

CATpiler

A Compiler for the LOLCODE language

Stephanie Stroka

`stephanie.stroka@sbg.ac.at`
Department of Applied Informatics
University of Salzburg

1 Syntax

2 EBNF

C-alike syntax	LOLCode syntax
//	BTW
/* ... */	OBTW ... TLDR
#include < module.h >	CAN HAS module
main()	HAI
exit()	KTHXBYE
< type > < var >	I HAS A < var >
...	< var > IS NOW A < type >
< var > == < value >	< var > R < value >
char	CHAR
char[]	CHARZ
int	NUMBR
int[]	NUMBRZ
boolean	TROOF
boolean[]	TROOFZ
untyped	NOOB
true	WIN
false	FAIL
\n	:)
"	:"
< x > + < y >	SUM OF < x > AN < y >
< x > - < y >	DIFF OF < x > AN < y >
< x > * < y >	PRODUKT OF < x > AN < y >
< x > / < y >	QUOSHUNT OF < x > AN < y >
max(< x >, < y >)	BIGGR OF < x > AN < y >
min(< x >, < y >)	SMALLR OF < x > AN < y >
< x > && < y >	BOTH OF < x > AN < y >
< x > < y >	EITHER OF < x > AN < y >
! < x >	NOT < x >
< x ₁ > && < x ₂ > && ... && < x _i >	ALL OF < x ₁ > AN < x ₂ > AN ... AN < x _i > MKAY
< x ₁ > < x ₂ > ... < x _i >	ANY OF < x ₁ > AN < x ₂ > AN ... AN < x _i > MKAY
< x > == < y >	BOTH SAEM < x > AN < y >
< x > != < y >	DIFFRINT < x > AN < y >
< x > > < y >	BOTH SAEM < x > AN BIGGR OF < x > AN < y >
< x > < < y >	BOTH SAEM < x > AN SMALLR OF < x > AN < y >
< x > > < y >	DIFFRINT < x > AN BIGGR OF < x > AN < y >
< x > < < y >	DIFFRINT < x > AN SMALLR OF < x > AN < y >
if	ORLY?
then	YA RLY
elseif	MEBBE
else	NO WAI
end - of - if	OIC
loop	IM IN YR < label > YR < var > [TIL—WILE < expr >]
loop - end	IM OUTTA YR < label >
function(< arg1 >, < arg2 > ...)	HOW DUZ I < label > [YR < arg1 > AN YR < arg2 > ...]
function - end	IF YOU SAY SO
struct < label >	STUFF < label >
struct - end	THATSIT
malloc()	DOWANT

Table 1: Syntax accepted by CATpiler

Non-Terminal	Production
<LETTER>	::= "a" ... "z" "A" ... "Z" .
<DIGIT_NO_ZERO>	::= "1" "2" "3" "4" "5" "6" "7" "8" "9" .
<DIGIT>	::= "0" <DIGIT_NO_ZERO> .
<NUM>	::= <DIGIT_NO_ZERO> { <DIGIT> } .
<UNDERSCORE>	::= " _ " .
<SPECIAL_CHAR>	::= " " <UNDERSCORE> <CTRL_CHAR> " - " " . " " , " " ; " " : " " : " " ; " " ! " " " " " % " " & " " / " " (" ") " " = " " ? " " \ " " / " " * " " + " " > " " < " .
<STRING>	::= " " ({ <LETTER> <DIGIT> <SPECIAL_CHAR> }) " " .
<BOOL>	::= "WIN" "FAIL" .
<IDENTIFIER>	::= <LETTER> { <LETTER> <DIGIT> <UNDERSCORE> } .
<VALUE>	::= <BOOL> <NUM> <STRING> .
<TYPE>	::= "TROOF" "NUMBR" "CHAR" .
<GEN_EXPR>	::= "TROOFZ" "NUMBRZ" "CHARZ" .
<INF_EXPR>	::= ("BOTH SAEM" "DIFFRINT") <OPERATION> "AN" <OPERATION> .
<BLEXP>	::= ("ALL OF" "ANY OF") <BOOL_OP> "AN" <BOOL_OP> .
<EXPR>	::= { "AN" <BOOL_OP> } "MKAY" .
<BOOL_OP>	::= <BOOL_OP> <GEN_EXPR> .
<STR_OP>	::= <BLEXP> <INF_EXPR> (<BOOL> <IDENTIFIER>) .
<NUM_OP>	::= ("BOTH OF" "EITHER OF") <BOOL_OP> "AN" <BOOL_OP> ("NOT" <BOOL_OP>) <EXPR> .
<OPERATION>	::= <STRING> <IDENTIFIER> .
<VAR_INIT>	::= ("SUM OF" "DIFF OF" "PRODUKT OF" "QUOSHUNT OF" "BIGGR OF" "SMALLR OF") <NUM_OP> "AN" <NUM_OP> (<NUM> <IDENTIFIER>) .
<VAR_DECL>	::= <NUM_OP> <BOOL_OP> <STR_OP> .
<VAR_ASSIGN>	::= "I HAS A" <IDENTIFIER> .
<IF>	::= <IDENTIFIER> "IS NOW A" <TYPE> .
<LOOP>	::= <IDENTIFIER> "R" <OPERATION> .
<FLOW_CONTROL>	::= <EXPR> ONLY? YA RLY { <STATEMENT> } { MEBBE <EXPR> { <STATEMENT> } } { NO WAI { <STATEMENT> } } OIC .
<FUNC_CALL>	::= "IM IN YR" <IDENTIFIER> [YR <IDENTIFIER>] [WILE TIL <EXPR>] { <STATEMENT> } "IM OUTTA YR" <IDENTIFIER> .
<STATEMENT>	::= <IDENTIFIER> { <EXPR> } .
<FUNCTION>	::= <VAR_INIT> <VAR_DECL> <VAR_ASSIGN> <OPERATION> <FLOW_CONTROL> <FUNC_CALL> .
<MODULE>	::= "HOW DUZ I" <IDENTIFIER> [YR <IDENTIFIER> { AN YR <IDENTIFIER> }] { <STATEMENT> } { "FOUND YR" <EXPR> "GTFO" } "IF YOU SAY SO" .
<STRUCT>	::= "CAN HAS" <IDENTIFIER> " ? " .
<MAIN>	::= "STUFF" { <VAR_DECL> } "THATSIT" .
<PROGRAMM>	::= "HAI" { <STATEMENT> } "KTHXBYE" .
	::= { <MODULE> } [<MAIN>] { <FUNCTION> } .

Table 2: Extended Backus-Naur-Form for LOLCODE