Charset Specification for the `N` Programming Language

Kavi Gupta

December 30, 2016

1 Representations

`N` can be represented in three different ways.

- Unicode: each command character is represented as a single code point, which can take anywhere from one to four bytes.
- Ascii: each command character is represented either as a single ascii codepoint or the character \ followed by an ascii codepoint
- Huffman: each command character is represented by a sequence of bits of variable length.

2 Command Character Types

- Parentheses: There are two types of parentheses, which must be matched `()` and {} are the two valid kinds of parentheses.
- `"`: begins a Haskell Style Strings
- `#`: Followed by a sequence of digits, which is treated as a single command.
- Action: Characters that perform an action directly
- Symbols: Push the associated value of the symbol onto the stack or a string with the symbol in it if there is no value associated with it

3 Commands

3.1 Symbols

Unicode	Ascii	Huffman Group	**Type**	**Action**	Library Implementation
$	$	B	Action	Pop and Execute	
&	&	C	Action	Address the stack	
!	!	A33	Symbol	Factorial	N@(N1N-$1*$)(1)ON=$?$$
"	"	A34	Special	N/A	
#	#	A35	Special	N/A	
‖	\"	a36	Symbol	Pop Two	
%	%	A37	Symbol	Modulus	
·	*	a38	Symbol	No operation	
’	’	A39	Symbol	Unmapped	
(	(	A40	Special	N/A	
)	)	A41	Special	N/A	
*	*	A42	Symbol	Multiply	
+	+	A43	Symbol	Add	
,	,	A44	Symbol	Construct Pair	
−	−	A45	Symbol	Subtract/Setminus	
.	.	A46	Symbol	Destruct Pair	
/	/	A47	Symbol	Divide	
:	:	A58	Symbol	Unmapped	
;	;	A59	Action	Pop and Push Twice	
<	<	A60	Symbol	Less than	
=	=	A61	Symbol	Equals	
>	>	A62	Symbol	Greater than	
?	?	A63	Action	Conditional Select	
@	@	A64	Action	Bind top to second	
[	[	A91	Symbol	Index	
]	]	A93	Symbol	Set To Value	
^	^	A94	Symbol	Exponentiate	
_	_	A95	Symbol	Compose	
`	`	A96	Symbol	Pop One	
{	{	A123	Special	N/A	
|	|	A124	Symbol	Push Current Block	
}	}	A125	Special	N/A	
~	~	A126	Action	Lookup Name	
¦	|	a127	Action	Pop number and get level	

3.2 Alphanumerics

Unicode	Ascii	Huffman Group	**Type**	**Action**	Library Implementation
<#>	<#>	A48-A57	Symbol	Push #	
<A..Z>	<A..Z>	A65-A90	Symbol	Unmapped	
<a..b>	<a..b>	A97-A98	Symbol	Unmapped	
c	c	A99	Symbol	List of Characters to String	
d	d	A100	Symbol	Unmapped	
e	e	A101	Symbol	Enumerate	0r$
f	f	A102	Symbol	Fold List	{ωAω(.$γA$γ)w$}$
<g..h>	<g..h>	A103-A104	Symbol	Unmapped	
i	i	A105	Symbol	Input	
<j..k>	<j..k>	A106-A107	Symbol	Unmapped	
l	l	A108	Symbol	Create a List	
m	m	A109	Symbol	Unmapped	
n	n	A110	Symbol	Nil	
o	o	A111	Symbol	Unmapped	
p	p	A112	Symbol	Print	
q	q	A113	Action	Push Code onto String	
r	r	A114	Symbol	range	
<s..v>	<s..v>	A115-A1118	Symbol	Unmapped	
w	w	A119	Symbol	While Loop	1&(β$β1¦)‖?$)
<x..z>	<x..z>	A120-A122	Symbol	Unmapped	

Note: <#> refers to any digit, and <f..h> refers to the range of characters from f to h.

3.3 Greek Alphabet

Unicode	Ascii	Huffman Group	**Type**	**Action**	Library Implementation
α	\a	a1	Symbol	Unmapped	
β	\b	a2	Symbol	Unmapped	
γ	\g	a3	Symbol	Unmapped	
δ	\d	a4	Symbol	Unmapped	
ϵ	\e	a5	Symbol	Unmapped	
ζ	\z	a6	Symbol	Unmapped	
η	\h	a7	Symbol	Unmapped	
θ	\c	a8	Symbol	Unmapped	
κ	\k	a9	Symbol	Unmapped	
λ	\l	a10	Symbol	Unmapped	
μ	\m	a11	Symbol	Unmapped	
ν	\n	a12	Symbol	Unmapped	
ξ	\x	a13	Symbol	Unmapped	
π	\p	a14	Symbol	Unmapped	
σ	\s	a15	Symbol	Unmapped	
τ	\t	a16	Symbol	Unmapped	
ϕ	\f	a17	Symbol	Unmapped	
χ	\j	a18	Symbol	Unmapped	
ψ	\q	a19	Symbol	Unmapped	
ω	\w	a20	Action	Swap Top	
Γ	\G	a21	Symbol	Unmapped	
Θ	\C	a22	Symbol	Unmapped	
Λ	\L	a23	Symbol	Unmapped	
Ξ	\X	a24	Symbol	Unmapped	
Π	\P	a25	Symbol	Unmapped	
Σ	\S	a26	Symbol	Sum List	0{N@(.$N+$1¦)(`N)2&n$?$}$
Φ	\F	a27	Symbol	Unmapped	
Ψ	\Q	a28	Symbol	Unmapped	
Ω	\W	a29	Symbol	Unmapped	