The Lua-UCA library

Michal Hoftich*

Version v0.1
2020-03-24

Contents

1 The Lua-UCA package 1
  1.1 Install .................................................. 1
  1.2 Usage ..................................................... 2
  1.3 Use with Xindex processor ............................... 2
  1.4 Change sorting rules .................................... 2
    1.4.1 Script reordering .................................. 4

2 What is missing ............................................ 4

3 Available languages ....................................... 4

4 License ..................................................... 4

5 Changelog .................................................. 5

1 The Lua-UCA package

This package adds support for the Unicode collation algorithm for Lua 5.3.

1.1 Install

The package needs to download Unicode collation data and convert it to a Lua table. It depends on wget and unzip utilities.

To install the package in the local TEXMF tree, run:

```make
make
make install
```

*<michal.h21@gmail.com>
1https://unicode.org/reports/tr10/
1.2 Usage

To sort a table using Czech collation rules:

```lua
kpse.set_program_name "luatex"
local ducet = require "lua-uca.lua-uca-ducet"
local collator = require "lua-uca.lua-uca-collator"
local languages = require "lua-uca.lua-uca-languages"
local collator_obj = collator.new(ducet)
-- load Czech rules
    collator_obj = languages.cs(collator_obj)
local t = {"cihla", "chochol", "hudba", "jasan", "čáp"}

table.sort(t, function(a,b)
    return collator_obj:compare_strings(a,b)
end)

for _, v in ipairs(t) do
    print(v)
end
```

The output:

```
   cihla čáp hudba chochol jasan
```

More samples of use can be found in the spec directory. tools/indexing-sample.lua is a simple indexing processor.

1.3 Use with Xindex processor

Xindex\footnote{https://www.ctan.org/pkg/xindex} is flexible index processor written in Lua by Herbert Voß. It supports Lua configuration files, which enables use of Lua-UCA for sorting of the index entries, as shown in this example\footnote{https://tex.stackexchange.com/a/524014/2891} for Norwegian text.

The xindex directory in the source repository\footnote{https://github.com/michal-h21/lua-uca/tree/master/xindex} contains more advanced version of such configuration file together with several examples. Run make xindex command to compile them.

1.4 Change sorting rules

The simplest way to change the default sorting order is to use the tailor_string method of the collator_obj object. It updates the collator object using special syntax which is subset of the format used by the Unicode locale data markup language\footnote{https://www.unicode.org/reports/tr35/tr35-collation.html#Orderings}. 

---

\footnote{<https://www.ctan.org/pkg/xindex>}
\footnote{<https://tex.stackexchange.com/a/524014/2891>}
\footnote{<https://github.com/michal-h21/lua-uca/tree/master/xindex>}
\footnote{<https://www.unicode.org/reports/tr35/tr35-collation.html#Orderings>}
collator_obj:tailor_string "&a<b"

Full example with Czech rules:

kpse.set_program_name "luatex"
local ducet = require "lua-uca.lua-uca-ducet"
local collator = require "lua-uca.lua-uca-collator"
local languages = require "lua-uca.lua-uca-languages"

local collator_obj = collator.new(ducet)
local tailoring = function(s) collator_obj:tailor_string(s) end
tailoring "&c<<ć<<<Č"
tailoring "&h<ch<<ch<<<Ch<<<CH"
tailoring "&R<f<<<Ř"
tailoring "&s<<§<<<§"
tailoring "&z<<ž<<<Ž"

Note that the sequence of letters ch, Ch, cH and CH will be sorted after h
It is also possible to expand a letter to multiple letters, like this example for DIN 2:
tailoring "&ö=Oe"
tailoring "&ő=oe"

Some languages, like Norwegian sort uppercase letters before lowercase.
This can be enabled using `collator_obj:uppercase_first()` function:

local tailoring = function(s) collator_obj:tailor_string(s) end
collator_obj:uppercase_first()
tailoring("&D<<d<<<D<<d<<<D")
tailoring("&th<<t"
tailoring("&Th<<t")
tailoring("&Y<<ü<<ü<<ü<<ü")
tailoring("&l<<æ<<å<<Å<<ã<<ã<<ã<<ã<<ã<<ã<<ã<<ã<<ã<<ã<<ã<<ã<<ã<<ã"
tailoring("&oe<<œ<<œ")

The data/common/collation/ directory contains files from the CLDR project.
They contain rules for many languages. The files needs to be normalized to
the NFC form for example using:
cat cs.xml | uconv -x any-nfc -o cs.xml

The uconv utility is a part of the ICU Project

---

7 <http://userguide.icu-project.org/>
1.4.1 Script reordering

Many languages sort different scripts after the script this language uses. As Latin based scripts are sorted first, it is necessary to reorder scripts in such cases.

The `collator_obj:reorder` function takes table with scripts that need to be reordered. For example Cyrillic can be sorted before Latin using:

```lua
collator_obj:reorder {"cyrillic"}
```

In German or Czech, numbers should be sorted after all other characters. This can be done using:

```lua
collator_obj:reorder {"others", "digits"}
```

The special keyword "others" means that the scripts that follows in the table will be sorted at the very end.

2 What is missing

- Tailorings for most languages.
- Algorithm for setting implicit sort weights of characters that are not explicitly listed in DUCET.
- Special handling of CJK scripts.

3 Available languages

The `lua-uca-languages` library provides the following languages: `cs`, `da`, `de`, `de_din2`, `no`, `ru`

4 License

Copyright 2020 Michal Hoftich

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
5 Changelog

2020-03-24
• version 0.1 released
• initial version for CTAN