EXPLORER™ GLOSSARY

MANUAL REVISION HISTORY

Explorer™ Glossary (2243134-0001 *A)

Original Issue	June 1985
Revision A	June 1987

The system-defined windows shown in this manual are examples of the software as this manual goes into production. Later changes in the software may cause the windows on your system to be different from those in the manual.

Texas Instruments Incorporated
ATTN: Data Systems Group, M/S 2151
P.O. Box 2909
Austin, Texas 78769-2909

THE EXPLORER™ SYSTEM SOFTWARE MANUALS

░	▥	▓
Little/No Interest	Medium Interest	Required

	First Day of Explorer Use	Casual or New Developer	Experienced Developer	Applications Programmer	Systems Manager
Introduction to the Explorer System	▓	▓	░	░	▥
Zmacs Editor Tutorial	▓	▥	░	░	░
Master Index	▥	▓	▓	▓	▥
Lisp Reference	▥	▓	▓	▓	▥
Input/Output Reference	▥	▓	▓	▓	▥
Tools and Utilities	▥	▓	▥	▓	▓
Zmacs Editor Reference	░	▓	▥	▥	░
Window System Reference	░	▥	▓	▓	░
Programming Concepts	▥	▓	▥	▥	▥
Networking Reference	▥	▥	▥	▥	▓
Glossary	▥	▥	░	░	░
System Software Installation	▥	░	░	░	▓
Technical Summary	▥	░	░	░	▥
System Software Design Notes	░	░	▥	▥	░

THE EXPLORER™ SYSTEM SOFTWARE MANUALS

Mastering the Explorer Environment	Explorer Technical Summary	2243189-0001
	Introduction to the Explorer System	2243190-0001
	Explorer Zmacs Editor Tutorial	2243191-0001
	Explorer Glossary	2243134-0001
	Explorer Networking Reference	2243206-0001
	Explorer Master Index to Software Manuals	2243198-0001
	Explorer System Software Installation Guide	2243205-0001
	Explorer Diagnostics	2533554-0001
Programming With the Explorer	Explorer Programming Concepts	2549830-0001
	Explorer Lisp Reference	2243201-0001
	Explorer Input/Output Reference	2549281-0001
	Explorer Zmacs Editor Reference	2243192-0001
	Explorer Tools and Utilities	2549831-0001
	Explorer Window System Reference	2243200-0001
Explorer Options	Explorer Natural Language Menu System User's Guide	2243202-0001
	Explorer Relational Table Management System User's Guide	2243203-0001
	Explorer Grasper User's Guide	2243135-0001
	Explorer Prolog User's Guide	2537248-0001
	Programming in Prolog, by Clocksin and Mellish	2537157-0001
	Explorer Color Graphics User's Guide	2537157-0001
	Explorer TCP/IP User's Guide	2537150-0001
	Explorer LX™ User's Guide	2537225-0001
	Explorer LX System Installation	2537227-0001
	Explorer NFS™ User's Guide	2546890-0001
	Explorer DECnet™ User's Guide	2537223-0001
	Personal Consultant™ Plus Explorer	2537259-0001
System Software Internals	Explorer System Software Design Notes	2243208-0001
	Release Information, Explorer System Software	2549844-0001

THE EXPLORER™ SYSTEM HARDWARE MANUALS

System Level Publications	Explorer 7-Slot System Installation	2243140-0001
	Explorer System Field Maintenance	2243141-0001
	Explorer System Field Maintenance Documentation Kit	2243222-0001
	Explorer System Field Maintenance Supplement	2537183-0001
	Explorer System Field Maintenance Supplement Documentation Kit	2549278-0001
	Explorer NuBus™ System Architecture General Description	2537171-0001
System Enclosure Equipment Publications	Explorer 7-Slot System Enclosure General Description	2243143-0001
	Explorer Memory General Description (8-megabytes)	2533592-0001
	Explorer 32-Megabyte Memory General Description	2537185-0001
	Explorer Processor General Description	2243144-0001
	68020-Based Processor General Description	2537240-0001
	Explorer II Processor and Auxiliary Processor Options General Description	2537187-0001
	Explorer System Interface General Description	2243145-0001
	Explorer NuBus Peripheral Interface General Description (NUPI board)	2243146-0001
Display Terminal Publications	Explorer Display Unit General Description	2243151-0001
	CRT Data Display Service Manual, Panasonic code number FTD85055057C	2537139-0001
	Model 924 Video Display Terminal User's Guide	2544365-0001
140-Megabyte Disk/Tape Enclosure Publications	Explorer Mass Storage Enclosure General Description	2243148-0001
	Explorer Winchester Disk Formatter (ADAPTEC) Supplement to Explorer Mass Storage Enclosure General Description	2243149-0001
	Explorer Winchester Disk Drive (Maxtor) Supplement to Explorer Mass Storage Enclosure General Description	2243150-0001
	Explorer Cartridge Tape Drive (Cipher) Supplement to Explorer Mass Storage Enclosure General Description	2243166-0001
	Explorer Cable Interconnect Board (2236120-0001) Supplement to Explorer Mass Storage Enclosure General Description	2243177-0001
140-Megabyte Disk Drive Vendor Publications	XT-1000 Service Manual, 5 1/4-inch Fixed Disk Drive, Maxtor Corporation, part number 20005 (5 1/4-inch Winchester disk drive, 112 megabytes)	2249999-0001
	ACB-5500 Winchester Disk Controller User's Manual, Adaptec, Inc., (formatter for the 5 1/4-inch Winchester disk drive)	2249933-0001

1/4-Inch Tape Drive Vendor Publications	Series 540 Cartridge Tape Drive Product Description, Cipher Data Products, Inc., Bulletin Number 01-311-0284-1K (1/4-inch tape drive)	2249997-0001
	MT01 Tape Controller Technical Manual, Emulex Corporation, part number MT0151001 (formatter for the 1/4-inch tape drive)	2243182-0001
182-Megabyte Disk/Tape Enclosure MSU II Publications	Mass Storage Unit (MSU II) General Description	2537197-0001
182-Megabyte Disk Drive Vendor Publications	Control Data® WREN™ III Disk Drive OEM Manual, part number 77738216, Magnetic Peripherals, Inc., a Control Data Company	2546867-0001
515-Megabyte Mass Storage Subsystem Publications	SMD/515-Megabyte Mass Storage Subsystem General Description (includes SMD/SCSI controller and 515-megabyte disk drive enclosure)	2537244-0001
515-Megabyte Disk Drive Vendor Publications	515-Megabyte Disk Drive Documentation Master Kit (Volumes 1, 2, and 3), Control Data Corporation	2246129-0002
	Volume 1, General Description, Operation, Installation and Checkout, and Part Data	2246125-0004
	Volume 2, Theory, General Maintenance, Trouble Analysis, Electrical Checks, and Repair Information	2246125-0005
	Volume 3, Diagrams	2246125-0006
1/2-Inch Tape Drive Publications	MT3201 1/2-Inch Tape Drive General Description	2537246-0001
1/2-Inch Tape Drive Vendor Publications	Cipher CacheTape® Documentation Manual Kit (Volumes 1 and 2 With SCSI Addendum and, Logic Diagram), Cipher Data products	2246130-0001
	1/2-Inch Tape Drive Operation and Maintenance (Volume 1), Cipher Data Products	2246126-0001
	1/2-Inch Tape Drive Theory of Operation (Volume 2), Cipher Data Products	2246126-0002
	SCSI Addendum With Logic Diagram, Cipher Data Products	2246126-0003

GLOSSARY

a

alist	See association list.
abnormal shutdown	See crash.
ABORT key	See program control keys.
access function	A function that can access each element of a defined structure. This function is usually created automatically when the structure is defined.
access operation	Any operation that recovers the value of a referenced object.
accessor function	See access function.
activation environment	The calling environment; the environment currently in force when a call to a function occurs.
active element	An element of a vector that is in use. That is, if a vector has a fill pointer, all elements with an index less than the fill pointer are considered to be active. If there is no fill pointer, all elements of the vector are active.
active process	A process that is either currently running or waiting to run; a process that has at least one run reason and no arrest reasons.
active window	An existing window that can be exposed if its superior window has a screen array, or any window that has had its window object presented to the screen manager for scheduling to be displayed on the screen. You can select any active window or create a new one as desired.
adjust	For text, to justify or add spaces until the right margins of each line align. For arrays, to change the length of the array.
advise	To add code before, after, or around a function so that you modify its behavior without actually changing its source definition and without its caller knowing there has been a change. This procedure is useful for testing a change to a function without committing to that change or for customizing a function that is also used elsewhere. Advise should be used for debugging purposes only—not for production work.
after daemon	A method called after the primary method of a combined method. See daemon method.
alu function	See arithmetic logic unit function.
ancestor	A higher-level, or preceding, node in a hierarchical structure such as the window system. For example, a window's ancestors are the superior window and all of its superiors windows.

Printer Publications	Model 810 Printer Installation and Operation Manual	2311356-9701
	Omni 800™ Electronic Data Terminals Maintenance Manual for Model 810 Printers	0994386-9701
	Model 850 RO Printer User's Manual	2219890-0001
	Model 850 RO Printer Maintenance Manual	2219896-0001
	Model 850 XL Printer User's Manual	2243250-0001
	Model 850 XL Printer Quick Reference Guide	2243249-0001
	Model 855 Printer Operator's Manual	2225911-0001
	Model 855 Printer Technical Reference Manual	2232822-0001
	Model 855 Printer Maintenance Manual	2225914-0001
	Model 860 XL Printer User's Manual	2239401-0001
	Model 860 XL Printer Maintenance Manual	2239427-0001
	Model 860 Xl Printer Quick Reference Guide	2239402-0001
	Model 860/859 Printer Technical Reference Manual	2239407-0001
	Model 865 Printer Operator's Manual	2239405-0001
	Model 865 Printer Maintenance Manual	2239428-0001
	Model 880 Printer User's Manual	2222627-0001
	Model 880 Printer Maintenance Manual	2222628-0001
	OmniLaser™ 2015 Page Printer Operator's Manual	2539178-0001
	OmniLaser 2015 Page Printer Technical Reference	2539179-0001
	OmniLaser 2015 Page Printer Maintenance Manual	2539180-0001
	OmniLaser 2108 Page Printer Operator's Manual	2539348-0001
	OmniLaser 2108 Page Printer Technical Reference	2539349-0001
	OmniLaser 2108 Page Printer Maintenance Manual	2539350-0001
	OmniLaser 2115 Page Printer Operator's Manual	2539344-0001
	OmniLaser 2115 Page Printer Technical Reference	2539345-0001
	OmniLaser 2115 Page Printer Maintenance Manual	2539356-0001

Communications Publications	990 Family Communications Systems Field Reference	2276579-9701
	EI990 Ethernet® Interface Installation and Operation	2234392-9701
	Explorer NuBus Ethernet Controller General Description	2243161-0001
	Communications Carrier Board and Options General Description	2537242-0001

Omni 800 is a trademark of Texas Instruments Incorporated.

OmniLaser is a trademark of Texas Instruments Incorporated.

Ethernet is a registered trademark of Xerox Corporation.

ABOUT THIS MANUAL

This manual contains definitions for terms used to describe the Lisp language and the Explorer system. The manual assumes that you are familiar with general computer and data processing terms, and thus, such terms are omitted from this glossary.

This manual contains mostly software terms with some hardware terms. Other manuals in the Explorer document set cover system concepts in detail and provide specific information on using various features. This manual is intended as a quick reference for unfamiliar terms you might encounter while reading the other manuals. To locate documentation on functions, flavors, commands, and so on, refer to the *Explorer Master Index to Software Manuals.*

The definitions contained in this manual are also available through the Glossary utility. Refer to the *Explorer Tools and Utilities* manual for instructions on using this utility.

application	1. The act of binding argument values to a function's parameters and executing the function within the context of those values. Normally, application is performed automatically whenever a function-call form is evaluated (implicit application). However, application can be explicitly invoked by the special forms **apply** and **funcall**. 2. A program that allows the user to solve a particular problem or to perform a particular task.
apropos	A facility that searches the system for all symbols with print names that contain a specified string and prints information about the symbols found. This facility provides features for performing complex, extended searches; for example, you can search for a combination of two strings. Different utilities have different apropos functions. For example, you can search for Lisp symbols in the Lisp Listener, and you can search for Zmacs command names when you are in Zmacs.
apropos completion	See completion.
area	A logical division of virtual memory that comprises one or more physical divisions (regions). An area can contain related objects of any type. You can specify which area(s) your program is to use for storage, or you can use the default area. Specifying areas gives you more control over garbage collection and paging operations for your program.
argument	An object in a function-call form whose value is bound to a corresponding parameter in the function's lambda list. This is analogous to an actual parameter in other programming languages.
arithmetic logic unit function	A graphics-operation function that controls the way the bits of the graphic object being drawn are combined with those already present in the window. This is also called an alu function.
array	A Lisp object consisting of multiple components or elements, dimensionally arranged according to a Cartesian coordinate system. An array can have any nonnegative number of dimensions; the components are indexed by numerical subscripts consisting of a sequence of integers. A general array (**art-q**) can hold any Lisp objects as components. A specialized array can hold only certain types of Lisp objects.
array leader	A one-dimensional array containing data about, and attached to, another array of arbitrary dimensions. An array that has a leader acts like two arrays joined together, which can be treated as one object. This, in effect, produces a property list for arrays in that it allows you to attach arbitrary information to an array.
arrest	To stop or halt a process. An arrested process is not allowed to run or compete for resources until it is made active again.
art-q array	A general array. Each element of an **art-q** array can be any type of Lisp object. The term **art** stands for array type. In Common Lisp, an **art-q** array is called a general array but has no specific type-specifier name.
association list	A data structure consisting of a list of pairs (conses) where each pair represents an association between its objects. The first element of the pair (the car) is the key and the second element (the cdr) is associated data. This is also referred to as an alist.

association pair	A cons used to represent an association between its car and its cdr.
atom	A single, indivisible Lisp object; specifically, any Lisp object that is not a cons.
atomic form	An object that can be evaluated but is not in a list form, such as a symbol, string, or number.
attribute list	See file attribute list.
autoexposure	A feature that automatically exposes a window when it falls entirely within unclaimed areas of the screen and can be made visible without deexposing other windows.
autoselect	A feature that allows the screen manager to automatically select a window if no window is selected.
aux variable	See auxiliary keyword variable.
auxiliary keyword variable	A variable (often called an aux variable) that appears in a function's lambda list. Aux variables are indicated by the lambda-list keyword &aux. However, aux variables are not parameters; they are not bound to any argument value during function application. Their purpose is to establish local variables within the function, and they are bound to values specified in the definition of the function. Aux variables behave identically to variables bound with a **let*** form.

b

backplane	A printed wiring board in the system enclosure that interconnects the system logic boards, the power supply, and the peripheral cable adapter (PCA) boards and cables. The Explorer backplane contains eight slots, each with three rows of female connectors to mate with the corresponding logic board connectors on one side and the PCA boards on the other side.
backquote facility	A facility that allows you to construct an expansion form without having to use a long sequence of list-manipulating functions. The backquote character (‘) allows the expansion function to more closely resemble a template for the expanded code and makes it easier to use.
backup system	A window-based software facility for backing up and restoring files on cartridge tapes.
band	A logical grouping of one or more physical partitions. See also partition.
bar	See foo.
base flavor	A flavor that is used to build a family of other related flavors.
baseline	A nonnegative fixnum that is the number of raster lines between the top of each character and the lowest point—excluding the descender—of the character.
baz	See foo.

beep	The output of a function that attracts the user's attention. Examples are an audible beep or a flashing screen.
before daemon	A method called before the primary method of a combined method. See daemon method.
bignum	A big number; any integer less than -2^{25} or greater than $(2^{25})-1$. A bignum requires more than one word of memory.
bind	To temporarily associate a value with a symbol. When the association is removed, the symbol assumes its previous value.
binding	The value currently assigned to a symbol.
bit	The smallest unit of information that can be represented in computer hardware.
bit array	An array whose elements consist of bits only. Bit arrays are primarily used for graphics operations.
bit map	A two-dimensional memory array on the system interface board that stores the values for the pixels on the physical display. Each element of the array is one bit, which corresponds to one pixel. Images are constructed on the display by setting the appropriate bits in memory.
bit vector	A vector whose elements consist of bits only.
bit-save array	An array used to hold the contents (pixel values) of a window when the window is not fully visible on the screen. When the window becomes fully visible again, the contents are copied from the bit-save array back into the bit map.
bitblt	A bit block transfer operation. The **bitblt** function copies a rectangular portion of an array to another array.
bits attribute	Extra flags associated with a character. A character with attribute bits is produced by a key used in combination with the SUPER, HYPER, META, or CTRL keys.
bitwise logical operation	A Boolean logical operation performed on matching subscripted elements of two or more bit arrays of matching rank and dimensions.
black plane	A buffer in the font editor that normally contains the character you are editing (compare with gray plane). Pixels that are set only in the black plane are displayed in black; pixels that are set in both the black plane and the gray plane are shown in medium gray.
blinker	A visible indicator used to mark a location in a window. Blinkers appear in various shapes and sizes. A window can use any number of blinkers. Some blinkers actually blink, but others are visible continuously. When blinkers are turned off, they become invisible. The keyboard cursor and the mouse cursor are examples of blinkers.
blip	A character in the input buffer that is actually a list. It is often used for program-generated input, especially mouse clicks. By convention, the car of the list indicates the type of the blip. For simple mouse click blips, the blip indicates not only which mouse button was clicked, but also where on the screen the mouse cursor was located.

blt	An abbreviation for block transfer.
BOA	See positional constructor functions.
BOLD LOCK key	See mode keys.
boot	The reinitialization of the system. See also cold boot and warm boot.
boot partition	A disk band that contains the code for the menu boot procedure.
border	A rectangular box drawn around the outside of a window. Borders are margin items and consist of straight lines of variable thickness unless you define them differently. See also border margin width.
border margin width	An area of whitespace that separates the border from the inside of the window or the next innermost margin item to keep them from merging with the border. See also border.
bound variable	A variable that has some value assigned to it. In particular, the parameters of a lambda list of a function definition are bound to argument values when the function definition is invoked.
box	To point to a mouse-sensitive item with the mouse cursor. When you do this, a box appears around the item and information about the item usually appears in the mouse documentation line. When an item is boxed, you can perform an operation on the item by clicking one of the mouse buttons.
break	To halt execution of a program and enter a Lisp Listener. You can set a breakpoint in your program or enter a breakpoint from the keyboard (by pressing the BREAK key) anytime during execution. You can also specify a break that invokes the debugger when a specified function is called or called under certain conditions.
BREAK key	See program control keys.
brick	The common name for the mass storage enclosure.
buffer	A temporary storage facility within the active system environment. For example, when you edit a file in Zmacs, the file is copied into a buffer, and you edit the copy in the buffer.
bump	To push the mouse cursor against the scroll bar margin of a window. This causes the scroll bar to be displayed.
bury	To deexpose a window and place it at the bottom of its priority group in the window ordering. Any other windows that it overlapped now overlap it.
bus	A group of signal lines used to transfer data from one or more sources to one or more destinations.
byte specifier	A pointer used to designate a specific byte position within any integer. (Here *byte* refers to an arbitrary bit string, not eight bits.) The representation of a byte specifier is a fixnum. The two lowest octal digits of the fixnum represent the number of bits of the byte, and the higher octal digits represent the position of the byte within the integer (0 means the least significant bit).

C

canonical	The usual, standard, or accepted way of doing something. In the file system, canonical file types, such as **:lisp**, are used because this extension could exist as lisp or lsp, depending on which host you are using. In this case, the canonical type provides a host-independent standard.
CAPS LOCK key	See mode keys.
car	The function that extracts the first component of a cons, and by common usage, the name of that component itself.
catch	A special form used with **throw** to perform nonlocal exits. The **catch** construct serves as a target for the transfer of control by a **throw**.
cdr	The function that extracts the second component of a cons, and by common usage, the name of that second component.
cdr-coding	A memory compaction scheme implemented in hardware and used during the creation of lists and conses. This scheme removes the need for two memory locations for each item of a list. Conventionally, the second location is a pointer to the next item in the list; with the cdr-coding scheme, the items of a list are in consecutive memory locations.
Chaosnet	A communications protocol that provides an Ethernet protocol between two user processes on different machines. The protocol is a full-duplex, packet-transmission channel. The term *Chaos* in the name refers to the fact that there is no central control.
Chaosnet server	A program that handles standard service requests received over the network using the Chaosnet protocol.
character	A unit of information in a computer representing one of the following:

- Alphabetic symbols, which include:
  - Uppercase and lowercase symbols of the alphabet
  - The numeric symbols 0 through 9
  - Punctuation symbols
  - Special graphic symbols, such as the tilde (~) and the dollar sign ($)
- Numerical digits (Note that a numeric digit is different from an alphabetic numerical digit.)
- Special nongraphic characters, such as the RETURN character or the space character

On the Explorer system, characters are represented in two ways. In Zetalisp mode, characters are represented as integers. In Common Lisp mode, characters are represented as character data objects.

character box	A box in the font editor that indicates the width and, along with the line below it, the height of a character.
character keys	The set of keys on the keyboard that includes the standard typewriter keys as well as the braces, brackets, and extra set of parentheses.
character spacing	The pitch of a printed line; the number of characters per inch (cpi). This term usually describes output from a printer.
choice box	A box in a multiple-choice menu or window margin that indicates a yes or no choice for an item. You can indicate yes or no by clicking the mouse within the choice box.
choice facilities	A set of facilities in the window system that allows you to create different types of menus for your applications.
choose-variable-values menu	A menu that displays a set of variables and their values, allowing you to change the values. In some cases, several different values for a variable may be shown, and you can select one with the mouse. The current value is usually shown in boldface type. In other cases, you can select the value with the mouse and enter a new value from the keyboard.
chording	Key combinations in which you must press and hold two or more keys simultaneously. Chorded sequences are denoted by a hyphen between the individual key names, for example, META-X.
circular list	A list in which the pointer of the last element of the list points to the first element of the list. An example is a list in which the cdr of the last cons is the first cons of the list.
CLEAR INPUT key	See user interaction keys.
CLEAR SCREEN key	See user interaction keys.
click	The pressing and releasing of one of the mouse buttons. Single clicks are denoted by L, M, and R (corresponding to the left, middle, and right buttons). Double clicks require you to quickly press and release the same mouse button twice. (Alternately, you can press CTRL and a single click.) These are denoted by L2, M2, and R2. Rubber banding operations require you to press a button and hold it while moving the mouse. These are denoted by LHOLD, MHOLD, and RHOLD.
closure	A functional object that is functionally similar to a flavor instance in that it retains state and maintains private values for certain variables whose initial values are determined when the object is created. There are two types of closures available in the Explorer environment: lexical and dynamic. Lexical closures (a standard Common Lisp construct) are created by a `#'(lambda ...)` form. A lexical closure maintains the complete existing lexical environment at the time the closure is created. A dynamic closure (an Explorer extension) is created by the **closure** function. Dynamic closures are closures over a user-specified list of (dynamic) special variables with respect to a function. Closures of either type can be **funcalled**, causing the function to be executed in an environment in which the values of the *closed-over* variables are obtained and updated.
code attribute	The ASCII character code associated with a character, as distinguished from its bits attributes or font attribute.

coercion	A concept that deals with converting objects to an equivalent object of another type. This usually refers to the conversion of numbers (for example, a real number becomes a complex number), but it can also refer to converting other objects (for example, a single character can become a string).
cold boot	The reloading of the load band and the microload band that were running when you pressed the keystroke sequence META-CTRL-META-CTRL-RUBOUT. Thus, you have a newly initialized Lisp system with no problems, but you have lost the contents of virtual memory. See also system reset, menu boot, and warm boot.
combined method	A single method that is automatically constructed for each operation when flavors are combined. The combined method is constructed out of all methods for that operation from all the components of the new flavor. The combined method consists of one primary method and any number of daemon methods. By default, the combined method is the primary method of the first specified component flavor that provides a primary method for the operation. The primary methods from other component flavors are ignored. If daemon methods are defined, the combined method calls all the daemons from all component flavors that provide them. The returned values from the combined method are the values returned by the primary method only; any values returned from the daemons are ignored. You can select other ways to combine methods when you define the combined flavors.
combining type specifier	A type specifier that defines a data type in terms of combinations of other data types.
command editor	A feature in the Universal Command Loop (UCL) that allows you to create new commands and command macros for a utility, to tailor existing commands, and to save them to disk for use in another session.
command history	A buffer that stores the commands you invoke within a particular window. Only certain windows implement command histories, and only significant commands are stored in the history. See also history.
command menu	A menu that allows you to invoke commands or operations by selecting items from the menu.
comment	A string within code that is ignored when the code is compiled or executed. For Lisp, the semicolon (;) is used to denote the beginning of a comment. You can also create a multiline comment, which begins with #| and ends with |#.
Common Lisp	The dialect of the Lisp language that is supported by the Explorer system. Common Lisp was defined by a joint group of industry and university representatives to establish a standard for portability and consistency among different machine implementations of the language. It was derived from Maclisp and influenced strongly by Zetalisp. In addition to Common Lisp, the Explorer supports higher-level extensions to the language.
Common Lisp mode	An editing mode that is context sensitive to Common Lisp. See Zmacs major mode.

compile	To translate a user-written program into machine language. Such code normally executes faster than code that has not been compiled.
compiler	See Lisp compiler.
completion	A feature that attempts to complete the name of a command, Lisp object, buffer, or file after you have typed a few characters. If you do not type enough characters to identify a unique string, you can request a menu or list of possible completions, from which you can select the one you want. Three types of completion are available: ■ Apropos — Searches for any substring that matches the characters you have entered. ■ Recognition — Searches for strings where the first letter (or first few letters) of each word matches the characters you have entered. ■ Spelling — Searches for a string that resembles the order of the characters you have entered.
complex numbers	A numbering system in which numbers are represented as ordered pairs (a,b), where a and b are elements of the real number set. Special operations can be performed on complex numbers that enable the square root of −1 to be defined.
component flavor	An existing flavor that is used as a constituent part in forming another flavor.
component option	A user-defined option that declares a component of a structure defined by **defstruct** to be of a particular data type or to be unsettable.
condition	Some noteworthy event that can occur during the execution of a program, such as an error.
condition handler	A function that is associated with certain condition names and called to handle those conditions when they are signaled.
condition instance	An instance of a particular type of condition.
condition name	1. A symbol used to convey a category of conditions. 2. Any flavor with **condition** as one of its base flavors.
condition signaling	A facility based on flavors that allows you to define signal-specific conditions in your programs.
conditional	A construct that allows a program to make a decision and perform one action or another based on some logical condition. Common Lisp supports three types of conditionals: one-way (**and** and **or**), two-way (**if**), and multiway (**cond** and **case**).
configuration ROM	Special read-only memory (ROM) located on each NuBus board, which allows the system to automatically configure itself according to the boards installed. Each configuration ROM contains information about the basic characteristics of that board, such as board type, required parameters, and an address pointer to diagnostic routines.

cons	A compound data object consisting of two components: the car and the cdr. Conceptually, the car represents the first component of the cons, and the cdr represents the next component of the cons. In actuality, a cons is a contiguous pair of pointers, and each can point to any data object. The first pointer is the car, and the next pointer is the cdr. Conses are used primarily to represent lists. In this case, the car of the cons points to an element in the list, and the cdr points to the next cons in the list. In a true list, the cdr of the last cons in the list points to the symbol **nil**. In a dotted list, the cdr of the last cons points to a non-**nil** atom.
console	The collection of hardware components through which you interact with the system. It consists of the monitor, the keyboard, and the mouse. This is also called the system console, and sometimes, the display unit.
constraint frame	A description of the conditions defining the layout or the configuration of panes within a frame. The description is stated symbolically rather than in explicit units of pixels; thus, when the window changes shape, the symbolic description can be recomputed, and the panes are reshaped appropriately.
constructor function	A function that creates an instance of a **defstruct**-defined structure. A constructor function is usually created automatically when the structure is defined.
contagion	A concept that deals with the automatic type conversion rules applied during numerical operations with arguments of different types. See coercion.
control-bit weights	The integer values given to the following named constants: **char-control-bit**, **char-meta-bit**, **char-super-bit**, and **char-hyper-bit**. These weights are used to manipulate the bits attribute of a character.
control bits	See bits attribute.
control stack	A stack that contains such information as the current function being run, its caller, the caller's caller, and the point of execution or the return address of each function.
control structure	A Lisp form or structure that allows you to control the flow of program execution within a function.
convenience receptacles	Power outlets in the back of some system enclosures that accept up to two power cords from the monitor or mass storage enclosures. These outlets are controlled by the on/off button on the front of the system enclosure and allow you to conveniently control power to the attached enclosures with one button.
Converse	An interactive window that allows you to send messages to other users on the network and receive messages from them. The messages are organized into groups, called conversations, with one conversation for each user with whom you are communicating. Editing commands are available for entering messages. The Converse window shows all the messages sent and received since the last cold boot operation.
copier function	A function that creates a new instance of a **defstruct**-defined structure that is a component copy of an already existing structure instance. A copier function is usually created automatically when the structure is defined.

coroutine	A term used in concurrent or parallel processing to describe a routine that is being executed concurrently with another routine.
Counters	A command, contained in the Peek facility, that lists the statistics of all the microcode meters and associated counters.
crash	System shutdown due to microcode detecting an irrevocable error. The **sys:report-last-shutdown** and **sys:report-all-shutdowns** functions can be used to display the crash records.
crash record	Information about a system crash, such as reason, time of crash, function running, and some microcode registers. See crash.
CTRL key	See modifier keys.
current font	The font currently being used to print output characters in a window. The current font of a window is always one of the fonts in the window's font map.
current package	The package that is currently in effect. This package is always identified by the value of the ***package*** variable.
current process	The process that is currently executing.
current stack group	The stack group associated with the current process. Also called the running stack group.
cursor pad	A set of keys labeled with arrows that, depending on the software, either move the cursor in the direction of the arrow or produce arrow characters.

d

daemon method	A method that is called by a combined method either before or after the primary method is called. Daemon methods are frequently provided by component flavors so that they know when a message is sent or that they can perform a part of the operation defined by the primary method. The daemon methods are passed the same arguments that the combined method was given, but values returned from the daemons are ignored.
data brick	See mass storage enclosure.
data type	A set of Lisp objects. Many Lisp objects belong to more than one set. Thus, an object is said to *belong* to a given data type rather than to be a particular type. Common Lisp provides a large set of predefined data types, including symbols, functions, lists, arrays, numbers, characters, and structures. You can also choose your own set of objects to define a new data type.
data type checking	Checking done automatically during run time to select the appropriate instruction for that type of data.
data type specifier	The name of a particular data type; often called a type specifier. A type specifier can be either a symbol or a list. Symbols name predefined classes of objects; lists usually indicate combinations or specializations of simpler data types.

deactivate	To remove a window from the list of active windows of its superior window. Deactivated windows are not managed by the window system; they can be reclaimed by the garbage collector. A deactivated window can be reactivated by the user.
debugger	A facility that helps you find and handle errors in your programs. The debugger lets you examine the environment in which an error (or some other condition) is signaled, take corrective action, and resume execution or abort the program.
declaration	A statement in certain special forms that allows you to specify additional information about objects in your program to aid in compilation and compiler optimization.
deexposed typeout action	A specified action to be taken when output is attempted to a window that is not exposed. For example, the specified action can be to signal an error or to expose the window.
deexposed window	A window whose superior window has no screen array (that is, there is no place to accept typeout for this window), or a window that is not ready to be exposed. Except for special circumstances, a process cannot send output to a deexposed window.
defaulting	See pathname defaulting.
definition environment	The environment in force when variables are defined.
defstruct macro	A macro that enables you to define record structures.
defstruct option	An option of the **defstruct** macro that allows you to define a specialized functionality for a particular record structure.
delete	1. In the Explorer file system, to mark a file or empty directory for removal. An empty directory contains no subdirectories or files. (A directory containing subdirectories or files cannot be deleted.) You must use an expunge operation to actually remove the file or empty directory. 2. When editing text, except for single characters, deleting does not actually remove the item from the system, and a delete operation can be undone.
demand paging	The act of moving a page that contains a referenced object not currently in primary storage (main memory) from secondary storage (disk) to main memory.
descendant	A lower-level node in a hierarchical structure such as the window system. For example, a window's inferior and all its inferior's inferiors are the window's descendants.
descender	The part of a character that falls below the baseline of the character.
deselected window	A window that is no longer enabled to accept input from the keyboard or mouse. See selected window.
destroyed argument	A sequence-type argument whose structure is altered in any manner during the processing of a function. If a copy of the argument is made and then altered by the function, the argument is not considered destroyed.

destructive counterpart function	A sequence-manipulating function that alters the structure of its arguments and has counterpart functions that perform the same operation but do not alter the structure of their arguments. For example, **nconc** is the destructive counterpart function to **append**.
diagnostic ROM	Special read-only memory (ROM) located on certain system boards that contains diagnostic routines for that board. The processor can read and execute the routines to test the board.
diagnostics partition	A disk partition that contains diagnostics microcode and test patterns used for testing the system.
DIN	The German standards organization comparable to ANSI in the United States.
directory	A group of related files and subdirectories.
directory editor	A facility within Zmacs that allows you to perform all your directory housekeeping. You can delete, copy, rename, edit, view, and print files. This is also referred to as Dired.
Dired	See directory editor.
disassemble	The converting of machine language code (that is, a compiled function) into a textual representation—the opposite of what an assembler does on most computers.
disk label	A description of the partitions of a disk. The disk label includes a name that uniquely identifies the system on a network.
displaced array	An array whose contents are located in another array. Therefore, the two arrays share array elements. If array X is displaced to array Y, when you reference elements of array X, you are actually referencing array Y's elements.
display unit	See console.
documentation string	A string within a function definition and several other forms that appears as the first element of the body. Unlike a comment, the documentation string is retained with the object code during compilation so that it can be accessed by various functions for online documentation.
dotted list	A list that is terminated by a non-**nil** atom rather than the symbol **nil**. Dotted lists are denoted by preceding the cdr of the last cons with a dot (surrounded by blank spaces). Example: `(a b c . d)`
dotted pair	A single cons whose cdr points to a non-**nil** atom. Dotted pairs are denoted by a dot (surrounded by blank spaces) between the car and the cdr. Example: `(a . b)`
double-colon package qualifier	A package qualifier (consisting of two colons) that allows you to externally access the internal symbols of another package. This accessing should be done with care because functions named by internal symbols of a package are not designed for public use and therefore may not have the exhaustive error-checking code that functions named by external symbols normally have. Thus, using these internal symbols can result in programming errors that are extremely difficult to find.

drag	To move a window or graphics object on the display by pressing and holding a mouse button while moving the mouse. As you move the mouse cursor, the object follows it until you release the button. The object then remains at the last position of the mouse cursor.
dribble file	A file used to log terminal interaction. When the dribble file is activated, all input and output is directed to the file as well as to the terminal.
dynamic binding	A type of binding in which a variable can be referenced anywhere outside its establishing function.
dynamic extent	A type of extent in which references can occur any time between the establishment of an object and the disestablishment of that object. An object is *established* and *disestablished* with the initiation and termination, respectively, of language constructs such as **with-open-file** and the binding of special variables.
dynamic item list menu	A menu that dynamically recomputes its menu item list at various times.
dynamic scope	Bindings having dynamic extent and indefinite scope. An example is the binding of special variables.

e

element	A component of a list or an array.
empty list	A list that has no elements. The empty list is denoted by the symbol **nil** or ().
encapsulation	A special type of form that surrounds another form and enhances the other form's operation without changing its basic functionality. A trace, for example, is an encapsulation.
END key	See user interaction keys.
end-test	A conditional test that stops iteration.
ENTER key	See number pad.
environment	A collection of variable bindings resulting from the invocation of a function or a program.
environment stack	The stack that contains all of the values saved by lambda binding. Also called the special push-down list (PDL).
EOF	An abbreviation for end of file.
eq	A condition of equality in which two or more objects are identical. This usually implies that they point to the same memory location.
eql	A condition of equality in which two or more objects are one of the following: ■ eq ■ Numbers of the same type and value ■ Character objects representing the same character

equal	A condition of equality in which two or more objects are similar in structure and value. Two numbers are equal if they have the same value and type. For conses, **equal** is defined as the two cars being equal and the two cdrs being equal. Two strings are **equal** if they have the same length and the characters composing them are the same. All other objects are **equal** if and only if they are **eq**.
equalp	The most general test of equality. Character case, font, and bits attributes are not considered when comparing characters using **equalp**. Numeric formats are not considered when comparing numbers.
error handler	A condition handler that deals with errors.
ESCAPE key	See user interaction keys.
Ethernet	The DEC™-INTEL™-XEROX™ (DIX) standard network communications system, version 1.0, 30 September, 1983.
Ethernet controller	An optional logic board installed in the system enclosure that provides an interface between the Explorer system and a local area network (LAN).
Ethernet network	A local area network (LAN) that employs Manchester-encoded data and uses coaxial cables to transmit data at speeds up to 10 megabits per second. The Ethernet uses the carrier sense multiple access/collision detect (CSMA/CD) protocol. An Ethernet segment can extend up to 1500 meters (4920 feet) and accommodate up to 100 transceivers.
Ethernet transceiver	The electronic device that attaches a host to the network coaxial cable. The transceiver provides the electronics to transmit and receive Manchester-encoded data on the Ethernet. It also provides the electrical isolation between the host and the network.
Eurocard	Any printed circuit board constructed according to the Eurocard format. Explorer logic boards are extended three-high Eurocards.
Eurocard format	A mechanical-design format for printed circuit boards developed as a European standard for high-performance board level products. This format defines a standard board outline and connector style (DIN 41612). The Eurocard format defines a family of board sizes, based on rules for combining multiples of the basic board outline and connector.
evaluation	The means by which the Explorer system extracts a value from a Lisp form. Execution of Lisp code is called evaluation because executing a piece of code normally results in a data object, called the returned value. The evaluation of a symbol returns the contents of the symbol's value cell. The evaluation of a function call also involves application, because the function must be applied to its arguments in order to extract a value from it. Normally, one level of evaluation is performed automatically with each entered Lisp object (implicit evaluation). You can inhibit this level of evaluation with the **quote** special form. For example, list forms are generally interpreted and evaluated as function calls, macro calls, or special forms. If you inhibit evaluation of the list form, it is treated strictly as data.

DEC is a trademark of Digital Equipment Corporation.
INTEL is a register trademark of Intel Corporation.
XEROX is a register trademark of Xerox Corporation.

	You can also explicitly invoke another level of evaluation with the **eval** or **eval-when** special form. This evaluation is useful when the value of a symbol is a function call and you want to invoke that function.
	The => notation is used in examples to indicate evaluation and the returned values.
evaluator	The part of the Lisp interpreter that evaluates Lisp forms.
event	A situation, circumstance, or state that the Lisp software or system microcode detects, records, and normally acts upon. For example, the event could be an end-of-file mark, an error condition, or a context change.
exit form	A form whose value is returned during the exiting procedure of an iterative form.
expanded function	A function resulting from the expansion of a macro.
explicit application	See application.
explicit evaluation	See evaluation.
explicitly specified structure	A structure whose representation type has been explicitly specified in the **defstruct** macro that defines the structure. For example, you can specify that when an instance of the structure is created, it will be created as a list or a specific type of vector.
exporting a symbol	Enabling a symbol internal to a particular package to become an external symbol.
exposed window	A window that is enabled to accept output data from a process. An exposed window need not be visible. In order to be exposed, the superior window of the window must have a screen array.
exposure priority	A value associated with a window to determine which window is displayed when more than one window is ready to be exposed simultaneously.
expunge	To actually remove an item, such as a file or empty directory, that has been marked for deletion from the system. An expunge operation cannot be undone.
extended command	A command in Zmacs invoked by pressing META-X and then entering the command name in the minibuffer.
extended search	A search facility that allows you to specify a pattern rather than just a character string for the search. For example, you can search for two words at the same time.
extent	The interval of time during which references to an object can occur.
external package mapping	An operation that occurs when code in the current package accesses a symbol that does not belong to the current package.
external symbol	A symbol in one package that can be accessed by functions of other packages. Such symbols are used as external interfaces to other packages.
external value cell	The memory location that contains the value of a symbol that is closed over by a closure.

f

F1, F2, F3, F4 keys	See function keys.
fiber-optic link	A cable that transmits data by means of light reflected through thin, flexible glass rods.
file	A sequence of character or binary data, stored as a group on disk or tape. A file of Lisp programs can be edited, compiled, and loaded. When you edit a file in Zmacs, the file is copied into a buffer and you edit the copy in the buffer.
file attribute list	A list (starting with -*-) in which you specify the attributes of a file. This list is located at the beginning of the file. Useful attributes might include base, package, editing mode, and font.
file partition	A disk partition (usually named FILE) that contains either all or a portion of a file system.
file server	A program on a host with a file system that handles standard requests for file operations from other hosts on the network.
file system	A set of directories, subdirectories, and files. Each file system is contained in one or more file partitions.
file type	An extension to a filename that indicates the type of information in the file. For example, the file type LISP denotes a Lisp source code file, the file type XLD denotes an Explorer file containing compiled code, and the file type TEXT denotes a general text file.
file version	See version number.
filename	See pathname.
fill	To put as much text as possible on the current line without exceeding the right margin. The fill column defines the maximum width of a line by setting the right margin. You can also define a group of characters, usually blanks, to start a line. You can use the prefix to start every line. Basically, the fill prefix is the left margin.
fill pointer	A number that indicates how many elements of an array are currently being used. See active element.
first	A term sometimes used to refer to the car of a list.

fixnum	A fixed-number; any integer in the range of -2^{25} through $(2^{25})-1$. Integers outside of this range are bignums.
flavor	An abstract data structure that describes a whole class of similar objects in object-oriented programming. A specific object is an instance of a particular flavor. The operations that can be performed on an object are defined by the methods attached to the flavor. The Explorer system includes a large number of predefined flavors, which you can instantiate and use in your programs. You can also define your own flavors or combine existing flavors to form new ones. See also base flavor and mixin flavor.
Flavor Inspector	A window-oriented program for examining flavors.
floating-point precision	The number of digits used to represent a floating-point number at a specified precision (short, single, double, or long).
flonum	A floating-point number; any real number.
font	A set of alphanumeric characters, usually all in one size and style of typeface. A font can include special characters, such as icons, as well as the regular character set. Each font is represented as a Lisp object consisting of an array indexed by character code. The Explorer system has a number of predefined fonts that you can use to display or print text. You can also define your own fonts or modify existing fonts with the font editor.
font attribute	A component of a character object that is intended to permit a different font style for each character, such as italic or boldface.
font editor	A facility that allows you to define new fonts or modify existing fonts by using the mouse to draw magnified characters on a grid of dots.
font map	An array that specifies the fonts available for use in a window. Each window has a font map.
foo	A meaningless expression used to represent an arbitrary symbol, variable, or function. When more than one expression is needed, the terms *bar* and *baz* are also used.
form	A Lisp object that can be evaluated. Forms divide into two broad categories: list forms and nonlist forms.
format partition	A disk partition that contains parameters for format and surface analysis of the disk.
frame	1. A collection of panes or inferior windows that can be manipulated as a single window. 2. A single processing environment or block of a run-time stack.
free list	A list of memory that has not been used and is accessible to the user. Also called free storage.
free storage	See free list.

free variable	A variable that is used within a function but does not appear in the function's parameter list or in any other local binding. This is commonly called a nonlocal variable.
frobboz	A window naming convention used to stand for any feature, attribute, or class of windows that can appear in a flavor name.
full-rubout	See over-rubout.
function	A Lisp object that can be invoked as a procedure. A function may take arguments, and it returns zero or more values. The Explorer system has thousands of predefined functions you can invoke (see function call). You can also define your own functions (see function definition).
function call	A list form consisting of a symbol that names a function followed by arguments to that function. This is also called a function call form or a function form. The form invokes the specified function by applying the function definition (obtained from the symbol's function definition cell) to the arguments. Any list form that has no other interpretation, such as a macro call or a special form, is considered a function call and is evaluated as a function unless it is quoted.
function definition	A procedure that creates a new function and defines its actions. Typically, you define a new function in one of two ways: ■ The **defun** special form, which defines a named function, assigns a specified symbol as the function name, and puts the function definition in the symbol's function definition cell. Named functions can be referenced and invoked at any time using a function call form. ■ A lambda expression defines an unnamed function within a function call form. Since the function is unnamed, it can only be used within the form in which it is defined. Use of lambda expressions reduces the number of symbols needed.
function definition cell	A component of a symbol that points to a function definition when the symbol is the name of a function.
function keys	A set of programmable keys (F1, F2, F3, and F4) that can be used in application software. They are not used by the system software. These keys are equivalent to the Roman I, II, III, and IV keys on other machines.
function spec	See function specification.
function specification	A Lisp object that describes the location of a function and, therefore, serves as the function name. In much of the Explorer documentation, this term is called a *function spec*. The most common function specs are symbols. Function specs can also be lists; for example, `(:method w:window :expose)` is a list function spec.
functional position	The first position in a list. This position is used as a function call.

g

garbage collector	A program that reclaims memory cells used to represent objects that are no longer accessible. Once the memory is recovered, it can be used again to represent new objects. You can turn the garbage collector on or off and control its operation in one of two modes: incremental garbage collection or stop-and-collect garbage collection.
general array	An array whose elements can be any type of data object.
general scroll window	An independent facility from text scroll windows. Although these windows are called scroll windows, the windows do not actually scroll. The contents of the windows are updated as items in the windows change. Peek is a good example of a general scroll window.
generalized variable	Any storage location that can be referenced by an operation, no matter how that location is named. For example, the form `(car x)` refers to a storage location and, therefore, can be considered a generalized variable.
generator	The usage of a closure that is called successively to obtain a unique value on each call.
generic function	A function that accepts any type of arguments and performs the appropriate action on them.
gettable	The ability to access an instance variable of a flavor by using an automatically generated method.
global variable	A special variable that has been assigned a global value. This variable is accessible to any function in any program throughout the system and is usually created with **defvar** or **defparameter**.
Glossary utility	An online facility for accessing and displaying definitions of terms. The utility provides features for accessing terms in different ways. A system glossary file is provided. You can add to this file or build your own glossary file to display terms appropriate to your application.
grabbing the mouse	A situation in which a process takes control of the mouse away from all windows.
graphics character	A character that has a standard textual representation as a single character, such as A, *, or =.
graphics editor	An interactive tool for creating pictures.
graphics object	A flavor instance used in the graphics window system to represent and display a graphics object, such as a circle, square, text string, or any user-defined object.
graphics window system	A flavor-based support for creating graphics objects such as lines, splines, arcs, circles, polygons, and text.

Grasper	An optional network representation tool built on top of Lisp. You can use it to describe knowledge that is naturally represented as a network. The Grasper primitives include functions to create, delete, bind, evaluate, and show the existence of elements in a network. Network elements include nodes to represent specific objects, edges to represent relationships between objects, and spaces to represent a collection of objects or a specific view of a collection of objects.
gray plane	A buffer in the font editor that displays characters in the editing area. You can use a character in the gray plane as a reference or template for the character you are editing in the black plane. Pixels set in the gray plane only are displayed in light gray; pixels set in both the gray plane and the black plane are displayed in medium gray.
gridify	To set the coordinates of a point to the coordinates of the nearest grid point. For example, if the x grid spacing is 20 and the y grid spacing is 15, a point with the coordinates (32,48) is moved to (40,45). Points are always and only gridified when the grid is turned on.
grind	To print or display Lisp code with line indentations that indicate the nesting level of functions, thus making the code easier to read. Grind is synonymous with pretty print.

h

hash code	The value produced by hashing an object. This value is usually an integer that is used as an index to a hash table.
hash function	A function that defines a hashing algorithm. This type of function accepts a hash key as an argument and returns a hash code. The hash code is then used as a value to access a hash table entry. See also hash value.
hash key	A symbol that names an entry in a hash table.
hash table	A table containing objects that have undergone hashing. Each hash table has a pair of entries associating a hash key with a hash value.
hash value	The value associated with a hash key in a hash table.
hashing	A technique used to provide fast retrieval of data in large tables. It consists of performing a computation on a set of objects to produce a hash code for each object. Each object becomes an entry in a hash table, which is indexed according to its hash code.
hashing collision	A condition that occurs when hashing two different objects results in the same hash code.
HELP key	See information and status keys.
herald	A listing of information about your system, which includes the name of your system, the memory available, the name and version of the software available, and the logical name of your machine. The herald is displayed in the Lisp Listener window when you boot the system. You can display the herald by executing the **print-herald** function.

history	A place to store information entered in one window so that it can be retrieved in a different spot in that window or in another window. Different histories are used for storing different types of information (see also command history, input history, output history, and kill history). In some cases, the system automatically places entries in a history; in other cases, you must use commands to do this. Commands are available in various windows to retrieve the entries. The entries stay in the history until the session ends or until the history becomes full. Once a history is full, an entry is deleted each time a new one is added. The number of entries that can be put in a history is user-modifiable.
hog	A file that has an excessive number of older versions present in the file system.
home directory	Generally, the directory where you keep your personal files, such as initialization files and your mail file. When you log in, you can specify the name of the host where you keep your home directory if it is on a machine other than the one you are using. Your home directory is identified by the function **user-homedir-pathname**.
home package	The package that owns a given symbol. A symbol can appear in more than one package, but it can have only one home package. The symbol's package cell contains the name of the home package.
hook	A piece of software or hardware that acts as an aid to debugging. A hook enables normal software evaluation or hardware execution to be interrupted at specified points in order to examine the program execution state or the hardware state.
horizontal index	A column subscript for a two-dimensional array. On the Explorer system, this is the second subscript.
host	Any computer or node on a network that can initiate a network operation or has a resource that can be used by other computers on the network.
host object	A Lisp object that represents a computer, usually connected to a network. On the Explorer system, a host object is a flavor instance that records information such as the name of the host, its operating system type, and its network address.
HYPER key	See modifier keys.
hysteresis	The distance, in pixels, that you can move the mouse blinker outside of a menu before the menu no longer owns the mouse.

i

I/O stream	See stream.
implicit application	See application.
implicit evaluation	See evaluation.

import shadowing	A type of shadowing that may occur when a particular package inherits symbols from another package. Ordinarily, if the names of the imported symbols conflict with those of the current package, an error is signaled by the Lisp Reader. However, the local symbols can be shadowed by the imported symbols through the use of the **shadowing-import** form, in which case the imported symbols take precedence over the local ones.
importing a symbol	Causing a symbol to become an internal symbol of a package. A symbol can be internal to more than one package although it can have only one home package.
incremental garbage collection	A type of garbage collection that occurs in parallel with computation. Incremental garbage collection is triggered when a user-specified threshold of available memory has been crossed.
incremental search	A Zmacs search facility that finds intermediate strings while you are typing. As the string is accumulated, this facility searches for what you have typed so far. That is, if you want to find the word CATS, this search facility finds the first C when you type it. When you type A, it finds the first CA and so on.
indefinite extent	A type of extent in which an established object continues to exist as long as the possibility of reference remains.
indefinite scope	A type of scope in which an established object can be referenced from anywhere within any program. This is similar to the concept of a global variable.
index offset	A number that is added to a subscript of an array to give the actual displacement of the element in the array.
index value	1. The integer value used to access an element of an array. 2. During list traversal, a value indicating how many list elements should be traversed before extracting the desired element.
index variable	A variable used in iterative forms, such as the **do** function, that is manipulated to control the amount of iteration.
indicator	A symbol that names an entry in a property list, usually a keyword.
indirect array	An array whose contents, instead of being located at a fixed place in virtual memory, are defined to be those of another array.
inferior	See inferior window.
inferior window	A subordinate window. An inferior window is contained entirely within its superior window and is positioned relative to the top left corner of the superior window.
information and status keys	The set of keys that supply information, invoke utilities, and control the keyboard: ■ HELP — Provides online documentation of various features. ■ NETWORK — Invokes functions in network utilities such as Telnet. ■ STATUS — Accesses the contents of the input, output, or kill histories.

- SYSTEM — Used in combination with character keys to invoke system utilities. SYSTEM HELP has other useful information.
- TERM — Used in combination with other keys to perform various functions on the console. TERM HELP has other useful information.

init file	See initialization file.
initialization file	A file you can create in your home directory or create with the Profile utility for customizations to be executed every time you log in. These customizations can set variables, load files, define functions, and so on. If your home directory is on an Explorer system, the pathname of this initialization file will be LOGIN-INIT.LISP.
initialization option	An option provided by the flavor system enabling the instance variables of a flavor instance to be initialized when a flavor instance is created. (Sometimes referred to as an init option.)
inittable	A characteristic of instance variables of a flavor that enables the instance variables to be initialized to a specified value when an instance of the flavor is created.
inline subroutine	A function in which the compiler incorporates the body forms of a subroutine into a function being compiled, substituting the argument forms for references to the variables in the inline subroutine's lambda list. This operation makes the code more efficient at the cost of losing some debugging ability. An inline subroutine is also called an open-coded subroutine.
input editor	A facility that allows you to edit input streams; formerly known as the rubout handler.
input history	A buffer that records the forms you enter in a Lisp Listener window. See also history.
insignificant command	A Universal Command Loop (UCL) command that performs a minor or insignificant operation. Insignificant commands are not placed in UCL command histories.
Inspector	A window-oriented program for examining complex data structures. It allows you to view and modify various components of Lisp objects. The Inspector has a history pane that maintains a list of all objects inspected so far and several inspection panes that allow you to view different objects at the same time. See Flavor Inspector.
instance	An individual occurrence of a flavor or structure.
instance variable	A part of the data structure of a flavor that specifies an attribute for that flavor. Each object (instance of the flavor) holds its own set of values for the instance variables. The flavor can establish default initial values for the instance variables, or you can supply them explicitly when you instantiate a flavor. When you combine flavors, the set of instance variables for the new flavor is the union of all the sets of instance variables from all the component flavors.
instantiate	To dynamically create an object, which can be an instance of a flavor or a defined structure.

internal package mapping	An operation that occurs when the current package accesses a symbol that belongs to the current package.
internal symbol	A symbol of a package that normally cannot be accessed by functions of other packages.
internal value cell	See value cell.
interned symbol	A symbol that belongs to a specific package, which is specified by the value of the symbol's package cell.
interning a symbol	Causing a symbol to belong to a package.
interpreted function	A function whose definition has been converted into an internal list representation by the Lisp Reader and which is then interpreted by the Lisp evaluator. Only the list representation of the function definition is saved in memory, not the printed representation.
interpreter	See Lisp interpreter.
interrupt	A timing signal that causes the processor to suspend execution of its current process and to execute another process. On the Explorer system, interrupts are implemented as events.
ITAL LOCK key	See mode keys.

j

justify	See adjust.

k

keyboard cursor	A blinking cursor appearing in the selected window that indicates where the text you type is echoed on the display.
keyboard macro	A series of Zmacs commands that are stored for use as a group. If you repeatedly use a particular series of command keystrokes, you can specify that the sequence of keystrokes be installed on a single key. You can specify several keyboard macros.
keystroke sequence	Any combination of keystrokes used to invoke a command or operation. Some sequences are chorded, which means you must press all the keys simultaneously. Chorded sequences are denoted by hyphens between key names. Sequences that are not chorded are denoted by spaces between key names; you must press these keys in the order specified.
keyword	A symbol that evaluates to itself and is contained in the keyword package. Keywords can be used as arguments to functions.
keyword argument	An argument passed to a keyword parameter. The argument is a pair consisting of a keyword and an associated value. The value is bound to the corresponding keyword parameter when the function is applied. The value of a keyword argument can itself be a keyword.

KEYWORD package	The package that contains all keyword symbols used by the system and by user-written code.
keyword package qualifier	A package qualifier consisting of a single colon (:) that is not preceded by a package name. This indicates that the symbol belongs to the KEYWORD package.
keyword parameter	A symbol in a lambda list indicating that this symbol is a function keyword. Keyword parameters always follow the lambda-list keyword &key. They allow an argument to be passed and matched to the corresponding parameter by a keyword rather than by position in the function call.
kill	1. To deactivate a window and make a positive effort to remove any other entities—such as processes or network connections—that may be associated with the killed window. If the killed window has associated entities, garbage collection alone may not be satisfactory. Therefore, killing a window is preferable to deactivating a window. 2. In Zmacs, to move a block of text into the kill history and remove it from your buffer. See also save.
kill history	A type of buffer that stores any block of text you have killed in a Zmacs buffer or in the input editor. The kill history is a mouse-sensitive display. You can retrieve the entries in the kill history by using the mouse or by using yank commands. See also history.

l

label	1. A text string in a window margin that identifies the window. Usually, the label appears in the lower left-hand corner. 2. The description of the partitions of a disk.
lambda binding	The binding of the parameters of a function to their corresponding argument values during application of the function.
lambda expression	A list that consists of the symbol **lambda** followed by a lambda list and a body containing any number of forms. Lambda expressions are used to define unnamed functions within a function call form. Lambda expressions by themselves are not forms and cannot be evaluated.
lambda list	A list that specifies names for the parameters of a function (sometimes called a *parameter list*). Lambda lists are contained within function definitions.
lambda-list keyword	A keyword used in a lambda list to indicate a parameter type such as an optional or rest parameter. The lambda list keywords are always preceded by an ampersand (&) and are provided in a function definition. The following are the available lambda-list keywords: ■ &optional ■ &rest ■ &key ■ &aux

- &special
- &local
- &functional
- &quote
- &eval

LAN	See local area network.
landscape	A particular orientation (shape) of a window or the display in which the area shown is wider than it is tall. The Explorer monitor uses a landscape orientation, instead of a portrait orientation.
leader	See array leader.
leaf	A component of a list that is not another list.
LEFT key	See mouse keys.
lexical binding	A type of binding in which a variable can be accessed only if its reference occurs textually within its establishing function (making it local).
lexical closure	A closure that saves the values of a function's lexical variables when the function has completed executing.
lexical scope	A type of scope in which the values of free variables are determined in the definition environment rather than in the run-time environment. In lexical scope, a reference to an established object occurs only within a particular portion of the program that is lexically contained within the establishing form.
lexical variable	A variable used in a function as a local variable.
lexicographic comparison	An alphanumeric comparison of two strings based on a collating sequence.
LINE FEED key	See line-positioning keys.
line height	The largest height of all the characters of the fonts in the font map plus the vertical spacing (VSP), which is the amount of space between lines. The line height determines the amount of space allowed for each line of text on the display. If the font map contains a large font and a small font, the line height is still large enough for the large font.
line-positioning keys	The set of keys that move the keyboard cursor within a line or to a new line. These keys are as follows:

- LINE FEED — Inserts a carriage return, moving the cursor to a new line and, if in the Zmacs editor, properly indenting the line for Lisp code.
- RETURN — Usually inserts a carriage return, moving the cursor to the left margin of the new line.

- RUBOUT — Backspaces and deletes a character at the current cursor position; can be used with modifier keys to rub out more than one character at a time.

- TAB — Moves the keyboard cursor a specified number of columns to the right.

line spacing	The number of lines per inch (lpi) printed. This term usually describes output from a printer.
linearization	A procedure involving the computation of an offset value from the subscripts of a multidimensional array. Real memory arrangement is linear. Therefore, when elements of a logical multidimensional array are accessed, there must be a mapping from a logical multidimensional subscript to a real single offset of linearly arranged memory.
linearized subscript	The result of multiplying each subscript of an array by an appropriate coefficient and adding them together.
Lisp	A programming language that is especially suited for manipulating symbolic structures in the form of linked lists. Lisp is widely used for artificial intelligence research and applications requiring symbolic processing. The Explorer system is written in Lisp, providing a rich, integrated environment for the development of Lisp programs.
Lisp compiler	A program that converts Lisp functions into machine code so that they run faster and require less memory. The compiler performs a number of different types of optimizations, checks for errors, and issues warnings when errors are found.
Lisp form	An object that can be evaluated.
Lisp interpreter	A program that evaluates Lisp forms.
Lisp Listener	A window through which you can interact directly with the Lisp interpreter. The Listener accepts Lisp forms as you type them and displays the results of evaluation. The Listener has a number of features to help you type Lisp forms, including completion and editing capabilities.
Lisp object	Any element of the Lisp language, such as a symbol, list, function, or structure. Objects include those that can be evaluated (forms) and those that cannot be evaluated (such as property lists). A set of similar objects comprises a data type.
Lisp Reader	A Lisp procedure that reads characters from an input stream, interprets them as the printed representation of a Lisp object, converts the printed representation into an internal binary representation, and and returns a corresponding object.
list	A data structure represented internally by a chain of conses (nodes) linked together by the cdr components of the conses. A true list (sometimes called an ordinary list) is terminated by the symbol **nil**. A dotted list is terminated by a non-nil atom. Most functions requiring lists as arguments require true lists.

	A list's external representation consists of a sequence of elements enclosed in parentheses. Each element in the list can be either an atom or another list. Lists are interpreted as function calls, macro calls, or special forms unless they are quoted.
Listener	See Lisp Listener.
lm	See local machine.
load	To bring a file into memory so that it is ready for execution. You can load the system, files, and patches with specially defined load functions or with load commands. To load a character (text) file is to read through it, evaluating each form within the file. Loading a compiled (xld) file, evaluates the compiled forms in the file. The usual reason for loading a file is to declare variables, functions, flavors, and so on. The best way to load any type of file is with the **load** function, which loads the file according to the file type (character or binary).
load partition	A disk partition that contains the initial environment loaded when you cold boot the system. Also called the *system partition*, *system band*, or *world load*.
local area network	A number of computer systems connected to a common data link; usually confined to a small area, such as an office or building.
local bus	A high-speed, 32-bit backplane bus for exclusive use of one processor. The Explorer processor uses the local bus for access to main memory and the graphics bit map. The local bus provides a high processor-to-memory bandwidth and leaves the NuBus device available for system-wide operations. Use of the local bus is transparent to the software.
local machine	The system you are currently working on as opposed to another system on the network. The local machine is usually abbreviated as lm in pathnames.
local pervasive declaration	A local declaration that affects all code of a special form.
locative	A low-level type of Lisp object used as a pointer to a memory location.
loop macro	A programmable iteration facility that consists of some initialization code, a body that can be executed a number of times, and some exit code. Each **loop** macro consists of clauses; you can choose and combine predefined clauses to specify the iteration path.

m

macro	A form that is converted into a different (usually, a more complex or more internally efficient) form before it is evaluated or compiled. Macros give Lisp more versatility by allowing you to write code in a form more suitable to your application. The ==> notation is used in examples to indicate macro expansion.
macro call	A list form that is similar to a function call except that it invokes a defined macro rather than a function.

macro definition	A procedure that creates a new macro and defines its actions.
macrocode	The actual machine code. If the Explorer system had an assembler, it would produce macrocode.
Mail	An interactive facility for sending and receiving mail over the network and for managing your mail files.
main memory	The physical address space located on the memory board(s). Main memory is the primary store in Explorer virtual memory.
main screen	The portion of the physical display area excluding the mouse documentation window and the status line. The main screen is used for displaying the windows of active processes you invoke.
major mode	See Zmacs major mode.
major version number	See version number.
map partition	A disk partition that lists all media defects on the disk.
mapping	A type of iteration in which a function is successively applied to elements of a list or a sequence. Examples are **mapcar** and **mapcon**. Mapping functions provide a control structure for applying a function to every element in a list and accumulating the results.
MAR	Memory Address Register, which provides the ability to trap on any reference to a word (or a set of words) in memory. If something is being written to by unknown agents, this facility can help track down the source of the unwanted input.
margin	The part of a window that is not the inside. A window has four margins, one for each edge. The inside and the margins of the window are kept separate. I/O can only be done to the inside.
margin choice	A facility that allows choice boxes to be placed in the bottom margin of a window. This allows certain choices to be independent of anything else in the window.
margin item	A component of a window that is contained within a margin (for example, a border, a label, or a scroll bar).
mark	1. To highlight (usually by underlining) a block of text by setting the two buffer pointers in Zmacs (that is, point and mark) at either end of the block of text. You can then manipulate the block of text as one unit. You can mark text of any size. The marked text is called a *region*. Some of the uses for marking text are as follows: ■ To put the region on the kill history by killing or saving it ■ To evaluate or compile the region ■ To change the font of the region 2. A buffer pointer in Zmacs used to specify one edge of a region. (The other edge is specified by point, which is the left edge of the keyboard cursor.)

mass storage controller	A logic board installed in the system enclosure that provides an interface between the NuBus device and the small computer systems interface (SCSI) bus for all mass storage devices.
mass storage enclosure	A compact enclosure that houses two storage devices; this can be any combination of Winchester disks and tape drives. (This is commonly called a *data brick*.)
mass storage subsystem	The set of hardware components that provides permanent storage for programs and data. The subsystem consists of the mass storage controller, the small computer systems interface (SCSI) bus and cables, and the mass storage enclosures.
menu	A window that presents a set of items from which you can select various choices using the mouse. A menu can be a permanent pane in a window, or it can be a temporary window.
menu boot	The invoking of a series of menus that allow you to specify the device, load band, and microload band from which the system boots. You execute a menu boot when you press the keystroke sequence META-CTRL-META-CTRL-M. See also system reset, cold boot, and warm boot.
menu geometry	The way in which a menu is displayed as described by six parameters: the number of columns, the number of rows, the inside width in pixels, the inside height in pixels, the maximum width in pixels, and the maximum height in pixels. Each parameter can be specified as a constraint or can be allowed to default based on the menu item list and the parameters that are constrained.
menu item list	A list of the items contained in a menu. Each item represents one of the choices offered and specifies what to display in the menu and what operation to perform when that item is selected.
merging of pathnames	The operation that takes as input a pathname, an association list of defaults (or another pathname), a default type, and a default version, and returns a pathname. Basically, the missing components in the pathname are filled in from the association list of defaults. However, if a name is specified but the type or version is not, then the type or version is treated specially.
message	In the flavor system, a call to perform an operation on an object and return a value. The message consists of the name of the object and the operation to be performed (and arguments). The operation is handled by the appropriate method attached to the flavor of which that object is an instance.
META key	See modifier keys.
metering	A facility that monitors your program so that you can find out which parts use the most time. When you run your program with metering, the system records each function call and its return, along with the time each occurred. Page faults are also recorded.
metering partition	A disk partition named METR that contains the information generated by metering.
method	A function that is attached to a flavor to define a generic operation that can be performed on any object of that flavor. The method handles the operation named in a message sent to the object.

microcode	Machine instructions that implement the Explorer system. Microcode is contained in the microload partition.
microload partition	A disk partition containing the microcode that is loaded when you cold boot the system.
MIDDLE key	See mouse keys.
minibuffer	See Zmacs minibuffer.
minor mode	See Zmacs minor mode.
minor version number	See version number.
mixin flavor	A flavor that defines one feature or characteristic of an object (often called a *mixin*). Mixins usually must be combined with a base flavor to be complete. Normally, mixins are used to slightly alter a base flavor by changing or adding methods.
mode keys	Keys that put the keyboard into a specific mode of operation when locked. The mode keys are locked electronically rather than mechanically; a small LED on the keycap lights up when the key is locked. The following are the mode keys: ■ BOLD LOCK — Reserved for future use. ■ CAPS LOCK — Produces uppercase letters for the letter keys. ■ ITAL LOCK — Reserved for future use. ■ MODE LOCK — Enables the cursor pad to move the mouse cursor.
mode line	A line in a Zmacs window, appearing just above the minibuffer, that contains information about the major and minor modes in effect, the name of the buffer, and the status of the buffer.
MODE LOCK key	See mode keys.
modifier keys	A set of keys used in combination with other keys to modify their behavior and invoke a variety of commands. Modifier keys do not themselves transmit characters. The modifier keys are HYPER, SUPER, META, CTRL, and SYMBOL. HYPER, SUPER, META, and CTRL are generally used to invoke commands. SYMBOL is used with letters or with other modifier keys to produce alternate character sets. SHIFT produces uppercase letters and is also used with other modifier keys to invoke commands.
module	A software subsystem that is loaded from one or more files and that can contain one or more packages.
more processing	Output exception processing that allows a programmer to specify what happens when typeout to a window would go beyond the bottom or edge of the window.

mouse	A hand-held pointing device that when moved across a flat surface causes movement of a special pointer (mouse cursor) on the display. The mouse on the Explorer system is an optical mouse. The mouse has three buttons that can cause operations to be performed when pressed.
mouse buttons	A set of buttons on the mouse that cause operations to be performed when pressed. The operation is performed on the item the mouse cursor is positioned over.
mouse click	See click.
mouse cursor	A blinker on the display that is positioned by using the mouse. The mouse cursor is usually in the shape of an arrow outside a menu or the shape of a period inside a menu, but it can change shape to indicate the type of operation for which the mouse is currently being used.
mouse documentation window	One or more lines (two in reverse video, by default) that appear just above the status line near the bottom of the display. The mouse documentation window describes the currently available mouse operations or gives information about the item to which the mouse cursor is pointing. Also called the mouse doc-line.
mouse keys	A set of keys (LEFT, MIDDLE, and RIGHT) on the keyboard that perform the same functions as the mouse buttons.
mouse pad	A special surface used with some optical mice to detect movement. For proper results, place the mouse pad in the same orientation (landscape) as the display.
mouse process	The process that tracks the position of the mouse and handles input from the mouse buttons.
mouse-sensitive item	An item in a menu or window that is highlighted in some way (usually boxed) when you point to it with the mouse cursor. You can then perform operations on a mouse-sensitive item by clicking one of the mouse buttons.
multiple returned value	More than one value returned by a function.
multiple values	A Lisp facility that enables an evaluated form to produce or return more than one value.
multiple-choice menu	A menu that displays a series of items, one per line, along with a set of choice boxes for each item. The set of choice boxes is the same for all items. You can select yes or no for each choice by clicking the mouse within the choice box.
multiple-item menu	A menu that allows you to select any combination of items rather than a single item. After you select the items, you need to select another item to perform the specified operations, such as the Do It item.

n

name conflict	A problem that occurs when a symbol is used to name two different objects. The package system helps prevent name conflicts.

name server	A host that serves a namespace. That is, a name server is responsible for accepting updates and responding to requests for information contained in the namespace.
named structure	An aggregated user-defined data type that is created by the **defstruct** macro and is named.
namespace	A database composed of objects that are identified by name and class within the namespace. These objects can have arbitrary attributes (or properties) attached to them. A namespace is used on the Explorer system to define the network configuration. Explorer namespaces can be used for other purposes as well.
namespace editor	A screen-oriented editor used to view and/or modify objects within a namespace.
namespace search list	A list of namespaces used as ordered defaults for looking up names when a particular namespace is not specified.
Natural Language Menu System	An optional set of tools for building a menu-driven, natural language interface to your applications. NLMenu presents users with a set of windows and menus containing words or phrases in English (or some other natural language). From these, the user selects the appropriate items to construct input sentences, such as queries to a database. NLMenu controls each step of the selection, allowing only valid sentences to be constructed. NLMenu includes tools for building the lexicon, grammar, and screen formats required for each specific application.
network	A complex that consists of two or more interconnected hosts.
NETWORK key	See information and status keys.
nil	A symbol that represents the empty list and a logical false value. The notation () is an alternate representation for the symbol **nil**. By convention, () is used to denote an empty list, and **nil** is used to denote a false value.
NLMenu	See Natural Language Menu System.
nonlocal exit	A facility for exiting a complex process in a nonlocal, dynamically scoped manner. This facility uses **catch** and **throw** forms, allowing you to backtrack in a search operation or recover from errors.
nonlocal variable	See free variable.
nonpervasive declaration	A declaration that does not affect all the code of a special form except when the code is defined to be within the scope of the variables affected by the declaration.
normal video	A display state in which black images are shown on a white background (negative contrast or positive image). By default, most windows appear in normal video (an exception is the mouse documentation window). You can toggle between normal and reverse video by changing the value of a system variable.
notification window	A temporary window that notifies you of an error or action taken by some process other than the process for the currently selected window.

notifications	A facility that displays messages that take place asynchronously and are not related to the process in the currently selected window (for example, file and print servers). Notifications work through the process in the selected window; each window can display a notification differently.
NuBus device	The backplane bus on which the Explorer system architecture is based. The NuBus device is a processor-independent, high-speed, asynchronous bus that multiplexes 32-bit data words with 32-bit address codes and uses a master/slave communications protocol.
number	See number object.
number object	A data object in Lisp that represents rational, irrational, or complex numbers. The different types of numbers require different data objects.
number pad	A set of keys that includes numeric digits and characters for entering arithmetic expressions, as well as the following: ■ ENTER — Currently not used, but can be programmed for use in application software. ■ SPACE — Performs the same function as the space bar. ■ TAB — Performs the same function as the TAB key on the main part of the keyboard.
numeric argument	An argument that you can give to many Zmacs commands, usually designating a repeat count (the number of times you want the command executed).

O

object	1. Any element of the Lisp language. See also Lisp object. 2. In the flavor system, a specific instance of a flavor.
one-origin indexing	A type of indexing for arrays or sequences in which the first index starts with one. All Explorer code uses zero-origin indexing.
open-coded subroutine	See inline subroutine.
optional parameter	A parameter that does not require a corresponding argument value in the call in order for the function to execute correctly. The lambda-list keyword &optional is used to indicate the optional parameters in a function.
ordinary list	See list.
output history	A buffer that stores the returned values of forms entered in a Lisp Listener window. See also history.
over-rubout	A condition that occurs when you press the RUBOUT key more times than necessary to rub out all the characters in the window's input buffer. Also called full-rubout.

owning the mouse	A term descriptive of the window that handles the mouse. This is usually the window in which the mouse is positioned. The window that owns the mouse is the one that receives the **:handle-mouse**, **:mouse-moves**, and **:mouse-click** messages.

P

P1, P2, P3 connectors	The set of connectors on the system logic boards that plug into the backplane. The P1 connectors plug into the top row of the backplane and provide NuBus device signals on all boards. The P2 connectors plug into the center row and are used for local bus access on some boards and are available for general I/O on other boards. The P3 connectors plug into the bottom row and are used for general I/O.
package	A collection of symbols that serves as a namespace. See also package system.
package cell	A component of a symbol that contains the name of the symbol's home package.
package name	A string that serves as a name for a particular package.
package nickname	A shortened or alternate package name.
package qualifier	A prefix to a symbol or object that specifies that the symbol or object belongs to a package other than the current package. The package qualifier consists of the name of the symbol's home package and a colon. For example, `user:x` indicates the symbol `x` is in the USER package.
package system	A facility that establishes a mapping from print names to symbols and helps prevent namespace conflicts. The package system allows different programs to use the same name for objects so that the programs and objects can coexist in the same environment. The different programs can be placed in separate packages. Each package provides a separate namespace for all the symbols used in the programs within that package. However, you can access symbols from different packages using package qualifiers.
packet	The basic unit of data that is transmitted over the network.
pad	See mouse pad.
page	1. A unit of storage for programs and data used to implement virtual memory. On the Explorer system, each page consists of 1K bytes. 2. In Zmacs, text separated by the page character, which is produced by pressing CTRL-Q CLEAR SCREEN.
page band	A group of one or more page partitions. If more memory is required than is physically present, pages are moved from physical memory to the page band, thus making space available in physical memory.
page partition	A disk partition that is used to implement address space for virtual memory.
pairwise comparison	A type of comparison in which elements are compared based on their positions in a sequence. For example, the first element of one sequence is compared only with the first element of another sequence.

pan	Panning moves a window in relation to the objects in the current graphics editor buffer, but does not change the apparent scale of the objects that are displayed. For example, panning left shows the graphics objects that are to the left of the current window.
pane	A subdivision of a window; an inferior window. A collection of panes comprises a frame, which can then be manipulated as a single window.
parameter	A variable named in a function's lambda list. When the function is applied, the parameter is bound to the value of the corresponding argument in the function call. This kind of parameter is analogous to a formal parameter in other programming languages. There are four types of parameters: required parameters, optional parameters, rest parameters, and keyword parameters. Required parameters must appear first in the function's lambda list. The other types of parameters are indicated by lambda-list keywords.
parameter list	See lambda list.
parse	To determine the syntactic structure of a sentence or some other object.
parsing of pathnames	An operation that converts a string into a pathname object.
partition	A contiguous, discrete region on a disk. Each partition has a unique name that indicates the function of that partition.
partition namestring	A unique name that is up to four characters long that indicates the function of a particular partition on the disk.
partition-table partition	A disk partition that contains information about each of the partitions on the disk.
patch	A small change or set of changes that updates a particular version of a system. Note that a patch replaces an entire function body, not just a few words within the function body.
patch system	A facility that keeps track of all the existing patch files and incorporates them into the system. Patch files are loaded to cause changes (bug fixes or improvements) in an existing program. The patch facility allows maintenance of both the Explorer system itself and applications that are large enough to be loaded and saved on a disk partition.
pathname	A string used to access a file or directory. On the Explorer system, all pathnames have five components, each of which is delineated by a special character, as follows: *host:directory;name.type#version* The host component can be either the name of the local machine or any remote host. The directory component can include subdirectory components, each of which is separated by a period (.). The name component is the name of the file followed by the file type component and the version number component of the file. Except for the host (which specifies the flavor of pathname object to use), all pathname components can be either a string or a symbol. The special symbol

	:wild can match any file with respect to that component. A wild component is read and written as an asterisk (*). See also parsing of pathnames and merging of pathnames.
pathname defaulting	To supply the missing components of a pathname. When the Explorer system prompts you for a pathname, it does not expect you to type a complete pathname that contains all five pathname components. Instead, default values allow the Explorer system to automatically supply whatever components are missing. Each program dealing with pathnames typically has its own set of defaults.
PCA board	See peripheral cable adapter board.
PDL	A push-down list; a stack.
Peek	A window-oriented utility that displays different types of system activities and their current states.
peripheral cable adapter board	A small board that plugs into the system backplane and provides an interface between the system and the I/O cables to peripheral devices. This is also called a PCA board.
permanent menu	A menu contained on one of the panes of a window.
pervasive declaration	A declaration that affects all the code within the body of a special form.
picture	The definition of an object or collection of objects within the graphics editor or graphics window system. You can save a picture for future use and integrate it with other pictures or subpictures.
pitch	See character spacing.
pixel	The smallest unit that can be displayed on a screen.
place	A term used to denote a generalized variable.
plane	An array whose bounds, in each dimension, are theoretically plus infinity and minus infinity. All integers are legal as indices for a plane.
plist	See property list.
point	A buffer pointer in Zmacs used to specify the location of the keyboard cursor or one edge of a region. (The other edge of a region is specified by mark.) Point is always between characters and is shown by the left edge of the keyboard cursor.
point PDL	A push-down list (PDL) in Zmacs that stores cursor locations (or point). Zmacs allows you to store cursor locations and return to them later. The cursor locations can be in different buffers, allowing you to jump between buffers.
pointer	An object whose value is the address or location of a piece of data.
pop-up menu	A temporary menu from which you must select an item before the menu goes away.

portrait	A particular orientation (shape) of a window or a display in which the area shown is taller than it is wide.
positional constructor function	A function that allows the components of a structure to be initialized in a manner that differs from regular constructor functions. With regular constructor functions, structure components are initialized by keywords. With positional constructor functions, structure components are initialized without keywords. Also known as *by order of arguments*, or BOA, constructor.
positional parameter	A parameter that is passed argument values according to its position in the lambda list.
possibility	In Zmacs certain commands, such as the Edit Definition command (META-.), perform operations that create a set of results. Each result is a possibility that can be individually examined. For example, if you wish to edit the definition of a function with META-. and that function was defined in a number of places, each definition becomes a possibility that you can examine.
predicate	A function that tests for a specific condition involving its arguments and returns a non-**nil** value (often the symbol **t**) if the condition is true or **nil** if it is not true. For example, the function **listp** returns **t** or **nil** depending on whether its argument is a list or not. Conventionally, the name of a predicate begins with a function name and ends with the letter **p**, such as with the function **listp**.
predicate-specified type specifier	A type specifier that defines a data type in terms of objects that satisfy a given predicate.
pretty print	See grind.
primary method	A flavor method that handles the main task of an operation. When methods are combined, only one primary method is used for a given operation.
print name	The printed representation of a symbol; the string used to identify a symbol. The print name is stored as a component of the symbol.
print server	A program on a host with printers that handles standard requests for printing from other hosts on the network.
print-print consistency	A concept stating that if two interned symbols are not **eq**, then their printed representations are different sequences of characters.
print-read consistency	A concept stating that an interned symbol that always prints as a sequence of characters yields the same (**eq**) symbol when read back in.
printed representation	The representation of Lisp objects as they appear in printed text.
process	A separate computation or activity consisting of a set of operations performed sequentially. The Explorer system allows a number of processes to be active concurrently. All processes run in the same virtual address space, sharing the same set of Lisp objects. However, each process has its own program counter, its own stack of function calls, and its own special-variable binding environment.
proclamation	A universal declaration; a declaration that is always in force unless locally shadowed by another declaration.

Profile utility	A series of windows that allow you to inspect and change the values of various system default variables, and to then save these customized changes in an initialization (usually the login) file.
progn-type form	See simple sequencing function.
program control keys	A set of keys that interrupt and resume the flow of control in program execution. These keys are the following: ■ ABORT — Stops the current program and returns to the next higher command level. This key is often used with modifier keys to leave a recursive function. ■ BREAK — Stops the current program. ■ RESUME — Resumes execution of a program that was stopped by using the BREAK key or by encountering a breakpoint. This key is also used to signal a proceed option when recovering from an error in the error handler.
Prolog	A logic programming language used alone or with Lisp for representing objects and solving object relationship problems. Prolog is an optional package available for the Explorer system that includes a Prolog compiler and a prototype expert system that uses a Prolog inference engine. You can extend the prototype expert system to suit your needs or use it as is.
Prolog window	The window in which a user develops and tests Prolog programs.
property list	A component of a symbol that effectively provides the symbol with many modifiable, named components. The property list has zero or more entries, with each entry consisting of a pair of elements. The first element of the pair is called the indicator and is used to name a particular property. Each indicator must be unique within that property list. The second element can be any Lisp object that represents the value of that property. Functions are available to manipulate a symbol's property list.
property name	See indicator.

q

quote	To inhibit the evaluation of an object. This is often called quoting because you generally use the **quote** special form to inhibit evaluation. You can abbreviate the special form to a single quote mark or apostrophe (').

r

radix	The base of the numbering system being used. For instance, the binary numbering system is of radix 2.
random number	A number obtained by chance from a uniformly probable distribution of numbers.

random-number generator	A function designed to produce random numbers.
rank	The number of subscripts used to refer to one of the elements of an array, or simply the number of dimensions of an array. On the Explorer system, arrays can have a maximum rank of 7.
raster height and raster width	The dimensions of the array of bits stored for each physical character in a font. Every array for one font has the same raster height and width.
read-eval-print loop	A continuous loop that reads a Lisp form, evaluates it, and prints the returned value(s).
read-read consistency	A concept stating that when reading the same print name you always receive the same (eq) symbol.
Reader	See Lisp Reader.
recognition completion	See completion.
record structure	See structure.
recursive function	A function that steps through its operation by continually invoking itself until a point of termination is reached. Then, a value is returned to the environment of each invocation of the function.
reduced ratio	A ratio number that is in a form such that further factoring with the denominator and numerator of the number is not possible.
region	1. A contiguous section of address space used as storage for an area. The area can be divided into more than one region. A given region can store only one data representation type (a list or a structure). 2. Any block of text in a Zmacs buffer defined by point and mark. Commands are available to manipulate regions. See also mark.
register	1. A Zmacs temporary storage area in which you can store and retrieve either text or cursor locations. You can also store text and a cursor location simultaneously in one register. A register is similar to the kill history except that you can only store one block of text and one cursor location in a register. However, they stay in a fixed place. In the kill history, text is moved down the stack as new entries are added. 2. In the font editor, buffer-like panes that can temporarily hold characters.
rehash threshold	A predetermined number of hash table entries. Once this threshold is exceeded, a hash table must grow in size and its current entries must be rehashed to accommodate more entries.
Relational Table Management System	An optional facility for building, accessing, and manipulating relational tables within the Lisp environment. It provides the basic features of a database while allowing as much flexibility as possible. All database manipulations occur in virtual memory; however, the database elements can be saved to and loaded from disk.

reparse attribute list	To permanently record editing changes made to the attribute list in a Zmacs buffer and to make those changes take effect. The changes are also saved when the file is written to disk. This operation is the inverse of the update attribute list operation, which records the changes in the attribute list from the current settings of the buffer (that is, buffer attributes that are changed but not recorded in the attribute list, such as changing modes with a mode command).
repeat form	A form that is repeatedly evaluated during iteration.
required parameter	A parameter that requires a corresponding argument value to be passed when the function is called in order for the function to execute correctly.
reset	To throw out the entire computation for a process and force the process to call its initial function again. Resetting a process clears its waiting condition, and if it is active, it starts running.
resource	A pool of interchangeable objects that are available to be used temporarily and then returned to the pool. Resources whose objects are windows are often useful. For example, a resource of windows serves as the System menu; when you invoke the System menu, a window is allocated from the resource, and it is returned to the resource's pool when it is deactivated.
rest	A term sometimes used to refer to the cdr.
rest parameter	A parameter that can be passed any number of arguments. The lambda-list keyword &rest is used to indicate a rest parameter. When the function is called, the required and optional parameters are bound to their respective argument values, and the rest parameter is bound to a list of any remaining argument values.
RESUME key	See program control keys.
resumption	See stack group switch.
RETURN key	See line-positioning keys.
reverse video	A display state in which white images are shown on a black background (reverse of normal video).
RIGHT key	See mouse keys.
root directory	The topmost level of directories in the file system, containing a number of user directories. Only directories (that is, no files) are allowed at the top level of an Explorer root.
row-major order	An ordering of multidimensional arrays in which memory locations represent the rows of the array. Therefore, the subscript representing rows is the first subscript, and the subscript representing columns is the last subscript. Arrays on the Explorer system are stored in row-major order.
RTMS	See Relational Table Management System.

rubber banding	An operation on a window or graphics object that is performed by pressing and holding a mouse button while you move the mouse. As the mouse cursor moves, the object stretches or shrinks in size, following the cursor until you release the button.
rubout handler	See input editor.
RUBOUT key	See line-positioning keys.
run bars	A set of three, thin horizontal lines that appear in the status line, indicating that the system is actively doing something. From left to right, the run bars correspond to the activities of garbage collecting, paging, and executing a process.
run reason	A Lisp object associated with a process. The Explorer system considers a process active if the process has at least one run reason and no arrest reasons associated with it.
running stack group	See current stack group.

S

salvager	A facility that verifies the integrity of a file system. The salvager verifies that all blocks and pointers are correct, that all files are closed, and that no dual pointers exist.
save	1. To write a Zmacs buffer to disk, automatically using the default filename. This differs from a write operation, which allows you to specify the filename. 2. To move a block of text into the kill history without removing it from your Zmacs buffer. See also kill.
scheduler	Software that manages the set of active processes. The scheduler repeatedly cycles through the active processes, determining for each process whether it is ready to run or is waiting. When the scheduler finds a process ready to run, that process is made the current process.
scope	The spatial or textual region of a program or form within which it is possible to refer to an object.
scoping	The rules or procedures involved in assigning values to variables when a program is implementing multiple levels of environments.
screen	The topmost node of the window system hierarchy, which defines an area of the physical display. The physical display area is divided logically into two screens: the main screen and the screen containing the status line and mouse documentation window. A screen has no superior windows, and windows in one screen cannot be extended into the other screen.
screen array	A special area in the hardware that is used to display the image on the screen.
screen formatter	A menu-based facility that allows you to change the shapes and sizes of windows and reposition windows within a screen. You can access the screen formatter through the System menu.

screen manager	The utility that controls which windows appear on a screen and where the windows appear on the screen. The screen manager can also select a window if no window is selected. See also autoselect.
scroll bar	A thin, vertical line that appears when you move the mouse cursor against the left margin of certain windows (such as Zmacs). The size and position of the scroll bar indicates the relative portion of the total buffer that is currently visible. For example, a short scroll bar appearing near the top of the displayed text indicates that you are looking at a small portion of the beginning of the buffer. When in the scroll region, the mouse cursor changes to the shape of a fat double-headed arrow, and you can then use mouse buttons to perform scrolling operations.
scrolling	The act of displaying different portions of a buffer whose contents are too big to all fit within a window at one time. Scrolling forward displays a portion toward the end of the buffer; scrolling backward displays a portion toward the top of the buffer.
scrolling window	See text scroll window.
SCSI bus	See small computer systems interface bus.
section	A top-level Lisp form in a Zmacs buffer. Usually, a section defines a variable or a function. Zmacs provides tools like List Sections that facilitate writing and debugging Lisp code. The only buffers that have sections are those containing Lisp code. Other buffers do not have sections. A section can contain any defining construct that begins with **def** after the initial opening parenthesis in the first column, such as **defvar**, **defflavor**, and **defun**. The defining construct can be user-defined. Each definition is a section. Actually, Zmacs looks for a matching set of parentheses starting with an opening parenthesis in the first column. Zmacs' detection of an opening parenthesis in the first column is unconditional. Even if the parenthesis is inside a quoted string or a block comment (that is, #| ... |#), Zmacs starts a new section. This feature may cause unexpected results.
sectionize	To read the Lisp code in a Zmacs buffer and update Zmacs internal data and property lists of the defined sections. This sectionizing is done automatically when a buffer is created. Certain Zmacs commands, such as Sectionize Buffer, allow you to resectionize the buffer after making changes. See also section.
seed	A number used to initiate a random-number generator.
select	1. To allow keyboard input to be sent to a window. 2. To box an item in a menu and click left to execute a command or invoke another menu. For example, if you select the System Menu item in the menu at the bottom of the Lisp Listener, you invoke the System menu.
selected window	A window that is enabled to accept input from the keyboard or mouse. Only one window can be selected at a time, and the selected window is made fully visible. The currently selected window is usually indicated by a blinking keyboard cursor.

self-evaluating form	A form that, when evaluated, returns itself (or a copy of itself) as the value of the form. Self-evaluating forms include all numbers, characters, strings, keywords, and bit vectors.
semilexicographic comparison	A string comparison where distinctions between lowercase and uppercase characters are ignored.
sequence	A data type that contains an ordered set of elements, which encompasses both lists and vectors.
server	A program that provides a standard service of some type on a network.
set	1. To permanently assign a value to a symbol or variable. 2. An abstract object containing zero or more related objects that can be represented in Lisp as a list or an array.
settable	The ability to set an instance variable of a flavor with a value.
shadow	To cause a symbol to take precedence over other symbols of the same name within a namespace.
SHIFT key	See modifier keys.
signaling	See condition signaling.
significant command	A command that performs a major or significant operation. Significant commands are placed in command histories. For example, deleting a line of text is significant, but deleting a single character is not.
simple array	An array that is not displaced to another array, has no fill pointer, and does not have its size altered dynamically.
simple sequencing function	A function that executes each form within the function in sequence and returns the value of the last form as the value of the function. This is also called a **progn**-type form.
single-colon package qualifier	A package qualifier (containing a single colon) that allows you to access the external symbols of another package.
small computer systems interface bus	An industry-standard bus used for access to mass storage devices. Use of this bus also allows for other commercially available devices.
sorting	The arranging of items in a sequence in some logical order, for example, alphabetic order or numeric order.
SPACE key	See number pad.
special form	A form that has its own idiosyncratic syntax. Special forms are generally environment and control constructs. The set of special forms is fixed in Common Lisp; you cannot define new ones.
special variable	A variable that has dynamic scope (indefinite scope and dynamic extent); similar to a global variable in other languages.
specialized array	An array whose elements are restricted to a specific data type.

specializing type specifier	A type specifier that defines specializations for a data type. These specializations are used to incorporate more efficient data type representations and allow for better compiler optimization.
spelling completion	See completion.
spline	A smooth curve through an ordered set of points. Splines are used to draw a curved line in utilities such as the font editor or the graphics editor.
split screen	A layout of the main screen in which two or more windows are completely visible. You can set up a split-screen configuration using the screen formatter.
stable rounding	A technique for rounding off floating-point numbers. This technique determines when to round up or down if a resulting value is exactly in the middle of two floating-point numbers. It rounds down if the least-significant bit is 0 and rounds up if the least-significant bit is 1. This technique ensures that no systematic bias is introduced.
stable sorting	A procedure that keeps equal items in their original order during a sorting operation. For example, when x and y are equal items being sorted with other items and x initially comes before y, x remains before y after the sorting operation.
stack	An area of memory used for temporary storage; sometimes referred to as the Lisp stack. The stack is implemented as a last-in/first-out data structure. As objects are pushed onto the stack, the stack pointer is incremented. As objects are popped from the stack, the stack pointer is decremented. In the Explorer system, special high-speed memories (caches) are reserved as stack buffers.
stack group	A type of Lisp object that represents a process and its internal state, including the Lisp stack. The stack group holds state information, including the local environment, special variables, and dynamic storage associated with the process. Stack groups are useful for implementing certain advanced control structures, such as coroutines and generators.
stack group switch	The operation of making a specified stack group be the current stack group; also called resumption.
stack list	A list that is discarded once the function that creates it is finished with it; also called a temporary list.
standalone editor	An editor that functions like a Zmacs editor but is not tied to the rest of the editing system. You can use a standalone editor in your program for terminal I/O.
starter kit	A file or set of files that comes with each system software option. The starter kits make use of the different features for each tool. They are intended to provide examples from which you can learn how to use a tool or upon which you can base your own application.
state information	Information on the values in an environment at any one quantum of time.
statements	The independently evaluated subforms of certain special forms.
static area	A specified area whose contents change slowly and, therefore, do not require regular garbage collecting. However, you can explicitly garbage collect a static area when necessary.

statistic counters	See Counters.
STATUS key	See information and status keys.
status line	A line in normal video at the bottom of the display that indicates the status of the system. From left to right the status line shows the date and time, your login name, the current package, various status messages, and the run bars. The status line can also show messages from the file server or the print server.
status message	A word or phrase appearing in the status line that describes the current state or activity. For example, the status message `Keyboard`, indicates that the process in the selected window is waiting for input from the keyboard.
Stepper	A facility that allows you to follow each step in the evaluation of a Lisp form and examine what is going on. The Stepper has a number of commands to control the stepping sequence.
sticky minor mode	See Zmacs minor mode.
stop-and-collect garbage collection	A type of garbage collection that occurs at a specified point in your program or whenever you turn on the garbage collector in this mode.
stream	A Lisp object that serves as a software channel for input and output. The use of streams allows I/O devices to be handled in a standard manner so that a program need not be concerned with the details of a particular device. A set of standard operations is available for every stream. Additional operations are defined for specific types of streams.
string	A one-dimensional array that represents a sequence of characters.
structure	A named, compound data object consisting of different components, which are also named; also called a record structure. In order to access the different components, both the structure name and component name must be used.
structure access function	See access function.
structure instance	A single occurrence of a record structure that is created from a predefined structure template.
subdirectory	A directory that is part of another superior directory. A subdirectory looks similar to any other file in the superior directory. On the Explorer system, the syntax of a pathname in a subdirectory is as follows: *host*:*superior-dir.sub-dir*;*name.type#version* On most file systems, any reasonable number of subdirectory levels can exist.
subwindow	See pane.
Suggestions menu	A type of menu that lists the commands available to a particular utility and/or types of things that can be done in the current situation. The menus change dynamically according to what you are doing. Many of the system utilities have a Suggestions menu, which you can turn on or off. You can tailor an existing Suggestions menu to meet your needs, or you can build a Suggestions menu on top of the UCL for your application.
SUPER key	See modifier keys.

superior window	A higher-level window. For example, a screen is the superior of a window, and a window is the superior of a pane. The position of a window is remembered in terms of where it is within its superior.
surface-mount technology	A technique in which chip carrier circuits are soldered directly to the surface of a board, allowing greater packing density, higher performance, and lower costs than previous methods.
symbol	A Lisp object used as an identifier to name other Lisp objects and variables. Internally, each symbol is represented as a structure with the following components (cells): ■ Print name ■ Value cell ■ Function definition cell ■ Property list ■ Package cell These cells contain information about the symbol, and functions are provided to manipulate this information, as well as the symbol itself.
symbol inheritance	To enable external symbols of a package to become accessible to another package without the use of package qualifiers.
SYMBOL key	See modifier keys.
syntax	The set of rules that govern the legal order of words and the meaning of special characters within a language. For Lisp, the **read** function reads a character from an input stream and interprets it, and possibly subsequent characters, depending on the type of the character.
system	1. The hardware and software that make up a single Explorer system. This overall system includes many standard utilities, including the Lisp Listener, the Zmacs editor, the Inspector, and so on. An Explorer system can also include optional toolkits such as the Relational Table Management System (RTMS). You can access most utilities in an Explorer system by selecting an item from the System menu or by pressing the SYSTEM key and a letter. You can display a list of the keys used to invoke utilities on your system by pressing SYSTEM HELP. 2. A collection of files that comprise a program or set of programs. Once a system is defined, the system definition facility handles the recompilation and loading of the files that make up the system.
system band	See load partition.
system console	See console.

system definition	A means to divide a large program into several files in order to keep the parts of the program organized. This makes items easier to find and results in smaller portions of code to edit and compile. The system facility, which uses the **defsystem** macro, allows you to specify which files make up the system, which ones depend on the presence of others, and so on. The system definition resides in its own file, which you load to inform the Lisp environment about your system and what files are in it. You can then use the **make-system** function to load all the files of the system, recompile all the files that need compiling, and so on.
system enclosure	A compact enclosure that houses the system logic boards, the backplane, power supplies, and PCA boards and cable connections to peripheral devices.
system interface board	A logic board installed in the system enclosure that provides an interface to the display unit, graphics control logic, both parallel and serial I/O ports, and various other system resources, such as clocks, timers, nonvolatile memory, and power failure event logic.
SYSTEM key	See information and status keys.
System menu	A top-level menu that can be invoked from any system utility by clicking the right mouse button once or twice. The System menu allows you to invoke and manipulate windows on the screen layout.
system partition	See load partition.
system reset	The complete initialization of the system, either when the system first powers up or when you press the keystroke sequence META-CTRL-META-CTRL-ABORT. During a system reset, the system performs hardware self-tests; loads the microcode contained in the microload partition; loads the load band, which destroys any contents of virtual memory; initializes a file system; performs other tasks defined in the initialization lists; and invokes a program, by default a Lisp Listener. At system reset, you have the option of specifying the device, load band, and microload band from which the system boots. In contrast to resetting the system, cold booting reloads the load band and microload band that you use when you reset but does not perform any hardware tests. Menu booting offers you the same options for booting as resetting the system does but, again, does not perform hardware tests. Both cold booting and menu booting reinitialize virtual memory. Warm booting resets and restarts all processes but does not destroy the contents of virtual memory or edited buffers.

t

t	A symbol used to represent the Boolean value true when no other value is more appropriate.
TAB key	See line-positioning keys and number pad.
tab-stop buffer	A Zmacs buffer that allows you to set tab stops.
tag	Analogous to a statement label in other languages like Fortran and PL/1. A tag enables the goto construct to be performed in Lisp.

Telnet	A facility that allows you to use its window as a terminal to another host. Telnet sends characters typed on the keyboard to the remote host's Telnet server.
temp-locking	Temporarily locks an exposed window so that output cannot be sent to it while a temporary window is exposed. When a window is covered—or partially covered—by a temporary window such as the System menu, output sent to the exposed window overwrites the temporary window; therefore, the exposed window is temp-locked until the temporary window disappears.
temporary window	A window that appears on the display for a short period of time and goes away after you have performed an action to remove it. You must remove a temporary window before you can manipulate any windows that the temporary window is covering (either partially or fully). A temporary window does not deexpose the window(s) it covers up.
TERM key	See information and status keys.
test zone partition	A partition that contains data patterns used in diagnostic testing.
Text mode	See Zmacs major mode.
text scroll window	A window that provides a simple means of displaying and scrolling a number of lines of similar material. For example, editor windows and menus can scroll.
throw	A special form used with **catch** to perform nonlocal exits. A **throw** transfers control to a matching **catch** construct. The **throw** form need not be within the body of the **catch** form, but it must occur within the extent of the evaluation of the body of the **catch**.
toggle	To complement or switch back and forth between two states of a condition.
top-level	A level of control that is not contained within another level. For example, a top-level function is one that has no other callers.
top-level directory	See root directory.
top-level loop	The highest level of control in a process, consisting of an endless read-eval-print loop that reads a form, evaluates it, and prints the results.
trace	A facility that allows you to examine certain functions. When a function is traced, certain special actions are taken when it is called and when it returns. The default tracing action is to print a message when the function is called, showing its name and arguments, and another message when the function returns, showing its name and value(s).
tracking the mouse	A term that describes how the mouse process examines the hardware mouse interface, notes how the mouse is moving, and adjusts the mouse cursor position and the mouse blinker accordingly.
transceiver	See Ethernet transceiver.
transformation	An operation, such as compiling or loading, that is to be performed on a set of files that comprise a system. The system definition facility has a number of options for specifying the transformations or the conditions for transformations to occur.

trap	A special software subroutine activated whenever a condition has been met that is not part of normal program execution. This subroutine usually halts execution immediately after recording certain information, such as the location of the program counter or the contents of various status registers.
tree editor	A tool that graphically displays any tree-structured object. To use the tree editor for a specific application, you write simple interface routines that enable it to traverse the tree structure, to handle node selection, and to provide editing functions on the tree structure or on the data in the nodes.
true list	See list.
type specifier	See data type specifier.
type-ahead buffer	A special buffer that stores keystrokes as you enter them. The buffer accommodates speedy typists, allowing you to continue typing even though the system cannot read and evaluate your input as quickly as you enter it. The system does not evaluate your input (echo it on the display) until you have stopped typing.
typeout window	A window that prints output, usually in response to individual commands, from other windows. A typeout window exposes itself when output is directed to it. The typeout window is an inferior of the window from which the output comes.

u

UCL	See Universal Command Loop.
unbind	To make a bound variable have either no value or the value it was formerly bound to if it was temporarily bound.
unbound symbol	A symbol that has no value; its value cell is empty.
UNDO key	See user interaction keys.
uninterned symbol	A symbol that does not belong to a package; the symbol's package cell contains **nil**. Typically, uninterned symbols are used simply as data objects.
uninterned symbol qualifier	A symbol whose package cell contains **nil** (that is, the symbol has no home package); when printed, the symbol is preceded by #:.
uninterning a symbol	To cause a symbol to belong to no package.
Universal Command Loop	A facility for constructing a command loop that accepts various forms of user input (menu selection with the mouse, keystroke sequence, or typed command name) and interprets the input as requests for command execution. It provides structures for command definitions and command tables, as well as a number of help facilities for the interface.
universal time	A standard way of indicating time that counts the number of seconds since midnight Greenwich Mean Time on January 1, 1900.
unnamed function	See lambda expression.

unnamed structure	An explicitly specified structure that, when defined by **defstruct**, does not use the **:named** option when using the **:type** option. This results in a structure that cannot be referred to by name.
unreduced ratio	A ratio number that is in a form such that further factoring with the denominator and numerator of the number is possible.
unsticky minor mode	See Zmacs minor mode.
update attribute list	To permanently record the current settings of the buffer attributes in the attribute list of a Zmacs buffer (that is, buffer attributes that are changed but not recorded in the attribute list, such as changing modes with a mode command). The changes are also saved when the file is written to disk. This operation is the inverse of the reparse attribute list operation, which records editing changes made to the attribute list and makes them take effect.
update operation	Any operation that alters the value of a generalized variable.
user directory	A directory that contains an arbitrary number (0 through *n*) of other user directories (called subdirectories) and files. It can also contain nothing (called an empty directory).
user interaction keys	The set of keys that enable you to modify the contents of the video display or to leave certain programs. These keys are the following: ■ CLEAR INPUT — Erases the characters that you have typed. Depending on the software, this key erases the characters on the current line or all the characters typed since the last complete function, or it redisplays a window obscured by an error handler. ■ CLEAR SCREEN — Clears the contents of the selected window. Zmacs refers to this as the PAGE key. ■ END — Depending on the software, this key is used to terminate a command, conversation, or application. ■ ESCAPE — Depending on the software, this key is used to escape from search mode. In Zmacs, this key is used with modifier keys to invoke command completion. ■ UNDO — Depending on the software, this key reverses an action taken.
user variable	See Zmacs user variable.
usurping the mouse	The act of setting aside the mouse process and explicitly doing all the mouse tracking. The mouse blinker is turned off, so the usurping process must provide a mouse blinker if a visual indicator is necessary.

V

value cell	A component of a symbol that contains the symbol's current value or binding. This is the value returned when the symbol is evaluated, and it can refer to any Lisp object. If the value cell is empty, the symbol is unbound.

vanilla flavor	A base flavor that, unless otherwise specified, is automatically included in every defined flavor. The **vanilla** flavor has no instance variables but provides methods useful for all flavors.
variable	An abstract object that signifies any actual value. In Lisp, variables are implemented as symbols.
vector	A one-dimensional array.
version number	1. A number associated with a file or system definition that indicates a sequence of updates. Each time a file is edited and saved, a new copy of the file is made with its version number incremented by 1. (You can also specify a version number not in sequence.) 2. The version number of a system definition consists of two parts: the minor version number and the major version number. The minor version number is increased every time a new patch is made to the system. The major version number is increased when the system is completely recompiled and the minor version number is reset to 0.
vertical index	A row subscript for a two-dimensional array. On the Explorer system, this is the first subscript.
vertical spacing	See VSP.
view	To cause Zmacs to display the contents of a file without reading it into a buffer first. You cannot edit a file while you are viewing it. See also visit.
virtual address space	The total amount of memory that can be addressed by the system. In the Explorer system, this amount is 128 megabytes.
virtual window	A window that is—or may be—too large to be displayed within an actual window. A virtual window is displayed in portions by a scrolling window.
visible window	A window that is currently displayed on a screen. A window is fully visible if no other window is covering any part of it; a window is partially visible if any other windows are covering part of it.
visit	To cause Zmacs to copy a file into a buffer so that you can edit the file. You work on the buffer as if it were the file. See also view.
VSP	The number of blank rows of pixels between lines of text in a window. VSP stands for vertical spacing.
VT100™ emulator	A window-based facility that allows you to use your Explorer monitor and keyboard as a VT100 terminal. When you are using the VT100 emulator, all alphabetic and nonalphabetic keys, cursor control keys, and most function keys transmit the VT100 key codes to a connected host.

VT100 is a trademark of Digital Equipment Corporation.

W

warm boot	The resetting and restarting of all processes in the system. Warm booting retains, unchanged, the contents of virtual memory and edited buffers. You can warm boot by pressing META-CTRL-META-CTRL-RETURN. See also cold boot, menu boot, and system reset.
warping the mouse	A term that describes how the program moves the mouse cursor and changes the logical position of the mouse. For example, double clicking left in the editor warps the mouse to where the editor cursor is currently located. Since there is no fixed association between positions of the physical mouse on an optical pad and positions of the mouse cursor on the screen, warping the mouse does not result in any inconsistency.
whitespace	One of the following characters: space, tab, return, linefeed, page, and newline.
who-line	A term previously used to refer to the combination of both the mouse documentation window and the status line.
wildcard	A special character (*) that can be substituted either for a missing pathname component or for missing characters in a pathname component.
window	A rectangular region that can be displayed on a screen. A window is associated with one process and acts as a stream, providing a place for the process to write to and read from. Each window is an instance of a specific flavor. The window's flavor governs how the window and process interact.
window status	A condition that indicates whether a window is exposed or selected.
window system	A collection of software that manages the display of various processes and directs input from the keyboard or mouse to the appropriate process.
window-based debugger	A window-oriented version of the debugger that includes several panes, allowing you to view different types of information at one time.
word	1. In the hardware of the Explorer system, a 32-bit memory cell. 2. A group of contiguous letters containing no blanks. Zmacs contains many commands that operate on the word level.
word abbreviation	A single word that Zmacs automatically expands after you type the abbreviation and a space (or any character that is not a letter or a number).
world load	See load partition.
wrapper	A construct dealing with flavors that allows you to implement advanced method accessing control. This construct can surround (or wrap around) a flavor's methods and allows specific operations to occur before and after the execution of any of the flavor's methods.

X

xfasl	On the Explorer system, the file type used for software releases prior to 3.0 into which compiled code is loaded. The x refers to Explorer system source code, and fasl stands for fast load.
xld	The file type of compiled code on releases of Explorer software 3.0 and later. The x refers to the Explorer system and ld stands for loadable code.

Y

yank	To retrieve an entry from the kill history.

Z

zero-origin indexing	A type of indexing for arrays or sequences in which the first index starts with 0.
Zetalisp	A dialect of Lisp developed at Massachusetts Institute of Technology (M.I.T.) that contributed significantly to the definition of Common Lisp. Zetalisp includes most of the features of Common Lisp, with some differences in syntax and functionality, as well as other features such as the flavor system and the **loop** macro. Zetalisp is fully supported on the Explorer system.
Zetalisp mode	See Zmacs major mode.
Zmacs	A real-time, window-oriented editor that provides extensive support for writing Lisp programs as well as other types of text. Zmacs is based on the M.I.T. Emacs editor and includes hundreds of features for manipulating text and supporting program development.
Zmacs buffer	See buffer.
Zmacs buffer history	A listing produced by the List Buffers command in Zmacs that lists all the buffers in their most recently used (not created) order.
Zmacs editor buffer window	The upper section of the Zmacs display in which you edit.
Zmacs major mode	A mode of operation that tells Zmacs what type of file is being edited. A major mode has its own distinct set of key bindings, indentation, and delimiters. Only one major mode can be in effect for a given buffer at a time. Some of the major modes available are the following: ■ Text — Designed for editing text. ■ Common Lisp — Designed for editing Common Lisp code. ■ Zetalisp — Designed for editing Zetalisp code.

Zmacs minibuffer A small window in Zmacs below the editor buffer window, where prompts for Zmacs commands appear and where you type responses to prompts.

Zmacs minor mode In Zmacs, a mode of operation that adds a specific customization to the way Zmacs behaves. Minor modes are optional and independent of the major mode in effect. You can have more than one minor mode active at a time. Some of the minor modes available are the following:

- Auto Fill — Zmacs automatically fills text. You can type text without worrying about the width of the Zmacs window. Return characters are inserted when needed to break lines.
- Electric Font Lock — Zmacs puts text in the first font, comments in the second font, documentations strings in the third font, other strings in the fourth font, and function specifications in the fifth font. Font IDs are read from the file attribute list.
- Electric Shift Lock — Zmacs uppercases everything except comments and strings.
- Overwrite — Typing over existing characters replaces them.
- RETURN Indents — The RETURN key indents, and the LINE FEED key does not.
- Word Abbreviation — Zmacs expands word abbreviations.

A minor mode can be either sticky or unsticky. A sticky minor mode remains active when you change major mode or go to another buffer. An unsticky minor mode does not remain active. Also, if you reparse the attribute list (to record a new major mode setting), the sticky minor modes remain active and the unsticky ones do not.

Zmacs user variable A type of variable in Zmacs that the user can easily access and change to customize some of the workings of Zmacs. Zmacs contains two types of user variables: global and local. A global user variable applies to all buffers. A local user variable applies to only one buffer.

zoom To change a window so that the scale of graphics objects that are displayed appears to change. Zooming in makes objects appear larger; zooming out makes objects appear smaller.

Data Systems Group – Austin
Documentation Questionnaire

Explorer Glossary

Do you use other TI manuals? If so, which one(s)?

______________________________ ______________________________

______________________________ ______________________________

______________________________ ______________________________

How would you rate the quality of our manuals?

	Excellent	Good	Fair	Poor
Accuracy	____	____	____	____
Organization	____	____	____	____
Clarity	____	____	____	____
Completeness	____	____	____	____
Overall design	____	____	____	____
Size	____	____	____	____
Illustrations	____	____	____	____
Examples	____	____	____	____
Index	____	____	____	____
Binding method	____	____	____	____

Was the quality of documentation a criterion in your selection of hardware or software?

☐ Yes ☐ No

How do you find the technical level of our manuals?

☐ Written for a more experienced user than yourself

☐ Written for a user with the same experience

☐ Written for a less experienced user than yourself

What is your experience using computers?

☐ Less than 1 year ☐ 1-5 years ☐ 5-10 years ☐ Over 10 years

We appreciate your taking the time to complete this questionnaire. If you have additional comments about the quality of our manuals, please write them in the space below. Please be specific.

____

____

____

____

____

Name ______________________ Title/Occupation ______________________

Company Name ____

Address ______________________ City/State/Zip ______________________

Telephone ______________________ Date ______________________

Manual Part No. 2243134-0001 *A

TAPE EDGE TO SEAL

FOLD

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST-CLASS PERMIT NO. 7284 DALLAS, TX

POSTAGE WILL BE PAID BY ADDRESSEE

TEXAS INSTRUMENTS INCORPORATED
DATA SYSTEMS GROUP
ATTN: PC SUSTAINING
P.O. Box 2909 M/S 2234
Austin, Texas 78769-9990

FOLD