

# MAlice Language Specification

Peter Hamilton

Sarah Tattersall

December 2, 2011

## 1 BNF Grammar

S	→	code_seperator
code_seperator	→	statement_list function_seperator functions   statement_list
statement_list	→	statement seperator   statement seperator statement_list
seperator	→	<b>'and'</b>   <b>'but'</b>   <b>'then'</b>   <b>'.'</b>   <b>'?'</b>   <b>'.'</b>
statement	→	expression <b>'spoke'</b>   expression <b>'said'</b> <b>'Alice'</b>   <b>'Alice found'</b> expression   <b>'what'</b> <b>'was'</b> expression   expression <b>'thought'</b> <b>'Alice'</b>   statement <b>'too'</b>   id <b>'was'</b> <b>'a'</b> type   id <b>'became'</b> expression   array_access <b>'became'</b> expression

	id <b>'had'</b> expression type   <b>'eventually'</b> '(' expression_logical ')' <b>'because'</b> statement_list <b>'enough'</b> <b>'times'</b>   <b>'either'</b> '(' expression_logical ')' <b>'so'</b> statement_list <b>'or'</b> statement_list <b>'Alice'</b> <b>'was'</b> <b>'unsure'</b> <b>'which'</b>   <b>'perhaps'</b> '(' expression_logical ')' <b>'so'</b> statement_list <b>'Alice'</b> <b>'was'</b> <b>'unsure'</b>   <b>'perhaps'</b> '(' expression_logical ')' <b>'so'</b> statement_list logical_clauses   expression
expression	→ id '(' function_arguments ')'   id <b>'went'</b> <b>'through'</b> id   '(' expression ')'   '~' expression   id <b>'drank'</b>   id <b>'ate'</b>   expression ' ' expression   expression '^' expression   expression <b>'&amp;'</b> expression   expression <b>'+'</b> expression   expression <b>'-'</b> expression   expression <b>'*'</b> expression   expression <b>'/'</b> expression   expression <b>'%'</b> expression   <b>'-'</b> expression   ( <i>Uses UMINUS precedence</i> ) expression_logical   array_access   factor
expression_logical	→ expression <b>'=='</b> expression   expression <b>'&lt;'</b> expression   expression <b>'&gt;'</b> expression   expression <b>'&gt;='</b> expression   expression <b>'&lt;='</b> expression   expression <b>'!='</b> expression   expression <b>'&amp;&amp;'</b> expression   expression <b>'  '</b> expression
array_access	→ id expression <b>'piece'</b>

factor	→	number   letter   id   sentence
type	→	<b>'number'</b>   <b>'letter'</b>   <b>'sentence'</b>
functions	→	function function_seperator functions   function
function_seperator	→	<b>'The'</b> <b>'room'</b>   <b>'The'</b> <b>'Looking-Glass'</b>
function	→	id <b>'('</b> arguments <b>)'</b> <b>'contained'</b> <b>'a'</b> type statement_list   id <b>'('</b> arguments <b>)'</b> <b>'contained'</b> <b>'a'</b> type id <b>'changed'</b> <b>'a'</b> type statement_list
logical_clauses	→	logical_clause logical_clauses   <b>'Alice'</b> <b>'was'</b> <b>'unsure'</b> <b>'which'</b>
logical_clause	→	<b>'or'</b> <b>'maybe'</b> <b>'('</b> expression_logical <b>)'</b> <b>'so'</b> statement_list   <b>'or'</b> statement_list
arguments	→	argument <b>'.'</b> arguments argument
argument	→	type id   <b>'spider'</b> type id
function_arguments	→	function_argument <b>'.'</b> function_arguments   function_argument
function_argument	→	expression

## 2 Precedences

precedence = (

```
('left', 'L_OR'),
('left', 'L_AND'),
('left', 'L_EQUAL', 'L_NOT_EQUAL'),
('left', 'L_LESS_THAN', 'L_LESS_THAN_EQUAL', 'L_GREATER_THAN', 'L_GREATER_THAN_EQUAL'),
('left', 'B_OR'),
('left', 'B_XOR'),
('left', 'B_AND'),
('left', 'PLUS', 'MINUS'),
('left', 'MULTIPLY', 'DIVIDE', 'MOD'),
('right', 'INCREMENT', 'DECREMENT', 'B_NOT', 'UMINUS'),
('left', 'L_PAREN', 'R_PAREN'),
)
```