

h— NUMBER — (— <ae> —) —————|

NUMBER returns the English text of the real value given by <ae>.

— OBJECTS — (— <MX situ context — : — <be> —) —|
| <PER situ context —|
| <PRINTS situ context> —|
| <TAPEDB situ context> —|
| <VDBS situ context> —|
| <VL situ context> —|

OBJECTS returns a comma-delimited string list of entities (e.g. mix numbers, unit numbers, tape serials etc.) that match the SITUATION.

— PAD — (— <se> — , — <ae> —) —————|

PAD returns the contents of the string <se> right-justified up to the size of <ae>.

— REPEAT — (— <se> — , — <ae> —) —————|

REPEAT returns a string that concatenates <ae> copies of the <se> together.

— STRING — (— <ae1> — , — <ae2> —) —————|
| * —|

STRING is identical to the same intrinsic in WFL. Note that, <ae1> is integerised before conversion.

— STRING8 — (— <ae1> — , — <ae2> —) —————|
| * —|

The OPAL STRING8 function is identical to the ALGOL STRING8 function. No integerisation is performed on <ae1> before conversion.

— TAIL — (— <se> , — <string constant> —) —|
| NOT —| ALPHA —|

TAIL is identical to the same intrinsic in WFL except that no run-time errors occur if the <se> has zero length.

— TAKE — (— <se> — , — <ae> —) —————|

The TAKE function works as in WFL except no run time errors occur if the value of <ae> is less than zero or greater than the length of the string.

— TIME — (— <ae> —) —————|

TIME generates a string in hours, minutes, and seconds separated by colons. <ae> is assumed to be a value in seconds.

— TRIM — (— <se> —) —————|

TRIM will remove from its string parameter any leading or trailing spaces.

— UPPER — (— <se> —) —————|

UPPER converts all lower case alpha characters in the given <se> into upper case.

— USERFN — (— <ae> — , — <se> —) —————|

USERFN allows the OPAL programmer to call a user-defined library procedure.

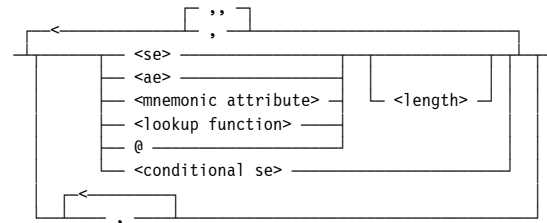
— VALID — (— <attribute primary> —) —————|

VALID returns FALSE if the value of an attribute is not defined.

— VIA — (— <ref attribute> — : — <expression> —) —|
| TASK(<ae>) —|
| TASKSTR(<se>) —|
| UNIT(<ae>) —|
| UNITSTR(<se>) —|
| PRINTS(<ae>) —|
| PRINTSSTR(<se>) —|
| TAPEDB(<se>) —|

VIA returns the <expression> evaluated for the object passed in the first parameter. Task expects a mix number, UNIT a unit number etc.

OPAL strings



Each element in an OPAL string is separated by a single comma ',' or a double-comma ',' (this inserts a space character between elements).

Hash-paren #() function

— # (— <OPAL string> —) —————|

The # () function allows the construction of a complex string expression using simple OPAL strings concatenated with commas.

E.g. \$MCPVAR:= #(MCP,TIME(TIMEOFDAY))

Lookup functions

— # — [— <character lookup> —] —|
| MX — = — <program name> —|
| PK — = — <family name> —|

A lookup function defines a target for possible text replacement in an OPAL string. In Supervisor, where MX or PK is used, mixnumber or unit number lists will be returned.

OPAL

Variables & Intrinsic Reference

Relative to OPAL version 510.12

Last modified 11th May 2005

<ae> = <arithmetic expression>

<be> = <boolean expression>

<se> = <string expression>

<arithmetic OPAL variable>

— # — <17-character identifier> —————|

<string OPAL variable>

— # — <17-character identifier> —————|

OPAL Variable Methods

— # <var> .ACCUM — (— <se> —) —————|

ACCUM adds the arithmetic value from the <ae> parameter to the variable named in the <se> parameter. It returns the <ae> as its result.

— # <var> .DELTA — (— <ae> —) —————|

DELTA stores a new arithmetic value from the <ae> parameter into the variable named in the <se> parameter. It returns the difference between the new value and the prior value.

— # <var> .SUM — (— <ae> —) —————|

SUM adds the arithmetic value in the <ae> parameter to the value held in variable <se> parameter and also returns the new value.

— \$ <var> .SPLIT — (— <ae> —) —|
| / —|
| <se> —|

SPLIT returns the string held in \$ <var> up to but not including the first occurrence of the string target (<se> or /); if no parameter is given, the delimiter ',' is used. The returned string is removed from the value held in \$ <var>.

If an arithmetic expression is passed, that number of leading characters is returned and removed from the string held in \$ <var>.

— \$ <var> .STORE — (— <se> —) —————|

STORE assigns a new string value into \$ <var> and returns an empty string ("") when used in an expression.

— # <var> .STORE — (— <ae> —) —————|

STORE assigns a new numeric value into # <var> and returns an empty string ("") when used in an expression.

OPAL Variable Properties

`# <var>` `.CONFIG`

The CONFIG property allows Metalogic configuration entities to be interrogated in OPAL; it cannot be used in the LHS of any assignment statement.

`# <var>` `.GLOBAL`

The GLOBAL property allows global data to be shared between multiple OPAL programs. Global values are preserved and available during that instance of Supervisor.

`# <var>` `.PERManent`

The PERM property allows global data to be shared between OPAL programs run by Supervisor and Flex. Both string and numeric values are retained over Supervisor restarts, H/L etc.

OPAL intrinsics

`COMS` (`<se>`)

COMS passes the string expression as a command to SYSTEM/COMS and returns the response as a string.

`COUNT` ((`<MX situ context>` : `<be>`) (`<PER situ context>` (`<PRINTS situ context>` (`<TAPEDB situ context>` (`<VL situ context>`))))

COUNT returns, as an integer value, the number of times that a SITUation returns TRUE for any of the above contexts. There are no restrictions on COUNT usage.

`DATETOTEXT` ((`<julian date>` , `<mnemonic mapping>`)

), (`NOSLASH` (`DELIM`))

DATETOTEXT is used to convert a <julian date> into a variety of string formats, determined by value of <mnemonic mapping>.

<mnemonic mapping>

DDMMYY ¹	MMDDYY ¹	YYMMDD ¹	DDMMYY ¹	MMDDYYYY ¹
YYYYMMDD ¹	YYDD ¹	YYYYDD ¹	DEFAULT ¹	DDMONYYYY ²
DDMONTHYYYY ²	DDMONYY ²	DDMONTHYY ²	YYYYMONDD ²	YYYYMONTHDD ²
YYMONTHDD ²	YYMONDD ²	MONDDYYYY ²	MONTHDDYYYY ²	MONDDYY ²
MONTHDDYY ²				

NOSLASH removes "/" from date only for these mappings marked ¹
DELIM inserts appropriate spaces into date using mappings marked ²

`DAYNAME` (`<ae>`)

DAYNAME returns name of the day of the week indicated by <ae>. For <ae> in range 1-70000, DAYNAME uses formula (D-1) MOD 7+1; for <ae> in range 70001-135365 and 1900001-2035365, <ae> is assumed to be a Julian date. Day 1=SUNDAY, 7=SATURDAY.

`DAYS` (`<ae>` , `<ae>`)

DAYS gives the number of days between two Julian dates.

`DBS` (`<ae>` , `<se>`)

DBS passes the string expression <se> as an SM command to the database mixnumber specified by <ae> and returns string response.

`DECAT` (`<sel>` , `<se2>` , `<N>`)

N	Binary string	Result
0	000	empty string (degenerate case)
1	001	following subset
2	010	target string
3	011	target & following substring
4	100	prior substring
5	101	prior & following substring
6	110	prior substring & target string
7	111	source string (degenerate case)

DECAT divides the source <sel> into three parts and returns combinations depending on the selector value specified by <N>.

`DECIMAL` (`<se>`)

DECIMAL converts number string <se> to an arithmetic value.

`DROP` (`<se>` , `<ae>`)

The DROP function works as in WFL, except no run time errors occur if the value of <ae> is less than zero or greater than the length of the string.

`FILEID` (`<se>` , `<ae>`)

FILEID returns the <ae> identifier of a file title string denoted by <se>.

`FILEIDS` (`<se>`)

FILEIDS returns the number of identifiers in the file title held in <se>.

`HEAD` (`<se>` , (`NOT` (`<string constant>`))

HEAD is identical to the same Intrinsic in WFL, except no run time errors occur if the <se> has zero length.

`HEX` (`<se>`)

HEX converts a hexadecimal string <se> to an arithmetic value.

`HEXSTRING` (`<ae>`)

HEXSTRING provides conversion of real <ae> into hex string format.

`INPUT`

INPUT allows external message or text to be passed to a waiting ODTs from a COMS window, returning a string of the user input.

`INDEXOF` (`<be>`)

The INDEXOF intrinsic returns an index value indicating which entity of a set of tests, denoted by <be>, has evaluated to TRUE.

`INTEGER` (`<ae>`)

INTEGER returns the integer value of <ae> (no fractional part).

`KEYIN` (`<se>`)

KEYIN allows the capture of command responses within an OPAL program, where <se> is a valid ODT or Supervisor command.

`JULIAN` (`<se>`)

JULIAN converts <se> to an integer Julian date in YYYYDDD format.

`LENGTH` (`<se>`)

LENGTH returns the number of characters contained in string <se>.

`LOWER` (`<se>`)

LOWER converts all upper case alpha characters in the given <se> into lower case.

`MAIL` (`<header string>` , `<se>`)

`<header string>`
" `SUBJECT: <text>` " `<hdr> : <address list>`
`<hdr> := BCC | CC | FROM | REPLY | TO`

MAIL statement allows the generation & sending of emails from Supervisor via Metalogic MAILLIB. Any OPAL program may use this function.

`MAX` (`<ae1>` , `<ae2>`)

MAX returns the value of the larger of the <ae1> and <ae2>.

`MIN` (`<ae>` , `<ae>`)

MIN returns the value of the smaller of the two <ae>s.

`MONTHNAME` (`<ae>`)

MONTHNAME returns as string the month name indicated by <ae>.

`NEWDATE` (`<ae1>` , `<ae2>`)

NEWDATE returns a Julian date a specified number of days <ae1> before or after a given Julian date <ae2>.