Loop Control Syntax in alcode

The following loop controls are available in alcode:

Iteration through properties of a structure:

FOR ALL | STRING | NUMBER PROPERTIES <simple_variable> IN <structure_variable> <statement>

```
$a.1 = “Part 1”;

$a.2 = “Part 2”;

$a.”Content” = “Contents”;

$i = “Index”;

$a.$i = “Index Page”;



for all properties $p in $a begin

   console(“{$p} value {$a.$p}”);

   end



// Prints out

1 value Part 1

2 value Part 2

Content value Contents

Index value Index Page
```

// while just extracting numeric properties

```
for  number properties $p in $a begin
   console(“Number {$p} has value {$a.$p}”);
   end
// Prints out
Number 1 has value Part 1
Number 2 has value Part 2
```

// and thus for strings

```
for string properties $p in $a begin
   console(“String {$p} has value {$a.$p}”);
   end

// Prints out
String Content has value Contents
String Index has value Index Page
```

WHILE loop

```
WHILE <Boolean_condition> <statement>



$a = 1; $b = 10;

while $a is < $b begin

   console($a);

   $a = $a + 1;

   end

// Prints out digits 1 to 9
```

FOR Loops

```
FOR <simple_variable> IS <expression> STEP <expression> UNTIL | TO
<expression> <statement>



for $a is 1 step 1 until 10 console($a);

// Prints out digits 1 to 9

for $a is 1 step 1 to 10 console($a);

// Prints out digits 1 to 10
```

Note: wary of side-effects: the initial expression and the step expression are only executed (resolved) once at the start of the for loop, however, the terminating expression (the until clause) is resolved each time around the loop.

```
FOR <simple_variable> ALL | EACH <expression_list> <statement>

<expression_list> :== <expression>[, <expression_list>]

for $a all 1,3,5 console($a);

// Prints out digits 1 3 and 5

$b = 6; $c = 11;

for $a each $b*$c, $b+4 console($a);

// Prints out digits 66 and 10
```

Note: each expression in the expression list is resolved once only and only when required to be assigned to the control variable. Keywords ALL and EACH are equivalent.
Be wary of side effects (or make use of them):

```
$b = 4; #c = 11;

for $a each 1, #c, get_a_value(), #c begin

   console($a);

   end


console("Final a is {$a}");

exit true;


function get_a_value() begin

   #c = 22;

   return "a side effect value {#c}";

   end


Request to execute process ref Loops at 2012-09-19 15:46:19

1

11

a side effect value 22

22

Final a is 22
```

Use of LIMIT

All loops can have the limit clause, the limit clause can be useful while in test mode to avoid unintended infinite loops, but can be used to dynamically restrict a structure or expression list, see examples. The Limit expression is only resolved once at the beginning of the loop.

```
FOR LIMIT 5  $a is 1 STEP 1 UNTIL 10 console($a);

// Only prints out digits 1 to 5
```

```
$a.1 = “Page 1”;

$a.2 = “Page 2”;

$a.3 = “Page 3”;

$a.4 = “Page 4”;

$a.5 = “Page 5”;

$page_limit = 2;

for limit $page_limit number properties $page in $a begin

   console(“Page {$page} says {$a.$page}”);

   end

// Prints out only

Page 1 says Page 1

Page 2 says Page 2
```

Limit in expression lists:

```
for limit limit_today() $report each 1, 2, 3, 4, 5, 6 begin
   console(“Reporting {$report}”);
   end
function limit_today() begin
   if date(‘w’) is 0 return 1;
   return 6;
   end
// Prints out
Reporting 1 // on Sundays
// Prints out
Reporting 1
Reporting 2
…
Reporting 6 // all other days
```