METVIEW MACRO SYNTAX

Introduction

These pages describe, though examples of usage, the basic syntax of Metview Macro. Loops, conditional tests and function usage are demonstrated. Significant keywords are highlighted.

Loops

```
# basic for loop
for i = 1 to 4 do
   print (i)
end for


# for loop with a list
for i = 1 to count(thisList) do
   print (i, " : ", thisList[i])
end for


# for loop using dates with a step
for day = 2003-01-24 to 2003-02-14 by 3 do
   print (day)
end for


# basic while loop
n = 1
while n <= 10 do
   print(n)
   n = n + 1
end while


# basic repeat loop
n = 1
```

```
repeat
   print(n)
   n = n + 1
until n > 10
```

```
# loop - can be used on lists, fieldsets and geopoints
loop element in thisList
   print(element)
end loop
```

Tests

```
# basic if test
if a = b then
   print('a and b are equal')
end if
```

```
# if test with an else condition
if a = b then
   print('a and b are equal')
else print('a and b are different')
end if
```

```
# if test with an else if and an else condition
if a > 0 then
   print('a is positive')
else if a < 0 then
   print('a is negative')
else print('a is null')
end if
```

```
# when statement. The code following the first true expression is
# executed.
when
   a > 0 :
        print('a is positive')
        end
   a < 0 :
        print('a is negative')
        end
   a = 0 :
        print('a is null')
        end
end when
```

```
# case statement
case type(x) of
   'number' :
```

```
        print('x is a number')
        end
    'date' :
        print('x is a date')
        end
    otherwise :
        stop('Unsupported type')
        end
end case
```

Functions

```
# declare a function that takes no parameters
function fn_no_params ()
    print ('This function takes no parameters')
end fn_no_params


# declare a function that takes two parameters of any type
function fn_2params_any_type (param1, param2)
    print ('Params are: ', param1, ' and ', param2)
    print ('Types are: ', type(param1), ' and ', type(param2))
end fn_2params_any_type


# declare a function that takes two parameters of specific
types
function fn_2params_specific_types (num_days:number, title:string)
    print (num_days, ' ', title)
end fn_2params_specific_types


# declare a function that can take any number of parameters
# also return the number of parameters passed
function fn_any_parameters
    loop n in arguments()
        print (n)
    end loop
    return count(arguments())
end fn_any_parameters


# call the above functions

fn_no_params

fn_2params_any_type (2, 'hello')
fn_2params_any_type ('hello', 2003-02-19)

fn_2params_specific_types (5, 'Title String')

num_params = fn_any_parameters (5, [3,3,4], 'hello')
print (num_params)
```