Global String Functions

Table of Contents

UCase(s as String) as String

Converts the string to all upper case.

```
print UCase("Hello")  ' prints: HELLO
```

LCase(s as String) as String

Converts the string to all lower case.

```
print LCase("Hello")  ' prints: hello
```

Asc(letter as String) as Integer

Returns the Unicode ("ASCII") value for the first character of the specified string.

An empty string argument will return 0.

```
print Asc("C") ' prints: 67
```

Chr(ch as Integer) as String

Performs the inverse of the Asc function: returns a one-character string whose character has the specified Unicode value.

Returns empty string if the specified value is 0 or an invalid Unicode value.

```
print Chr(67) ' prints: C
```

By using Chr, you can create strings containing characters which cannot be contained in quotes, such as newline or the quote character itself.

```
print (Chr(34) + "hello" + Chr(34))  ' prints: "hello"
```

Instr(start as Integer, text as String, substring as String) as Integer

Returns the position of the first instances of substring within text, starting at the specified start position.

Returns 0 if the substring is not found. Unlike the ifString.Instr() method, the first position is 1.

```
print Instr(1, "this is a test", "t")  ' prints: 1
print Instr(2, "this is a test", "t")  ' prints: 11
print Instr(1, "this is a test", "is") ' prints: 3
```

Left(s as String, n as Integer) as String

Returns the first *n* characters of *s.*

```
print Left("timothy", 3) ' prints: tim
```

Len(s as String) as Integer

Returns the number of characters in the specified string.

```
print Len("timothy") ' prints: 7
```

Mid(s as String, p as Integer, [n as Integer]) as String

Returns a substring of *s* with length *n* and starting at position *p*.

n may be omitted, in which case the string starting at p and ending at the end of the string is returned.

Unlike the ifString.Mid() method, the first character in the string is position 1.

```
print mid("timothy", 4, 3) ' prints: oth
```

Right(s as String, n as Integer) as String

Returns the last n characters of s.

```
print right("timothy", 3) ' prints: thy
```

Str(value as Float) as String

Strl(value as Integer) as String

Converts a *value* to a string. Str(A), for example, returns a string equal to the decimal representation of the numeric value of A.

For example, if A# = 58.5 then Str(A#) equals the string " 58.5". If A# = -58.5 then Str(A#) equals the string "-58.5".

Note: for non-negative numbers, a leading blank is inserted before the value string as a sign placeholder.

(For integer values, *value*.ToStr() may be used instead if a leading blank for non-negative numbers is not desired. See ifIntOpts in roInt).

Note: see the ifStringOps interface in roString for the corresponding inverse / string-to-value functions ToInt() and ToFloat().

StrI(value as Integer, radix as Integer) as String

Converts the integer value into a string representation using the given radix.

If radix is not 2 .. 36 then an empty string is returned.

Note that the returned string does not include a base prefix and uses lowercase letters to represent those digits in bases greater than 10.

Example:

```
print StrI(255, 16) '= "ff"
print StrI(9, 2) '= "1001"
```

String(n as Integer, str as String) as String

Returns a string composed of n copies of the second argument concatenated together.

For example,

```
print String(4, "ab") ' prints: abababab
```

StringI(n as Integer, ch as Integer) as String

Returns a string composed of n copies of the character whose Unicode value is the second argument.

For example,

```
print StringI(5, 67) ' prints: CCCCC
```

Val(s as String) as Float

Performs the inverse of the STR function: returns the number represented by the characters in a string argument.

For example, if A$="12" and B$="34" then VAL(A$+ "."+B$) returns the number 12.34.

Val(str as String, radix as Integer) as Integer

Returns the integer value from parsing the string with the specified radix.

Radix should be 2 .. 36 or the special value 0 (which automatically identified hexadecimal or octal numbers based on 0x or 0 prefixes respectively).

Leading whitespace is ignored then as much of the rest of the string will be parsed as valid.

Example:

```
print Val("0x80", 0) '= 128
print Val("FF", 16) '= 255
print Val("1001", 2) '= 9
```

Substitute(str as String, arg0 as String, arg1 = "" as String, arg2 = "" as String, arg3 = "" as String) as

String

Replaces all instances of {0} or ^0 in str with arg0. Similarly, replaces all instances of {1} or ^1 with arg1, {2} or ^2 with arg2, and {3} or ^3 with arg3.

Example

```
thing = "book"
color = "red"
print Substitute("My {0} is {1}.", thing, color) ' prints "My book is red."
```