MSX Article

MARMSX

Functions in Basic

Summary

This article aims at presenting the function resource in MSX Basic language, from native functions to those created by user.

1- Introduction

A function is a subroutine separated from the main code, which is dedicated for solving a specific and recurrent problem, avoiding duplicated code. Generally a function receives a set of parameters as input, performs some operations and returns a value as output.

In structured languages as C and Pascal, a function is a piece of code explicitly separated from the main program body. In these languages, a function is composed by a name, the input parameters, the return data type and the code for solving the problem. Example:

```
return_data_type function_name(parameter_list)
{
      // code
}
```

Once the Basic is not a structured language, the functions are created logically inside the main body.

There are two types of functions in Basic: native and created by user. As an example of native functions, we have:

abs(arg)	atn(arg)	bin$(arg)	hex$(arg)
peek(arg)	poke(arg)	sin(arg)	right$(args)

Could you identify the return data type from a Basic function?

When the function name ends with a “$”, the returned value is a string. Otherwise, the returned value is a number. By observing the above native functions examples, we conclude that the function “abs” returns a number while the function “hex$” returns a string.

Whenever a function is called in Basic, we must store the result in a variable or set it as an input for another function or command (like PRINT). The function called alone is not allowed and an syntax error happens. See the following examples:

i = abs(-1)	✔	n$ = bin$(10)	✔
print abs(2)	✔	print bin$(10)	✔
abs(2)	✖	bin$(10)	✖
c$ = abs(-1)	✖	f = bin$(10)	✖

Storing the returned value in variable with different data type is also not allowed.

2- Functions created by user: DEF FN

The user may create his own functions in Basic. There are two ways to achieve that:

- Simple functions using DEF FN command.
- More complex functions using subroutines (GOSUB-RETURN).

The command DEF FN creates simple functions where only expressions are supported. The syntax for DEF FN is:

```
DEF FN function_name (parameter_list) = expression
```

The Basic interpreter do not accept the command DEF FN as a direct command, returning a illegal direct error. In that case, DEF FN must be always inside a program.

In our first example, we will create a function called “squared” which returns the square for a given number.

```
10 DEF FN SQUARE(X) = X*X
```

How to call the function created? We must call the function using its name preceded by the term “FN”, and the parameter list fulfilled. Example:

```
20 PRINT FN SQUARE(4)
```

Parameter data type must be the same as defined on DEF FN declaration. In this case, a number.

The complete program:

example1.bas

```
10 DEF FN QUAD(X) = X*X
20 PRINT FN QUAD(4)
```

Exit:
16

The function expression allows to call other functions in cascade. See the next example:

```
10 DEF FN CALC(X) = X*SIN(X)
```

The next example uses three parameters as input.

example2.bas

```
10 DEF FN FUNXYZ(X,Y,Z) = (X+Y)*Z
20 PRINT FN FUNXYZ(1,2,3)
```

Exit:
9

The parameter list must strictly follow the definition order. So, the X variable gets the value 1, Y the value 2 and Z the value 3.

In the next example we will concatenate the input string with itself.

example3.bas

```
10 DEF FN CAT$(S$) = S$+S$
20 PRINT FN CAT$("BYE")
```

Exit:
BYEBYE

It is possible to define the parameter list with different data types. Example:

```
10 DEF FN F(S$, C) = LEN(LEFT$(S$,C)+RIGHT$(S$,C))
```

We also may have one data type on the input parameter and another data type on the returned value. See the next example.

In this example, we want to format a number with preceding zeros, displaying always a certain number of digits. Suppose we want to use 2 digits.

example4.bas

```
10 DEF FN FRMT$(N) = RIGHT$("0"+RIGHT$(STR$(N),LEN(STR$(N))-1),2)
20 PRINT FN FRMT$(4)
30 PRINT FN FRMT$(10)
```

Exit:
04
10

In the next example, the number of digits is passed through the input parameter “C”.

example5.bas

```
10 DEF FN FRMT$(N,C) = RIGHT$(STRING$(C-1,"0")+RIGHT$(STR$(N),LEN(STR$(N))-1),C)
20 PRINT FN FRMT$(5,3)
30 PRINT FN FRMT$(32,5)
```

Exit:
005
00032

In the last example for DEF FN, we will format the output data from a table, using the numeric format from the example4.bas.

```
example6.bas
```

```
10 DEF FN FRMT$(N) = RIGHT$("0"+RIGHT$(STR$(N),LEN(STR$(N))-1),2)
20 DIM T$(2):DIM NA(2,3)
30 T$(1) = "Patty":NA(1,1)=4:NA(1,2)=6:NA(1,3)=1985
40 T$(2) = "Daniel":NA(2,1)=25:NA(2,2)=2:NA(2,3)=1988
50 FOR I=1 TO 2
60 PRINT USING "NAME: \        \ - BIRTH: &/&/####";T$(I);FN FRMT$(NA(I,1));FN FRMT$(NA(I,2));NA(I,3)
70 NEXT I
```

Exit:

```
NAME:  Patty – BIRTH: 04/06/1985
NAME:  Daniel  – BIRTH: 25/02/1988
```

3- Functions created by user: GOSUB-RETURN

Functions can also be created in Basic using GOSUB-RETURN. Different from the DE FN, is it possible now to write a subroutine to solve a problem. Nevertheless, it is not possible to use neither input parameters nor receive a return value. In this case, global variables are used to communicate with the function.

The “example1.bas” calculates the square of a given number. Now we will re-create this function using GOSUB-RETURN.

```
example7.bas
```

```
10 N=4
20 GOSUB 500
30 PRINT "The square of ";N;"is: ";Q
40 END
500 REM
510 REM FUNCTION SQUARE
520 REM
530 Q = N*N
540 RETURN
```

Exit:

```
The square of 4 is: 16
```

As told before, the functions in Basic are created logically inside the program. In order to help on identifying it, we located the function at the bottom of the list as well introduced some comments at the beginning of the function.

The GOSUB-RETURN function equivalent to the “example4.bas” is the following:

example8.bas

```
10 N=4
20 GOSUB 500
30 PRINT NF$
40 END
500 REM
510 REM FUNCTION FRMT
520 REM
530 NF$ = STR$(N)
540 NF$ = RIGHT$("0"+RIGHT$(NF$,LEN(NF$)-1),2)
550 RETURN
```

Exit:
04

4- Credits

This article was written by Marcelo Silveira, on April 2017.

Home-page: http://marmsx.msxall.com
E-mail: flamar98@hotmail.com

Note: this is a translation from the original article titled “Funcoes em Basic”, in portuguese, written by the same author.