZ_REPUBLIC** = freetypy.TT_MS_LANGID.KIRGHIZ_KIRGHIZ_REPUBLIC

freetypy.TT_MS_LANGID.KONKANI_INDIA

TT_MS_LANGID.**KONKANI_INDIA** = freetypy.TT_MS_LANGID.KONKANI_INDIA

freetypy.TT_MS_LANGID.KOREAN_EXTENDED_WANSUNG_KOREA

TT_MS_LANGID.**KOREAN_EXTENDED_WANSUNG_KOREA** = freetypy.TT_MS_LANGID.KOREAN_EXTENDED_WANS

freetypy.TT_MS_LANGID.KOREAN_JOHAB_KOREA

TT_MS_LANGID.**KOREAN_JOHAB_KOREA** = freetypy.TT_MS_LANGID.KOREAN_JOHAB_KOREA

freetypy.TT_MS_LANGID.LAO_LAOS

TT_MS_LANGID.**LAO_LAOS** = freetypy.TT_MS_LANGID.LAO_LAOS

freetypy.TT_MS_LANGID.LATIN

TT_MS_LANGID.**LATIN** = freetypy.TT_MS_LANGID.LATIN

freetypy.TT_MS_LANGID.LATVIAN_LATVIA

TT_MS_LANGID.**LATVIAN_LATVIA** = freetypy.TT_MS_LANGID.LATVIAN_LATVIA

freetypy.TT_MS_LANGID.LITHUANIAN_LITHUANIA

TT_MS_LANGID.**LITHUANIAN_LITHUANIA** = freetypy.TT_MS_LANGID.LITHUANIAN_LITHUANIA

freetypy.TT_MS_LANGID.MACEDONIAN_MACEDONIA

TT_MS_LANGID.**MACEDONIAN_MACEDONIA** = freetypy.TT_MS_LANGID.MACEDONIAN_MACEDONIA

freetypy.TT_MS_LANGID.MALAYALAM_INDIA

TT_MS_LANGID.**MALAYALAM_INDIA** = freetypy.TT_MS_LANGID.MALAYALAM_INDIA

freetypy.TT_MS_LANGID.MALAY_BRUNEI_DARUSSALAM

TT_MS_LANGID.**MALAY_BRUNEI_DARUSSALAM** = freetypy.TT_MS_LANGID.MALAY_BRUNEI_DARUSSALAM

freetypy.TT_MS_LANGID.MALAY_MALAYSIA

TT_MS_LANGID.**MALAY_MALAYSIA** = freetypy.TT_MS_LANGID.MALAY_MALAYSIA

freetypy.TT_MS_LANGID.MALTESE_MALTA

TT_MS_LANGID.**MALTESE_MALTA** = freetypy.TT_MS_LANGID.MALTESE_MALTA

freetypy.TT_MS_LANGID.MANIPURI_INDIA

TT_MS_LANGID.**MANIPURI_INDIA** = freetypy.TT_MS_LANGID.MANIPURI_INDIA

freetypy.TT_MS_LANGID.MAORI_NEW_ZEALAND

TT_MS_LANGID.**MAORI_NEW_ZEALAND** = freetypy.TT_MS_LANGID.MAORI_NEW_ZEALAND

freetypy.TT_MS_LANGID.MARATHI_INDIA

TT_MS_LANGID.**MARATHI_INDIA** = freetypy.TT_MS_LANGID.MARATHI_INDIA

freetypy.TT_MS_LANGID.MOLDAVIAN_MOLDAVIA

TT_MS_LANGID.**MOLDAVIAN_MOLDAVIA** = freetypy.TT_MS_LANGID.MOLDAVIAN_MOLDAVIA

freetypy.TT_MS_LANGID.MONGOLIAN_MONGOLIA

TT_MS_LANGID.**MONGOLIAN_MONGOLIA** = freetypy.TT_MS_LANGID.MONGOLIAN_MONGOLIA

freetypy.TT_MS_LANGID.MONGOLIAN_MONGOLIA_MONGOLIAN

TT_MS_LANGID.**MONGOLIAN_MONGOLIA_MONGOLIAN** = freetypy.TT_MS_LANGID.MONGOLIAN_MONGOLIA_M

freetypy.TT_MS_LANGID.NEPALI_INDIA

TT_MS_LANGID.**NEPALI_INDIA** = freetypy.TT_MS_LANGID.NEPALI_INDIA

freetypy.TT_MS_LANGID.NEPALI_NEPAL

TT_MS_LANGID.**NEPALI_NEPAL** = freetypy.TT_MS_LANGID.NEPALI_NEPAL

freetypy.TT_MS_LANGID.NORWEGIAN_NORWAY_BOKMAL

TT_MS_LANGID.**NORWEGIAN_NORWAY_BOKMAL** = freetypy.TT_MS_LANGID.NORWEGIAN_NORWAY_BOKMAL

freetypy.TT_MS_LANGID.NORWEGIAN_NORWAY_NYNORSK

TT_MS_LANGID.**NORWEGIAN_NORWAY_NYNORSK** = freetypy.TT_MS_LANGID.NORWEGIAN_NORWAY_NYNORSK

freetypy.TT_MS_LANGID.ORIYA_INDIA

TT_MS_LANGID.**ORIYA_INDIA** = freetypy.TT_MS_LANGID.ORIYA_INDIA

freetypy.TT_MS_LANGID.OROMO_ETHIOPIA

TT_MS_LANGID.**OROMO_ETHIOPIA** = freetypy.TT_MS_LANGID.OROMO_ETHIOPIA

freetypy.TT_MS_LANGID.PAPIAMENTU_NETHERLANDS_ANTILLES

TT_MS_LANGID.**PAPIAMENTU_NETHERLANDS_ANTILLES** = freetypy.TT_MS_LANGID.PAPIAMENTU_NETHERLA

freetypy.TT_MS_LANGID.PASHTO_AFGHANISTAN

TT_MS_LANGID.**PASHTO_AFGHANISTAN** = freetypy.TT_MS_LANGID.PASHTO_AFGHANISTAN

freetypy.TT_MS_LANGID.POLISH_POLAND

TT_MS_LANGID.**POLISH_POLAND** = freetypy.TT_MS_LANGID.POLISH_POLAND

freetypy.TT_MS_LANGID.PORTUGUESE_BRAZIL

TT_MS_LANGID.**PORTUGUESE_BRAZIL** = freetypy.TT_MS_LANGID.PORTUGUESE_BRAZIL

freetypy.TT_MS_LANGID.PORTUGUESE_PORTUGAL

TT_MS_LANGID.**PORTUGUESE_PORTUGAL** = freetypy.TT_MS_LANGID.PORTUGUESE_PORTUGAL

freetypy.TT_MS_LANGID.PUNJABI_ARABIC_PAKISTAN

TT_MS_LANGID.**PUNJABI_ARABIC_PAKISTAN** = freetypy.TT_MS_LANGID.PUNJABI_ARABIC_PAKISTAN

freetypy.TT_MS_LANGID.PUNJABI_INDIA

TT_MS_LANGID.**PUNJABI_INDIA** = freetypy.TT_MS_LANGID.PUNJABI_INDIA

freetypy.TT_MS_LANGID.QUECHUA_BOLIVIA

TT_MS_LANGID.**QUECHUA_BOLIVIA** = freetypy.TT_MS_LANGID.QUECHUA_BOLIVIA

freetypy.TT_MS_LANGID.QUECHUA_ECUADOR

TT_MS_LANGID.**QUECHUA_ECUADOR** = freetypy.TT_MS_LANGID.QUECHUA_ECUADOR

freetypy.TT_MS_LANGID.QUECHUA_PERU

TT_MS_LANGID.**QUECHUA_PERU** = freetypy.TT_MS_LANGID.QUECHUA_PERU

freetypy.TT_MS_LANGID.RHAETO_ROMANIC_SWITZERLAND

TT_MS_LANGID.**RHAETO_ROMANIC_SWITZERLAND** = freetypy.TT_MS_LANGID.RHAETO_ROMANIC_SWITZERL

freetypy.TT_MS_LANGID.ROMANIAN_ROMANIA

TT_MS_LANGID.**ROMANIAN_ROMANIA** = freetypy.TT_MS_LANGID.ROMANIAN_ROMANIA

freetypy.TT_MS_LANGID.RUSSIAN_MOLDAVIA

TT_MS_LANGID.**RUSSIAN_MOLDAVIA** = freetypy.TT_MS_LANGID.RUSSIAN_MOLDAVIA

freetypy.TT_MS_LANGID.RUSSIAN_RUSSIA

TT_MS_LANGID.**RUSSIAN_RUSSIA** = freetypy.TT_MS_LANGID.RUSSIAN_RUSSIA

freetypy.TT_MS_LANGID.SAAMI_LAPONIA

TT_MS_LANGID.**SAAMI_LAPONIA** = freetypy.TT_MS_LANGID.SAAMI_LAPONIA

freetypy.TT_MS_LANGID.SAMI_INARI_FINLAND

TT_MS_LANGID.**SAMI_INARI_FINLAND** = freetypy.TT_MS_LANGID.SAMI_INARI_FINLAND

freetypy.TT_MS_LANGID.SAMI_LULE_NORWAY

TT_MS_LANGID.**SAMI_LULE_NORWAY** = freetypy.TT_MS_LANGID.SAMI_LULE_NORWAY

freetypy.TT_MS_LANGID.SAMI_LULE_SWEDEN

TT_MS_LANGID.**SAMI_LULE_SWEDEN** = freetypy.TT_MS_LANGID.SAMI_LULE_SWEDEN

freetypy.TT_MS_LANGID.SAMI_NORTHERN_FINLAND

TT_MS_LANGID.**SAMI_NORTHERN_FINLAND** = freetypy.TT_MS_LANGID.SAMI_NORTHERN_FINLAND

freetypy.TT_MS_LANGID.SAMI_NORTHERN_NORWAY

TT_MS_LANGID.**SAMI_NORTHERN_NORWAY** = freetypy.TT_MS_LANGID.SAAMI_LAPONIA

freetypy.TT_MS_LANGID.SAMI_NORTHERN_SWEDEN

TT_MS_LANGID.**SAMI_NORTHERN_SWEDEN** = freetypy.TT_MS_LANGID.SAMI_NORTHERN_SWEDEN

freetypy.TT_MS_LANGID.SAMI_SKOLT_FINLAND

TT_MS_LANGID.**SAMI_SKOLT_FINLAND** = freetypy.TT_MS_LANGID.SAMI_SKOLT_FINLAND

freetypy.TT_MS_LANGID.SAMI_SOUTHERN_NORWAY

TT_MS_LANGID.**SAMI_SOUTHERN_NORWAY** = freetypy.TT_MS_LANGID.SAMI_SOUTHERN_NORWAY

freetypy.TT_MS_LANGID.SAMI_SOUTHERN_SWEDEN

TT_MS_LANGID.**SAMI_SOUTHERN_SWEDEN** = freetypy.TT_MS_LANGID.SAMI_SOUTHERN_SWEDEN

freetypy.TT_MS_LANGID.SANSKRIT_INDIA

TT_MS_LANGID.**SANSKRIT_INDIA** = freetypy.TT_MS_LANGID.SANSKRIT_INDIA

freetypy.TT_MS_LANGID.SCOTTISH_GAELIC_UNITED_KINGDOM

TT_MS_LANGID.**SCOTTISH_GAELIC_UNITED_KINGDOM** = freetypy.TT_MS_LANGID.SCOTTISH_GAELIC_UNITE

freetypy.TT_MS_LANGID.SEPEDI_SOUTH_AFRICA

TT_MS_LANGID.**SEPEDI_SOUTH_AFRICA** = freetypy.TT_MS_LANGID.SOTHO_SOUTHERN_SOUTH_AFRICA

freetypy.TT_MS_LANGID.SERBIAN_BOSNIA_HERZ_CYRILLIC

TT_MS_LANGID.**SERBIAN_BOSNIA_HERZ_CYRILLIC** = freetypy.TT_MS_LANGID.SERBIAN_BOSNIA_HERZ_CYI

freetypy.TT_MS_LANGID.SERBIAN_BOSNIA_HERZ_LATIN

TT_MS_LANGID.**SERBIAN_BOSNIA_HERZ_LATIN** = freetypy.TT_MS_LANGID.SERBIAN_BOSNIA_HERZ_CYRILI

freetypy.TT_MS_LANGID.SERBIAN_SERBIA_CYRILLIC

TT_MS_LANGID.**SERBIAN_SERBIA_CYRILLIC** = freetypy.TT_MS_LANGID.SERBIAN_SERBIA_CYRILLIC

freetypy.TT_MS_LANGID.SERBIAN_SERBIA_LATIN

TT_MS_LANGID.**SERBIAN_SERBIA_LATIN** = freetypy.TT_MS_LANGID.SERBIAN_SERBIA_LATIN

freetypy.TT_MS_LANGID.SINDHI_INDIA

TT_MS_LANGID.**SINDHI_INDIA** = **freetypy.TT_MS_LANGID.SINDHI_INDIA**

freetypy.TT_MS_LANGID.SINDHI_PAKISTAN

TT_MS_LANGID.**SINDHI_PAKISTAN** = **freetypy.TT_MS_LANGID.SINDHI_PAKISTAN**

freetypy.TT_MS_LANGID.SINHALESE_SRI_LANKA

TT_MS_LANGID.**SINHALESE_SRI_LANKA** = **freetypy.TT_MS_LANGID.SINHALESE_SRI_LANKA**

freetypy.TT_MS_LANGID.SLOVAK_SLOVAKIA

TT_MS_LANGID.**SLOVAK_SLOVAKIA** = **freetypy.TT_MS_LANGID.SLOVAK_SLOVAKIA**

freetypy.TT_MS_LANGID.SLOVENE_SLOVENIA

TT_MS_LANGID.**SLOVENE_SLOVENIA** = **freetypy.TT_MS_LANGID.SLOVENE_SLOVENIA**

freetypy.TT_MS_LANGID.SOMALI_SOMALIA

TT_MS_LANGID.**SOMALI_SOMALIA** = **freetypy.TT_MS_LANGID.SOMALI_SOMALIA**

freetypy.TT_MS_LANGID.SORBIAN_GERMANY

TT_MS_LANGID.**SORBIAN_GERMANY** = **freetypy.TT_MS_LANGID.SORBIAN_GERMANY**

freetypy.TT_MS_LANGID.SOTHO_SOUTHERN_SOUTH_AFRICA

TT_MS_LANGID.**SOTHO_SOUTHERN_SOUTH_AFRICA** = **freetypy.TT_MS_LANGID.SOTHO_SOUTHERN_SOUTH_A**

freetypy.TT_MS_LANGID.SPANISH_ARGENTINA

TT_MS_LANGID.**SPANISH_ARGENTINA** = **freetypy.TT_MS_LANGID.SPANISH_ARGENTINA**

freetypy.TT_MS_LANGID.SPANISH_BOLIVIA

TT_MS_LANGID.**SPANISH_BOLIVIA** = **freetypy.TT_MS_LANGID.SPANISH_BOLIVIA**

freetypy.TT_MS_LANGID.SPANISH_CHILE

TT_MS_LANGID.**SPANISH_CHILE** = **freetypy.TT_MS_LANGID.SPANISH_CHILE**

freetypy.TT_MS_LANGID.SPANISH_COLOMBIA

TT_MS_LANGID.**SPANISH_COLOMBIA** = freetypy.TT_MS_LANGID.SPANISH_COLOMBIA

freetypy.TT_MS_LANGID.SPANISH_COSTA_RICA

TT_MS_LANGID.**SPANISH_COSTA_RICA** = freetypy.TT_MS_LANGID.SPANISH_COSTA_RICA

freetypy.TT_MS_LANGID.SPANISH_DOMINICAN_REPUBLIC

TT_MS_LANGID.**SPANISH_DOMINICAN_REPUBLIC** = freetypy.TT_MS_LANGID.SPANISH_DOMINICAN_REPUBL

freetypy.TT_MS_LANGID.SPANISH_ECUADOR

TT_MS_LANGID.**SPANISH_ECUADOR** = freetypy.TT_MS_LANGID.SPANISH_ECUADOR

freetypy.TT_MS_LANGID.SPANISH_EL_SALVADOR

TT_MS_LANGID.**SPANISH_EL_SALVADOR** = freetypy.TT_MS_LANGID.SPANISH_EL_SALVADOR

freetypy.TT_MS_LANGID.SPANISH_GUATEMALA

TT_MS_LANGID.**SPANISH_GUATEMALA** = freetypy.TT_MS_LANGID.SPANISH_GUATEMALA

freetypy.TT_MS_LANGID.SPANISH_HONDURAS

TT_MS_LANGID.**SPANISH_HONDURAS** = freetypy.TT_MS_LANGID.SPANISH_HONDURAS

freetypy.TT_MS_LANGID.SPANISH_LATIN_AMERICA

TT_MS_LANGID.**SPANISH_LATIN_AMERICA** = freetypy.TT_MS_LANGID.SPANISH_LATIN_AMERICA

freetypy.TT_MS_LANGID.SPANISH_MEXICO

TT_MS_LANGID.**SPANISH_MEXICO** = freetypy.TT_MS_LANGID.SPANISH_MEXICO

freetypy.TT_MS_LANGID.SPANISH_NICARAGUA

TT_MS_LANGID.**SPANISH_NICARAGUA** = freetypy.TT_MS_LANGID.SPANISH_NICARAGUA

freetypy.TT_MS_LANGID.SPANISH_PANAMA

TT_MS_LANGID.**SPANISH_PANAMA** = freetypy.TT_MS_LANGID.SPANISH_PANAMA

freetypy.TT_MS_LANGID.SPANISH_PARAGUAY

TT_MS_LANGID.**SPANISH_PARAGUAY** = freetypy.TT_MS_LANGID.SPANISH_PARAGUAY

freetypy.TT_MS_LANGID.SPANISH_PERU

TT_MS_LANGID.**SPANISH_PERU** = freetypy.TT_MS_LANGID.SPANISH_PERU

freetypy.TT_MS_LANGID.SPANISH_PUERTO_RICO

TT_MS_LANGID.**SPANISH_PUERTO_RICO** = freetypy.TT_MS_LANGID.SPANISH_PUERTO_RICO

freetypy.TT_MS_LANGID.SPANISH_SPAIN_INTERNATIONAL_SORT

TT_MS_LANGID.**SPANISH_SPAIN_INTERNATIONAL_SORT** = freetypy.TT_MS_LANGID.SPANISH_SPAIN_INTERN

freetypy.TT_MS_LANGID.SPANISH_SPAIN_TRADITIONAL_SORT

TT_MS_LANGID.**SPANISH_SPAIN_TRADITIONAL_SORT** = freetypy.TT_MS_LANGID.SPANISH_SPAIN_TRADITIO

freetypy.TT_MS_LANGID.SPANISH_UNITED_STATES

TT_MS_LANGID.**SPANISH_UNITED_STATES** = freetypy.TT_MS_LANGID.SPANISH_UNITED_STATES

freetypy.TT_MS_LANGID.SPANISH_URUGUAY

TT_MS_LANGID.**SPANISH_URUGUAY** = freetypy.TT_MS_LANGID.SPANISH_URUGUAY

freetypy.TT_MS_LANGID.SPANISH_VENEZUELA

TT_MS_LANGID.**SPANISH_VENEZUELA** = freetypy.TT_MS_LANGID.SPANISH_VENEZUELA

freetypy.TT_MS_LANGID.SUTU_SOUTH_AFRICA

TT_MS_LANGID.**SUTU_SOUTH_AFRICA** = freetypy.TT_MS_LANGID.SUTU_SOUTH_AFRICA

freetypy.TT_MS_LANGID.SWAHILI_KENYA

TT_MS_LANGID.**SWAHILI_KENYA** = freetypy.TT_MS_LANGID.SWAHILI_KENYA

freetypy.TT_MS_LANGID.SWEDISH_FINLAND

TT_MS_LANGID.**SWEDISH_FINLAND** = freetypy.TT_MS_LANGID.SWEDISH_FINLAND

freetypy.TT_MS_LANGID.SWEDISH_SWEDEN

`TT_MS_LANGID.`**`SWEDISH_SWEDEN`** = freetypy.TT_MS_LANGID.SWEDISH_SWEDEN

freetypy.TT_MS_LANGID.SYRIAC_SYRIA

`TT_MS_LANGID.`**`SYRIAC_SYRIA`** = freetypy.TT_MS_LANGID.SYRIAC_SYRIA

freetypy.TT_MS_LANGID.TAJIK_TAJIKISTAN

`TT_MS_LANGID.`**`TAJIK_TAJIKISTAN`** = freetypy.TT_MS_LANGID.TAJIK_TAJIKISTAN

freetypy.TT_MS_LANGID.TAMAZIGHT_MOROCCO

`TT_MS_LANGID.`**`TAMAZIGHT_MOROCCO`** = freetypy.TT_MS_LANGID.TAMAZIGHT_MOROCCO

freetypy.TT_MS_LANGID.TAMAZIGHT_MOROCCO_LATIN

`TT_MS_LANGID.`**`TAMAZIGHT_MOROCCO_LATIN`** = freetypy.TT_MS_LANGID.TAMAZIGHT_MOROCCO_LATIN

freetypy.TT_MS_LANGID.TAMIL_INDIA

`TT_MS_LANGID.`**`TAMIL_INDIA`** = freetypy.TT_MS_LANGID.TAMIL_INDIA

freetypy.TT_MS_LANGID.TATAR_TATARSTAN

`TT_MS_LANGID.`**`TATAR_TATARSTAN`** = freetypy.TT_MS_LANGID.TATAR_TATARSTAN

freetypy.TT_MS_LANGID.TELUGU_INDIA

`TT_MS_LANGID.`**`TELUGU_INDIA`** = freetypy.TT_MS_LANGID.TELUGU_INDIA

freetypy.TT_MS_LANGID.THAI_THAILAND

`TT_MS_LANGID.`**`THAI_THAILAND`** = freetypy.TT_MS_LANGID.THAI_THAILAND

freetypy.TT_MS_LANGID.TIBETAN_BHUTAN

`TT_MS_LANGID.`**`TIBETAN_BHUTAN`** = freetypy.TT_MS_LANGID.TIBETAN_BHUTAN

freetypy.TT_MS_LANGID.TIBETAN_CHINA

`TT_MS_LANGID.`**`TIBETAN_CHINA`** = freetypy.TT_MS_LANGID.TIBETAN_CHINA

freetypy.TT_MS_LANGID.TIGRIGNA_ERYTHREA

TT_MS_LANGID.**TIGRIGNA_ERYTHREA** = freetypy.TT_MS_LANGID.TIGRIGNA_ERYTREA

freetypy.TT_MS_LANGID.TIGRIGNA_ERYTREA

TT_MS_LANGID.**TIGRIGNA_ERYTREA** = freetypy.TT_MS_LANGID.TIGRIGNA_ERYTREA

freetypy.TT_MS_LANGID.TIGRIGNA_ETHIOPIA

TT_MS_LANGID.**TIGRIGNA_ETHIOPIA** = freetypy.TT_MS_LANGID.TIGRIGNA_ETHIOPIA

freetypy.TT_MS_LANGID.TSONGA_SOUTH_AFRICA

TT_MS_LANGID.**TSONGA_SOUTH_AFRICA** = freetypy.TT_MS_LANGID.TSONGA_SOUTH_AFRICA

freetypy.TT_MS_LANGID.TSWANA_SOUTH_AFRICA

TT_MS_LANGID.**TSWANA_SOUTH_AFRICA** = freetypy.TT_MS_LANGID.TSWANA_SOUTH_AFRICA

freetypy.TT_MS_LANGID.TURKISH_TURKEY

TT_MS_LANGID.**TURKISH_TURKEY** = freetypy.TT_MS_LANGID.TURKISH_TURKEY

freetypy.TT_MS_LANGID.TURKMEN_TURKMENISTAN

TT_MS_LANGID.**TURKMEN_TURKMENISTAN** = freetypy.TT_MS_LANGID.TURKMEN_TURKMENISTAN

freetypy.TT_MS_LANGID.UIGHUR_CHINA

TT_MS_LANGID.**UIGHUR_CHINA** = freetypy.TT_MS_LANGID.UIGHUR_CHINA

freetypy.TT_MS_LANGID.UKRAINIAN_UKRAINE

TT_MS_LANGID.**UKRAINIAN_UKRAINE** = freetypy.TT_MS_LANGID.UKRAINIAN_UKRAINE

freetypy.TT_MS_LANGID.URDU_INDIA

TT_MS_LANGID.**URDU_INDIA** = freetypy.TT_MS_LANGID.URDU_INDIA

freetypy.TT_MS_LANGID.URDU_PAKISTAN

TT_MS_LANGID.**URDU_PAKISTAN** = freetypy.TT_MS_LANGID.URDU_PAKISTAN

freetypy.TT_MS_LANGID.UZBEK_UZBEKISTAN_CYRILLIC

`TT_MS_LANGID.`**`UZBEK_UZBEKISTAN_CYRILLIC`** = freetypy.TT_MS_LANGID.UZBEK_UZBEKISTAN_CYRILLIC

freetypy.TT_MS_LANGID.UZBEK_UZBEKISTAN_LATIN

`TT_MS_LANGID.`**`UZBEK_UZBEKISTAN_LATIN`** = freetypy.TT_MS_LANGID.UZBEK_UZBEKISTAN_LATIN

freetypy.TT_MS_LANGID.VENDA_SOUTH_AFRICA

`TT_MS_LANGID.`**`VENDA_SOUTH_AFRICA`** = freetypy.TT_MS_LANGID.VENDA_SOUTH_AFRICA

freetypy.TT_MS_LANGID.VIETNAMESE_VIET_NAM

`TT_MS_LANGID.`**`VIETNAMESE_VIET_NAM`** = freetypy.TT_MS_LANGID.VIETNAMESE_VIET_NAM

freetypy.TT_MS_LANGID.WELSH_WALES

`TT_MS_LANGID.`**`WELSH_WALES`** = freetypy.TT_MS_LANGID.WELSH_WALES

freetypy.TT_MS_LANGID.XHOSA_SOUTH_AFRICA

`TT_MS_LANGID.`**`XHOSA_SOUTH_AFRICA`** = freetypy.TT_MS_LANGID.XHOSA_SOUTH_AFRICA

freetypy.TT_MS_LANGID.YIDDISH_GERMANY

`TT_MS_LANGID.`**`YIDDISH_GERMANY`** = freetypy.TT_MS_LANGID.YIDDISH_GERMANY

freetypy.TT_MS_LANGID.YI_CHINA

`TT_MS_LANGID.`**`YI_CHINA`** = freetypy.TT_MS_LANGID.YI_CHINA

freetypy.TT_MS_LANGID.YORUBA_NIGERIA

`TT_MS_LANGID.`**`YORUBA_NIGERIA`** = freetypy.TT_MS_LANGID.YORUBA_NIGERIA

freetypy.TT_MS_LANGID.ZULU_SOUTH_AFRICA

`TT_MS_LANGID.`**`ZULU_SOUTH_AFRICA`** = freetypy.TT_MS_LANGID.ZULU_SOUTH_AFRICA

freetypy.TT_ADOBE_ID

class freetypy.**TT_ADOBE_ID**

Adobe-specific encoding values.

- *STANDARD*: Adobe standard encoding.
- *EXPERT*: Adobe expert encoding.
- *CUSTOM*: Adobe custom encoding.
- *LATIN_1*: Adobe Latin 1 encoding.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

CUSTOM
EXPERT
LATIN_1
STANDARD

freetypy.TT_ADOBE_ID.CUSTOM

TT_ADOBE_ID.**CUSTOM** = **freetypy.TT_ADOBE_ID.CUSTOM**

freetypy.TT_ADOBE_ID.EXPERT

TT_ADOBE_ID.**EXPERT** = **freetypy.TT_ADOBE_ID.EXPERT**

freetypy.TT_ADOBE_ID.LATIN_1

TT_ADOBE_ID.**LATIN_1** = **freetypy.TT_ADOBE_ID.LATIN_1**

freetypy.TT_ADOBE_ID.STANDARD

TT_ADOBE_ID.**STANDARD** = **freetypy.TT_ADOBE_ID.STANDARD**

freetypy.TT_NAME_ID

class freetypy.**TT_NAME_ID**

The type of value stored in a *SfntName* record.

- *COPYRIGHT*
- *FONT_FAMILY*
- *FONT_SUBFAMILY*
- *UNIQUE_ID*
- *FULL_NAME*

- *VERSION_STRING*
- *PS_NAME*
- *TRADEMARK*

The following values are from the OpenType spec:

- *MANUFACTURER*
- *DESIGNER*
- *DESCRIPTION*
- *VENDOR_URL*
- *DESIGNER_URL*
- *LICENSE*
- *LICENSE_URL*
- *PREFERRED_FAMILY*
- *PREFERRED_SUBFAMILY*
- *MAC_FULL_NAME*
- *SAMPLE_TEXT*

This is new in OpenType 1.3:

- *CID_FINDFONT_NAME*

This is new in OpenType 1.5:

- *WWS_FAMILY*
- *WWS_SUBFAMILY*

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

CID_FINDFONT_NAME
COPYRIGHT
DESCRIPTION
DESIGNER
DESIGNER_URL
FONT_FAMILY
FONT_SUBFAMILY
FULL_NAME
LICENSE
LICENSE_URL
MAC_FULL_NAME
MANUFACTURER
PREFERRED_FAMILY
PREFERRED_SUBFAMILY
PS_NAME
SAMPLE_TEXT

Continued on next page

Table 2.56 – continued from previous page

TRADEMARK
UNIQUE_ID
VENDOR_URL
VERSION_STRING
WWS_FAMILY
WWS_SUBFAMILY

freetypy.TT_NAME_ID.CID_FINDFONT_NAME

`TT_NAME_ID.`**`CID_FINDFONT_NAME`** = **freetypy.TT_NAME_ID.CID_FINDFONT_NAME**

freetypy.TT_NAME_ID.COPYRIGHT

`TT_NAME_ID.`**`COPYRIGHT`** = **freetypy.TT_NAME_ID.COPYRIGHT**

freetypy.TT_NAME_ID.DESCRIPTION

`TT_NAME_ID.`**`DESCRIPTION`** = **freetypy.TT_NAME_ID.DESCRIPTION**

freetypy.TT_NAME_ID.DESIGNER

`TT_NAME_ID.`**`DESIGNER`** = **freetypy.TT_NAME_ID.DESIGNER**

freetypy.TT_NAME_ID.DESIGNER_URL

`TT_NAME_ID.`**`DESIGNER_URL`** = **freetypy.TT_NAME_ID.DESIGNER_URL**

freetypy.TT_NAME_ID.FONT_FAMILY

`TT_NAME_ID.`**`FONT_FAMILY`** = **freetypy.TT_NAME_ID.FONT_FAMILY**

freetypy.TT_NAME_ID.FONT_SUBFAMILY

`TT_NAME_ID.`**`FONT_SUBFAMILY`** = **freetypy.TT_NAME_ID.FONT_SUBFAMILY**

freetypy.TT_NAME_ID.FULL_NAME

`TT_NAME_ID.`**`FULL_NAME`** = **freetypy.TT_NAME_ID.FULL_NAME**

freetypy.TT_NAME_ID.LICENSE

`TT_NAME_ID.`**`LICENSE`** = **freetypy.TT_NAME_ID.LICENSE**

freetypy.TT_NAME_ID.LICENSE_URL

`TT_NAME_ID.`**`LICENSE_URL`** = **freetypy.TT_NAME_ID.LICENSE_URL**

freetypy.TT_NAME_ID.MAC_FULL_NAME

TT_NAME_ID.**MAC_FULL_NAME** = freetypy.TT_NAME_ID.MAC_FULL_NAME

freetypy.TT_NAME_ID.MANUFACTURER

TT_NAME_ID.**MANUFACTURER** = freetypy.TT_NAME_ID.MANUFACTURER

freetypy.TT_NAME_ID.PREFERRED_FAMILY

TT_NAME_ID.**PREFERRED_FAMILY** = freetypy.TT_NAME_ID.PREFERRED_FAMILY

freetypy.TT_NAME_ID.PREFERRED_SUBFAMILY

TT_NAME_ID.**PREFERRED_SUBFAMILY** = freetypy.TT_NAME_ID.PREFERRED_SUBFAMILY

freetypy.TT_NAME_ID.PS_NAME

TT_NAME_ID.**PS_NAME** = freetypy.TT_NAME_ID.PS_NAME

freetypy.TT_NAME_ID.SAMPLE_TEXT

TT_NAME_ID.**SAMPLE_TEXT** = freetypy.TT_NAME_ID.SAMPLE_TEXT

freetypy.TT_NAME_ID.TRADEMARK

TT_NAME_ID.**TRADEMARK** = freetypy.TT_NAME_ID.TRADEMARK

freetypy.TT_NAME_ID.UNIQUE_ID

TT_NAME_ID.**UNIQUE_ID** = freetypy.TT_NAME_ID.UNIQUE_ID

freetypy.TT_NAME_ID.VENDOR_URL

TT_NAME_ID.**VENDOR_URL** = freetypy.TT_NAME_ID.VENDOR_URL

freetypy.TT_NAME_ID.VERSION_STRING

TT_NAME_ID.**VERSION_STRING** = freetypy.TT_NAME_ID.VERSION_STRING

freetypy.TT_NAME_ID.WWS_FAMILY

TT_NAME_ID.**WWS_FAMILY** = freetypy.TT_NAME_ID.WWS_FAMILY

freetypy.TT_NAME_ID.WWS_SUBFAMILY

TT_NAME_ID.**WWS_SUBFAMILY** = **freetypy.TT_NAME_ID.WWS_SUBFAMILY**

freetypy.TT_MAC_STYLE

class freetypy.**TT_MAC_STYLE**

Bit flags for the style of the face.

- *BOLD*
- *ITALIC*
- *UNDERLINE*
- *OUTLINE*
- *SHADOW*
- *CONDENSED*
- *EXTENDED*

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

BOLD
CONDENSED
EXTENDED
ITALIC
OUTLINE
SHADOW
UNDERLINE

freetypy.TT_MAC_STYLE.BOLD

TT_MAC_STYLE.**BOLD** = **freetypy.TT_MAC_STYLE.BOLD**

freetypy.TT_MAC_STYLE.CONDENSED

TT_MAC_STYLE.**CONDENSED** = **freetypy.TT_MAC_STYLE.CONDENSED**

freetypy.TT_MAC_STYLE.EXTENDED

TT_MAC_STYLE.**EXTENDED** = **freetypy.TT_MAC_STYLE.EXTENDED**

freetypy.TT_MAC_STYLE.ITALIC

TT_MAC_STYLE.**ITALIC** = **freetypy.TT_MAC_STYLE.ITALIC**

freetypy.TT_MAC_STYLE.OUTLINE

TT_MAC_STYLE.**OUTLINE** = **freetypy.TT_MAC_STYLE.OUTLINE**

freetypy.TT_MAC_STYLE.SHADOW

TT_MAC_STYLE.**SHADOW** = **freetypy.TT_MAC_STYLE.SHADOW**

freetypy.TT_MAC_STYLE.UNDERLINE

TT_MAC_STYLE.**UNDERLINE** = **freetypy.TT_MAC_STYLE.UNDERLINE**

freetypy.TT_HEADER_FLAGS

class freetypy.**TT_HEADER_FLAGS**

Bit flags for global information about a *Face*.

- *BASELINE_AT_ZERO*: If set, the baselines for the font is at $y = 0$ (that is, the x-axis)
- *LEFT_BLACK_BIT_IS_LSB*: If set, the x-position of the leftmost black bit is assumed to be the left side bearing.
- *NONLINEAR_SCALE*: If set, instructions may use point size explicitly in place of pixels per em. Scaled point size and actual point size will differ (i.e. 24 point glyph differs from 12 point glyph scaled by factor of 2).
- *INTEGER_SCALING*: Use integer scaling instead of fractional.
- *VERTICAL*: Set in fonts that are intended to ne laid out vertically, and in which the glyphs have been drawn such that an x-coordinate of 0 corresponds to the desired vertical baseline.
- *REQUIRES_LAYOUT*: Set if the font requires layout for correct linguistic rendering (e.g. Arabic fonts).
- *METAMORPHOSIS_EFFECTS*: Set for a GX font which has one or more metamorphosis effects designated as happening by default.
- *RIGHT_TO_LEFT_GLYPHS*: Set if the font contains any strong right-to-left glyphs.
- *INDIC_REARRANGEMENT_EFFECTS*: Set if the font contains Indic-style rearrangement effects.

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

BASELINE_AT_ZERO
INDIC_REARRANGEMENT_EFFECTS
INTEGER_SCALING
LEFT_BLACK_BIT_IS_LSB
METAMORPHOSIS_EFFECTS
NONLINEAR_SCALE
REQUIRES_LAYOUT
RIGHT_TO_LEFT_GLYPHS
VERTICAL

freetypy.TT_HEADER_FLAGS.BASELINE_AT_ZERO

TT_HEADER_FLAGS.**BASELINE_AT_ZERO** = freetypy.TT_HEADER_FLAGS.BASELINE_AT_ZERO

freetypy.TT_HEADER_FLAGS.INDIC_REARRANGEMENT_EFFECTS

TT_HEADER_FLAGS.**INDIC_REARRANGEMENT_EFFECTS** = freetypy.TT_HEADER_FLAGS.INDIC_REARRANGEM

freetypy.TT_HEADER_FLAGS.INTEGER_SCALING

TT_HEADER_FLAGS.**INTEGER_SCALING** = freetypy.TT_HEADER_FLAGS.INTEGER_SCALING

freetypy.TT_HEADER_FLAGS.LEFT_BLACK_BIT_IS_LSB

TT_HEADER_FLAGS.**LEFT_BLACK_BIT_IS_LSB** = freetypy.TT_HEADER_FLAGS.LEFT_BLACK_BIT_IS_LSB

freetypy.TT_HEADER_FLAGS.METAMORPHOSIS_EFFECTS

TT_HEADER_FLAGS.**METAMORPHOSIS_EFFECTS** = freetypy.TT_HEADER_FLAGS.METAMORPHOSIS_EFFECTS

freetypy.TT_HEADER_FLAGS.NONLINEAR_SCALE

TT_HEADER_FLAGS.**NONLINEAR_SCALE** = freetypy.TT_HEADER_FLAGS.NONLINEAR_SCALE

freetypy.TT_HEADER_FLAGS.REQUIRES_LAYOUT

TT_HEADER_FLAGS.**REQUIRES_LAYOUT** = freetypy.TT_HEADER_FLAGS.REQUIRES_LAYOUT

freetypy.TT_HEADER_FLAGS.RIGHT_TO_LEFT_GLYPHS

TT_HEADER_FLAGS.**RIGHT_TO_LEFT_GLYPHS** = freetypy.TT_HEADER_FLAGS.RIGHT_TO_LEFT_GLYPHS

freetypy.TT_HEADER_FLAGS.VERTICAL

TT_HEADER_FLAGS.**VERTICAL** = freetypy.TT_HEADER_FLAGS.VERTICAL

freetypy.TT_WIDTH_CLASS

class freetypy.**TT_WIDTH_CLASS**

Width values for the *TT_OS2.width_class* property.

- *ULTRA_CONDENSED*: 50% of normal
- *EXTRA_CONDENSED*: 62.5% of normal
- *CONDENSED*: 75% of normal
- *SEMI_CONDENSED*: 87.5% of normal
- *MEDIUM*: 100% of normal (aliased to *NORMAL*)

- *SEMI_EXPANDED*: 112.5% of normal
- *EXPANDED*: 125% of normal
- *EXTRA_EXPANDED*: 150% of normal
- *ULTRA_EXPANDED*: 200% of normal

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

CONDENSED
EXPANDED
EXTRA_CONDENSED
EXTRA_EXPANDED
MEDIUM
NORMAL
SEMI_CONDENSED
SEMI_EXPANDED
ULTRA_CONDENSED
ULTRA_EXPANDED

freetypy.TT_WIDTH_CLASS.CONDENSED

TT_WIDTH_CLASS.**CONDENSED = freetypy.TT_WIDTH_CLASS.CONDENSED**

freetypy.TT_WIDTH_CLASS.EXPANDED

TT_WIDTH_CLASS.**EXPANDED = freetypy.TT_WIDTH_CLASS.EXPANDED**

freetypy.TT_WIDTH_CLASS.EXTRA_CONDENSED

TT_WIDTH_CLASS.**EXTRA_CONDENSED = freetypy.TT_WIDTH_CLASS.EXTRA_CONDENSED**

freetypy.TT_WIDTH_CLASS.EXTRA_EXPANDED

TT_WIDTH_CLASS.**EXTRA_EXPANDED = freetypy.TT_WIDTH_CLASS.EXTRA_EXPANDED**

freetypy.TT_WIDTH_CLASS.MEDIUM

TT_WIDTH_CLASS.**MEDIUM = freetypy.TT_WIDTH_CLASS.NORMAL**

freetypy.TT_WIDTH_CLASS.NORMAL

TT_WIDTH_CLASS.**NORMAL = freetypy.TT_WIDTH_CLASS.NORMAL**

freetypy.TT_WIDTH_CLASS.SEMI_CONDENSED

TT_WIDTH_CLASS.**SEMI_CONDENSED** = **freetypy.TT_WIDTH_CLASS.SEMI_CONDENSED**

freetypy.TT_WIDTH_CLASS.SEMI_EXPANDED

TT_WIDTH_CLASS.**SEMI_EXPANDED** = **freetypy.TT_WIDTH_CLASS.SEMI_EXPANDED**

freetypy.TT_WIDTH_CLASS.ULTRA_CONDENSED

TT_WIDTH_CLASS.**ULTRA_CONDENSED** = **freetypy.TT_WIDTH_CLASS.ULTRA_CONDENSED**

freetypy.TT_WIDTH_CLASS.ULTRA_EXPANDED

TT_WIDTH_CLASS.**ULTRA_EXPANDED** = **freetypy.TT_WIDTH_CLASS.ULTRA_EXPANDED**

freetypy.TT_WEIGHT_CLASS

class freetypy.**TT_WEIGHT_CLASS**

Weight values for the *TT_OS2.weight_class* property.

- *ULTRA_LIGHT*
- *EXTRA_LIGHT*
- *LIGHT*
- *SEMI_LIGHT*
- *MEDIUM* (aliased to *NORMAL*)
- *SEMI_BOLD*
- *BOLD*
- *EXTRA_BOLD*
- *ULTRA_BOLD*

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

BOLD
EXTRA_BOLD
EXTRA_LIGHT
LIGHT
MEDIUM
NORMAL
SEMI_BOLD
SEMI_LIGHT
ULTRA_BOLD

Continued on next page

Table 2.60 – continued from previous page

ULTRA_LIGHT

freetypy.TT_WEIGHT_CLASS.BOLD

TT_WEIGHT_CLASS.**BOLD** = **freetypy.TT_WEIGHT_CLASS.BOLD**

freetypy.TT_WEIGHT_CLASS.EXTRA_BOLD

TT_WEIGHT_CLASS.**EXTRA_BOLD** = **freetypy.TT_WEIGHT_CLASS.EXTRA_BOLD**

freetypy.TT_WEIGHT_CLASS.EXTRA_LIGHT

TT_WEIGHT_CLASS.**EXTRA_LIGHT** = **freetypy.TT_WEIGHT_CLASS.EXTRA_LIGHT**

freetypy.TT_WEIGHT_CLASS.LIGHT

TT_WEIGHT_CLASS.**LIGHT** = **freetypy.TT_WEIGHT_CLASS.LIGHT**

freetypy.TT_WEIGHT_CLASS.MEDIUM

TT_WEIGHT_CLASS.**MEDIUM** = **freetypy.TT_WEIGHT_CLASS.NORMAL**

freetypy.TT_WEIGHT_CLASS.NORMAL

TT_WEIGHT_CLASS.**NORMAL** = **freetypy.TT_WEIGHT_CLASS.NORMAL**

freetypy.TT_WEIGHT_CLASS.SEMI_BOLD

TT_WEIGHT_CLASS.**SEMI_BOLD** = **freetypy.TT_WEIGHT_CLASS.SEMI_BOLD**

freetypy.TT_WEIGHT_CLASS.SEMI_LIGHT

TT_WEIGHT_CLASS.**SEMI_LIGHT** = **freetypy.TT_WEIGHT_CLASS.SEMI_LIGHT**

freetypy.TT_WEIGHT_CLASS.ULTRA_BOLD

TT_WEIGHT_CLASS.**ULTRA_BOLD** = **freetypy.TT_WEIGHT_CLASS.ULTRA_BOLD**

freetypy.TT_WEIGHT_CLASS.ULTRA_LIGHT

TT_WEIGHT_CLASS.**ULTRA_LIGHT** = **freetypy.TT_WEIGHT_CLASS.ULTRA_LIGHT**

freetypy.TT_FS_SELECTION

class freetypy.**TT_FS_SELECTION**

Bit flag for font style.

- *ITALIC*
- *UNDERSCORE*
- *NEGATIVE*
- *OUTLINED*
- *STRIKEOUT*
- *BOLD*

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

BOLD
ITALIC
NEGATIVE
OUTLINED
STRIKEOUT
UNDERSCORE

freetypy.TT_FS_SELECTION.BOLD

TT_FS_SELECTION.**BOLD** = **freetypy.TT_FS_SELECTION.BOLD**

freetypy.TT_FS_SELECTION.ITALIC

TT_FS_SELECTION.**ITALIC** = **freetypy.TT_FS_SELECTION.ITALIC**

freetypy.TT_FS_SELECTION.NEGATIVE

TT_FS_SELECTION.**NEGATIVE** = **freetypy.TT_FS_SELECTION.NEGATIVE**

freetypy.TT_FS_SELECTION.OUTLINED

TT_FS_SELECTION.**OUTLINED** = **freetypy.TT_FS_SELECTION.OUTLINED**

freetypy.TT_FS_SELECTION.STRIKEOUT

TT_FS_SELECTION.**STRIKEOUT** = **freetypy.TT_FS_SELECTION.STRIKEOUT**

freetypy.TT_FS_SELECTION.UNDERSCORE

TT_FS_SELECTION.**UNDERSCORE** = **freetypy.TT_FS_SELECTION.UNDERSCORE**

LCD Filtering

set_lcd_filter	Apply color filtering to LCD decimated bitmaps.
set_lcd_filter_weights	Enable LCD filter with custom weights.

freetypy.set_lcd_filter

freetypy.**set_lcd_filter**()

Apply color filtering to LCD decimated bitmaps.

Works when called *Glyph.render* with *RENDER_MODE.LCD* or *RENDER_MODE.LCD_V*.

Parameters **filter** (LCD_FILTER constant) –

Notes

This feature is always disabled by default. Clients must make an explicit call to this function with a filter value other than LCD_FILTER.NONE in order to enable it.

Due to PATENTS covering subpixel rendering, this function doesn't do anything except raising NotImplementedError if the configuration macro FT_CONFIG_OPTION_SUBPIXEL_RENDERING is not defined in your build of the library, which should correspond to all default builds of FreeType.

freetypy.set_lcd_filter_weights

freetypy.**set_lcd_filter_weights**()

Enable LCD filter with custom weights.

Parameters **b, c, d, e** (*a,*) – The filter weights

Notes

Due to PATENTS covering subpixel rendering, this function doesn't do anything except raising NotImplementedError if the configuration macro FT_CONFIG_OPTION_SUBPIXEL_RENDERING is not defined in your build of the library, which should correspond to all default builds of FreeType.

Basic Types

BBox	An outline's bounding box.
Matrix	A 2x2 matrix.
Vector	A 2D vector.

freetypy.BBox

class freetypy.**BBox**

An outline's bounding box.

The bounding box is specified with the coordinates of the lower left and the upper right corner. In PostScript, those values are often called `(llx, lly)` and `(urx, ury)`, respectively.

If *y_min* is negative, this value gives the glyph's descender. Otherwise, the glyph doesn't descend below the baseline. Similarly, if *y_max* is positive, this value gives the glyph's ascender.

x_min gives the horizontal distance from the glyph's origin to the left edge of the glyph's bounding box. If *x_min* is negative, the glyph extends to the left of the origin.

BBox also works as a Python sequence, so it is easy to do:

```
x_min, y_min, x_max, y_max = bbox
```

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

ascent	The height of the bounding box above the baseline.
depth	The depth of the bounding box below the baseline.
height	The height of the bounding box (`y_max - y_min`).
width	The width of the bounding box (`x_max - x_min`).
x_max	The horizontal maximum (right-most).
x_min	The horizontal minimum (left-most).
y_max	The vertical maximum (top-most).
y_min	The vertical minimum (bottom-most).

freetypy.BBox.ascent

BBox.**ascent**

The height of the bounding box above the baseline. This is an alias for *y_max*.

freetypy.BBox.depth

BBox.**depth**

The depth of the bounding box below the baseline. This is an alias for *y_min*.

freetypy.BBox.height

BBox.**height**

The height of the bounding box (`y_max - y_min`).

freetypy.BBox.width

BBox.**width**

The width of the bounding box (`x_max - x_min`).

freetypy.BBox.x_max

BBox.**x_max**

The horizontal maximum (right-most).

freetypy.BBox.x_min

BBox.**x_min**

The horizontal minimum (left-most).

freetypy.BBox.y_max

BBox.**y_max**

The vertical maximum (top-most).

freetypy.BBox.y_min

BBox.**y_min**

The vertical minimum (bottom-most).

freetypy.Matrix

class freetypy.**Matrix**

A 2x2 matrix.

The computation performed is:

```
x' = x*xx + y*xy
y' = x*yx + y*yy
```

__init__()

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

xx	Matrix coefficient.
xy	Matrix coefficient.
yx	Matrix coefficient.
yy	Matrix coefficient.

freetypy.Matrix.xx

Matrix.**xx**

Matrix coefficient.

freetypy.Matrix.xy

Matrix.**xy**

Matrix coefficient.

freetypy.Matrix.yx

Matrix.**yx**

Matrix coefficient.

freetypy.Matrix.yy

`Matrix.yy`

Matrix coefficient.

freetypy.Vector

class `freetypy.Vector`

A 2D vector.

Vector also works as a Python sequence of length 2, so it's easy to do:

```
x, y = vector
```

`__init__()`

x.__init__(...) initializes x; see help(type(x)) for signature

Attributes

x	The x coordinate.
y	The y coordinate.

freetypy.Vector.x

`Vector.x`

The x coordinate.

freetypy.Vector.y

`Vector.y`

The y coordinate.

Miscellaneous Utilities

util

freetypy.util

Functions

bitmap_to_ascii(a)	Converts a single Glyph to a string with an ASCII drawing of that glyph.
vera_path()	The path to the copy of Bitstream Vera Sans that ships with freetypy for testing purposes.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index