Comrite

Flex Lex cheat sheet

mike (created at : June 29, 2016 11:34 | Updated at : June 29, 2016 14:12)

flex/lex cheat sheet

Min Wang <mingewang at gmail.com >

flex -o lexer.c mylex.l (will auto generate lexer.c as lexical analyzer)

here is the sample mylex.l

```
==========================================================================
/* definitions,
could be used in the rule sections
format : name definition
*/
LETTER [a-zA-Z]
CAPITAL [A-Z]
SMALL [a-z]
DIGIT [0-9]
IDENT [a-zA-Z0-9'_]


%top {
/*  put the code to the top of the generated file, before any flex definitions */
}

/*
see details at:
http://flex.sourceforge.net/manual/Scanner-Options.html#Scanner-Options
reentrant http://flex.sourceforge.net/manual/Reentrant.html#Reentrant
*/
%option noyywrap reentrant (‘-R’)

%{
  /* c code/ define, include, functions etc */
  #define yylval filterlval
  #include "y.tab.h"
%}

/* start conditions/state.
http://flex.sourceforge.net/manual/Start-Conditions.html#Start-Conditions
defined by ‘%s’ or ‘%x’ or '%START'
‘%s’ declars inclusive start conditions,
'%x’ declars exclusive start conditions.
.e.g: */
```

%s YYINITIAL
%START YYINITIAL COMMENT CHAR CHARESC CHAREND STRING ESCAPED

%x comment

```
================================================================
%%
/*
```
Rules Section
format is: pattern actions

the match will be put in: yytext, yyleng, then run the actions

some pre-defined actions:
  ECHO, BEGIN, REJECT, yymore(), yyless(n)
  input(), YY_FLUSH_BUFFER, yyterminate()

some predfined variables:

File* yyin: default input
yyrestart: may be called to point yyin at the new input file
FILE* yyout: which ECHO actions are done
YY_CURRENT_BUFFER: returns a YY_BUFFER_STATE handle to the current buffer.
YY_START: int of current start condition
*/

```
username    printf( "%s", getlogin() );

/*
Interface with bison/yacc:
  bison/yacc parsers call  `yylex()' to find the next input token
  yylex will return the type of the next token and associated value in the global yylval.
  The global variable yylval which lex
  uses to return token values is declared as a YYSTYPE union.
e.g:
%union {
      long int4;            /* Constant integer value */
      float fp;             /* Constant floating point value */
      char *str;            /* Ptr to constant string (strings are malloc'd) */
      exprT expr;           /* Expression -  constant or address */
      operatorT *operatorP;   /* Pointer to run-time expression operator */
   };
e.g:
*/
[0-9]+       yylval.int4 = atoi( yytext ); return TOK_NUMBER;



/* will only be active when the scanner is in the start condition name
```

```
  comment
  <*> matches every start condition.
  flex provide: YY_START or YYSTATE to access those condition
  to manipulate stacks of start conditions: yy_push_state(), yy_pop_state(),yy_top_state()
*/

"/*"        BEGIN(comment);  /* start comment condition */

<comment>[^*\n]*        /* eat anything that's not a '*' */
<comment>"*"+[^*/\n]*   /* eat up '*'s not followed by '/'s */
<comment>\n             ++line_num;
<comment>"*"+"/"        BEGIN(INITIAL); /* end comment, and start INITIAL condition */




%%
=============================================================================
/* user code ( copied to lex.yy.c) */
void initialize_lexer(FILE *inp) { yyin = inp; BEGIN YYINITIAL; }
```

This document is generated by comrite.com