Modul:Convert/text

This page defines text used by Module:Convert. All documentation (from Module:Convert/doc) is at that module. The text includes messages and categories output by the module, and parameters used as input.

This is a separate module to simplify translation for use on another wiki. For example, see `translation_table` and the other tables in bn:Module:Convert/text. Documentation is at Template:Convert/Transwiki guide.

Any changes should first be tested at Module:Convert/text/sandbox—see Template:Convert/testcases#Sandbox testcases.

```
-- Text used by Module:Convert for enwiki.
-- This is a separate module to simplify translation for use on another wiki.
-- See [[:en:Template:Convert/Transwiki guide]] if copying to another wiki.

-- Some units accept an SI prefix before the unit code, such as "kg" for kilogram
local SIprefixes = {
        -- The prefix field is what the prefix should be, if different from the p
        ['Y'] = { exponent = 24, name = 'yotta',                  },
        ['Z'] = { exponent = 21, name = 'zetta',                  },
        ['E'] = { exponent = 18, name = 'exa'  ,                  },
        ['P'] = { exponent = 15, name = 'peta' ,                  },
        ['T'] = { exponent = 12, name = 'tera' ,                  },
        ['G'] = { exponent =  9, name = 'giga' ,                  },
        ['M'] = { exponent =  6, name = 'mega' ,                  },
        ['k'] = { exponent =  3, name = 'kilo' ,                  },
        ['h'] = { exponent =  2, name = 'hecto',                  },
        ['da']= { exponent =  1, name = 'deca' , name_us = 'deka' },
        ['d'] = { exponent = -1, name = 'deci' ,                  },
        ['c'] = { exponent = -2, name = 'centi',                  },
        ['m'] = { exponent = -3, name = 'milli',                  },
        ['µ'] = { exponent = -6, name = 'micro',                  },  -- key = 'G
        ['μ'] = { exponent = -6, name = 'micro', prefix = 'µ'     },  -- key = 'M
        ['u'] = { exponent = -6, name = 'micro', prefix = 'µ'     },  -- not an S
        ['n'] = { exponent = -9, name = 'nano' ,                  },
        ['p'] = { exponent =-12, name = 'pico' ,                  },
        ['f'] = { exponent =-15, name = 'femto',                  },
        ['a'] = { exponent =-18, name = 'atto' ,                  },
        ['z'] = { exponent =-21, name = 'zepto',                  },
        ['y'] = { exponent =-24, name = 'yocto',                  },
}

-- Some units can be qualified with one of the following prefixes, when linked.
local customary_units = {
        { "US", link = "United States customary units" },
        { "U.S.", link = "United States customary units" },
        { "imperial", link = "Imperial units" },
        { "imp", link = "Imperial units" },
}

-- Names when using engineering notation (a prefix of "eN" where N is a number; e
-- key = { "name", link = "article title", exponent = numeric_key_value }
-- If lk=on and link is defined, the name of the number will appear as a link.
local eng_scales = {
        ["3"]  = { "thousand", exponent = 3 },
```

```
        ["6"]  = { "million", exponent = 6 },
        ["9"]  = { "billion", link = "1000000000 (number)", exponent = 9 },
        ["12"] = { "trillion", link = "1000000000000 (number)", exponent = 12 },
        ["15"] = { "quadrillion", link = "1000000000000000 (number)", exponent =
}

local all_categories = {
        unit = "[[Category:Convert errors]]",
        option = "[[Category:Convert errors]]",
        warning = '[[Category:Convert invalid options]]',
        tracking = '[[Category:Convert tracking]]',
}

-- For some error messages, the following puts the wanted style around
-- each unit code marked like '...%{ft%}...'.
local unitcode_regex = '%%([{}])'
local unitcode_replace = { ['{'] = '"', ['}'] = '"' }  -- no longer need the more

-- All messages that may be displayed if a problem occurs.
local all_messages = {
        -- Message format string: $1=title, $2=text, $3=category, $4=anchor.
        -- Each displayed message starts with "Convert:" so can easily locate by
        cvt_format = '<sup class="noprint Inline-Template" style="white-space:nov
        cvt_format2 = '<sup class="noprint Inline-Template" style="white-space:no
        cvt_format_preview = '<strong class="error">Error in convert: $1 [[Help:C
        -- Each of following messages is a table:
        -- { [1] = 'title',         -- mouseover title text
        --   [2] = 'text',          -- link text displayed in article
        --   [3] = 'category key',  -- key to lookup category in all_categories
        --   [4] = 'anchor',        -- anchor for link to relevant section on hel
        --   regex = gsub_regex,
        --   replace = gsub_table,
        -- }                    Mouseover title text                          Li
        cvt_bad_input      = { 'input "$1" must be a number and unit'       , 'in
        cvt_bad_num        = { 'Value "$1" must be a number'               , 'in
        cvt_big_prec       = { 'Precision "$1" is too large'               , 'pr
        cvt_invalid_num    = { 'Number has overflowed'                     , 'nu
        cvt_no_num         = { 'Needs the number to be converted'          , 'ne
        cvt_no_num2        = { 'Needs another number for a range'          , 'ne
        cvt_bad_altitude   = { '"$1" needs an integer'                     , 'in
        cvt_bad_frac       = { '"$1" needs an integer above 1'             , 'in
        cvt_bad_prec       = { 'Precision "$1" must be an integer'         , 'in
        cvt_bad_sigfig     = { '"$1" needs a positive integer'             , 'in
        cvt_empty_option   = { 'Ignored empty option "$1"'                 , 'em
        cvt_deprecated     = { 'Option "$1" is deprecated'                 , '*'
        cvt_no_spell       = { 'Spelling is not available'                 , 'bu
        cvt_unknown_option = { 'Ignored invalid option "$1"'               , 'in
        cvt_wd_fail        = { 'Unable to access Wikidata'                 , 'wi
        cvt_bad_default    = { 'Unit "$1" has an invalid default'          , 'bu
        cvt_bad_unit       = { 'Unit "$1" is invalid here'                 , 'un
        cvt_no_default     = { 'Unit "$1" has no default output unit'      , 'bu
        cvt_no_unit        = { 'Needs name of unit'                        , 'ne
        cvt_unknown        = { 'Unit name "$1" is not known'               , 'un
        cvt_should_be      = { '$1'                                        , 'an
        cvt_mismatch       = { 'Cannot convert "$1" to "$2"'               , 'un
        cvt_bug_convert    = { 'Bug: Cannot convert between specified units', 'bu
        cvt_lookup         = { 'Unit "$1" is incorrectly defined'          , 'bu
}

-- Text to join input value/unit with output value/unit.
local disp_joins = {
        -- [1]=before output, [2]=after output, [3]=between outputs in a combinat
        -- [wantname] gives default abbr=off
        ["or"]         = { " or "    , "" , " or ", wantname = true },
```

```
        ["sqbr-sp"]    = { " ["      , "]" },
        ["sqbr-nbsp"]  = { " [" , "]" },
        ["comma"]      = { ", "      , "" , ", " },
        ["slash-sp"]   = { " / "     , "" , wantname = true },
        ["slash-nbsp"] = { " / ", "" , wantname = true },
        ["slash-nosp"] = { "/"       , "" , wantname = true },
        ["b"]          = { " ("      , ")" },
        ["(or)"]       = { " ("      , ")", " or " },
        ["br"]         = { "<br />"  , "" , wantname = true },
        ["br()"]       = { "<br />(" , ")", wantname = true },
}

-- Text to separate values in a range.
local range_types = {
        -- Specifying a table requires either:
        -- * "off" and "on" values (for "abbr=off" and "abbr=on"), or
        -- * "input" and "output" values (for LHS and RHS);
        -- other fields are optional.
        -- When "adj=on|abbr=off" applies, spaces in range text are replaced with
        -- With "exception = true", that also occurs with "adj=on|abbr=on".
        -- If "adj" is defined here, that text (unchanged) is used with "adj=on"
        ["+"]      = " + ",
        [","]      = ", ",
        [", and"]  = ", and ",
        [", or"]   = ", or ",
        ["by"]     = " by ",
        ["-"]      = "–",
        ["to about"] = " to about ",
        ["and"]    = { off = " and ", on = " and ", exception = true },
        ["and(-)"] = { input = " and ", output = "–" },
        ["or"]     = { off = " or " , on = " or " , exception = true },
        ["to"]     = { off = " to " , on = " to " , exception = true },
        ["to(-)"]  = { input = " to ", output = "–" },
        ["+/-"]    = { off = " ± ", on = " ± ", adj = " 
        ["by(x)"]  = { input = " by ", output = " × ", out_range_x = true },
        ["x"]      = { off = " by ", on = " × ", abbr_range_x = true },
        ["xx"]     = " × ",
        ["*"]      = "x",
        ["/"]      = " / ",  -- for a table of high/low temperature
}

local range_aliases = {
        -- ["alternative name for a range"] = "standard range name"
        ["–"]        = "-",
        ["&ndash;"]  = "-",
        ["x"]        = "x",
        ["&times;"]  = "x",
        ["±"]        = "+/-",
        ["&plusmn;"] = "+/-",
}

-- Convert accepts range text delimited with whitespace, for example, {{convert|1
-- In addition, the following "words" are accepted without spaces, for example, {
-- Words must be in correct order for searching, for example, 'x' after 'xx'.
local range_words = { '-', '–', 'xx', 'x', '*' }

local ranges = {
        types = range_types,
        aliases = range_aliases,
        words = range_words,
}

-- Valid option names.
local en_option_name = {
```

```
        -- ["local text for option name"] = "en name used in this module"
        ["$"] = "$",
        ["abbr"] = "abbr",
        ["adj"] = "adj",
        ["altitude_ft"] = "altitude_ft",
        ["altitude_m"] = "altitude_m",
        ["comma"] = "comma",
        ["debug"] = "debug",
        ["disp"] = "disp",
        ["frac"] = "frac",
        ["input"] = "input",
        ["lang"] = "lang",
        ["lk"] = "lk",
        ["order"] = "order",
        ["qid"] = "qid",
        ["qual"] = "qual",
        ["qualifier"] = "qual",
        ["round"] = "round",
        ["sigfig"] = "sigfig",
        ["sing"] = "adj",                   -- "sing" is an old alias for "adj"
        ["sortable"] = "sortable",
        ["sp"] = "sp",
        ["spell"] = "spell",
        ["stylein"] = "stylein",
        ["styleout"] = "styleout",
        ["tracking"] = "tracking",
}

-- Valid option values.
-- Convention: parms.opt_xxx refers to an option that is set here
-- (not intended to be set by the template which invokes this module).
-- Example: At enwiki, "abbr" includes:
--     ["values"] = "opt_values"
-- As a result, if the template uses abbr=values, Module:Convert sets:
--     parms["opt_values"] = true
--     parms["abbr"] = nil
-- Therefore parms.abbr will be nil, or will have one of the listed values
-- that do not start with "opt_".
-- An option value of form "xxx?" is the same as "xxx" but shows the input as dep
local en_option_value = {
        ["$"] = 'TEXT',                     -- TEXT should be a currency symbol t
        ["abbr"] = {
                -- ["local text for option value"] = "en value used in this modul
                ["def"] = "",                   -- ignored (some wrapper template
                ["h"] = "on",                   -- abbr=on + use "h" for hand uni
                ["hh"] = "opt_hand_hh",         -- abbr=on + use "hh" for hand un
                ["in"] = "in",                  -- use symbol for LHS unit
                ["none"] = "off",               -- old name for "off"
                ["off"] = "off",                -- use name for all units
                ["on"] = "on",                  -- use symbol for all units
                ["out"] = "out",                -- use symbol for RHS unit (defau
                ["unit"] = "unit",              -- abbr=on but abbreviate units c
                ["values"] = "opt_values",      -- show only input and output num
                ["~"] = "opt_also_symbol",      -- show input unit symbol as well
        },
        ["adj"] = {
                ["mid"] = "opt_adjectival, opt_adj_mid",  -- adj=on with user-spe
                ["off"] = "",                   -- ignored (off is the default)
                ["on"] = "opt_adjectival",      -- unit name is singular and hyph
                ["pre"] = "opt_one_preunit",    -- user-specified text before inp
                ["ri0"] = "opt_ri=0",           -- round input with precision = 0
                ["ri1"] = "opt_ri=1",           -- round input with precision = 1
                ["ri2"] = "opt_ri=2",           -- round input with precision = 2
                ["ri3"] = "opt_ri=3",           -- round input with precision = 3
```

```
        },
        ["altitude_ft"] = 'INTEGER',
        ["altitude_m"] = 'INTEGER',
        ["comma"] = {
                ["5"] = "opt_comma5",           -- only use numsep grouping if 5
                ["gaps"] = "opt_gaps",          -- use gaps, not numsep, to separ
                ["gaps3"] = "opt_gaps, opt_gaps3",  -- group only in threes rathe
                ["off"] = "opt_nocomma",        -- no numsep in input or output r
        },
        ["debug"] = {
                ["yes"] = "opt_sortable_debug", -- make the normally hidden sort
        },
        ["disp"] = {
                ["5"] = "opt_round=5?",         -- round output value to nearest
                ["b"] = "b",                    -- join: '(...)'
                ["(or)"] = "(or)",              -- join: '(...)' with 'or' betwee
                ["br"] = "br",                  -- join: '<br />'
                ["br()"] = "br()",              -- join: '<br />(...)'
                ["comma"] = "comma",            -- join: ','
                ["flip"] = "opt_flip",          -- reverse order of input/output
                ["number"] = "opt_output_number_only",  -- display output value (
                ["or"] = "or",                  -- join: 'or'
                ["out"] = "opt_output_only",
                ["output number only"] = "opt_output_number_only",
                ["output only"] = "opt_output_only",
                ["preunit"] = "opt_two_preunits",    -- user-specified text befor
                ["sqbr"] = "sqbr",              -- join: '[...]'
                ["table"] = "opt_table",        -- output is suitable for a table
                ["tablecen"] = "opt_tablecen",  -- output is suitable for a table
                ["unit"] = "opt_input_unit_only", -- display input symbol/name (r
                ["unit or text"] = "opt_input_unit_only, opt_ignore_error", -- di
                ["unit2"] = "opt_output_unit_only",
                ["x"] = "x",                    -- join: <first>...<second> (user
        },
        ["frac"] = 'INTEGER',
        ["input"] = 'TEXT',                 -- TEXT should be value><space><unit
        ["lang"] = {                        -- language for output digits (both e
                ["en"] = "opt_lang_en",         -- use en digits for numbers, reg
                ["local"] = "opt_lang_local",   -- use local digits for numbers (
        },
        ["lk"] = {
                ["in"] = "in",                  -- link LHS unit name or symbol
                ["off"] = "off",                -- do not link: same as default e
                ["on"] = "on",                  -- link all unit names or symbols
                ["out"] = "out",                -- link RHS unit name or symbol
        },
        ["order"] = {
                ["flip"] = "opt_flip",          -- reverse order of input/output
                ["out"] = "opt_order_out",      -- do not show input; instead, us
        },
        ["qid"] = 'TEXT',                   -- TEXT should be a Wikidata Q item i
        ["qual"] = 'TEXT',                  -- TEXT should be a Wikidata Q item i
        ["round"] = {
                ["0.5"] = "opt_round=0.5",      -- round output value to nearest
                ["5"] = "opt_round=5",          -- round output value to nearest
                ["10"] = "opt_round=10",        -- round output value to nearest
                ["25"] = "opt_round=25",        -- round output value to nearest
                ["50"] = "opt_round=50",        -- round output value to nearest
                ["each"] = "opt_round_each",    -- using default precision in a r
        },
        ["sigfig"] = 'INTEGER',
        ["sortable"] = {
                ["off"] = "",                   -- ignored (off is the default)
                ["on"] = "opt_sortable_on",     -- output sort key for use in a s
```

```
                ["debug"] = "opt_sortable_on, opt_sortable_debug",  -- |sortable=
        },
        ["sp"] = {
                ["us"] = "opt_sp_us",           -- use U.S. spelling (like "meter
        },
        ["spell"] = {                       -- only English spelling is supported
                ["in"] = "opt_spell_in",        -- spell input value in words
                ["In"] = "opt_spell_in, opt_spell_upper",               -- spell
                ["on"] = "opt_spell_in, opt_spell_out",                 -- spell
                ["On"] = "opt_spell_in, opt_spell_out, opt_spell_upper", -- same;
        },
        ["stylein"] = 'TEXT',
        ["styleout"] = 'TEXT',
        ["tracking"] = 'TEXT',
}

local titles = {
        ["frac"] = "Fraction/styles.css",
        ["sfrac"] = "Sfrac/styles.css",
}

return {
        SIprefixes = SIprefixes,
        all_categories = all_categories,
        all_messages = all_messages,
        currency = { ['$'] = true, ['£'] = true, ['€'] = true, ['₱'] = true, ['']
        customary_units = customary_units,
        disp_joins = disp_joins,
        en_option_name = en_option_name,
        en_option_value = en_option_value,
        eng_scales = eng_scales,
        ranges = ranges,
        titles = titles,
}
```