

AutoLISP Functions

This describes all of the functions provided by AutoLISP. It consists of a synopsis and a catalogue of the functions. In the synopsis, function names are grouped by topic, and each is followed by a brief description. In the catalogue, function names appear in alphabetical order, and the functions are described in detail.

Synopsis of Functions

(defun *sym argument-list expr ...*)
Defines an external function (Subr).

Error Handling

(*error* *string*)
Prints an error message.

(alert *string*)
Displays a dialogue box alerting the user with *string*.

AutoCAD Queries and Commands

(command [*arguments*] ...)
Executes one or more AutoCAD commands.

(getvar *varname*)
Gets the current value of an AutoCAD system variable.

(setvar *varname value*)
Sets the value of an AutoCAD system variable.

(findfile *filename*)
Searches for a filename.

(getfiled *title filename ext flags*)
Prompts the user for a filename via the standard AutoCAD file dialogue box.

(osnap *pt1 mode-string*)
Finds a point via object snap.

Geometric Utilities

(distance *pt1 pt2*)
Finds the distance between two points.

(angle *pt1 pt2*)
Finds the angle between two lines.

(polar *pt angle dist*)
Finds a point via polar co-ordinates.

(inters *pt1 pt2 pt3 pt4 [onseg]*)
Finds the intersection of two lines.

(textbox *elist*)
Returns the diagonal co-ordinates of a box that encloses a text entity.

User Input

(initget [*bits*] [*string*])
Determines valid user input for the next call to a **get xxx function**.

(getreal [*prompt*])
Prompts for user input of a real (floating-point) number.

(getstring [*cr*] [*prompt*])
Prompts for user input of a string.

(getpoint [*pt*] [*prompt*])
Prompts for user input of a point.

(getcorner *pt* [*prompt*])
Prompts for user input of the corner of a rectangle.

(getdist [*pt*] [*prompt*])
Prompts for user input of a distance.

(getangle [*pt*] [*prompt*])
Prompts for user input of an angle.

(getorient [*pt*] [*prompt*])
Similar to getangle, but takes into account the current value of the ANGBASE system variable.

(getkeyword [*prompt*])
Prompts for user input of a keyword.

(getint [*prompt*])
Prompts for user input of an integer.

Conversion

(rtos *number [mode [precision]]*)
Formats a real (floating-point) value as a string.

(dtof *string [mode]*)
Converts a string that displays a real value into a real (floating-point) value.

(angtos *angle [mode [precision]]*)
Formats an angle as a string.

(angtof *string [mode]*)
Converts a string that displays an angle into a real (floating-point) value.

(cvunit *value from to*)
Converts between real-world units.

Co-ordinate System Transformation

(trans *pt from to [disp]*)
Translates a point or displacement from one co-ordinate system to another.

Display Control

(prin1 [*expr [file-desc]*])
Prints a message on the text screen or to an open file.

(princ [*expr [file-desc]*])
Prints a message on the text screen or to an open file.

(print [*expr [file-desc]*])
Prints a message on the text screen or to an open file.

(prompt *msg*)
Displays a message on the prompt line.

(menucmd *string*)
Displays and activates menus.

(redraw [ename [mode]])
Redraws the current graphics screen.

(graphscr)
Displays the current graphics screen.

(textscr)
Displays the current text screen.

(textpage)
Same as textscr, but clears the text screen first.

Low-level Graphics

(grclear)
Clears the graphics screen.

(grdraw from to colour [highlight])
Draws a vector in the current viewport.

(grvecs vlist [trans])
Draws multiple vectors in the current viewport.

(grread [track] [allkeys [curtype]])
Reads from an input device.

(grtext [box text [highlight]])
Displays text in the menu, mode, or status area of the graphics screen.

Wild Card Matching

(wcmatch string pattern)
Matches a string to a wild card pattern.

Selection Sets

(ssget [mode] [pt1 [pt2]] [pt-list] [filter-list])
Gets a selection set.

(ssadd [ename [ss]])
Adds an entity to a selection set (or creates a new set).

(ssdel ename ss)
Deletes an entity from a selection set.

(sslenght ss)
Returns the number of entities in a selection set.

(ssname ss index)
Returns the name of an entity in a selection set.

(ssmemb ename ss)
Checks whether an entity is a member of a selection set.

Entity Handling

(entget ename [applist])
Gets the definition data of an entity.

(entmod elist)
Modifies the definition data of an entity.

(entmake [elist])
Makes a new entity and appends it to the drawing database.

(entdel ename)
Deletes (and undeletes) entities in the drawing.

(entnext [ename])
Finds the next entity in the drawing.

(entlast)
Finds the last entity in the drawing.

(handent handle)
Finds an entity by its handle.

(entsel [prompt])
Prompts user to select an entity by specifying a point.

(nentsel [prompt])
Like entsel, but returns additional data for nested entities.

(nentselp [prompt] [pt])
Similar to nentsel but returns a full 3D 4x4 matrix and enables the program to specify the pick point.

(entupd ename)
Updates the screen image of an entity.

Extended Entity Data

(regapp application)
Registers the application's extended entity data.

(xdsizel list)
Returns the amount of memory (in bytes) that a list of extended entity data will occupy.

(xdroom ename)
Returns the amount of memory (in bytes) that an entity has available for extended data.

Symbol Tables

(tblnext table-name [rewind])
Finds the next item in a symbol table.

(tblsearch table-name symbol [setnext])
Searches for a symbol in a symbol table.

General Functions Arithmetic

(+ number number ...)
Returns the sum of all numbers.

(- number [number])
Subtracts the second number from the first and returns the difference.

(* number [number ...])
Returns the product of all numbers.

(/ number [number ...])
Divides the first number by the second and returns the quotient.

(~ number)
Returns the bitwise NOT of *number*.

(1+ number)
Returns *number* incremented by 1.

(1- number)	Returns <i>number</i> decremented by 1.	(atom item)	Verifies that <i>item</i> is an atom.
(abs number)	Returns the absolute value of <i>number</i> .	(atoms-family format [symlist])	Returns a list of previously defined functions.
(atan num1 [num2])	Returns the arctangent of a number in radians.	(boundp atom)	Verifies that a value has been bound to an atom.
(cos angle)	Returns the cosine of an angle.	(not item)	Verifies that <i>item</i> is nil.
(exp number)	Returns a value raised to the <i>number</i> power (natural antilog).	(null item)	Verifies that <i>item</i> is bound to fill.
(expt base power)	Returns <i>base</i> raised to <i>power</i> .	(numberp item)	Verifies that <i>item</i> is a real or an integer.
(fix number)	Returns the conversion of a number into an integer.	(quote expr ...)	Returns an expression unevaluated.
(float number)	Returns the conversion of a number into a real value.	(set sym expr)	Sets the value of a quoted symbol to that of an expression.
(gcd num1 num2)	Returns the greatest common denominator of two numbers.	(setq sym1 expr1 [sym2 expr2] ...)	Sets the value of one or more symbols to that of an expression.
(log number)	Returns the natural log of a number as a real value.	(type item)	Returns the type of <i>item</i> .
(logand number number ...)	Returns the result of a logical bitwise AND of a list of numbers.	Text Strings	
(logior integer ...)	Returns the result of a logical bitwise inclusive OR of a list of numbers.	(read string)	Returns the first list or atom obtained from the string.
(lsh num1 numbits)	Returns the logical bitwise shift of a number by a given number of bits.	(read-char [file-desc])	Reads a single character from the keyboard or from an open file.
(max number number ...)	Returns the largest of the numbers given.	(read-line [file-desc])	Reads a string from the keyboard or from an open file.
(min number number ...)	Returns the smallest of the numbers given.	(strcase string [which])	Returns a copy of a string with all characters converted to upper or lowercase.
(minusp item)	Verifies that <i>item</i> is a real or integer and evaluates to a negative value.	(strcat string1 [string2])	Returns the concatenation of one or more strings.
pi	Evaluates to constant π .	(strlen [string])	Returns the length, in characters, of a string.
(rem num1 num2 ...)	Divides two numbers and returns the remainder.	(substr string start [length])	Returns a substring of a string.
(sin angle)	Returns the sine of an angle as a real value.	(write-char num [file-desc])	Writes one character, described by an ASCII code, to the screen or an open file.
(sqrt number)	Returns the square root of a number as a real value.	(write-line string [file-desc])	Writes a string to the screen or to an open file.
(zerop item)	Verifies that <i>item</i> is a real number or an integer that evaluates to zero.		

Symbol Handling

Conversion

(ascii string)	Returns the conversion of the first character of a string into its ASCII character code.
(atof string)	Returns the conversion of a string into a real value.
(atoi string)	Returns the conversion of a string into an integer.
(chr integer)	Returns the conversion of an integer representing an ASCII character code into a single character string.
(itoa int)	Returns the conversion of an integer into a string.

Equality/Conditional

(= atom atom . .)	The <i>equal to</i> relational function.
(/= atom atom . .)	The <i>not equal to</i> relational function.
(< atom atom ...)	The <i>less than</i> relational function.
(<= atom atom)	The <i>less than or equal to</i> relational function.
(> atom atom . .)	The <i>greater than</i> relational function.
(>= atom atom)	The <i>greater than or equal to</i> relational function.
(and expr ...)	Returns the logical AND of a list of expressions.
(Boole func int1 int2 ...)	A general bitwise Boolean function.
(cond (test1 result1.....),.....)	Primary conditional function in AutoLisp.
(eq expr1 expr2)	Determines whether two expressions are identical.
(equal expr1 expr2 [fuzz])	Determines whether two expressions evaluate to the same thing.
(if testexpr thenexpr [elseexpr])	Conditionally evaluates expressions.
(or expr ...)	Returns the logical OR of a list of expressions.
(repeat number expr ...)	Evaluates each expression a given number of times.
(while testexpr expr ...)	Repeats the enclosed expressions while the test expression remains true.

List Manipulation

(append expr)	Takes any number of lists and runs them together as one list.
(assoc item alist)	Searches an association list using <i>Item</i> as a key, and returns the associated entry.
(car list)	Returns the first element of a list.
(cdr list)	Returns a list containing all but the first element of the list.
(caar list), (cadr list), (caddr list), (cadar list), etc.	Concatenations up to four levels deep are supported.
(cons new-first-element list)	Returns a list with the new element added to the beginning.
(foreach name list expr ...)	Steps through a list and evaluates each expression for every element in the list.
(list expr ...)	Creates a list from any number of expressions.
(listp item)	Verifies that <i>Item</i> is a list.
(mapcar function list1 ... listn)	Returns a list as the result of executing a function with the elements of lists supplied.
(member expr list)	Searches a list for an occurrence of an expression and returns the remainder of the list starting with the first occurrence of the expression.
(nth n list)	Returns the <i>n</i> th element of a list.
(reverse list)	Returns a list with its elements reversed.
(subst newitem olditem list)	Returns a copy of a list with <i>newitem</i> in place of every <i>olditem</i> .

File Handling

(close file-desc)	Closes a file.
(load filename [on failure])	Loads a file of AutoLISP expressions.
(open filename mode)	Opens a file for access by the AutoLISP I/O functions.

Display

(terpri)

Prints a newline on the screen.

(vports)

Returns a list of viewport descriptors for the current viewport configuration.

Function Handling

(apply *function list*)

Executes a function with the arguments given.

(eval *expr*)

Returns the result of evaluating any AutoLISP expression.

(exit)

Forces the current application to quit.

(lambda *arguments expr ...*)

Defines an anonymous function.

(progn *expr ...*)

Evaluates each expression sequentially.

(trace *function ...*)

Sets the trace flag for the specified functions.

(quit)

Forces the current application to quit.

(untrace *function ...*)

Clears the trace flag for specified functions.

Memory Management

(alloc *number*)

Sets the segment size to a given number of nodes.

(expand *number*)

Allocates node space by requesting a specified number of segments.

(gc)

Forces a garbage collection.

(mem)

Displays the current state of AutoLISP's memory.

Miscellaneous

(getenv *variable-name*)

Returns the string value assigned to a system environment variable.

(ver)

Returns a string containing the current AutoLISP version.

Programmable Dialogue Box Functions

Detailed explanations of the following AutoLISP functions, which handle user-defined, customised dialogue boxes, are available in chapter 9 of the *AutoCAD Customisation Manual*

This section summarises the functions in the Programmable Dialogue Box (PDB) package, grouping them by functionality. These functions call an associated DCL (Dialogue Control Language) file to display the desired dialogue box. It shows the arguments to each function.

Opening and Closing DCL Files

(load_dialog filename)

Loads the specified DCL file.

(unload_dialog dcl_id)

Unloads the specified DCL file.

Opening and Closing Dialogue Boxes

(new_dialog dlgname dcl_Id [[action-expression] screen-pt])

Initialises a dialogue box and displays it.

(start_dialog)

Begins accepting user input from the dialogue box initialised by the `new_dialog` call.

(done_dialog [status])

Terminates the current dialogue box and stops displaying it. Must be called from within an action expression or call-back function. This function also returns the current (X,Y) position of the dialogue box.

(term_dialog)

Terminates *all* current dialogue boxes as if the user had cancelled them.

Initialising Action Expressions or Call-back Functions

(action_tile key action-expression)

Associates the specified tile with the action expression or call-back function.

Handling Tiles and Attributes

(mode_tile key mode)

Sets the *mode* of the specified tile.

(get_attr key attribute)

Gets the DCL *value* of the specified attribute.

(get_tile key)

Gets the run-time value of the specified tile.

(set_tile key value)

Sets the run-time value of the specified tile.

Setting Up List Boxes and Popup Lists

(start_list key [operation [index]])

Starts processing the specified list box or popup list.

(add_list item)

Adds the specified string to the current list.

(end_list)

Ends processing of the current list.

Creating Images

(dimx_tile key) (dimy_tile key)

Retrieves dimensions of the specified tile.

(start_image key)

Starts creating the specified image.

(vector_image x1 y1 x2 y2 colour)

Draws a vector in the currently active image.

(fill_image x1 y1 x2 y2 colour)

Draws a filled rectangle in the currently active image.

(slide_image x1 y1 x2 y2 slidename)

Draws an AutoCAD slide in the currently active image.

(end_image)

Ends creation of the currently active image.

Application-specific Data

(client_data_tile key clientdata)

Associates application managed data with the specified tile.