16

Lua et l'Unicode

Sujet récurrent sur les mailing-lists Lua depuis quelques années, un support élémentaire de l'Unicode a été ajouté dans la version 5.3. Cet ajout se matérialise par la gestion de l'encodage UTF-8 sous la forme d'un nouveau module `utf8` comprenant six fonctions permettant de lire ou de créer des chaînes UTF-8.

16. Créer une chaîne UTF-8 à partir de codes

La fonction `utf8.char(...)` reçoit une chaîne de 0 à N point(s) de code de caractères Unicode et les convertit en séquences d'octets UTF-8, puis retourne leur concaténation.

```
local mystr = utf8.char(0xBA, 0x53, 0x9A, 0xBB) -- Chaîne : <<Sé>>
print(#mystr, mystr) -- La chaîne fait 7 caractères : 2 octets pour
 chaque caractère de valeur supérieure à 127
```

17. Parcours des chaînes UTF-8

La fonction `utf8.codes(utf8_string)` permet d'itérer sur une chaîne UTF-8 en retournant la position en octets du caractère courant et son point de code, c'est-à-dire le code Unicode attribué au caractère moins 1 (par exemple le caractère Unicode 876 a pour point de code la valeur 875).

```
local mystr = utf8.char(0xBA, 0x53, 0x9A, 0xBB) -- Chaîne : <<Sé>>
for pos, code in utf8.codes(mystr) do
    print(pos, code)
end
--[[ Affiche :
    1 186 -- Position 1, code 186, codé sur 2 octets
    3 83 -- Position 3, code 83, codé sur 1 octet
    4 154 -- Position 4, code 154, codé sur 2 octets
    6 187 -- Position7, code 187 , codé sur 2 octets
--]]
```

18. Obtenir les points de code d'une chaîne

La fonction `utf8.codepoint(utf8_string [,start [, end]])` retourne la liste des points de code des caractères trouvés dans la chaîne `utf8_string`, sous forme d'entiers et à partir de `start` et jusqu'à `end`. L'argument `start` vaut `1` par défaut et `stop` vaut aussi `start` s'il est omis. Si une séquence d'octets incompatible avec le format UTF-8 est trouvée, alors la fonction retourne une erreur.

```
local mystr = utf8.char(0xBA, 0x53, 0x9A, 0xBB) -- Chaîne : <<Sé>>
print(utf8.codepoint(mystr)) -- Affiche 186, car start = 1 et stop =
 1 par défaut
print( utf8.codepoint(mystr, 2)) -- Affiche une erreur, car pas de
 codepoint valide à la position 2
print( utf8.codepoint(mystr, 3)) -- Affiche 83
print( utf8.codepoint(mystr, 1, #mystr)) -- Affiche 182 83 154 187
```

19. Longueur d'une chaîne UTF-8

La méthode `utf8.len(utf8_string [,start [, end]])` retourne la longueur en caractère Unicode de la chaîne UTF-8 donnée en argument, à partir de la position `start` et jusqu'à `end`. En cas d'erreur de codage, la fonction retourne `nil` ainsi que la position de l'erreur dans la chaîne.

```
local mystr = utf8.char(0xBA, 0x53, 0x9A, 0xBB) -- Chaîne : <<Sé>>
print("UTF8 string len :", utf8.len( mystr ) ) -- Affiche 4, car la
 chaîne contient 4 codepoints codé sur 7 octets
```

20. Obtenir la position d'un caractère Unicode

La fonction `utf8.offset(utf8_string, n [, start])` permet de connaître la position en octets du caractère `n` dans la chaîne UTF-8. Si `start` est spécifié, alors `n` est ajouté à `start`. Sinon, `start` vaut 1 si `n` est positif et `#utf8_string + 1` si `n` est négatif.

```
local mystr = utf8.char(0xBA, 0x53, 0x9A, 0xBB) -- Chaîne : <<Sé>>
print("Position du caractère 3 :", utf8.offset( mystr, 3 ) ) --
 Affiche 4
print("Position du dernier caractère", utf8.offset( mystr, -0 ) ) --
 Affiche 6
```

21. Extraire un pattern UTF-8

Le champ `pattern` du module `utf8` contient le motif permettant d'extraire une chaîne UTF-8, grâce aux fonctions `string.match`, `string.gmatch` et `string.gsub`.