AutoCAD®

autodesk®

DXF Reference

2004

February 2003

1 2 3 4 5 6 7 8 9 10

Contents

DXF Format

The DXF™ format is a tagged data representation of all the information contained in an AutoCAD® drawing file. *Tagged data* means that each data element in the file is preceded by an integer number that is called a *group code*. A group code's value indicates what type of data element follows. This value also indicates the meaning of a data element for a given object (or record) type. Virtually all user-specified information in a drawing file can be represented in DXF format.

In this chapter

Organization of This Reference

The *DXF Reference* presents the DXF™ group codes found in DXF files and encountered by AutoLISP® and ObjectARX™ applications. This chapter describes the general DXF conventions. The remaining chapters list the group codes organized by object type. The group codes are presented in the order in which they are found in a DXF file, and each chapter is named according to the associated section of a DXF file. Although the DXF file format is used as the organizing mechanism for this reference, specific information on the actual formatting of DXF files is found in "Drawing Interchange File Formats." Advanced concepts relating to DXF group codes as they pertain to both applications and DXF files are found in "Advanced DXF Issues."

For descriptions of the AutoLISP functions that use group codes, see "Using AutoLISP to Manipulate AutoCAD Objects," in the *AutoLISP Developer's Guide*.

Revisions to the DXF Reference

This topic lists revisions since the last update of the *DXF Reference*. The version number of this *DXF Reference* is u18.1.01.

Header Section
Variables with group codes and descriptions have been added to the "HEADER Section Group Codes."

Formatting Conventions in This Reference

Each group code listed in this reference is presented by a numeric group code value and a description. All group codes can apply to DXF™ files, applications (AutoLISP or ObjectARX), or both. When the description of a code is different for applications and DXF files (or applies to only one or the other), the description is preceded by the following indicators:

- **APP.** Application-specific description.
- **DXF.** DXF file-specific description.

If the description is common to both DXF files and applications, no indicator is provided.

Optional codes are indicated as "optional" in the description.

Object and Entity Codes

In the DXF™ format, the definition of objects differs from entities: objects have no graphical representation and entities do. For example, dictionaries are objects, and not entities. Entities are also referred to as *graphical objects* while objects are referred to as *nongraphical objects*.

Entities appear in both the BLOCK and ENTITIES sections of the DXF file. The use of group codes in the two sections is identical.

Some group codes that define an entity always appear; others are optional and appear only if their values differ from the defaults.

Do not write programs that rely on the order given here. The end of an entity is indicated by the next 0 group, which begins the next entity or indicates the end of the section.

Note Accommodating DXF files from future releases of AutoCAD® will be easier if you write your DXF processing program in a table-driven way, ignore undefined group codes, and make no assumptions about the order of group codes in an entity. With each new AutoCAD release, new group codes will be added to entities to accommodate additional features.

Group Code Value Types

Group codes define the type of the associated value as an integer, a floating-point number, or a string, according to the following table of group code ranges. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Group code value types

Code range	Group value type
0–9	String (with the introduction of extended symbol names in AutoCAD 2000, the 255-character limit has been lifted. There is no explicit limit to the number of bytes per line, although most lines should fall within 2049 bytes)
10–39	Double precision 3D point value
40–59	Double-precision floating-point value

Group code value types (*continued*)

Code range	Group value type
60–79	16-bit integer value
90–99	32-bit integer value
100	String (255-character maximum; less for Unicode strings)
102	String (255-character maximum; less for Unicode strings)
105	String representing hexadecimal (hex) handle value
110–119	Double precision floating-point value
120–129	Double precision floating-point value
130–139	Double precision floating-point value
140–149	Double precision scalar floating-point value
170–179	16-bit integer value
210–239	Double-precision floating-point value
270–279	16-bit integer value
280–289	16-bit integer value
290–299	Boolean flag value
300–309	Arbitrary text string
310–319	String representing hex value of binary chunk
320–329	String representing hex handle value
330–369	String representing hex object IDs
370–379	16-bit integer value
380–389	16-bit integer value
390–399	String representing hex handle value
400–409	16-bit integer value
410–419	String

Group code value types (*continued*)	
Code range	**Group value type**
420-429	32-bit integer value
430-439	String
440-449	32-bit integer value
450-459	Long
460-469	Double-precision floating-point value
470-479	String
999	Comment (string)
1000–1009	String (same limits as indicated with 0–9 code range)
1010–1059	Double-precision floating-point value
1060–1070	16-bit integer value
1071	32-bit integer value

Group Codes in Numerical Order

The following table gives the group code or group code range accompanied by an explanation of the group code value. In the table, "fixed" indicates that the group code always has the same purpose. If a group code isn't fixed, its purpose depends on the context. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Group codes by number	
Group code	**Description**
–5	APP: persistent reactor chain
–4	APP: conditional operator (used *only* with **`ssget`**)
–3	APP: extended data (XDATA) sentinel (fixed)
–2	APP: entity name reference (fixed)

Group codes by number (*continued*)

Group code	Description
–1	APP: entity name. The name changes each time a drawing is opened. It is never saved (fixed)
0	Text string indicating the entity type (fixed)
1	Primary text value for an entity
2	Name (attribute tag, block name, and so on)
3–4	Other text or name values
5	Entity handle; text string of up to 16 hexadecimal digits (fixed)
6	Linetype name (fixed)
7	Text style name (fixed)
8	Layer name (fixed)
9	DXF: variable name identifier (used only in HEADER section of the DXF file)
10	Primary point; this is the start point of a line or text entity, center of a circle, and so on DXF: *X* value of the primary point (followed by *Y* and *Z* value codes 20 and 30) APP: 3D point (list of three reals)
11–18	Other points DXF: *X* value of other points (followed by *Y* value codes 21–28 and *Z* value codes 31–38) APP: 3D point (list of three reals)
20, 30	DXF™*Y* and *Z* values of the primary point
21–28, 31–37	DXF: *Y* and *Z* values of other points
38	DXF: entity's elevation if nonzero
39	Entity's thickness if nonzero (fixed)
40–48	Double-precision floating-point values (text height, scale factors, and so on)
48	Linetype scale; double precision floating point scalar value; default value is defined for all entity types

Group codes by number (*continued*)	
Group code	**Description**
49	Repeated double-precision floating-point value. Multiple 49 groups may appear in one entity for variable-length tables (such as the dash lengths in the LTYPE table). A 7*x* group always appears *before* the first 49 group to specify the table length
50–58	Angles (output in degrees to DXF files and radians through AutoLISP and ObjectARX applications)
60	Entity visibility; integer value; absence or 0 indicates visibility; 1 indicates invisibility
62	Color number (fixed)
66	"Entities follow" flag (fixed)
67	Space—that is, model or paper space (fixed)
68	APP: identifies whether viewport is on but fully off screen; is not active or is off
69	APP: viewport identification number
70–78	Integer values, such as repeat counts, flag bits, or modes
90–99	32-bit integer values
100	Subclass data marker (with derived class name as a string). Required for all objects and entity classes that are derived from another concrete class. The subclass data marker segregates data defined by different classes in the inheritance chain for the same object. This is in addition to the requirement for DXF names for each distinct concrete class derived from ObjectARX (see "Subclass Markers" on page 176)
102	Control string, followed by "{<arbitrary name>" or "}". Similar to the xdata 1002 group code, except that when the string begins with "{", it can be followed by an arbitrary string whose interpretation is up to the application. The only other control string allowed is "}" as a group terminator. AutoCAD does not interpret these strings except during drawing audit operations. They are for application use
105	Object handle for DIMVAR symbol table entry
110	UCS origin (appears only if code 72 is set to 1) DXF: *X* value; APP: 3D point
111	UCS *X*-axis (appears only if code 72 is set to 1) DXF: *X* value; APP: 3D vector

Group codes by number (*continued*)

Group code	Description
112	UCS *Y*-axis (appears only if code 72 is set to 1) DXF: *X* value; APP: 3D vector
120–122	DXF: *Y* value of UCS origin, UCS *X*-axis, and UCS *Y*-axis
130–132	DXF: Z value of UCS origin, UCS *X*-axis, and UCS *Y*-axis
140–149	Double-precision floating-point values (points, elevation, and DIMSTYLE settings, for example)
170–179	16-bit integer values, such as flag bits representing DIMSTYLE settings
210	Extrusion direction (fixed) DXF: *X* value of extrusion direction APP: 3D extrusion direction vector
220, 230	DXF: *Y* and *Z* values of the extrusion direction
270–279	16-bit integer values
280–289	16-bit integer values
290–299	Boolean flag value
300–309	Arbitrary text strings
310–319	Arbitrary binary chunks with same representation and limits as 1004 group codes: hexadecimal strings of up to 254 characters represent data chunks of up to 127 bytes
320–329	Arbitrary object handles; handle values that are taken "as is." They are not translated during INSERT and XREF operations
330–339	Soft-pointer handle; arbitrary soft pointers to other objects within same DXF file or drawing. Translated during INSERT and XREF operations
340–349	Hard-pointer handle; arbitrary hard pointers to other objects within same DXF file or drawing. Translated during INSERT and XREF operations
350–359	Soft-owner handle; arbitrary soft ownership links to other objects within same DXF file or drawing. Translated during INSERT and XREF operations
360–369	Hard-owner handle; arbitrary hard ownership links to other objects within same DXF file or drawing. Translated during INSERT and XREF operations

Group codes by number (*continued*)

Group code	Description
370–379	Lineweight enum value (AcDb::LineWeight). Stored and moved around as a 16-bit integer. Custom non-entity objects may use the full range, but entity classes only use 371–379 DXF group codes in their representation, because AutoCAD® and AutoLISP both always assume a 370 group code is the entity's lineweight. This allows 370 to behave like other "common" entity fields
380–389	PlotStyleName type enum (AcDb::PlotStyleNameType). Stored and moved around as a 16-bit integer. Custom non-entity objects may use the full range, but entity classes only use 381–389 DXF group codes in their representation, for the same reason as the Lineweight range above
390–399	String representing handle value of the PlotStyleName object, basically a hard pointer, but has a different range to make backward compatibility easier to deal with. Stored and moved around as an object ID (a handle in DXF files) and a special type in AutoLISP. Custom non-entity objects may use the full range, but entity classes only use 391–399 DXF group codes in their representation, for the same reason as the lineweight range above
400–409	16-bit integers
410–419	String
420-427	32-bit integer value. When used with True Color; a 32-bit integer representing a 24-bit color value. The high-order byte (8 bits) is 0, the low-order byte an unsigned char holding the Blue value (0-255), then the Green value, and the next-to-high order byte is the Red Value. Convering this integer value to hexadecimal yields the following bit mask: 0x00RRGGBB. For example, a true color with Red==200, Green==100 and Blue==50 is 0x00C86432, and in DXF, in decimal, 13132850
430-437	String; when used for True Color, a string representing the name of the color
440-447	32-bit integer value. When used for True Color, the transparency value
450-459	Long
460-469	Double-precision floating-point value
470-479	String

Group codes by number (*continued*)

Group code	Description
999	DXF: The 999 group code indicates that the line following it is a comment string. SAVEAS does not include such groups in a DXF output file, but OPEN honors them and ignores the comments. You can use the 999 group to include comments in a DXF file that you've edited
1000	ASCII string (up to 255 bytes long) in extended data
1001	Registered application name (ASCII string up to 31 bytes long) for extended data
1002	Extended data control string ("{" or "}")
1003	Extended data layer name
1004	Chunk of bytes (up to 127 bytes long) in extended data
1005	Entity handle in extended data; text string of up to 16 hexadecimal digits
1010	A point in extended data DXF: *X* value (followed by 1020 and 1030 groups) APP: 3D point
1020, 1030	DXF: *Y* and *Z* values of a point
1011	A 3D world space position in extended data DXF: *X* value (followed by 1021 and 1031 groups) APP: 3D point
1021, 1031	DXF: *Y* and *Z* values of a world space position
1012	A 3D world space displacement in extended data DXF: *X* value (followed by 1022 and 1032 groups) APP: 3D vector
1022, 1032	DXF: *Y* and *Z* values of a world space displacement
1013	A 3D world space direction in extended data DXF: *X* value (followed by 1022 and 1032 groups) APP: 3D vector
1023, 1033	DXF: *Y* and *Z* values of a world space direction
1040	Extended data double-precision floating-point value
1041	Extended data distance value

Group codes by number (*continued*)	
Group code	**Description**
1042	Extended data scale factor
1070	Extended data 16-bit signed integer
1071	Extended data 32-bit signed long

HEADER Section

2

The group codes described in this chapter pertain only to DXF™ files. The HEADER section of a DXF file contains the settings of variables associated with the drawing. Each variable is specified by a 9 group code giving the variable's name, followed by groups that supply the variable's value. This chapter lists only the variables that are saved in the drawing file.

In this chapter

HEADER Section Group Codes

The following table lists the variables that are represented in the HEADER section of a DXF™ file. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

DXF header variables

Variable	Group code	Description
$ACADMAINTVER	70	Maintenance version number (should be ignored)
$ACADVER	1	The AutoCAD® drawing database version number: AC1006 = R10; AC1009 = R11 and R12; AC1012 = R13; AC1014 = R14; AC1015 = AutoCAD 2000; AC1018 = AutoCAD 2004
$ANGBASE	50	Angle 0 direction
$ANGDIR	70	1 = Clockwise angles 0 = Counterclockwise angles
$ATTMODE	70	Attribute visibility: 0 = None 1 = Normal 2 = All
$AUNITS	70	Units format for angles
$AUPREC	70	Units precision for angles
$CECOLOR	62	Current entity color number: 0 = BYBLOCK; 256 = BYLAYER
$CELTSCALE	40	Current entity linetype scale
$CELTYPE	6	Entity linetype name, or BYBLOCK or BYLAYER
$CELWEIGHT	370	Lineweight of new objects
$CEPSNID	390	Plotstyle handle of new objects; if CEPSNTYPE is 3, then this value indicates the handle
$CEPSNTYPE	380	Plot style type of new objects: 0 = Plot style by layer 1 = Plot style by block 2 = Plot style by dictionary default 3 = Plot style by object ID/handle

DXF header variables (*continued*)		
Variable	**Group code**	**Description**
$CHAMFERA	40	First chamfer distance
$CHAMFERB	40	Second chamfer distance
$CHAMFERC	40	Chamfer length
$CHAMFERD	40	Chamfer angle
$CLAYER	8	Current layer name
$CMLJUST	70	Current multiline justification: 0 = Top; 1 = Middle; 2 = Bottom
$CMLSCALE	40	Current multiline scale
$CMLSTYLE	2	Current multiline style name
$DIMADEC	70	Number of precision places displayed in angular dimensions
$DIMALT	70	Alternate unit dimensioning performed if nonzero
$DIMALTD	70	Alternate unit decimal places
$DIMALTF	40	Alternate unit scale factor
$DIMALTRND	40	Determines rounding of alternate units
$DIMALTTD	70	Number of decimal places for tolerance values of an alternate units dimension
$DIMALTTZ	70	Controls suppression of zeros for alternate tolerance values: 0 = Suppresses zero feet and precisely zero inches 1 = Includes zero feet and precisely zero inches 2 = Includes zero feet and suppresses zero inches 3 = Includes zero inches and suppresses zero feet
$DIMALTU	70	Units format for alternate units of all dimension style family members except angular: 1 = Scientific; 2 = Decimal; 3 = Engineering; 4 = Architectural (stacked); 5 = Fractional (stacked); 6 = Architectural; 7 = Fractional

DXF header variables (*continued*)

Variable	Group code	Description
$DIMALTZ	70	Controls suppression of zeros for alternate unit dimension values: 0 = Suppresses zero feet and precisely zero inches 1 = Includes zero feet and precisely zero inches 2 = Includes zero feet and suppresses zero inches 3 = Includes zero inches and suppresses zero feet
$DIMAPOST	1	Alternate dimensioning suffix
$DIMASO	70	1 = Create associative dimensioning 0 = Draw individual entities
$DIMASSOC	280	Controls the associativity of dimension objects 0 = Creates exploded dimensions; there is no association between elements of the dimension, and the lines, arcs, arrowheads, and text of a dimension are drawn as separate objects 1 = Creates non-associative dimension objects; the elements of the dimension are formed into a single object, and if the definition point on the object moves, then the dimension value is updated 2 = Creates associative dimension objects; the elements of the dimension are formed into a single object and one or more definition points of the dimension are coupled with association points on geometric objects
$DIMASZ	40	Dimensioning arrow size
$DIMATFIT	70	Controls dimension text and arrow placement when space is not sufficient to place both within the extension lines: 0 = Places both text and arrows outside extension lines 1 = Moves arrows first, then text 2 = Moves text first, then arrows 3 = Moves either text or arrows, whichever fits best AutoCAD adds a leader to moved dimension text when DIMTMOVE is set to 1
$DIMAUNIT	70	Angle format for angular dimensions: 0 = Decimal degrees; 1 = Degrees/minutes/seconds; 2 = Gradians; 3 = Radians; 4 = Surveyor's units
$DIMAZIN	70	Controls suppression of zeros for angular dimensions: 0 = Displays all leading and trailing zeros 1 = Suppresses leading zeros in decimal dimensions 2 = Suppresses trailing zeros in decimal dimensions 3 = Suppresses leading and trailing zeros
$DIMBLK	1	Arrow block name

DXF header variables (*continued*)

Variable	Group code	Description
$DIMBLK1	1	First arrow block name
$DIMBLK2	1	Second arrow block name
$DIMCEN	40	Size of center mark/lines
$DIMCLRD	70	Dimension line color: range is 0 = BYBLOCK; 256 = BYLAYER
$DIMCLRE	70	Dimension extension line color: range is 0 = BYBLOCK; 256 = BYLAYER
$DIMCLRT	70	Dimension text color: range is 0 = BYBLOCK; 256 = BYLAYER
$DIMDEC	70	Number of decimal places for the tolerance values of a primary units dimension
$DIMDLE	40	Dimension line extension
$DIMDLI	40	Dimension line increment
$DIMDSEP	70	Single-character decimal separator used when creating dimensions whose unit format is decimal
$DIMEXE	40	Extension line extension
$DIMEXO	40	Extension line offset
$DIMFAC	40	Scale factor used to calculate the height of text for dimension fractions and tolerances. AutoCAD multiplies DIMTXT by DIMTFAC to set the fractional or tolerance text height
$DIMGAP	40	Dimension line gap
$DIMJUST	70	Horizontal dimension text position: 0 = Above dimension line and center-justified between extension lines 1 = Above dimension line and next to first extension line 2 = Above dimension line and next to second extension line 3 = Above and center-justified to first extension line 4 = Above and center-justified to second extension line
$DIMLDRBLK	1	Arrow block name for leaders
$DIMLFAC	40	Linear measurements scale factor

DXF header variables (*continued*)

Variable	Group code	Description
$DIMLIM	70	Dimension limits generated if nonzero
$DIMLUNIT	70	Sets units for all dimension types except Angular: 1 = Scientific; 2 = Decimal; 3 = Engineering; 4 = Architectural; 5 = Fractional; 6 = Windows desktop
$DIMLWD	70	Dimension line lineweight: –3 = Standard –2 = ByLayer –1 = ByBlock 0–211 = an integer representing 100th of mm
$DIMLWE	70	Extension line lineweight: –3 = Standard –2 = ByLayer –1 = ByBlock 0–211 = an integer representing 100th of mm
$DIMPOST	1	General dimensioning suffix
$DIMRND	40	Rounding value for dimension distances
$DIMSAH	70	Use separate arrow blocks if nonzero
$DIMSCALE	40	Overall dimensioning scale factor
$DIMSD1	70	Suppression of first extension line: 0 = Not suppressed; 1 = Suppressed
$DIMSD2	70	Suppression of second extension line: 0 = Not suppressed; 1 = Suppressed
$DIMSE1	70	First extension line suppressed if nonzero
$DIMSE2	70	Second extension line suppressed if nonzero
$DIMSHO	70	1 = Recompute dimensions while dragging 0 = Drag original image
$DIMSOXD	70	Suppress outside-extensions dimension lines if nonzero
$DIMSTYLE	2	Dimension style name
$DIMTAD	70	Text above dimension line if nonzero
$DIMTDEC	70	Number of decimal places to display the tolerance values

DXF header variables (*continued*)

Variable	Group code	Description
$DIMTFAC	40	Dimension tolerance display scale factor
$DIMTIH	70	Text inside horizontal if nonzero
$DIMTIX	70	Force text inside extensions if nonzero
$DIMTM	40	Minus tolerance
$DIMTMOVE	70	Dimension text movement rules: 0 = Moves the dimension line with dimension text 1 = Adds a leader when dimension text is moved 2 = Allows text to be moved freely without a leader
$DIMTOFL	70	If text is outside extensions, force line extensions between extensions if nonzero
$DIMTOH	70	Text outside horizontal if nonzero
$DIMTOL	70	Dimension tolerances generated if nonzero
$DIMTOLJ	70	Vertical justification for tolerance values: 0 = Top; 1 = Middle; 2 = Bottom
$DIMTP	40	Plus tolerance
$DIMTSZ	40	Dimensioning tick size: 0 = No ticks
$DIMTVP	40	Text vertical position
$DIMTXSTY	7	Dimension text style
$DIMTXT	40	Dimensioning text height
$DIMTZIN	70	Controls suppression of zeros for tolerance values: 0 = Suppresses zero feet and precisely zero inches 1 = Includes zero feet and precisely zero inches 2 = Includes zero feet and suppresses zero inches 3 = Includes zero inches and suppresses zero feet
$DIMUPT	70	Cursor functionality for user-positioned text: 0 = Controls only the dimension line location 1 = Controls the text position as well as the dimension line location

DXF header variables (*continued*)

Variable	Group code	Description
$DIMZIN	70	Controls suppression of zeros for primary unit values: 0 = Suppresses zero feet and precisely zero inches 1 = Includes zero feet and precisely zero inches 2 = Includes zero feet and suppresses zero inches 3 = Includes zero inches and suppresses zero feet
$DISPSILH	70	Controls the display of silhouette curves of body objects in Wireframe mode: 0 = Off; 1 = On
$DWGCODEPAGE	3	Drawing code page; set to the system code page when a new drawing is created, but not otherwise maintained by AutoCAD
$ELEVATION	40	Current elevation set by ELEV command
$ENDCAPS	280	Lineweight endcaps setting for new objects: 0 = none; 1 = round; 2 = angle; 3 = square
$EXTMAX	10, 20, 30	*X*, *Y*, and *Z* drawing extents upper-right corner (in WCS)
$EXTMIN	10, 20, 30	*X*, *Y*, and *Z* drawing extents lower-left corner (in WCS)
$EXTNAMES	290	Controls symbol table naming: 0 = Release 14 compatibility. Limits names to 31 characters in length. Names can include the letters A to Z, the numerals 0 to 9, and the special characters dollar sign ($), underscore (_), and hyphen (–). 1 = AutoCAD 2000. Names can be up to 255 characters in length, and can include the letters A to Z, the numerals 0 to 9, spaces, and any special characters not used for other purposes by Microsoft Windows and AutoCAD
$FILLETRAD	40	Fillet radius
$FILLMODE	70	Fill mode on if nonzero
$FINGERPRINTGUID	2	Set at creation time, uniquely identifies a particular drawing
$HALOGAP	280	Specifies a gap to be displayed where an object is hidden by another object; the value is specified as a percent of one unit and is independent of the zoom level. A haloed line is shortened at the point where it is hidden when HIDE or the Hidden option of SHADEMODE is used
$HANDSEED	5	Next available handle

DXF header variables (*continued*)

Variable	Group code	Description
$HIDETEXT	290	Specifies HIDETEXT system variable: 0 = HIDE ignores text objects when producing the hidden view 1 = HIDE does not ignore text objects
$HYPERLINKBASE	1	Path for all relative hyperlinks in the drawing. If null, the drawing path is used
$INDEXCTL	280	Controls whether layer and spatial indexes are created and saved in drawing files: 0 = No indexes are created 1 = Layer index is created 2 = Spatial index is created 3 = Layer and spatial indexes are created
$INSBASE	10, 20, 30	Insertion base set by BASE command (in WCS)
$INSUNITS	70	Default drawing units for AutoCAD DesignCenter blocks: 0 = Unitless; 1 = Inches; 2 = Feet; 3 = Miles; 4 = Millimeters; 5 = Centimeters; 6 = Meters; 7 = Kilometers; 8 = Microinches; 9 = Mils; 10 = Yards; 11 = Angstroms; 12 = Nanometers; 13 = Microns; 14 = Decimeters; 15 = Decameters; 16 = Hectometers; 17 = Gigameters; 18 = Astronomical units; 19 = Light years; 20 = Parsecs
$INTERSECTIONCOLOR	70	Specifies the entity color of intersection polylines: Values 1-255 designate an AutoCAD color index (ACI) 0 = Color BYBLOCK 256 = Color BYLAYER 257 = Color BYENTITY
$INTERSECTIONDISPLAY	290	Specifies the display of intersection polylines: 0 = Turns off the display of intersection polylines 1 = Turns on the display of intersection polylines
$JOINSTYLE	280	Lineweight joint setting for new objects: 0=none; 1= round; 2 = angle; 3 = flat
$LIMCHECK	70	Nonzero if limits checking is on
$LIMMAX	10, 20	*XY* drawing limits upper-right corner (in WCS)
$LIMMIN	10, 20	*XY* drawing limits lower-left corner (in WCS)
$LTSCALE	40	Global linetype scale
$LUNITS	70	Units format for coordinates and distances

DXF header variables (*continued*)

Variable	Group code	Description
$LUPREC	70	Units precision for coordinates and distances
$LWDISPLAY	290	Controls the display of lineweights on the Model or Layout tab: 0 = Lineweight is not displayed 1 = Lineweight is displayed
$MAXACTVP	70	Sets maximum number of viewports to be regenerated
$MEASUREMENT	70	Sets drawing units: 0 = English; 1 = Metric
$MENU	1	Name of menu file
$MIRRTEXT	70	Mirror text if nonzero
$OBSCOLOR	70	Specifies the color of obscured lines. An obscured line is a hidden line made visible by changing its color and linetype and is visible only when the HIDE or SHADEMODE command is used. The OBSCUREDCOLOR setting is visible only if the OBSCUREDLTYPE is turned ON by setting it to a value other than 0. 0 and 256 = Entity color 1-255 = An AutoCAD color index (ACI)
$OBSLTYPE	280	Specifies the linetype of obscured lines. Obscured linetypes are independent of zoom level, unlike regular AutoCAD linetypes. Value 0 turns off display of obscured lines and is the default. Linetype values are defined as follows: 0 = Off 1 = Solid 2 = Dashed 3 = Dotted 4 = Short Dash 5 = Medium Dash 6 = Long Dash 7 = Double Short Dash 8 = Double Medium Dash 9 = Double Long Dash 10 = Medium Long Dash 11 = Sparse Dot
$ORTHOMODE	70	Ortho mode on if nonzero
$PDMODE	70	Point display mode
$PDSIZE	40	Point display size
$PELEVATION	40	Current paper space elevation

DXF header variables (*continued*)

Variable	Group code	Description
$PEXTMAX	10, 20, 30	Maximum *X*, *Y*, and *Z* extents for paper space
$PEXTMIN	10, 20, 30	Minimum *X*, *Y*, and *Z* extents for paper space
$PINSBASE	10, 20, 30	Paper space insertion base point
$PLIMCHECK	70	Limits checking in paper space when nonzero
$PLIMMAX	10, 20	Maximum *X* and *Y* limits in paper space
$PLIMMIN	10, 20	Minimum *X* and *Y* limits in paper space
$PLINEGEN	70	Governs the generation of linetype patterns around the vertices of a 2D polyline: 1 = Linetype is generated in a continuous pattern around vertices of the polyline 0 = Each segment of the polyline starts and ends with a dash
$PLINEWID	40	Default polyline width
$PROJECTNAME	1	Assigns a project name to the current drawing. Used when an external reference or image is not found on its original path. The project name points to a section in the registry that can contain one or more search paths for each project name defined. Project names and their search directories are created from the Files tab of the Options dialog box
$PROXYGRAPHICS	70	Controls the saving of proxy object images
$PSLTSCALE	70	Controls paper space linetype scaling: 1 = No special linetype scaling 0 = Viewport scaling governs linetype scaling
$PSTYLEMODE	290	Indicates whether the current drawing is in a Color-Dependent or Named Plot Style mode: 0 =Uses named plot style tables in the current drawing 1 = Uses color-dependent plot style tables in the current drawing
$PSVPSCALE	40	View scale factor for new viewports: 0 = Scaled to fit >0 = Scale factor (a positive real value)
$PUCSBASE	2	Name of the UCS that defines the origin and orientation of orthographic UCS settings (paper space only)
$PUCSNAME	2	Current paper space UCS name

DXF header variables (*continued*)

Variable	Group code	Description
$PUCSORG	10, 20, 30	Current paper space UCS origin
$PUCSORGBACK	10, 20, 30	Point which becomes the new UCS origin after changing paper space UCS to BACK when PUCSBASE is set to WORLD
$PUCSORGBOTTOM	10, 20, 30	Point which becomes the new UCS origin after changing paper space UCS to BOTTOM when PUCSBASE is set to WORLD
$PUCSORGFRONT	10, 20, 30	Point which becomes the new UCS origin after changing paper space UCS to FRONT when PUCSBASE is set to WORLD
$PUCSORGLEFT	10, 20, 30	Point which becomes the new UCS origin after changing paper space UCS to LEFT when PUCSBASE is set to WORLD
$PUCSORGRIGHT	10, 20, 30	Point which becomes the new UCS origin after changing paper space UCS to RIGHT when PUCSBASE is set to WORLD
$PUCSORGTOP	10, 20, 30	Point which becomes the new UCS origin after changing paper space UCS to TOP when PUCSBASE is set to WORLD
$PUCSORTHOREF	2	If paper space UCS is orthographic (PUCSORTHOVIEW not equal to 0), this is the name of the UCS that the orthographic UCS is relative to. If blank, UCS is relative to WORLD
$PUCSORTHOVIEW	70	Orthographic view type of paper space UCS: 0 = UCS is not orthographic; 1 = Top; 2 = Bottom; 3 = Front; 4 = Back; 5 = Left; 6 = Right
$PUCSXDIR	10, 20, 30	Current paper space UCS *X* axis
$PUCSYDIR	10, 20, 30	Current paper space UCS *Y* axis
$QTEXTMODE	70	Quick Text mode on if nonzero
$REGENMODE	70	REGENAUTO mode on if nonzero
$SHADEDGE	70	0 = Faces shaded, edges not highlighted 1 = Faces shaded, edges highlighted in black 2 = Faces not filled, edges in entity color 3 = Faces in entity color, edges in black
$SHADEDIF	70	Percent ambient/diffuse light; range 1–100; default 70
$SKETCHINC	40	Sketch record increment

DXF header variables (*continued*)

Variable	Group code	Description
$SKPOLY	70	0 = Sketch lines; 1 = Sketch polylines
$SORTENTS	280	Controls the object sorting methods; accessible from the Options dialog box User Preferences tab. SORTENTS uses the following bitcodes: 0 = Disables SORTENTS 1 = Sorts for object selection 2 = Sorts for object snap 4 = Sorts for redraws 8 = Sorts for MSLIDE command slide creation 16 = Sorts for REGEN commands 32 = Sorts for plotting 64 = Sorts for PostScript output
$SPLFRAME	70	Spline control polygon display: 1 = On; 0 = Off
$SPLINESEGS	70	Number of line segments per spline patch
$SPLINETYPE	70	Spline curve type for PEDIT Spline
$SURFTAB1	70	Number of mesh tabulations in first direction
$SURFTAB2	70	Number of mesh tabulations in second direction
$SURFTYPE	70	Surface type for PEDIT Smooth
$SURFU	70	Surface density (for PEDIT Smooth) in *M* direction
$SURFV	70	Surface density (for PEDIT Smooth) in *N* direction
$TDCREATE	40	Local date/time of drawing creation (see "Special Handling of Date/Time Variables")
$TDINDWG	40	Cumulative editing time for this drawing (see "Special Handling of Date/Time Variables")
$TDUCREATE	40	Universal date/time the drawing was created (see "Special Handling of Date/Time Variables")
$TDUPDATE	40	Local date/time of last drawing update (see "Special Handling of Date/Time Variables")
$TDUSRTIMER	40	User-elapsed timer
$TDUUPDATE	40	Universal date/time of the last update/save (see "Special Handling of Date/Time Variables")

DXF header variables (*continued*)

Variable	Group code	Description
$TEXTSIZE	40	Default text height
$TEXTSTYLE	7	Current text style name
$THICKNESS	40	Current thickness set by ELEV command
$TILEMODE	70	1 for previous release compatibility mode; 0 otherwise
$TRACEWID	40	Default trace width
$TREEDEPTH	70	Specifies the maximum depth of the spatial index
$UCSBASE	2	Name of the UCS that defines the origin and orientation of orthographic UCS settings
$UCSNAME	2	Name of current UCS
$UCSORG	10, 20, 30	Origin of current UCS (in WCS)
$UCSORGBACK	10, 20, 30	Point which becomes the new UCS origin after changing model space UCS to BACK when UCSBASE is set to WORLD
$UCSORGBOTTOM	10, 20, 30	Point which becomes the new UCS origin after changing model space UCS to BOTTOM when UCSBASE is set to WORLD
$UCSORGFRONT	10, 20, 30	Point which becomes the new UCS origin after changing model space UCS to FRONT when UCSBASE is set to WORLD
$UCSORGLEFT	10, 20, 30	Point which becomes the new UCS origin after changing model space UCS to LEFT when UCSBASE is set to WORLD
$UCSORGRIGHT	10, 20, 30	Point which becomes the new UCS origin after changing model space UCS to RIGHT when UCSBASE is set to WORLD
$UCSORGTOP	10, 20, 30	Point which becomes the new UCS origin after changing model space UCS to TOP when UCSBASE is set to WORLD
$UCSORTHOREF	2	If model space UCS is orthographic (UCSORTHOVIEW not equal to 0), this is the name of the UCS that the orthographic UCS is relative to. If blank, UCS is relative to WORLD
$UCSORTHOVIEW	70	Orthographic view type of model space UCS: 0 = UCS is not orthographic; 1 = Top; 2 = Bottom; 3 = Front; 4 = Back; 5 = Left; 6 = Right

DXF header variables (*continued*)

Variable	Group code	Description
$UCSXDIR	10, 20, 30	Direction of the current UCS *X* axis (in WCS)
$UCSYDIR	10, 20, 30	Direction of the current UCS *Y* axis (in WCS)
$UNITMODE	70	Low bit set = Display fractions, feet-and-inches, and surveyor's angles in input format
$USERI1 – 5	70	Five integer variables intended for use by third-party developers
$USERR1 – 5	40	Five real variables intended for use by third-party developers
$USRTIMER	70	0 = Timer off; 1 = Timer on
$VERSIONGUID	2	Uniquely identifies a particular version of a drawing. Updated when the drawing is modified
$VISRETAIN	70	0 = Don't retain xref-dependent visibility settings 1 = Retain xref-dependent visibility settings
$WORLDVIEW	70	1 = Set UCS to WCS during DVIEW/VPOINT 0 = Don't change UCS
$XCLIPFRAME	290	Controls the visibility of xref clipping boundaries: 0 = Clipping boundary is not visible 1 = Clipping boundary is visible
$XEDIT	290	Controls whether the current drawing can be edited in-place when being referenced by another drawing. 0 = Can't use in-place reference editing 1 = Can use in-place reference editing

Revised VPORT Header Variables

The following header variables existed before AutoCAD® Release 11 but now have independent settings for each active viewport. OPEN honors these variables when read from DXF™ files. If a VPORT symbol table with *ACTIVE entries is present (as is true for any DXF file produced by Release 11 or later), the values in the VPORT table entries override the values of these header variables.

Revised VPORT header variables

Variable	Group code	Description
$FASTZOOM	70	Fast zoom enabled if nonzero
$GRIDMODE	70	Grid mode on if nonzero
$GRIDUNIT	10, 20	Grid *X* and *Y* spacing
$SNAPANG	50	Snap grid rotation angle
$SNAPBASE	10, 20	Snap/grid base point (in UCS)
$SNAPISOPAIR	70	Isometric plane: 0 = Left; 1 = Top; 2 = Right
$SNAPMODE	70	Snap mode on if nonzero
$SNAPSTYLE	70	Snap style: 0 = Standard; 1 = Isometric
$SNAPUNIT	10, 20	Snap grid *X* and *Y* spacing
$VIEWCTR	10, 20	*XY* center of current view on screen
$VIEWDIR	10, 20, 30	Viewing direction (direction from target in WCS)
$VIEWSIZE	40	Height of view

Special Handling of Date/Time Variables

The CDATE and DATE system variables provide access to the current date and time. The TDCREATE, TDINDWG, TDUPDATE, and TDUSRTIMER system variables (and the $TDCREATE, $TDUCREATE, $TDUPDATE, and $TDUUPDATE DXF header variables) provide access to times and dates associated with the current drawing. The values are represented as real numbers with special meanings, as described below.

DATE is the current date and time represented as a Julian date and fraction of a day in a real number.

<Julian date>.<Fraction of day>

For example, on December 31, 1999, at 9:58:35 p.m. GMT, the DATE variable contains

2451544.91568287

The date and time are taken from the computer's clock when the variable is read. The time is represented as a fraction of a day, and the times returned by DATE may be truly subtracted to compute differences in time. To extract the seconds since midnight from the value returned by DATE, use the AutoLISP expressions

```
(setq s (getvar "DATE"))
(setq seconds (* 86400.0 (- s (fix s))))
```

Note that DATE returns only a true Julian date if the system's clock is set to UTC/Zulu (Greenwich Mean Time). TDCREATE and TDUPDATE have the same format as DATE, but their values represent the creation time and last update time of the current drawing.

TDINDWG and TDUSRTIMER (and the $TDINDWG and $TDUSRTIMER DXF header variables) use a format similar to that of DATE, but their values represent elapsed times, as in

<Number of days>.<Fraction of day>

CDATE is the current date and time in calendar and clock format. The value is returned as a real number in the form

YYYYMMDD.HHMMSShsec

where

YYYY = year
MM = month (01-12)
DD = day (01-31)
HH = hour (00-23)
MM = minute (00-59)
SS = second (00-59)
hsec = hundredths of a second (00-99)

For example, if the current date is December 31, 1999, and the time is 9:58:35.75 p.m., CDATE would return the value:

19991231.21583575

Note that CDATE values can be compared for later and earlier values but that subtracting them yields numbers that are not meaningful.

CLASSES Section

3

The group codes described in this chapter are found only in DXF™ files. The CLASSES section holds the information for application-defined classes whose instances appear in the BLOCKS, ENTITIES, and OBJECTS sections of the database. It is assumed that a class definition is permanently fixed in the class hierarchy. All fields are required.

In this chapter

CLASSES Section Group Codes

Each entry in the CLASSES section contains the groups described in the following table.

CLASSES section group codes

Group code	Description
0	Record type (CLASS). Identifies beginning of a CLASS record
1	Class DXF record name; always unique
2	C++ class name. Used to bind with software that defines object class behavior; always unique
3	Application name. Posted in Alert box when a class definition listed in this section is not currently loaded
90	Proxy capabilities flag. Bit-coded value that indicates the capabilities of this object as a proxy: 0 = No operations allowed (0) 1 = Erase allowed (0x1) 2 = Transform allowed (0x2) 4 = Color change allowed (0x4) 8 = Layer change allowed (0x8) 16 = Linetype change allowed (0x10) 32 = Linetype scale change allowed (0x20) 64 = Visibility change allowed (0x40) 128 = Cloning allowed (0x80) 256 = Lineweight change allowed (0x100) 512 = Plot Style Name change allowed (0x200) 895 = All operations except cloning allowed (0x37F) 1023 = All operations allowed (0x3FF) 1024 = Disables proxy warning dialog (0x400) 32768 = R13 format proxy (0x8000)
91	Instance count for a custom class
280	Was-a-proxy flag. Set to 1 if class was not loaded when this DXF file was created, and 0 otherwise
281	Is-an-entity flag. Set to 1 if class was derived from the AcDbEntity class and can reside in the BLOCKS or ENTITIES section. If 0, instances may appear only in the OBJECTS section

Default Class Values

AutoCAD registers the classes listed in the following table. (This may not be a complete list of the classes found in a DXF file. It depends on the applications currently in use by AutoCAD®.)

Default class values

DXF record name code 1	C++ class name code 2	Code 90	Code 280	Code 281
ACDBDICTIONARYWDFLT	AcDbDictionaryWithDefault	0	0	0
ACDBPLACEHOLDER	AcDbPlaceHolder	0	0	0
ARCALIGNEDTEXT	AcDbArcAlignedText	0	0	1
DICTIONARYVAR	AcDbDictionaryVar	0	0	0
HATCH	AcDbHatch	0	0	1
IDBUFFER	AcDbIdBuffer	0	0	0
IMAGE	AcDbRasterImage	127	0	1
IMAGEDEF	AcDbRasterImageDef	0	0	0
IMAGEDEF_REACTOR	AcDbRasterImageDefReactor	1	0	0
LAYER_INDEX	AcDbLayerIndex	0	0	0
LAYOUT	AcDbLayout	0	0	0
LWPOLYLINE	AcDbPolyline	0	0	1
OBJECT_PTR	CAseDLPNTableRecord	1	0	0
OLE2FRAME	AcDbOle2Frame	0	0	1
PLOTSETTINGS	AcDbPlotSettings	0	0	0
RASTERVARIABLES	AcDbRasterVariables	0	0	0
RTEXT	RText	0	0	1
SORTENTSTABLE	AcDbSortentsTable	0	0	0
SPATIAL_INDEX	AcDbSpatialIndex	0	0	0

Default class values (*continued*)

DXF record name code 1	C++ class name code 2	Code 90	Code 280	Code 281
SPATIAL_FILTER	AcDbSpatialFilter	0	0	0
WIPEOUT	AcDbWipeout	127	0	1
WIPEOUTVARIABLES	AcDbWipeoutVariables	0	0	0

TABLES Section

4

The group codes described in this chapter are found in DXF™ files and used by applications. The TABLES section contains several tables, each of which can contain a variable number of entries. These codes are also used by AutoLISP® and ObjectARX™ applications in entity definition lists.

In this chapter

Symbol Table Group Codes

The order of the tables may change, but the LTYPE table always precedes the LAYER table. Each table is introduced with a 0 group code with the label TABLE. This is followed by a 2 group code identifying the particular table (APPID, DIMSTYLE, LAYER, LTYPE, STYLE, UCS, VIEW, VPORT, or BLOCK_RECORD), a 5 group code (a handle), a 100 group code (AcDbSymbolTable subclass marker), and a 70 group code that specifies the maximum number of table entries that may follow. Table names are output in uppercase. The DIMSTYLE handle is a 105 group code, and not a 5 group code.

The tables in a drawing can contain deleted items, but these are not written to the DXF file. As a result, fewer table entries may follow the table header than are indicated by the 70 group code, so do not use the count in the 70 group code as an index to read in the table. This group code is provided so that a program that reads DXF files can allocate an array large enough to hold all the table entries that follow.

Following this header for each table are the table entries. Each table entry consists of a 0 group identifying the item type (same as table name, such as LTYPE or LAYER), a 2 group giving the name of the table entry, a 70 group specifying flags relevant to the table entry (defined for each following table), and additional groups that give the value of the table entry. The end of each table is indicated by a 0 group with the value ENDTAB.

Both symbol table records and symbol tables are database objects. At a very minimum, with all prevailing usage within AutoCAD®, this implies that a handle is present, positioned after the 2 group codes for both the symbol table record objects and the symbol table objects.

The DIMSTYLE table is the only record type in the system with a handle code of 105 because of its earlier usage of group code 5. As a rule, programmers should not be concerned about this exception unless it is in the context of the DIMSTYLE table section. This is the only context in which this exception should occur.

Common Symbol Table Group Codes

The following table shows group codes that apply to all symbol tables. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Group codes that apply to all symbol tables

Group code	Description
–1	APP: entity name (changes each time a drawing is opened)
0	Object type (TABLE)
2	Table name
5	Handle
102	"{ACAD_XDICTIONARY" indicates the start of an extension dictionary group. This group exists only if persistent reactors have been attached to this object (optional)
360	Hard owner ID/handle to owner dictionary (optional)
102	End of group, "}" (optional)
330	Soft-pointer ID/handle to owner object
100	Subclass marker (AcDbSymbolTable)
70	Maximum number of entries in table

Common Group Codes for Symbol Table Entries

The following table shows group codes that apply to all symbol table entries. When you refer to the table of group codes by entity type, which lists the codes associated with specific entities, keep in mind that the codes shown here can also be present. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Group codes that apply to all symbol table entries

Group code	Description
–1	APP: entity name (changes each time a drawing is opened)
0	Entity type (table name)
5	Handle (all except DIMSTYLE)
105	Handle (DIMSTYLE table only)
102	Start of application-defined group "{*application_name*". For example, "{ACAD_REACTORS" indicates the start of the AutoCAD persistent reactors group (optional)
application-defined codes	Codes and values within the 102 groups are application defined (optional)
102	End of group, "}" (optional)
102	"{ACAD_REACTORS" indicates the start of the AutoCAD persistent reactors group. This group exists only if persistent reactors have been attached to this object (optional)
330	Soft-pointer ID/handle to owner dictionary (optional)
102	End of group, "}" (optional)
102	"{ACAD_XDICTIONARY" indicates the start of an extension dictionary group. This group exists only if persistent reactors have been attached to this object (optional)
360	Hard-owner ID/handle to owner dictionary (optional)
102	End of group, "}" (optional)
330	Soft-pointer ID/handle to owner object
100	Subclass marker (AcDbSymbolTableRecord)

APPID

The following group codes apply to APPID symbol table entries. In addition to the group codes described here, see “Common Group Codes for Symbol Table Entries” on page 38. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

APPID group codes

Group codes	Description
100	Subclass marker (AcDbRegAppTableRecord)
2	User-supplied (or application-supplied) application name (for extended data). These table entries maintain a set of names for all registered applications
70	Standard flag values (bit-coded values): 1 = If set, xdata associated with this APPID is not written when SAVEASR12 is performed 16 = If set, table entry is externally dependent on an xref 32 = If both this bit and bit 16 are set, the externally dependent xref has been successfully resolved 64 = If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD commands. It can be ignored by most programs that read DXF files and need not be set by programs that write DXF files)

BLOCK_RECORD

The following group codes apply to BLOCK_RECORD symbol table entries. In addition to the group codes described here, see “Common Group Codes for Symbol Table Entries” on page 38. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

BLOCK_RECORD group codes

Group codes	Description
100	Subclass marker (AcDbBlockTableRecord)
2	Block name
340	Hard-pointer ID/handle to associated LAYOUT object
310	DXF: Binary data for bitmap preview (optional)
1001	Xdata application name “ACAD” (optional)
1000	Xdata string data “DesignCenter Data” (optional)
1002	Begin xdata “{“ (optional)
1070	Autodesk Design Center version number
1070	Insert units: 0 = Unitless; 1 = Inches; 2 = Feet; 3 = Miles; 4 = Millimeters; 5 = Centimeters; 6 = Meters; 7 = Kilometers; 8 = Microinches; 9 = Mils; 10 = Yards; 11 = Angstroms; 12 = Nanometers; 13 = Microns; 14 = Decimeters; 15 = Decameters; 16 = Hectometers; 17 = Gigameters; 18 = Astronomical units; 19 = Light years; 20 = Parsecs
1002	End xdata “}“

DIMSTYLE

The following group codes apply to DIMSTYLE symbol table entries. The DIMSTYLE system variables are described in "System Variables," in the *Command Reference*. In addition to the group codes described here, see "Common Group Codes for Symbol Table Entries" on page 38. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

DIMSTYLE group codes

Group codes	Description
100	Subclass marker (AcDbDimStyleTableRecord)
2	Dimension style name
70	Standard flag values (bit-coded values): 16 = If set, table entry is externally dependent on an xref 32 = If both this bit and bit 16 are set, the externally dependent xref has been successfully resolved 64 = If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD® commands. It can be ignored by most programs that read DXF™ files and need not be set by programs that write DXF files)
3	DIMPOST
4	DIMAPOST
5	DIMBLK (obsolete, now object ID)
6	DIMBLK1 (obsolete, now object ID)
7	DIMBLK2 (obsolete, now object ID)
40	DIMSCALE
41	DIMASZ
42	DIMEXO
43	DIMDLI
44	DIMEXE
45	DIMRND

DIMSTYLE group codes (*continued*)	
Group codes	**Description**
46	DIMDLE
47	DIMTP
48	DIMTM
140	DIMTXT
141	DIMCEN
142	DIMTSZ
143	DIMALTF
144	DIMLFAC
145	DIMTVP
146	DIMTFAC
147	DIMGAP
148	DIMALTRND
71	DIMTOL
72	DIMLIM
73	DIMTIH
74	DIMTOH
75	DIMSE1
76	DIMSE2
77	DIMTAD
78	DIMZIN
79	DIMAZIN
170	DIMALT
171	DIMALTD

DIMSTYLE group codes (*continued*)

Group codes	Description
172	DIMTOFL
173	DIMSAH
174	DIMTIX
175	DIMSOXD
176	DIMCLRD
177	DIMCLRE
178	DIMCLRT
179	DIMADEC
270	DIMUNIT (obsolete, now use DIMLUNIT AND DIMFRAC)
271	DIMDEC
272	DIMTDEC
273	DIMALTU
274	DIMALTTD
275	DIMAUNIT
276	DIMFRAC
277	DIMLUNIT
278	DIMDSEP
279	DIMTMOVE
280	DIMJUST
281	DIMSD1
282	DIMSD2
283	DIMTOLJ
284	DIMTZIN

DIMSTYLE group codes (*continued*)

Group codes	Description
285	DIMALTZ
286	DIMALTTZ
287	DIMFIT (obsolete, now use DIMATFIT and DIMTMOVE)
288	DIMUPT
289	DIMATFIT
340	DIMTXSTY (handle of referenced STYLE)
341	DIMLDRBLK (handle of referenced BLOCK)
342	DIMBLK (handle of referenced BLOCK)
343	DIMBLK1 (handle of referenced BLOCK)
344	DIMBLK2 (handle of referenced BLOCK)
371	DIMLWD (lineweight enum value)
372	DIMLWE (lineweight enum value)

LAYER

The following group codes apply to LAYER symbol table entries. In addition to the group codes described here, see "Common Group Codes for Symbol Table Entries" on page 38. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

LAYER group codes

Group codes	Description
100	Subclass marker (AcDbLayerTableRecord)
2	Layer name

LAYER group codes (*continued*)

Group codes	Description
70	Standard flags (bit-coded values): 1 = Layer is frozen; otherwise layer is thawed 2 = Layer is frozen by default in new viewports 4 = Layer is locked 16 = If set, table entry is externally dependent on an xref 32 = If both this bit and bit 16 are set, the externally dependent xref has been successfully resolved 64 = If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD commands. It can be ignored by most programs that read DXF files and need not be set by programs that write DXF files)
62	Color number (if negative, layer is off)
6	Linetype name
290	Plotting flag. If set to 0, do not plot this layer
370	Lineweight enum value
390	Hard-pointer ID/handle of PlotStyleName object

Xref-dependent layers are output during SAVEAS. For these layers, the associated linetype name in the DXF file is always CONTINUOUS.

LTYPE

The following group codes apply to LTYPE symbol table entries. In addition to the group codes described here, see “Common Group Codes for Symbol Table Entries” on page 38. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

LTYPE group codes

Group codes	Description
100	Subclass marker (AcDbLinetypeTableRecord)
2	Linetype name

LTYPE group codes (*continued*)

Group codes	Description
70	Standard flag values (bit-coded values): 16 = If set, table entry is externally dependent on an xref 32 = If both this bit and bit 16 are set, the externally dependent xref has been successfully resolved 64 = If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD commands. It can be ignored by most programs that read DXF files and need not be set by programs that write DXF files)
3	Descriptive text for linetype
72	Alignment code; value is always 65, the ASCII code for A
73	The number of linetype elements
40	Total pattern length
49	Dash, dot or space length (one entry per element)
74	Complex linetype element type (one per element). Default is 0 (no embedded shape/text) The following codes are bit values: 1 = If set, code 50 specifies an absolute rotation; if not set, code 50 specifies a relative rotation 2 = Embedded element is a text string 4 = Embedded element is a shape
75	Shape number (one per element) if code 74 specifies an embedded shape If code 74 specifies an embedded text string, this value is set to 0 If code 74 is set to 0, code 75 is omitted
340	Pointer to STYLE object (one per element if code 74 > 0)
46	S = Scale value (optional); multiple entries can exist
50	R = (relative) or A = (absolute) rotation value in radians of embedded shape or text; one per element if code 74 specifies an embedded shape or text string
44	X = *X* offset value (optional); multiple entries can exist
45	Y = *Y* offset value (optional); multiple entries can exist
9	Text string (one per element if code 74 = 2)

The group codes 74, 75, 340, 46, 50, 44, 45, and 9 are not returned by the **tblsearch** or **tblnext** functions. You must use **tblobjname** to retrieve these values within an application.

STYLE

The following group codes apply to STYLE symbol table entries. In addition to the group codes described here, see "Common Group Codes for Symbol Table Entries" on page 38. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

STYLE group codes

Group codes	Description
100	Subclass marker (AcDbTextStyleTableRecord)
2	Style name
70	Standard flag values (bit-coded values): 1 = If set, this entry describes a shape 4 = Vertical text 16 = If set, table entry is externally dependent on an xref 32 = If both this bit and bit 16 are set, the externally dependent xref has been successfully resolved 64 = If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD commands. It can be ignored by most programs that read DXF files and need not be set by programs that write DXF files)
40	Fixed text height; 0 if not fixed
41	Width factor
50	Oblique angle
71	Text generation flags: 2 = Text is backward (mirrored in *X*) 4 = Text is upside down (mirrored in *Y*)
42	Last height used
3	Primary font file name
4	Bigfont file name; blank if none

A STYLE table item is also used to record shape file LOAD command requests. In this case the first bit (1) is set in the 70 group flags and only the 3 group (shape file name) is meaningful (all the other groups are output, however).

UCS

The following group codes apply to UCS symbol table entries. In addition to the group codes described here, see "Common Group Codes for Symbol Table Entries" on page 38. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

UCS group codes

Group codes	Description
100	Subclass marker (AcDbUCSTableRecord)
2	UCS name
70	Standard flag values (bit-coded values): 16 = If set, table entry is externally dependent on an xref 32 = If both this bit and bit 16 are set, the externally dependent xref has been successfully resolved 64 = If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD commands. It can be ignored by most programs that read DXF files and need not be set by programs that write DXF files)
10	Origin (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of origin (in WCS)
11	*X*-axis direction (in WCS) DXF: *X* value; APP: 3D vector
21, 31	DXF: *Y* and *Z* values of *X*-axis direction (in WCS)
12	*Y*-axis direction (in WCS) DXF: *X* value; APP: 3D vector
22, 32	DXF: *Y* and *Z* values of *Y*-axis direction (in WCS)
79	Always 0
146	Elevation

UCS group codes (*continued*)

Group codes	Description
346	ID/handle of base UCS if this is an orthographic. This code is not present if the 79 code is 0. If this code is not present and 79 code is non-zero, then base UCS is assumed to be WORLD
71	Orthographic type (optional; always appears in pairs with the 13, 23, 33 codes): 1 = Top; 2 = Bottom 3 = Front; 4 = Back 5 = Left; 6 = Right
13	Origin for this orthographic type relative to this UCS DXF: *X* value of origin point; APP: 3D point
23, 33	DXF: *Y* and *Z* values of origin point

Each 71/13,23,33 pair defines the UCS origin for a particular orthographic type relative to this UCS. For instance if the following pair is present, then invoking the UCS/LEFT command when UCSBASE is set to this UCS will cause the new UCS origin to become (1,2,3).

```
71: 5
13: 1.0
23: 2.0
33: 3.0
```

If this pair were not present, then invoking the UCS/LEFT command would cause the new UCS origin to be set to this UCS's origin point.

VIEW

The following group codes apply to VIEW symbol table entries. In addition to the group codes described here, see "Common Group Codes for Symbol Table Entries" on page 38. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

VIEW group codes

Group codes	Description
100	Subclass marker (AcDbViewTableRecord)

VIEW group codes (*continued*)

Group codes	Description
2	Name of view
70	Standard flag values (bit-coded values): 1 = If set, this is a paper space view 16 = If set, table entry is externally dependent on an xref 32 = If both this bit and bit 16 are set, the externally dependent xref has been successfully resolved 64 = If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD commands. It can be ignored by most programs that read DXF files and need not be set by programs that write DXF files)
40	View height (in DCS)
10	View center point (in DCS) DXF: *X* value; APP: 2D point
20	DXF: *Y* value of view center point (in DCS)
41	View width (in DCS)
11	View direction from target (in WCS) DXF: *X* value; APP: 3D vector
21, 31	DXF: *Y* and *Z* values of view direction from target (in WCS)
12	Target point (in WCS) DXF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of target point (in WCS)
42	Lens length
43	Front clipping plane (offset from target point)
44	Back clipping plane (offset from target point)
50	Twist angle
71	View mode (see VIEWMODE system variable)

VIEW group codes (*continued*)

Group codes	Description
281	Render mode: 0 = 2D Optimized (classic 2D) 1 = Wireframe 2 = Hidden line 3 = Flat shaded 4 = Gouraud shaded 5 = Flat shaded with wireframe 6 = Gouraud shaded with wireframe All rendering modes other than 2D Optimized engage the new 3D graphics pipeline. These values directly correspond to the SHADEMODE command and the AcDbAbstractViewTableRecord::RenderMode enum
72	1 if there is a UCS associated to this view, 0 otherwise

The following codes appear only if code 72 is set to 1. They define the UCS that is associated to this view. This UCS will become the current UCS whenever this view is restored (if code 72 is 0, the UCS is unchanged).

VIEW with UCS group codes

Group codes	Description
110	UCS origin (appears only if code 72 is set to 1) DXF: *X* value; APP: 3D point
120, 130	DXF: *Y* and *Z* values of UCS origin
111	UCS *X*-axis (appears only if code 72 is set to 1) DXF: *X* value; APP: 3D vector
121, 131	DXF: *Y* and *Z* values of UCS *X*-axis
112	UCS *Y*-axis (appears only if code 72 is set to 1) DXF: *X* value; APP: 3D vector
122, 132	DXF: *Y* and *Z* values of UCS *Y*-axis
79	Orthographic type of UCS (appears only if code 72 is set to 1): 0 = UCS is not orthographic; 1 = Top; 2 = Bottom; 3 = Front; 4 = Back; 5 = Left; 6 = Right
146	UCS Elevation (appears only if code 72 is set to 1)

VIEW with UCS group codes (*continued*)	
Group codes	**Description**
345	ID/handle of AcDbUCSTableRecord if UCS is a named UCS. If not present, then UCS is unnamed (appears only if code 72 is set to 1)
346	ID/handle of AcDbUCSTableRecord of base UCS if UCS is orthographic (79 code is non-zero). If not present and 79 code is non-zero, then base UCS is taken to be WORLD (appears only if code 72 is set to 1)

VPORT

The following group codes apply to VPORT symbol table entries. The VPORT table is unique: it may contain several entries with the same name (indicating a multiple-viewport configuration). The entries corresponding to the active viewport configuration all have the name *ACTIVE. The first such entry describes the current viewport. In addition to the group codes described here, see "Common Group Codes for Symbol Table Entries" on page 38. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

VPORT group codes	
Group codes	**Description**
100	Subclass marker (AcDbViewportTableRecord)
2	Viewport name
70	Standard flag values (bit-coded values): 16 = If set, table entry is externally dependent on an xref 32 = If both this bit and bit 16 are set, the externally dependent xref has been successfully resolved 64 = If set, the table entry was referenced by at least one entity in the drawing the last time the drawing was edited. (This flag is for the benefit of AutoCAD commands. It can be ignored by most programs that read DXF files and need not be set by programs that write DXF files)
10	Lower-left corner of viewport DXF: *X* value; APP: 2D point
20	DXF: *Y* value of lower-left corner of viewport
11	Upper-right corner of viewport DXF: *X* value; APP: 2D point

VPORT group codes (*continued*)

Group codes	Description
21	DXF: *Y* value of upper-right corner of viewport
12	View center point (in DCS) DXF: *X* value; APP: 2D point
22	DXF: *Y* value of view center point (in DCS)
13	Snap base point DXF: *X* value; APP: 2D point
23	DXF: *Y* value of snap base point
14	Snap spacing *X* and *Y* DXF: *X* value; APP: 2D point
24	DXF: *Y* value of snap spacing *X* and *Y*
15	Grid spacing *X* and *Y* DXF: *X* value; APP: 2D point
25	DXF: *Y* value of grid spacing *X* and *Y*
16	View direction from target point (in WCS) DXF: *X* value; APP: 3D point
26, 36	DXF: *Y* and *Z* values of view direction from target point (in WCS)
17	View target point (in WCS) DXF: *X* value; APP: 3D point
27, 37	DXF: *Y* and *Z* values of view target point (in WCS)
40	View height
41	Viewport aspect ratio
42	Lens length
43	Front clipping plane (offset from target point)
44	Back clipping plane (offset from target point)
50	Snap rotation angle
51	View twist angle

VPORT group codes (*continued*)	
Group codes	**Description**
68	APP: Status field (never saved in DXF)
69	APP: ID (never saved in DXF)
71	View mode (see VIEWMODE system variable)
72	Circle zoom percent
73	Fast zoom setting
74	UCSICON setting
75	Snap on/off
76	Grid on/off
77	Snap style
78	Snap isopair
281	Render mode: 0 = 2D Optimized (classic 2D) 1 = Wireframe 2 = Hidden line 3 = Flat shaded 4 = Gouraud shaded 5 = Flat shaded with wireframe 6 = Gouraud shaded with wireframe All rendering modes other than 2D Optimized engage the new 3D graphics pipeline. These values directly correspond to the SHADEMODE command and the AcDbAbstractViewTableRecord::RenderMode enum
65	Value of UCSVP for this viewport. If set to 1, then viewport stores its own UCS which will become the current UCS whenever the viewport is activated. If set to 0, UCS will not change when this viewport is activated
110	UCS origin DXF: *X* value; APP: 3D point
120, 130	DXF: *Y* and *Z* values of UCS origin
111	UCS *X*-axis DXF: *X* value; APP: 3D vector
121, 131	DXF: *Y* and *Z* values of UCS *X*-axis

VPORT group codes (*continued*)

Group codes	Description
112	UCS *Y*-axis DXF: *X* value; APP: 3D vector
122, 132	DXF: *Y* and *Z* values of UCS *Y*-axis
79	Orthographic type of UCS 0 = UCS is not orthographic; 1 = Top; 2 = Bottom 3 = Front; 4 = Back 5 = Left; 6 = Right
146	Elevation
345	ID/handle of AcDbUCSTableRecord if UCS is a named UCS. If not present, then UCS is unnamed
346	ID/handle of AcDbUCSTableRecord of base UCS if UCS is orthographic (79 code is non-zero). If not present and 79 code is non-zero, then base UCS is taken to be WORLD

BLOCKS Section

5

The group codes described in this chapter are found in DXF™ files and used by applications. The BLOCKS section contains an entry for each block reference in the drawing.

In this chapter

BLOCKS Section Group Codes

The BLOCKS section of the DXF file contains all the block definitions, including anonymous blocks generated by the HATCH command and by associative dimensioning. Each block definition contains the entities that make up that block as it is used in the drawing. The format of the entities in this section is identical to those in the ENTITIES section. All entities in the BLOCKS section appear between block and endblk entities. Block and endblk entities appear only in the BLOCKS section. Block definitions are never nested (that is, no block or endblk entity ever appears within another block-endblk pair), although a block definition can contain an insert entity.

External references are written in the DXF file as block definitions, except that they also include a string (group code 1) that specifies the path and file name of the external reference.

The block table handle, along with any xdata and persistent reactors, appears in each block definition immediately following the BLOCK record, which contains all of the specific information that a block table record stores.

BLOCK

The following group codes apply to block entities. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Block group codes

Group codes	Description
0	Entity type (BLOCK)
5	Handle
102	Start of application-defined group "{*application_name*". For example, "{ACAD_REACTORS" indicates the start of the AutoCAD persistent reactors group (optional)
application-defined codes	Codes and values within the 102 groups are application defined (optional)
102	End of group, "}" (optional)
330	Soft-pointer ID/handle to owner object

Block group codes (*continued*)

Group codes	Description
100	Subclass marker (AcDbEntity)
8	Layer name
100	Subclass marker (AcDbBlockBegin)
2	Block name
70	Block-type flags (bit-coded values, may be combined): 0 = Indicates none of the following flags apply 1 = This is an anonymous block generated by hatching, associative dimensioning, other internal operations, or an application 2 = This block has non-constant attribute definitions (this bit is not set if the block has any attribute definitions that are constant, or has no attribute definitions at all) 4 = This block is an external reference (xref) 8 = This block is an xref overlay 16 = This block is externally dependent 32 = This is a resolved external reference, or dependent of an external reference (ignored on input) 64 = This definition is a referenced external reference (ignored on input)
10	Base point DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of base point
3	Block name
1	Xref path name
4	Block description (optional)

The UCS in effect when a block definition is created becomes the WCS for all entities in the block definition. The new origin for these entities is shifted to match the base point defined for the block definition. All entity data is translated to fit this new WCS.

Model Space and Paper Space Block Definitions
Three empty definitions always appear in the BLOCKS section. They are titled *Model_Space, *Paper_Space and *Paper_Space0. These definitions manifest the representations of model space and paper space as block definitions internally. The internal name of the first paper space layout is *Paper_Space, the second is *Paper_Space0, the third is *Paper_Space1, and so on.

Model Space and Paper Space Entity Segregation
The interleaving between model space and paper space no longer occurs. Instead, all paper space entities are output, followed by model space entities. The flag distinguishing them is the group code 67.

ENDBLK

The following group codes apply to endblk objects. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Endblk group codes

Group codes	Description
0	Entity type (ENDBLK)
5	Handle
102	Start of application-defined group "{*application_name*". For example, "{ACAD_REACTORS" indicates the start of the AutoCAD persistent reactors group (optional)
application-defined codes	Codes and values within the 102 groups are application defined (optional)
102	End of group, "}" (optional)
330	Soft-pointer ID/handle to owner object
100	Subclass marker (AcDbEntity)
8	Layer name
100	Subclass marker (AcDbBlockEnd)

ENTITIES Section

6

This chapter presents the group codes that apply to graphical objects. These codes are found in the ENTITIES section of a DXF™ file and are used by AutoLISP® and ObjectARX™ applications in entity definition lists.

In this chapter

Common Group Codes for Entities

The following table shows group codes that apply to virtually all graphical objects. Some of the group codes shown here are included with an entity definition only if the entity has nondefault values for the property. When you refer to the group codes by entity type, the lists of codes associated with *specific* entities, keep in mind that the codes shown here are also present.

Note Do not write programs that rely on the order shown in these DXF code tables. Although these tables show the order of group codes as they usually appear, the order can change under certain conditions or may be changed in a future AutoCAD® release. The code that controls an entity should be driven by a case (switch) or a table so that it can process each group correctly even if the order is unexpected.

When a group is omitted, its default value upon input (when using OPEN) is indicated in the third column. If the value of a group code is equal to the default, it is omitted upon output (when using SAVEAS). For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Group codes that apply to all graphical objects

Group code	Description	If omitted, defaults to…
–1	APP: entity name (changes each time a drawing is opened)	not omitted
0	Entity type	not omitted
5	Handle	not omitted
102	Start of application-defined group "{*application_name*" (optional)	no default
application-defined codes	Codes and values within the 102 groups are application-defined (optional)	no default
102	End of group, "}" (optional)	no default
102	"{ACAD_REACTORS" indicates the start of the AutoCAD persistent reactors group. This group exists only if persistent reactors have been attached to this object (optional)	no default

Group codes that apply to all graphical objects (*continued*)

Group code	Description	If omitted, defaults to…
330	Soft-pointer ID/handle to owner dictionary (optional)	no default
102	End of group, "}" (optional)	no default
102	"{ACAD_XDICTIONARY" indicates the start of an extension dictionary group. This group exists only if an extension dictionary has been attached to the object (optional)	no default
360	Hard-owner ID/handle to owner dictionary (optional)	no default
102	End of group, "}" (optional)	no default
330	Soft-pointer ID/handle to owner BLOCK_RECORD object	not omitted
100	Subclass marker (AcDbEntity)	not omitted
67	Absent or zero indicates entity is in model space. 1 indicates entity is in paper space (optional)	0
410	APP: layout tab name	not omitted
8	Layer name	not omitted
6	Linetype name (present if not BYLAYER). The special name BYBLOCK indicates a floating linetype (optional)	BYLAYER
62	Color number (present if not BYLAYER); zero indicates the BYBLOCK (floating) color; 256 indicates BYLAYER; a negative value indicates that the layer is turned off (optional)	BYLAYER
370	Lineweight enum value. Stored and moved around as a 16-bit integer.	not omitted
48	Linetype scale (optional)	1.0
60	Object visibility (optional): 0 = Visible; 1 = Invisible	0
92	The number of bytes in the proxy entity graphics represented in the subsequent 310 groups, which are binary chunk records (optional)	no default

Group codes that apply to all graphical objects (*continued*)		
Group code	**Description**	**If omitted, defaults to…**
310	Proxy entity graphics data (multiple lines; 256 characters max. per line) (optional)	no default
420	A 24-bit color value that should be dealt with in terms of bytes with values of 0 to 255. The lowest byte is the blue value, the middle byte is the green value, and the third byte is the red value. The top byte is always 0. The group code cannot be used by custom entities for their own data because the group code is reserved for AcDbEntity, class-level color data and AcDbEntity, class-level transparency data	no default
430	The color name. The group code cannot be used by custom entities for their own data because the group code is reserved for AcDbEntity, class-level color data and AcDbEntity, class-level transparency data	no default
440	The transparency value. The group code cannot be used by custom entities for their own data because the group code is reserved for AcDbEntity, class-level color data and AcDbEntity, class-level transparency data	no default

3DFACE

The following group codes apply to 3dface entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

3dface group codes	
Group codes	**Description**
100	Subclass marker (AcDbFace)
10	First corner (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of first corner (in WCS)

3dface group codes (*continued*)	
Group codes	**Description**
11	Second corner (in WCS) DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of second corner (in WCS)
12	Third corner (in WCS) DXF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of third corner (in WCS)
13	Fourth corner (in WCS). If only three corners are entered, this is the same as the third corner DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of fourth corner (in WCS)
70	Invisible edge flags (optional; default = 0): 1 = First edge is invisible 2 = Second edge is invisible 4 = Third edge is invisible 8 = Fourth edge is invisible

3DSOLID

The following group codes apply to 3dsolid entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

3dsolid group codes	
Group codes	**Description**
100	Subclass marker (AcDbModelerGeometry)
70	Modeler format version number (currently = 1)
1	Proprietary data (multiple lines < 255 characters each)
3	Additional lines of proprietary data (if previous group 1 string is greater than 255 characters) (optional)

ACAD_PROXY_ENTITY

The following group codes apply to proxy entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Acad_proxy_entity group codes

Group codes	Description
100	DXF™AcDbProxyEntity
90	DXF: Proxy entity class ID (always 498)
91	DXF: Application entity’s class ID. Class IDs are based on the order of the class in the CLASSES section. The first class is given the ID of 500, the next is 501, and so on
92	DXF: Size of graphics data in bytes
310	DXF: Binary graphics data (multiple entries can appear) (optional)
93	DXF: Size of entity data in bits
310	DXF: Binary entity data (multiple entries can appear) (optional)
330 or 340 or 350 or 360	DXF: An object ID (multiple entries can appear) (optional)
94	DXF: 0 (indicates end of object ID section)
95	DXF: Object drawing format when it becomes a proxy (a 32-bit unsigned integer): Low word is AcDbDwgVersion High word is MaintenanceReleaseVersion
70	DXF: Original custom object data format: 0 = DWG format 1 = DXF format

ARC

The following group codes apply to arc entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Arc group codes

Group codes	Description
100	Subclass marker (AcDbCircle)
39	Thickness (optional; default = 0)
10	Center point (in OCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of center point (in OCS)
40	Radius
100	Subclass marker (AcDbArc)
50	Start angle
51	End angle
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

ATTDEF

The following group codes apply to attdef (attribute definition) entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Attdef group codes

Group codes	Description
100	Subclass marker (AcDbText)
39	Thickness (optional; default = 0)
10	First alignment point (in OCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of text start point (in OCS)
40	Text height
1	Default value (string)
100	Subclass marker (AcDbAttributeDefinition)
50	Text rotation (optional; default = 0)
41	Relative X scale factor (width) (optional; default = 1). This value is also adjusted when fit-type text is used
51	Oblique angle (optional; default = 0)
7	Text style name (optional; default = STANDARD)
71	Text generation flags (optional; default = 0); see TEXT group codes
72	Horizontal text justification type (optional; default = 0); see TEXT group codes
11	Second alignment point (in OCS) (optional) DXF: *X* value; APP: 3D point Meaningful only if 72 or 74 group values are nonzero
21, 31	DXF: *Y* and *Z* values of second alignment point (in OCS) (optional)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector

Attdef group codes (*continued*)	
Group codes	**Description**
220, 230	DXF: *Y* and *Z* values of extrusion direction
100	Subclass marker (AcDbAttributeDefinition)
3	Prompt string
2	Tag string
70	Attribute flags: 1 = Attribute is invisible (does not appear) 2 = This is a constant attribute 4 = Verification is required on input of this attribute 8 = Attribute is preset (no prompt during insertion)
73	Field length (optional; default = 0) (not currently used)
74	Vertical text justification type (optional, default = 0); see group code 73 in TEXT

If group 72 and/or 74 values are nonzero then the first alignment point values are ignored and AutoCAD calculates new values based on the second alignment point and the length and height of the text string itself (after applying the text style). If the 72 and 74 values are zero or missing, then the second alignment point is meaningless.

ATTRIB

The following group codes apply to attrib (attribute) entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Attrib group codes	
Group codes	**Description**
100	Subclass marker (AcDbText)
39	Thickness (optional; default = 0)

Attrib group codes (*continued*)

Group codes	Description
10	Text start point (in OCS) DXF™: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of text start point (in OCS)
40	Text height
1	Default value (string)
100	Subclass marker (AcDbAttribute)
2	Attribute tag (string)
70	Attribute flags: 1 = Attribute is invisible (does not appear) 2 = This is a constant attribute 4 = Verification is required on input of this attribute 8 = Attribute is preset (no prompt during insertion)
73	Field length (optional; default = 0) (not currently used)
50	Text rotation (optional; default = 0)
41	Relative X scale factor (width) (optional; default = 1). This value is also adjusted when fit-type text is used
51	Oblique angle (optional; default = 0)
7	Text style name (optional; default = STANDARD)
71	Text generation flags (optional; default = 0). See TEXT group codes
72	Horizontal text justification type (optional; default = 0). See TEXT group codes
74	Vertical text justification type (optional; default = 0). See group code 73 in TEXT
11	Alignment point (in OCS) (optional) DXF: *X* value; APP: 3D point Present only if 72 or 74 group is present and nonzero
21, 31	DXF: *Y* and *Z* values of alignment point (in OCS) (optional)
210	Extrusion direction. Present only if the entity's extrusion direction is not parallel to the WCS Z axis (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector

Attrib group codes (*continued*)

Group codes	Description
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

If group 72 and/or 74 values are nonzero then the text insertion point values are ignored, and AutoCAD calculates new values based on the text alignment point and the length of the text string itself (after applying the text style). If the 72 and 74 values are zero or missing, then the text alignment point is ignored and recalculated based on the text insertion point and the length of the text string itself (after applying the text style).

BODY

The following group codes apply to body entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Body group codes

Group codes	Description
100	Subclass marker (AcDbModelerGeometry)
70	Modeler format version number (currently = 1)
1	Proprietary data (multiple lines < 255 characters each)
3	Additional lines of proprietary data (if previous group 1 string is greater than 255 characters) (optional)

CIRCLE

The following group codes apply to circle entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Circle group codes

Group codes	Description
100	Subclass marker (AcDbCircle)
39	Thickness (optional; default = 0)
10	Center point (in OCS) DXF™: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of center point (in OCS)
40	Radius
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

DIMENSION

Dimension entity definitions consist of group codes that are common to all dimension types, followed by codes specific to the type.

Common Dimension Group Codes

The following group codes apply to all dimension entity types. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Common dimension group codes

Group codes	Description
100	Subclass marker (AcDbDimension)
2	Name of the block that contains the entities that make up the dimension picture
10	Definition point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of definition point (in WCS)
11	Middle point of dimension text (in OCS) DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of middle point of dimension text (in OCS)
70	Dimension type: Values 0–6 are integer values that represent the dimension type. Values 32, 64, and 128 are bit values, which are added to the integer values (value 32 is always set in R13 and later releases) 0 = Rotated, horizontal, or vertical; 1 = Aligned 2 = Angular; 3 = Diameter; 4 = Radius 5 = Angular 3 point; 6 = Ordinate 32 = Indicates that the block reference (group code 2) is referenced by this dimension only 64 = Ordinate type. This is a bit value (bit 7) used only with integer value 6. If set, ordinate is X-type; if not set, ordinate is Y-type 128 = This is a bit value (bit 8) added to the other group 70 values if the dimension text has been positioned at a user-defined location rather than at the default location
71	Attachment point: 1 = Top left; 2 = Top center; 3 = Top right 4 = Middle left; 5 = Middle center; 6 = Middle right 7 = Bottom left; 8 = Bottom center; 9 = Bottom right
72	Dimension text line-spacing style (optional): 1 (or missing) = At least (taller characters will override) 2 = Exact (taller characters will not override)

Common dimension group codes (*continued*)

Group codes	Description
41	Dimension text-line spacing factor (optional): Percentage of default (3-on-5) line spacing to be applied. Valid values range from 0.25 to 4.00
42	Actual measurement (optional; read-only value)
1	Dimension text explicitly entered by the user. Optional; default is the measurement. If null or "<>", the dimension measurement is drawn as the text, if " " (one blank space), the text is suppressed. Anything else is drawn as the text
53	The optional group code 53 is the rotation angle of the dimension text away from its default orientation (the direction of the dimension line) (optional)
51	All dimension types have an optional 51 group code, which indicates the horizontal direction for the dimension entity. The dimension entity determines the orientation of dimension text and lines for horizontal, vertical, and rotated linear dimensions This group value is the negative of the angle between the OCS *X* axis and the UCS *X* axis. It is always in the *XY* plane of the OCS
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)
3	Dimension style name

Xdata belonging to the application ID `"ACAD"` follows a dimension entity if any dimension overrides have been applied to this entity. See "Dimension Style Overrides" on page 79.

For all dimension types, the following group codes represent 3D WCS points:

- (10, 20, 30)
- (13, 23, 33)
- (14, 24, 34)
- (15, 25, 35)

For all dimension types, the following group codes represent 3D OCS points:

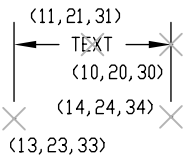
- (11, 21, 31)
- (12, 22, 32)
- (16, 26, 36)

Aligned Dimension Group Codes

The following group codes apply to aligned dimensions. In addition to the group codes described here, those listed in "Common Group Codes for Entities" on page 62 and "Common Dimension Group Codes" on page 73 can also be present. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Aligned dimension group codes

Group codes	Description
100	Subclass marker (AcDbAlignedDimension)
12	Insertion point for clones of a dimension—Baseline and Continue (in OCS) DXF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of insertion point for clones of a dimension—Baseline and Continue (in OCS)
13	Definition point for linear and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of definition point for linear and angular dimensions (in WCS)
14	Definition point for linear and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
24, 34	DXF: *Y* and *Z* values of definition point for linear and angular dimensions (in WCS)



The point (13,23,33) specifies the start point of the first extension line and the point (14,24,34) specifies the start point of the second extension line. Point (10,20,30) specifies the dimension line location. The point (11,21,31) specifies the midpoint of the dimension text.

Linear and Rotated Dimension Group Codes

The following group codes apply to linear and rotated dimensions (note that linear and rotated dimensions are part of the AcDbAlignedDimension subclass). In addition to the group codes described here, those listed in "Common Group Codes for Entities" on page 62 and "Common Dimension Group

Codes" on page 73 can also be present. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Linear and rotated dimension group codes

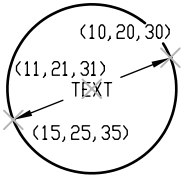
Group codes	Description
100	Subclass marker (AcDbAlignedDimension)
12	Insertion point for clones of a dimension—Baseline and Continue (in OCS) DXF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of insertion point for clones of a dimension—Baseline and Continue (in OCS)
13	Definition point for linear and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of definition point for linear and angular dimensions (in WCS)
14	Definition point for linear and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
24, 34	DXF: *Y* and *Z* values of definition point for linear and angular dimensions (in WCS)
50	Angle of rotated, horizontal, or vertical dimensions
52	Linear dimension types with an oblique angle have an optional group code 52. When added to the rotation angle of the linear dimension (group code 50), it gives the angle of the extension lines
100	Subclass marker (AcDbRotatedDimension)

Radial and Diameter Dimension Group Codes

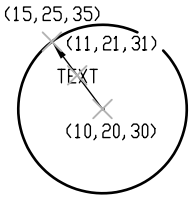
The following group codes apply to radial and diameter dimensions. In addition to the group codes described here, those listed in “Common Group Codes for Entities” on page 62 and “Common Dimension Group Codes” on page 73 can also be present. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Radial and diameter dimension group codes

Group codes	Description
100	Subclass marker (AcDbRadialDimension or AcDbDiametricDimension)
15	Definition point for diameter, radius, and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
25, 35	DXF: *Y* and *Z* values of definition point for diameter, radius, and angular dimensions (in WCS)
40	Leader length for radius and diameter dimensions



The point (15,25,35) specifies the first point of the dimension line on the circle/arc and the point (10,20,30) specifies the point opposite the first point. The point (11,21,31) specifies the midpoint of the dimension text.



The point (15,25,35) specifies the first point of the dimension line on the circle/arc and the point (10,20,30) specifies the center of the circle/arc. The point (11,21,31) specifies the midpoint of the dimension text.

Angular Dimension Group Codes

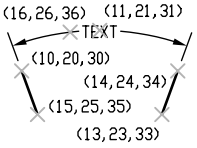
The following group codes apply to angular dimensions. In addition to the group codes described here, those listed in “Common Group Codes for Entities” on page 62 and “Common Dimension Group Codes” on page 73 can also be present. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Angular dimension group codes

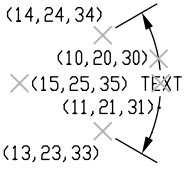
Group codes	Description
100	Subclass marker (AcDb3PointAngularDimension)

Angular dimension group codes (*continued*)

Group codes	Description
13	Definition point for linear and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of definition point for linear and angular dimensions (in WCS)
14	Definition point for linear and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
24, 34	DXF: *Y* and *Z* values of definition point for linear and angular dimensions (in WCS)
15	Definition point for diameter, radius, and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
25, 35	DXF: *Y* and *Z* values of definition point for diameter, radius, and angular dimensions (in WCS)
16	Point defining dimension arc for angular dimensions (in OCS) DXF: *X* value; APP: 3D point
26, 36	DXF: *Y* and *Z* values of point defining dimension arc for angular dimensions (in OCS)



The points (13,23,33) and (14,24,34) specify the endpoints of the line used to determine the first extension line. Points (10,20,30) and (15,25,35) specify the endpoints of the line used to determine the second extension line. Point (16,26,36) specifies the location of the dimension line arc. The point (11,21,31) specifies the midpoint of the dimension text.



The point (15,25,35) specifies the vertex of the angle. The points (13,23,33) and (14,24,34) specify the endpoints of the extension lines. The point (10,20,30) specifies the location of the dimension line arc and the point (11,21,31) specifies the midpoint of the dimension text.

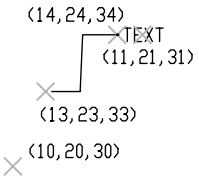
Ordinate Dimension Group Codes

The following group codes apply to ordinate dimensions. In addition to the group codes described here, those listed in “Common Group Codes for Entities” on page 62 and “Common Dimension Group Codes” on page 73 can

also be present. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Ordinate dimension group codes

Group codes	Description
100	Subclass marker (AcDbOrdinateDimension)
13	Definition point for linear and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of definition point for linear and angular dimensions (in WCS)
14	Definition point for linear and angular dimensions (in WCS) DXF: *X* value; APP: 3D point
24, 34	DXF: *Y* and *Z* values of definition point for linear and angular dimensions (in WCS)



The point (13,23,33) specifies the feature location and the point (14,24,34) specifies the leader endpoint. The point (11,21,31) specifies the midpoint of the dimension text. Point (10,20,30) is placed at the origin of the UCS that is current when the dimension is created.

Dimension Style Overrides

Dimension style overrides can be applied to dimension, leader, and tolerance entities. Any overrides applied to these entities are stored in the entity as xdata. The overridden dimension variable group codes and the related values are contained within group 1002 control strings. The following example shows the xdata of a dimension entity where the DIMTOL and DIMCLRE variables have been overridden.

```
(setq diment (car (entsel)))          ; Select dimension entity
(setq elst (entget diment '("ACAD"))) ; Get entity definition list
(assoc -3 elst)                       ; Extract xdata only
```

This code returns the following:

```
(-3 ("ACAD"                  Start of the ACAD APPID section of xdata
  (1000 . "DSTYLE") (1002 . "{") Beginning of the dimstyle subsection
  (1070 . 177) (1070 . 3)        The DIMCLRE (code 177) override +
value (3)
  (1070 . 71) (1070 . 1)          The DIMTOL (code 71) override +
value (1)
  (1002 . "}") ))                 End dimstyle subsection and ACAD
section
```

ELLIPSE

The following group codes apply to ellipse entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Ellipse group codes

Group codes	Description
100	Subclass marker (AcDbEllipse)
10	Center point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of center point (in WCS)
11	Endpoint of major axis, relative to the center (in WCS) DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of endpoint of major axis, relative to the center (in WCS)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)
40	Ratio of minor axis to major axis
41	Start parameter (this value is 0.0 for a full ellipse)
42	End parameter (this value is 2pi for a full ellipse)

HATCH

The following group codes apply to hatch and MPolygon entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Hatch group codes

Group codes	Description
100	Subclass marker (AcDbHatch)
10	Elevation point (in OCS) DXF: *X* value = 0; APP: 3D point (*X* and *Y* always equal 0, *Z* represents the elevation)
20, 30	DXF: *Y* and *Z* values of elevation point (in OCS) *Y* value = 0, *Z* represents the elevation
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction
2	Hatch pattern name
70	Solid fill flag (solid fill = 1; pattern fill = 0); for MPolygon, the version of MPolygon
63	For MPolygon, pattern fill color as the ACI
71	Associativity flag (associative = 1; non-associative = 0); for MPolygon, solid-fill flag (has solid fill = 1; lacks solid fill = 0)
91	Number of boundary paths (loops)
varies	Boundary path data. Repeats number of times specified by code 91. See "Boundary Path Data" on page 83
75	Hatch style: 0 = Hatch "odd parity" area (Normal style) 1 = Hatch outermost area only (Outer style) 2 = Hatch through entire area (Ignore style)
76	Hatch pattern type: 0 = User-defined; 1 = Predefined; 2 = Custom

Hatch group codes (*continued*)	
Group codes	**Description**
52	Hatch pattern angle (pattern fill only)
41	Hatch pattern scale or spacing (pattern fill only)
73	For MPolygon, boundary annotation flag (boundary is an annotated boundary = 1; boundary is not an annotated boundary = 0)
77	Hatch pattern double flag (pattern fill only): 0 = not double; 1 = double
78	Number of pattern definition lines
varies	Pattern line data. Repeats number of times specified by code 78. See "Pattern Data" on page 86
47	Pixel size used to determine the density to perform various intersection and ray casting operations in hatch pattern computation for associative hatches and hatches created with the Flood method of hatching
98	Number of seed points
11	For MPolygon, offset vector
99	For MPolygon, number of degenerate boundary paths (loops), where a degenerate boundary path is a border that is ignored by the hatch
10	Seed point (in OCS) DXF: *X* value; APP: 2D point (multiple entries)
20	DXF: *Y* value of seed point (in OCS); (multiple entries)
450	Indicates solid hatch or gradient; if solid hatch, the values for the remaining codes are ignored but must be present. Optional; if code 450 is in the file, then the following codes must be in the file: 451, 452, 453, 460, 461, 462, and 470. If code 450 is not in the file, then the following codes must not be in the file: 451, 452, 453, 460, 461, 462, and 470 0 = Solid hatch 1 = Gradient
451	Zero is reserved for future use
452	Records how colors were defined and is used only by dialog code: 0 = Two-color gradient 1 = Single-color gradient

Hatch group codes (*continued*)

Group codes	Description
453	Number of colors: 0 = Solid hatch 2 = Gradient
460	Rotation angle in radians for gradients (default = 0, 0)
461	Gradient definition; corresponds to the Centered option on the Gradient Tab of the Boundary Hatch and Fill dialog box. Each gradient has two definitions, shifted and unshifted. A Shift value describes the blend of the two definitions that should be used. A value of 0.0 means only the unshifted version should be used, and a value of 1.0 means that only the shifted version should be used.
462	Color tint value used by dialog code (default = 0, 0; range is 0.0 to 1.0). The color tint value is a gradient color and controls the degree of tint in the dialog when the Hatch group code 452 is set to 1.
463	Reserved for future use: 0 = First value 1 = Second value
470	String (default = LINEAR)

Boundary Path Data

The boundary of each hatch object is defined by a path (or *loop*) that consists of one or more segments. Path segment data varies depending on the entity type (or types) that make up the path. Each path segment is defined by its own set of group codes. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Hatch boundary path data group codes

Group codes	Description
92	Boundary path type flag (bit coded): 0 = Default; 1 = External; 2 = Polyline 4 = Derived; 8 = Textbox; 16 = Outermost
varies	Polyline boundary type data (only if boundary = polyline). See Polyline boundary data table below
93	Number of edges in this boundary path (only if boundary is not a polyline)

Hatch boundary path data group codes (*continued*)	
Group codes	**Description**
72	Edge type (only if boundary is not a polyline): 1 = Line; 2 = Circular arc; 3 = Elliptic arc; 4 = Spline
varies	Edge type data (only if boundary is not a polyline). See appropriate Edge data table below
97	Number of source boundary objects
330	Reference to source boundary objects (multiple entries)

Polyline boundary data group codes	
Group codes	**Description**
72	Has bulge flag
73	Is closed flag
93	Number of polyline vertices
10	Vertex location (in OCS) DXF: *X* value; APP: 2D point (multiple entries)
20	DXF: *Y* value of vertex location (in OCS) (multiple entries)
42	Bulge (optional, default = 0)

Line edge data group codes	
Group codes	**Description**
10	Start point (in OCS) DXF: *X* value; APP: 2D point
20	DXF: *Y* value of start point (in OCS)
11	Endpoint (in OCS) DXF: *X* value; APP: 2D point
21	DXF: *Y* value of endpoint (in OCS)

Arc edge data group codes

Group codes	Description
10	Center point (in OCS) DXF: *X* value; APP: 2D point
20	DXF: *Y* value of center point (in OCS)
40	Radius
50	Start angle
51	End angle
73	Is counterclockwise flag

Ellipse edge data group codes

Group codes	Description
10	Center point (in OCS) DXF: *X* value; APP: 2D point
20	DXF: *Y* value of center point (in OCS)
11	Endpoint of major axis relative to center point (in OCS) DXF: *X* value; APP: 2D point
21	DXF: *Y* value of endpoint of major axis (in OCS)
40	Length of minor axis (percentage of major axis length)
50	Start angle
51	End angle
73	Is counterclockwise flag

Spline edge data group codes

Group codes	Description
94	Degree
73	Rational
74	Periodic

Spline edge data group codes (*continued*)	
Group codes	**Description**
95	Number of knots
96	Number of control points
40	Knot values (multiple entries)
10	Control point (in OCS) DXF: *X* value; APP: 2D point
20	DXF: *Y* value of control point (in OCS)
42	Weights (optional, default = 1)

Pattern Data

The following pattern data codes repeat for each pattern definition line. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Hatch pattern data group codes	
Group codes	**Description**
53	Pattern line angle
43	Pattern line base point, *X* component
44	Pattern line base point, *Y* component
45	Pattern line offset, *X* component
46	Pattern line offset, *Y* component
79	Number of dash length items
49	Dash length (multiple entries)

IMAGE

The following group codes apply to image entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Image group codes

Group codes	Description
100	Subclass marker (AcDbRasterImage)
90	Class version
10	Insertion point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of insertion point (in WCS)
11	U-vector of a single pixel (points along the visual bottom of the image, starting at the insertion point) (in WCS) DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values U-vector (in WCS)
12	V-vector of a single pixel (points along the visual left side of the image, starting at the insertion point) (in WCS) DXF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of V-vector (in WCS)
13	Image size in pixels DXF: *U* value; APP: 2D point (*U* and *V* values)
23	DXF: *V* value of image size in pixels
340	Hard reference to imagedef object
70	Image display properties: 1 = Show image 2 = Show image when not aligned with screen 4 = Use clipping boundary 8 = Transparency is on
280	Clipping state: 0 = Off; 1 = On
281	Brightness value (0-100; default = 50)

Image group codes (*continued*)	
Group codes	**Description**
282	Contrast value (0-100; default = 50)
283	Fade value (0-100; default = 0)
360	Hard reference to imagedef_reactor object
71	Clipping boundary type. 1 = Rectangular; 2 = Polygonal
91	Number of clip boundary vertices that follow
14	Clip boundary vertex (in OCS) DXF: *X* value; APP: 2D point (multiple entries) NOTE 1) For rectangular clip boundary type, two opposite corners must be specified. Default is (–0.5,–0.5), (size.x–0.5, size.y–0.5). 2) For polygonal clip boundary type, three or more vertices must be specified. Polygonal vertices must be listed sequentially
24	DXF: *Y* value of clip boundary vertex (in OCS) (multiple entries)

INSERT

The following group codes apply to insert (block reference) entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Insert group codes	
Group codes	**Description**
100	Subclass marker (AcDbBlockReference)
66	Variable attributes-follow flag (optional; default = 0); if the value of attributes-follow flag is 1, a series of attribute entities is expected to follow the insert, terminated by a seqend entity
2	Block name
10	Insertion point (in OCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of insertion point (in OCS)

Insert group codes (*continued*)	
Group codes	**Description**
41	*X* scale factor (optional; default = 1)
42	*Y* scale factor (optional; default = 1)
43	*Z* scale factor (optional; default = 1)
50	Rotation angle (optional; default = 0)
70	Column count (optional; default = 1)
71	Row count (optional; default = 1)
44	Column spacing (optional; default = 0)
45	Row spacing (optional; default = 0)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

LEADER

The following group codes apply to leader entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Leader group codes	
Group codes	**Description**
100	Subclass marker (AcDbLeader)
3	Dimension style name
71	Arrowhead flag: 0 = Disabled; 1 = Enabled
72	Leader path type: 0 = Straight line segments; 1 = Spline

Leader group codes (*continued*)

Group codes	Description
73	Leader creation flag (default = 3): 0 = Created with text annotation 1 = Created with tolerance annotation 2 = Created with block reference annotation 3 = Created without any annotation
74	Hookline direction flag: 0 = Hookline (or end of tangent for a splined leader) is the opposite direction from the horizontal vector 1 = Hookline (or end of tangent for a splined leader) is the same direction as horizontal vector (see code 75)
75	Hookline flag: 0 = No hookline; 1 = Has a hookline
40	Text annotation height
41	Text annotation width
76	Number of vertices in leader (ignored for OPEN)
10	Vertex coordinates (one entry for each vertex) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of vertex coordinates
77	Color to use if leader's DIMCLRD = BYBLOCK
340	Hard reference to associated annotation (mtext, tolerance, or insert entity)
210	Normal vector DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of normal vector
211	"Horizontal" direction for leader DXF: *X* value; APP: 3D vector
221, 231	DXF: *Y* and *Z* values of "horizontal" direction for leader
212	Offset of last leader vertex from block reference insertion point DXF: *X* value; APP: 3D vector
222, 232	DXF: *Y* and *Z* values of offset
213	Offset of last leader vertex from annotation placement point DXF: *X* value; APP: 3D vector

Leader group codes (*continued*)

Group codes	Description
223, 233	DXF: *Y* and *Z* values of offset

Xdata belonging to the application ID "ACAD" follows a leader entity if any dimension overrides have been applied to this entity. See "Dimension Style Overrides" on page 79.

LINE

The following group codes apply to line entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Line group codes

Group codes	Description
100	Subclass marker (AcDbLine)
39	Thickness (optional; default = 0)
10	Start point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of start point (in WCS)
11	Endpoint (in WCS) DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of endpoint (in WCS)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

LWPOLYLINE

The following group codes apply to lwpolyline entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Lwpolyline group codes

Group codes	Description
100	Subclass marker (AcDbPolyline)
90	Number of vertices
70	Polyline flag (bit-coded); default is 0: 1 = Closed; 128 = Plinegen
43	Constant width (optional; default = 0). Not used if variable width (codes 40 and/or 41) is set
38	Elevation (optional; default = 0)
39	Thickness (optional; default = 0)
10	Vertex coordinates (in OCS), multiple entries; one entry for each vertex DXF: *X* value; APP: 2D point
20	DXF: *Y* value of vertex coordinates (in OCS), multiple entries; one entry for each vertex
40	Starting width (multiple entries; one entry for each vertex) (optional; default = 0; multiple entries). Not used if constant width (code 43) is set
41	End width (multiple entries; one entry for each vertex) (optional; default = 0; multiple entries). Not used if constant width (code 43) is set
42	Bulge (multiple entries; one entry for each vertex) (optional; default = 0)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

MLINE

The following group codes apply to mline entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Mline group codes

Group codes	Description
100	Subclass marker (AcDbMline)
2	String of up to 32 characters. The name of the style used for this mline. An entry for this style must exist in the MLINESTYLE dictionary. Do not modify this field without also updating the associated entry in the MLINESTYLE dictionary
340	Pointer-handle/ID of MLINESTYLE object
40	Scale factor
70	Justification: 0 = Top; 1 = Zero; 2 = Bottom
71	Flags (bit-coded values): 1 = Has at least one vertex (code 72 is greater than 0) 2 = Closed 4 = Suppress start caps 8 = Suppress end caps
72	Number of vertices
73	Number of elements in MLINESTYLE definition
10	Start point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of start point (in WCS)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)
11	Vertex coordinates (multiple entries; one entry for each vertex) DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of vertex coordinates

Mline group codes (*continued*)

Group codes	Description
12	Direction vector of segment starting at this vertex (multiple entries; one for each vertex) DXF: *X* value; APP: 3D vector
22, 32	DXF: *Y* and *Z* values of direction vector of segment starting at this vertex
13	Direction vector of miter at this vertex (multiple entries: one for each vertex) DXF: *X* value; APP: 3D vector
23, 33	DXF: *Y* and *Z* values of direction vector of miter
74	Number of parameters for this element (repeats for each element in segment)
41	Element parameters (repeats based on previous code 74)
75	Number of area fill parameters for this element (repeats for each element in segment)
42	Area fill parameters (repeats based on previous code 75)

The group code 41 parameterization is a list of real values, one real per group code 41. The list may contain zero or more items. The first group code 41 value is the distance from the segment vertex along the miter vector to the point where the line element's path intersects the miter vector. The next group code 41 value is the distance along the line element's path from the point defined by the first group 41 to the actual start of the line element. The next is the distance from the start of the line element to the first break (or cut) in the line element. The successive group code 41 values continue to list the start and stop points of the line element in this segment of the mline. Linetypes do not affect group 41 lists.

The group code 42 parameterization is also a list of real values. Similar to the 41 parameterization, it describes the parameterization of the fill area for this mline segment. The values are interpreted identically to the 41 parameters and when taken as a whole for all line elements in the mline segment, they define the boundary of the fill area for the mline segment.

A common example of the use of the group code 42 mechanism is when an unfilled mline crosses over a filled mline and mledit is used to cause the filled mline to appear unfilled in the crossing area. This would result in two group 42s for each line element in the affected mline segment; one for the fill stop and one for the fill start.

The 2 group codes in mline entities and mlinestyle objects are redundant fields. These groups should not be modified under any circumstances, although it is safe to read them and use their values. The correct fields to modify are as follows:

Mline	The 340 group in the same object, which indicates the proper MLINESTYLE object.
Mlinestyle	The 3 group value in the MLINESTYLE dictionary, which precedes the 350 group that has the handle or entity name of the current mlinestyle.

MTEXT

The following group codes apply to mtext entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Mtext group codes

Group codes	Description
100	Subclass marker (AcDbMText)
10	Insertion point DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of insertion point
40	Nominal (initial) text height
41	Reference rectangle width
71	Attachment point: 1 = Top left; 2 = Top center; 3 = Top right 4 = Middle left; 5 = Middle center; 6 = Middle right 7 = Bottom left; 8 = Bottom center; 9 = Bottom right

Mtext group codes (*continued*)

Group codes	Description
72	Drawing direction: 1 = Left to right 3 = Top to bottom 5 = By style (the flow direction is inherited from the associated text style)
1	Text string. If the text string is less than 250 characters, all characters appear in group 1. If the text string is greater than 250 characters, the string is divided into 250-character chunks, which appear in one or more group 3 codes. If group 3 codes are used, the last group is a group 1 and has fewer than 250 characters
3	Additional text (always in 250-character chunks) (optional)
7	Text style name (STANDARD if not provided) (optional)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)
11	*X*-axis direction vector (in WCS) DXF: *X* value; APP: 3D vector A group code 50 (rotation angle in radians) passed as DXF input is converted to the equivalent direction vector (if both a code 50 and codes 11, 21, 31 are passed, the last one wins). This is provided as a convenience for conversions from text objects
21, 31	DXF: *Y* and *Z* values of *X*-axis direction vector (in WCS)
42	Horizontal width of the characters that make up the mtext entity. This value will always be equal to or less than the value of group code 41 (read-only, ignored if supplied)
43	Vertical height of the mtext entity (read-only, ignored if supplied)
50	Rotation angle in radians
73	Mtext line spacing style (optional): 1 = At least (taller characters will override) 2 = Exact (taller characters will not override)
44	Mtext line spacing factor (optional): Percentage of default (3-on-5) line spacing to be applied. Valid values range from 0.25 to 4.00

Xdata with the "DCO15" application ID may follow an mtext entity. This contains information related to the dbConnect feature.

OLEFRAME

The following group codes apply to oleframe entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Oleframe group codes

Group codes	Description
100	Subclass marker (AcDbOleFrame)
70	OLE version number
90	Length of binary data
310	Binary data (multiple lines)
1	End of OLE data (the string “OLE”)

OLE2FRAME

The following group codes apply to ole2frame entities. This information is read-only. During OPEN, the values are ignored because they are part of the OLE binary object, and are obtained via access functions. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Ole2frame group codes

Group codes	Description
100	Subclass marker (AcDbOle2Frame)
70	OLE version number
3	Length of binary data
10	Upper-left corner (WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of upper-left corner (in WCS)
11	Lower-right corner (WCS) DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of lower-right corner (in WCS)
71	OLE object type, 1 = Link; 2 = Embedded; 3 = Static
72	Tile mode descriptor: 0 = Object resides in model space 1 = Object resides in paper space
90	Length of binary data
310	Binary data (multiple lines)
1	End of OLE data (the string "OLE")

Sample DXF output:

```
    OLE2FRAME
      5
    2D
    100
    AcDbEntity
     67
         1
      8
    0
    100
    AcDbOle2Frame
     70
        2
      3
    Paintbrush Picture
     10
    4.43116
     20
    5.665992
     30
    0.0
     11
    6.4188
     21
    4.244939
     31
    0.0
     71
        2
     72
        1
     90
       23680
    310
0155764BD60082B91140114B08C8F9A916400000000000000000506DC0D0D9AC

    310

1940114B08C8F9A916400000000000000000506DC0D0D9AC194002303E5CD1FA

    310

1040000000000000000764BD60082B9114002303E5CD1FA1040000000000000
    ...
    ...
```

AutoLISP **entnext** function sample output:

```
Command:  (setq e (entget e3))
    ((-1 . <Entity name: 7d50428>) (0 . "OLE2FRAME") (5 . "2D")
    (100 . "AcDbEntity") (67 . 1) (8 . "0") (100 . "AcDbOle2Frame")
    (70 . 2) (3 "Paintbrush Picture") (10 4.43116 5.66599 0.0)
    (11 6.4188 4.24494 0.0) (71 . 2) (72 . 1))
```

POINT

The following group codes apply to point entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Point group codes

Group codes	Description
100	Subclass marker (AcDbPoint)
10	Point location (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of point location (in WCS)
39	Thickness (optional; default = 0)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)
50	Angle of the *X* axis for the UCS in effect when the point was drawn (optional, default = 0); used when PDMODE is nonzero

POLYLINE

The following group codes apply to polyline entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Polyline group codes

Group codes	Description
100	Subclass marker (AcDb2dPolyline or AcDb3dPolyline)
66	Obsolete; formerly an “entities follow flag” (optional; ignore if present)

Polyline group codes (*continued*)

Group codes	Description
10	DXF: always 0 APP: a "dummy" point; the *X* and *Y* values are always 0, and the *Z* value is the polyline's elevation (in OCS when 2D, WCS when 3D)
20	DXF: always 0
30	DXF: polyline's elevation (in OCS when 2D; WCS when 3D)
39	Thickness (optional; default = 0)
70	Polyline flag (bit-coded; default = 0): 1 = This is a closed polyline (or a polygon mesh closed in the M direction) 2 = Curve-fit vertices have been added 4 = Spline-fit vertices have been added 8 = This is a 3D polyline 16 = This is a 3D polygon mesh 32 = The polygon mesh is closed in the N direction 64 = The polyline is a polyface mesh 128 = The linetype pattern is generated continuously around the vertices of this polyline
40	Default start width (optional; default = 0)
41	Default end width (optional; default = 0)
71	Polygon mesh M vertex count (optional; default = 0)
72	Polygon mesh N vertex count (optional; default = 0)
73	Smooth surface M density (optional; default = 0)
74	Smooth surface N density (optional; default = 0)
75	Curves and smooth surface type (optional; default = 0); integer codes, not bit-coded: 0 = No smooth surface fitted 5 = Quadratic B-spline surface 6 = Cubic B-spline surface 8 = Bezier surface
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

Xdata with the "AUTOCAD_POSTSCRIPT_FIGURE" application ID may follow a polyline entity. This contains information related to PostScript images and PostScript fill information.

Polyface Meshes

A polyface mesh is represented in DXF as a variant of a polyline entity. The polyline header is identified as introducing a polyface mesh by the presence of the 64 bit in the polyline flags (70) group. The 71 group specifies the number of vertices in the mesh, and the 72 group specifies the number of faces. Although these counts are correct for all meshes created with the PFACE command, applications are not required to place correct values in these fields. Following the polyline header is a sequence of vertex entities that specify the vertex coordinates, followed by faces that compose the mesh.

The AutoCAD entity structure imposes a limit on the number of vertices that a given face entity can specify. You can represent more complex polygons by decomposing them into triangular wedges. Their edges should be made invisible to prevent visible artifacts of this subdivision from being drawn. The PFACE command performs this subdivision automatically, but when applications generate polyface meshes directly, the applications must do this themselves. The number of vertices per face is the key parameter in this subdivision process. The PFACEVMAX system variable provides an application with the number of vertices per face entity. This value is read-only and is set to 4.

Polyface meshes created with the PFACE command are always generated with all the vertex coordinate entities first, followed by the face definition entities. The code within AutoCAD that processes polyface meshes requires this ordering. Programs that generate polyface meshes in DXF should generate all the vertices, and then all the faces. However, programs that read polyface meshes from DXF should be tolerant of odd vertex and face ordering.

RAY

The following group codes apply to ray entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Ray group codes

Group codes	Description
100	Subclass marker (AcDbRay)
10	Start point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of start point (in WCS)
11	Unit direction vector (in WCS) DXF: *X* value; APP: 3D vector
21, 31	DXF: *Y* and *Z* values of unit direction vector (in WCS)

REGION

The following group codes apply to region entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Region group codes

Group codes	Description
100	Subclass marker (AcDbModelerGeometry)
70	Modeler format version number (currently = 1)
1	Proprietary data (multiple lines < 255 characters each)
3	Additional lines of proprietary data (if previous group 1 string is greater than 255 characters) (optional)

SEQEND

The following group codes apply to seqend entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Seqend group codes

Group codes	Description
–2	APP: name of entity that began the sequence. This entity marks the end of vertex (vertex type name) for a polyline, or the end of attribute entities (attrib type name) for an insert entity that has attributes (indicated by 66 group present and nonzero in insert entity). This code is not saved in a DXF file

SHAPE

The following group codes apply to shape entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Shape group codes

Group codes	Description
100	Subclass marker (AcDbShape)
39	Thickness (optional; default = 0)
10	Insertion point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of insertion point (in WCS)
40	Size
2	Shape name
50	Rotation angle (optional; default = 0)
41	Relative X scale factor (optional; default = 1)

Shape group codes (*continued*)	
Group codes	**Description**
51	Oblique angle (optional; default = 0)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

SOLID

The following group codes apply to solid entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Solid group codes	
Group codes	**Description**
100	Subclass marker (AcDbTrace)
10	First corner DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of first corner
11	Second corner DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of second corner
12	Third corner XF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of third corner
13	Fourth corner. If only three corners are entered to define the SOLID, then the fourth corner coordinate is the same as the third. DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of fourth corner
39	Thickness (optional; default = 0)

Solid group codes (*continued*)	
Group codes	**Description**
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

SPLINE

The following group codes apply to spline entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Spline group codes	
Group codes	**Description**
100	Subclass marker (AcDbSpline)
210	Normal vector (omitted if the spline is nonplanar) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of normal vector (optional)
70	Spline flag (bit coded): 1 = Closed spline 2 = Periodic spline 4 = Rational spline 8 = Planar 16 = Linear (planar bit is also set)
71	Degree of the spline curve
72	Number of knots
73	Number of control points
74	Number of fit points (if any)
42	Knot tolerance (default = 0.0000001)
43	Control-point tolerance (default = 0.0000001)
44	Fit tolerance (default = 0.0000000001)

Spline group codes (*continued*)	
Group codes	**Description**
12	Start tangent—may be omitted (in WCS) DXF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of start tangent—may be omitted (in WCS)
13	End tangent—may be omitted (in WCS) DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of end tangent—may be omitted (in WCS)
40	Knot value (one entry per knot)
41	Weight (if not 1); with multiple group pairs, they are present if all are not 1
10	Control points (in WCS); one entry per control point DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of control points (in WCS); one entry per control point
11	Fit points (in WCS); one entry per fit point DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of fit points (in WCS); one entry per fit point

TEXT

The following group codes apply to text entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Text group codes	
Group codes	**Description**
100	Subclass marker (AcDbText)
39	Thickness (optional; default = 0)
10	First alignment point (in OCS) DXF: *X* value; APP: 3D point

Text group codes (*continued*)

Group codes	Description
20, 30	DXF: *Y* and *Z* values of first alignment point (in OCS)
40	Text height
1	Default value (the string itself)
50	Text rotation (optional; default = 0)
41	Relative *X* scale factor—width (optional; default = 1) This value is also adjusted when fit-type text is used
51	Oblique angle (optional; default = 0)
7	Text style name (optional, default = STANDARD)
71	Text generation flags (optional, default = 0): 2 = Text is backward (mirrored in X) 4 = Text is upside down (mirrored in Y)
72	Horizontal text justification type (optional, default = 0) integer codes (not bit-coded) 0 = Left; 1= Center; 2 = Right 3 = Aligned (if vertical alignment = 0) 4 = Middle (if vertical alignment = 0) 5 = Fit (if vertical alignment = 0) See the Group 72 and 73 integer codes table for clarification
11	Second alignment point (in OCS) (optional) DXF: *X* value; APP: 3D point This value is meaningful only if the value of a 72 or 73 group is nonzero (if the justification is anything other than baseline/left)
21, 31	DXF: *Y* and *Z* values of second alignment point (in OCS) (optional)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)
100	Subclass marker (AcDbText)
73	Vertical text justification type (optional, default = 0): integer codes (not bit-coded): 0 = Baseline; 1 = Bottom; 2 = Middle; 3 = Top See the Group 72 and 73 integer codes table for clarification

The following table describes the group codes 72 (horizontal alignment) and 73 (vertical alignment) in greater detail.

Group 72 and 73 integer codes

Group 73	Group 72 0	1	2	3	4	5
3 (top)	TLeft	TCenter	TRight			
2 (middle)	MLeft	MCenter	MRight			
1 (bottom)	BLeft	BCenter	BRight			
0 (baseline)	Left	Center	Right	Aligned	Middle	Fit

If group 72 and/or 73 values are nonzero then the first alignment point values are ignored and AutoCAD calculates new values based on the second alignment point and the length and height of the text string itself (after applying the text style). If the 72 and 73 values are zero or missing, then the second alignment point is meaningless.

TOLERANCE

The following group codes apply to tolerance entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Tolerance group codes

Group codes	Description
100	Subclass marker (AcDbFcf)
3	Dimension style name
10	Insertion point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of insertion point (in WCS)
1	String representing the visual representation of the tolerance

Tolerance group codes (*continued*)

Group codes	Description
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)
11	X-axis direction vector (in WCS) DXF: *X* value; APP: 3D vector
21, 31	DXF: *Y* and *Z* values of X-axis direction vector (in WCS)

TRACE

The following group codes apply to trace entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Trace group codes

Group codes	Description
100	Subclass marker (AcDbTrace)
10	First corner (in OCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of first corner (in OCS)
11	Second corner (in OCS) DXF: *X* value; APP: 3D point
21, 31	DXF: *Y* and *Z* values of second corner (in OCS)
12	Third corner (in OCS) DXF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of third corner (in OCS)
13	Fourth corner (in OCS) DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of fourth corner (in OCS)

Trace group codes (*continued*)	
Group codes	**Description**
39	Thickness (optional; default = 0)
210	Extrusion direction (optional; default = 0, 0, 1) DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction (optional)

VERTEX

The following group codes apply to vertex entities. In addition to the group codes described here, see "Common Group Codes for Entities" on page 62. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Vertex group codes	
Group codes	**Description**
100	Subclass marker (AcDbVertex)
100	Subclass marker (AcDb2dVertex or AcDb3dPolylineVertex)
10	Location point (in OCS when 2D, and WCS when 3D) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of location point (in OCS when 2D, and WCS when 3D)
40	Starting width (optional; default is 0)
41	Ending width (optional; default is 0)
42	Bulge (optional; default is 0). The bulge is the tangent of one fourth the included angle for an arc segment, made negative if the arc goes clockwise from the start point to the endpoint. A bulge of 0 indicates a straight segment, and a bulge of 1 is a semicircle

Vertex group codes (*continued*)

Group codes	Description
70	Vertex flags: 1 = Extra vertex created by curve-fitting 2 = Curve-fit tangent defined for this vertex. A curve-fit tangent direction of 0 may be omitted from DXF output but is significant if this bit is set 4 = Not used 8 = Spline vertex created by spline-fitting 16 = Spline frame control point 32 = 3D polyline vertex 64 = 3D polygon mesh 128 = Polyface mesh vertex
50	Curve fit tangent direction
71	Polyface mesh vertex index (optional; present only if nonzero)
72	Polyface mesh vertex index (optional; present only if nonzero)
73	Polyface mesh vertex index (optional; present only if nonzero)
74	Polyface mesh vertex index (optional; present only if nonzero)

Every vertex that is part of a polyface mesh has its vertex flag 128 bit set. If the entity supplies the coordinate of a vertex of the mesh, its 64 bit is set as well, and the 10, 20, 30 groups give the vertex coordinate. The vertex index values are determined by the order in which the vertex entities appear within the polyline, with the first being numbered 1.

If the vertex defines a face of the mesh, its vertex flags group has the 128 bit set but not the 64 bit. In this case, the 10, 20, 30 (location) groups of the face entity are irrelevant and are always written as 0 in a DXF file. The vertex indexes that define the mesh are given by 71, 72, 73, and 74 group codes, the values of which specify one of the previously defined vertexes by index. If the index is negative, the edge that begins with that vertex is invisible. The first 0 vertex marks the end of the vertices of the face.

VIEWPORT

The following group codes apply to viewport entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Viewport group codes

Group codes	Description
100	Subclass marker (AcDbViewport)
10	Center point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of center point (in WCS)
40	Width in paper space units
41	Height in paper space units
68	Viewport status field: –1 = On, but is fully off screen, or is one of the viewports that is not active because the $MAXACTVP count is currently being exceeded. 0 = Off <positive value > = On and active. The value indicates the order of stacking for the viewports, where 1 is the active viewport, 2 is the next, and so forth
69	Viewport ID
12	View center point (in DCS) DXF: *X* value; APP: 2D point
22	DXF: View center point *Y* value (in DCS)
13	Snap base point DXF: *X* value; APP: 2D point
23	DXF: Snap base point *Y* value
14	Snap spacing DXF: *X* value; APP: 2D point
24	DXF: Snap spacing *Y* value

Viewport group codes (*continued*)

Group codes	Description
15	Grid spacing DXF: *X* value; APP: 2D point
25	DXF: Grid spacing *Y* value
16	View direction vector (in WCS) DXF: *X* value; APP: 3D vector
26, 36	DXF: *Y* and *Z* values of view direction vector (in WCS)
17	View target point (in WCS) DXF: *X* value; APP: 3D vector
27, 37	DXF: *Y* and *Z* values of view target point (in WCS)
42	Perspective lens length
43	Front clip plane *Z* value
44	Back clip plane *Z* value
45	View height (in model space units)
50	Snap angle
51	View twist angle
72	Circle zoom percent
331	Frozen layer object ID/handle (multiple entries may exist) (optional)

Viewport group codes (*continued*)	
Group codes	**Description**
90	Viewport status bit-coded flags: 1 (0x1) = Enables perspective mode 2 (0x2) = Enables front clipping 4 (0x4) = Enables back clipping 8 (0x8) = Enables UCS follow 16 (0x10) = Enables front clip not at eye 32 (0x20) = Enables UCS icon visibility 64 (0x40) = Enables UCS icon at origin 128 (0x80) = Enables fast zoom 256 (0x100) = Enables snap mode 512 (0x200) = Enables grid mode 1024 (0x400) = Enables isometric snap style 2048 (0x800) = Enables hide plot mode 4096 (0x1000) = kIsoPairTop. If set and kIsoPairRight is not set, then isopair top is enabled. If both kIsoPairTop and kIsoPairRight are set, then isopair left is enabled 8192 (0x2000) = kIsoPairRight. If set and kIsoPairTop is not set, then isopair right is enabled 16384 (0x4000) = Enables viewport zoom locking 32768 (0x8000) = Currently always enabled 65536 (0x10000) = Enables non-rectangular clipping 131072 (0x20000) = Turns the viewport off
340	Hard-pointer ID/handle to entity that serves as the viewport's clipping boundary (only present if viewport is non-rectangular)
1	Plot style sheet name assigned to this viewport
281	Render mode: 0 = 2D Optimized (classic 2D) 1 = Wireframe 2 = Hidden line 3 = Flat shaded 4 = Gouraud shaded 5 = Flat shaded with wireframe 6 = Gouraud shaded with wireframe All rendering modes other than 2D Optimized engage the new 3D graphics pipeline. These values directly correspond to the SHADEMODE command and the AcDbAbstractViewTableRecord::RenderMode enum
71	UCS per viewport flag: 0 = The UCS will not change when this viewport becomes active. 1 = This viewport stores its own UCS which will become the current UCS whenever the viewport is activated

Viewport group codes (*continued*)	
Group codes	**Description**
74	Display UCS icon at UCS origin flag: Controls whether UCS icon represents viewport UCS or current UCS (these will be different if UCSVP is 1 and viewport is not active). However, this field is currently being ignored and the icon always represents the viewport UCS
110	UCS origin DXF: *X* value; APP: 3D point
120, 130	DXF: *Y* and *Z* values of UCS origin
111	UCS *X*-axis DXF: *X* value; APP: 3D vector
121, 131	DXF: *Y* and *Z* values of UCS *X*-axis
112	UCS *Y*-axis DXF: *X* value; APP: 3D vector
122, 132	DXF: *Y* and *Z* values of UCS *Y*-axis
345	ID/handle of AcDbUCSTableRecord if UCS is a named UCS. If not present, then UCS is unnamed
346	ID/handle of AcDbUCSTableRecord of base UCS if UCS is orthographic (79 code is non-zero). If not present and 79 code is non-zero, then base UCS is taken to be WORLD
79	Orthographic type of UCS: 0 = UCS is not orthographic 1 = Top; 2 = Bottom 3 = Front; 4 = Back 5 = Left; 6 = Right
146	Elevation
170	ShadePlot mode: 0 = As Displayed 1 = Wireframe 2 = Hidden 3 = Rendered

Note The ZOOM XP factor is calculated with the following formula: group_41 / group_45 (or pspace_height / mspace_height).

XLINE

The following group codes apply to xline entities. In addition to the group codes described here, see “Common Group Codes for Entities” on page 62. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Xline group codes

Group codes	Description
100	Subclass marker (AcDbXline)
10	First point (in WCS) DXF: *X* value; APP: 3D point
20, 30	DXF: *Y* and *Z* values of first point (in WCS)
11	Unit direction vector (in WCS) DXF: *X* value; APP: 3D vector
21, 31	DXF: *Y* and *Z* values of unit direction vector (in WCS)

OBJECTS Section

7

This chapter presents the group codes that apply to non-graphical objects. These codes are found in the OBJECTS section of a DXF™ file and are used by AutoLISP® and ObjectARX™ applications in entity definition lists.

In this chapter

OBJECT Section Group Codes

Objects are similar to entities, except that they have no graphical or geometric meaning. All objects that are not entities or symbol table records or symbol tables are stored in this section. This section represents a homogeneous heap of objects with topological ordering of objects by ownership, such that the owners always appear before the objects they own.

Object Ownership

The root owner of most objects appearing in the OBJECTS section is the named object dictionary, which is, therefore, always the first object that appears in this section. Objects that are not owned by the named object dictionary are owned by other entities, objects, or symbol table entries. Objects in this section may be defined by AutoCAD® or by applications with access to ObjectARX™ API. The DXF names of application-defined object types should always be associated with a class name in the CLASS section of the DXF file, or else the object record cannot be bound to the application that will interpret it.

As with other dictionaries, the named-object dictionary record consists solely of associated pairs of entry names and hard ownership pointer references to the associated object.

To avoid name collision between objects, developers should always use their registered developer prefix for their entries.

Common Group Codes for Objects

The following table shows group codes that apply to virtually all nongraphical objects. When you refer to a table of group codes by object type, a list of codes associated with a *specific* object, keep in mind that the codes shown here can also be present. Some of the group codes are included with an object only if the object has nondefault values for those group code properties. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

Common object group codes

Group codes	Description
0	Object type

Common object group codes (*continued*)	
Group codes	**Description**
5	Handle
102	Start of application-defined group "{*application_name*" (optional)
application-defined codes	Codes and values within the 102 groups are application defined (optional)
102	End of group, "}" (optional)
102	"{ACAD_REACTORS" indicates the start of the AutoCAD® persistent reactors group. This group exists only if persistent reactors have been attached to this object (optional)
330	Soft-pointer ID/handle to owner dictionary (optional)
102	End of group, "}" (optional)
102	"{ACAD_XDICTIONARY" indicates the start of an extension dictionary group. This group exists only if persistent reactors have been attached to this object (optional)
360	Hard-owner ID/handle to owner dictionary (optional)
102	End of group, "}" (optional)
330	Soft-pointer ID/handle to owner object

ACAD_PROXY_OBJECT

The following group codes apply to ACAD_PROXY_OBJECT objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

ACAD_PROXY_OBJECT group codes	
Group codes	**Description**
100	DXF™ Subclass marker (AcDbProxyObject)
90	DXF: Proxy object class ID (always 499)

ACAD_PROXY_OBJECT group codes (*continued*)

Group codes	Description
91	DXF: Application object's class ID. Class IDs are based on the order of the class in the CLASSES section. The first class is given the ID of 500, the next is 501, and so on
93	DXF: Size of object data in bits
310	DXF: Binary object data (multiple entries can appear) (optional)
330 or 340 or 350 or 360	DXF: An object ID (multiple entries can appear) (optional)
94	DXF: 0 (indicates end of object ID section)
95	DXF: Object drawing format when it becomes a proxy (a 32-bit unsigned integer): Low word is AcDbDwgVersion High word is MaintenanceReleaseVersion
70	DXF: Original custom object data format: 0 = DWG format 1 = DXF format

The 92 field is not used for AcDbProxyObject. Objects of this class never have graphics.

ACDBDICTIONARYWDFLT

The following group codes are used by ACDBDICTIONARYWDFLT objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

ACDBDICTIONARYWDFLT group codes

Group codes	Description
0	Object name (ACDBDICTIONARYWDFLT)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"

ACDBDICTIONARYWDFLT group codes (*continued*)	
Group codes	**Description**
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always "}"
330	Soft-owner ID/handle to owner object
100	Subclass marker (AcDbDictionary)
281	Duplicate record cloning flag (determines how to merge duplicate entries): 0 = Not applicable 1 = Keep existing 2 = Use clone 3 = <xref>0<name> 4 = 0<name> 5 = Unmangle name
3	Entry name (one for each entry)
350	Soft-owner ID/handle to entry object (one for each entry)
100	Subclass marker (AcDbDictionaryWithDefault)
340	Hard pointer to default object ID/handle (currently only used for plot style dictionary's default entry, named "Normal")

ACDBPLACEHOLDER

The following group codes are used by the ACDBPLACEHOLDER objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

ACDBPLACEHOLDER group codes	
Group codes	**Description**
0	Object name (ACDBPLACEHOLDER)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"

ACDBPLACEHOLDER group codes (*continued*)	
Group codes	**Description**
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always "}"
330	Soft-pointer ID/handle to owner object

DICTIONARY

The following group codes are used by DICTIONARY objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

DICTIONARY group codes	
Group codes	**Description**
0	Object name (DICTIONARY)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always "}"
330	Soft-pointer ID/handle to owner object
100	Subclass marker (AcDbDictionary)
280	Hard-owner flag. If set to 1, indicates that elements of the dictionary are to be treated as hard-owned
281	Duplicate record cloning flag (determines how to merge duplicate entries): 0 = Not applicable 1 = Keep existing 2 = Use clone 3 = <xref>0<name> 4 = 0<name> 5 = Unmangle name

DICTIONARY group codes (*continued*)

Group codes	Description
3	Entry name (one for each entry) (optional)
350	Soft-owner ID/handle to entry object (one for each entry) (optional)

AutoCAD® maintains items such as mline styles and group definitions as objects in dictionaries. The following sections describe the AutoCAD object group codes maintained in dictionaries; however, other applications are free to create and use their own dictionaries as they see fit. The prefix "ACAD_" is reserved for use by AutoCAD applications.

DICTIONARYVAR

The following group codes are used by DICTIONARYVAR objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

DICTIONARYVAR group codes

Group codes	Description
0	Object name (DICTIONARYVAR)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"
330	Soft-pointer ID/handle to owner dictionary (ACDBVARIABLEDICTIONARY)
102	End of persistent reactors group, always "}"
100	Subclass marker (DictionaryVariables)
280	Object schema number (currently set to 0)
1	Value of variable

DICTIONARYVAR objects are used by AutoCAD as a means to store named values in the database for **setvar/getvar** purposes without the need to add entries to the DXF™ HEADER section. Currently, the system variables that are stored as DICTIONARYVAR objects are: DIMADEC, DIMDSEP, INDEXCTL, PROJECTNAME, and XCLIPFRAME.

DIMASSOC

The following group codes are used by DIMASSOC objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

DIMASSOC group codes

Group codes	Description
0	Object name (DIMASSOC)
5	Handle
102	Persistent reactors group; always "{ACAD_REACTORS}"
330	Soft-pointer ID
100	Subclass marker (AcDbDimAssoc)
330	ID of dimension object
90	Associativity flag 1 = First point reference 2 = Second point reference 4 = Third point reference 8 = Fourth point reference
70	Trans-space flag (true/false)
71	Rotated Dimension type (parallel, perpendicular)
1	Class name (AcDbOsnapPointRef)
72	Object Osnap type (Start, End, Mid, Cen, etc.)
331	ID of main object (geometry)
73	SubentType of main object (edge, face)

DIMASSOC group codes (*continued*)

Group codes	Description
91	GsMarker of main object (index)
301	Handle (string) of Xref object
40	Geometry parameter for Near Osnap
10	Osnap point in WCS; X value
20	Osnap point in WCS; Y value
30	Osnap point in WCS; Z value
332	ID of intersection object (geometry)
74	SubentType of intersction object (edge/face)
92	GsMarker of intersection object (index)
302	Handle (string) of intersection Xref object
75	hasLastPointRef flag (true/false)

DIMASSOC objects implement associative dimensions by specifying an association between a dimension object and drawing geometry objects. An associative dimension is a dimension that will automatically update when the associated geometry is modified.

GROUP

The following group codes are used by GROUP objects. In addition to the group codes described here, see “Common Group Codes for Objects” on page 120. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

GROUP group codes

Group codes	Description
0	Object name (GROUP)
5	Handle

GROUP group codes (*continued*)

Group codes	Description
102	Start of persistent reactors group; always "{ACAD_REACTORS" (persistent reactors group appears in all dictionaries except the main dictionary)
330	Soft-pointer ID/handle to owner dictionary. For GROUP objects this is always the ACAD_GROUP entry of the named object dictionary
102	End of persistent reactors group, always "}"
330	Soft-pointer ID/handle to owner object
100	Subclass marker (AcDbGroup)
300	Group description
70	"Unnamed" flag: 1 = Unnamed; 0 = Named
71	Selectability flag: 1 = Selectable; 0 = Not selectable
340	Hard-pointer handle to entity in group (one entry per object)

IDBUFFER

The following group codes are used by IDBUFFER objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

IDBUFFER group codes

Group codes	Description
100	Subclass marker (AcDbIdBuffer)
330	Soft-pointer reference to entity (multiple entries may exist)

The IDBUFFER object is a utility object that is just a list of references to objects.

IMAGEDEF

The following group codes are used by IMAGEDEF objects. In addition to the group codes described here, see “Common Group Codes for Objects” on page 120. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

IMAGEDEF group codes

Group codes	Description
0	Object name (IMAGEDEF)
5	Handle
102	Start of persistent reactors group; always “{ACAD_REACTORS”
330	Soft-pointer ID/handle to the ACAD_IMAGE_DICT dictionary
330	Soft-pointer ID/handle to IMAGEDEF_REACTOR object (multiple entries; one for each instance)
102	End of persistent reactors group, always “}”
100	Subclass marker (AcDbRasterImageDef)
90	Class version 0
1	File name of image
10	Image size in pixels DXF: *U* value; APP: 2D point (*U* and *V* values)
20	DXF: *V* value of image size in pixels
11	Default size of one pixel in AutoCAD units DXF: *U* value; APP: 2D point (*U* and *V* values)
12	DXF: *V* value of pixel size
280	Image-is-loaded flag. 0 = Unloaded; 1 = Loaded
281	Resolution units. 0 = No units; 2 = Centimeters; 5 = Inch

IMAGEDEF_REACTOR

The following group codes are used by IMAGEDEF_REACTOR objects. In addition to the group codes described here, see “Common Group Codes for Objects” on page 120. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

IMAGEDEF_REACTOR group codes

Group codes	Description
0	Object name (IMAGEDEF_REACTOR)
5	Handle
100	Subclass marker (AcDbRasterImageDefReactor)
90	Class version 2
330	Object ID for associated image object

LAYER_INDEX

The following group codes are used by LAYER_INDEX objects. In addition to the group codes described here, see “Common Group Codes for Objects” on page 120. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

LAYER_INDEX group codes

Group codes	Description
0	Object name (LAYER_INDEX)
5	Handle
102	Start of persistent reactors group; always “{ACAD_REACTORS”
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always “}”
100	Subclass marker (AcDbIndex)

LAYER_INDEX group codes (*continued*)

Group codes	Description
40	Time stamp (Julian date)
100	Subclass marker (AcDbLayerIndex)
8	Layer name (multiple entries may exist)
360	Hard-owner reference to IDBUFFER (multiple entries may exist)
90	Number of entries in the IDBUFFER list (multiple entries may exist)

LAYER_FILTER

The following group codes are used by LAYER_FILTER objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

LAYER_FILTER group codes

Group codes	Description
0	Object name (LAYER_FILTER)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always "}"
100	Subclass marker (AcDbFilter)
100	Subclass marker (AcDbLayerFilter)
8	Layer name (multiple entries may exist)

LAYOUT

The following group codes are used by LAYOUT objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

LAYOUT group codes

Group codes	Description
0	Object name (LAYOUT)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always "}"
330	Soft-pointer ID/handle to owner object
100	Subclass marker (AcDbPlotSettings)
plotsettings object group codes	For group codes and descriptions following the AcDbPlotSettings marker, see "PLOTSETTINGS" on page 141
100	Subclass marker (AcDbLayout)
1	Layout name
70	Flag (bit-coded) to control the following: 1 = Indicates the PSLTSCALE value for this layout when this layout is current 2 = Indicates the LIMCHECK value for this layout when this layout is current
71	Tab order. This number is an ordinal indicating this layout's ordering in the tab control that is attached to the AutoCAD drawing frame window. Note that the "Model" tab always appears as the first tab regardless of its tab order
10	Minimum limits for this layout (defined by LIMMIN while this layout is current) DXF: *X* value; APP: 2D point
20	DXF: *Y* value of minimum limits

LAYOUT group codes (*continued*)

Group codes	Description
11	Maximum limits for this layout (defined by LIMMAX while this layout is current): DXF: *X* value; APP: 2D point
21	DXF: *Y* value of maximum limits
12	Insertion base point for this layout (defined by INSBASE while this layout is current): DXF: *X* value; APP: 3D point
22, 32	DXF: *Y* and *Z* values of the insertion base point
14	Minimum extents for this layout (defined by EXTMIN while this layout is current): DXF: *X* value; APP: 3D point
24, 34	DXF: *Y* and *Z* values of the minimum extents
15	Maximum extents for this layout (defined by EXTMAX while this layout is current): DXF: *X* value; APP: 3D point
25, 35	DXF: *Y* and *Z* values of the maximum extents
146	Elevation
13	UCS origin DXF: *X* value; APP: 3D point
23, 33	DXF: *Y* and *Z* values of UCS origin
16	UCS *X*-axis DXF: *X* value; APP: 3D vector
26, 36	DXF: *Y* and *Z* values of UCS *X*-axis
17	UCS *Y*-axis DXF: *X* value; APP: 3D vector
27, 37	DXF: *Y* and *Z* values of UCS *Y*-axis
76	Orthographic type of UCS 0 = UCS is not orthographic 1 = Top; 2 = Bottom 3 = Front; 4 = Back 5 = Left; 6 = Right

LAYOUT group codes (*continued*)

Group codes	Description
330	ID/handle to this layout's associated paper space block table record
331	ID/handle to the viewport that was last active in this layout when the layout was current
345	ID/handle of AcDbUCSTableRecord if UCS is a named UCS. If not present, then UCS is unnamed
346	ID/handle of AcDbUCSTableRecord of base UCS if UCS is orthographic (76 code is non-zero). If not present and 76 code is non-zero, then base UCS is taken to be WORLD

MATERIAL

The following group codes are used by MATERIAL objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

MATERIAL group codes

Group codes	Description
0	Object name (MATERIAL)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS" (The persistent reactors group appears in all dictionaries except the main dictionary.)
330	Soft-pointer ID/handle to owner dictionary. For MATERIAL objects, this is always the ACAD_MATERIAL entry of the named object dictionary
102	End of persistent reactors group; always "}"
100	Subclass marker (AcDbMaterial)
1	Material name (string)
2	Description (string, default null string)

MATERIAL group codes (*continued*)

Group codes	Description
70	Ambient color method (default = 0): 0 = Use current color 1 = Override current color
40	Ambient color factor (real, default = 1.0, valid range is 0.0 to 1.0)
90	Ambient color value (unsigned 32-bit integer representing an AcCmEntityColor)
71	Diffuse color method (default = 0): 0 = Use current color 1 = Override current color
41	Diffuse color factor (real, default = 1.0, valid range is 0.0 to 1.0)
91	Diffuse color value (unsigned 32-bit integer representing an AcCmEntityColor)
42	Diffuse map blend factor (real, default = 1.0)
72	Diffuse map source (default = 1): 0 = Use current scene 1 = Use image file (specified by file name; null file name specifies no map)
3	Diffuse map file name (string, default = null string)
73	Projection method of diffuse map mapper (default = 1): 1 = Planar 2 = Box 3 = Cylinder 4 = Sphere
74	Tiling method of diffuse map mapper (default = 1): 1 = Tile 2 = Crop 3 = Clamp
75	Auto transform method of diffuse map mapper (bitset, default = 1): 1= No auto transform 2 = Scale mapper to current entity extents; translate mapper to entity origin 4 = Include current block transform in mapper transform
43	Transform matrix of diffuse map mapper (16 reals; row major format; default = identity matrix)
44	Specular gloss factor (real, default = 0.5)

MATERIAL group codes (*continued*)	
Group codes	**Description**
76	Specular color method (default = 0): 0 = Use current color 1 = Override current color
45	Specular color factor (real, default = 1.0; valid range is 0.0 to 1.0)
92	Specular color value (unsigned 32-bit integer representing an AcCmEntityColor)
46	Specular map blend factor (real; default = 1.0)
77	Specular map source (default = 1): 0 = Use current scene 1 = Use image file (specified by file name; null file name specifies no map)
4	Specular map file name (string; default = null string)
78	Projection method of specular map mapper (default = 1): 1 = Planar 2 = Box 3 = Cylinder 4 = Sphere
79	Tiling method of specular map mapper (default = 1): 1 = Tile 2 = Crop 3 = Clamp
170	Auto transform method of specular map mapper (bitset; default = 1): 1 = No auto transform 2 = Scale mapper to current entity extents; translate mapper to entity origin 4 = Include current block transform in mapper transform
47	Transform matrix of specular map mapper (16 reals; row major format; default = identity matrix)
48	Blend factor of reflection map (real, default = 1.0)
171	Reflection map source (default = 1): 0 = Use current scene 1 = Use image file (specified by file name; null file name specifies no map)
6	Reflection map file name (string; default = null string)

MATERIAL group codes (*continued*)

Group codes	Description
172	Projection method of reflection map mapper (default = 1): 1 = Planar 2 = Box 3 = Cylinder 4 = Sphere
173	Tiling method of reflection map mapper (default = 1): 1 = Tile 2 = Crop 3 = Clamp
174	Auto transform method of reflection map mapper (bitset; default = 1): 1 = No auto transform 2 = Scale mapper to current entity extents; translate mapper to entity origin 4 = Include current block transform in mapper transform
49	Transform matrix of reflection map mapper (16 reals; row major format; default = identity matrix)
140	Opacity percent (real; default = 1.0)
141	Blend factor of opacity map (real; default = 1.0)
175	Opacity map source (default = 1): 0 = Use current scene 1 = Use image file (specified by file name; null file name specifies no map)
7	Opacity map file name (string; default = null string)
176	Projection method of opacity map mapper (default = 1): 1 = Planar 2 = Box 3 = Cylinder 4 = Sphere
177	Tiling method of opacity map mapper (default = 1): 1 = Tile 2 = Crop 3 = Clamp
178	Auto transform method of opacity map mapper (bitset; default = 1): 1 = No auto transform 2 = Scale mapper to current entity extents; translate mapper to entity origin 4 = Include current block transform in mapper transform

MATERIAL group codes (*continued*)	
Group codes	**Description**
142	Transform matrix of opacity map mapper (16 reals; row major format; default = identity matrix)
143	Blend factor of bump map (real; default = 1.0)
179	Bump map source (default = 1): 0 = Use current scene 1 = Use image file (specified by file name; null file name specifies no map)
8	Bump map file name (string; default = null string)
270	Projection method of bump map mapper (default = 1): 1 = Planar 2 = Box 3 = Cylinder 4 = Sphere
271	Tiling method of bump map mapper (default = 1): 1 = Tile 2 = Crop 3 = Clamp
272	Auto transform method of bump map mapper (bitset; default = 1): 1 = No auto transform 2 = Scale mapper to current entity extents; translate mapper to entity origin 4 = Include current block transform in mapper transform
144	Transform matrix of bump map mapper (16 reals; row major format; default = identity matrix)
145	Refraction index (real; default = 1.0)
146	Blend factor of refraction map (real; default = 1.0)
273	Refraction map source (default = 1): 0 = Use current scene 1 = Use image file (specified by file name; null file name specifies no map)
9	Refraction map file name (string; default = null string)
274	Projection method of refraction map mapper (default = 1): 1 = Planar 2 = Box 3 = Cylinder 4 = Sphere

MATERIAL group codes (*continued*)	
Group codes	**Description**
275	Tiling method of refraction map mapper (default = 1): 1 = Tile 2 = Crop 3 = Clamp
276	Auto transform method of refraction map mapper (bitset; default = 1): 1 = No auto transform 2 = Scale mapper to current entity extents; translate mapper to entity origin 4 = Include current block transform in mapper transform
147	Transform matrix of refraction map mapper (16 reals; row major format; default = identity matrix)

MLINESTYLE

The following group codes are used by MLINESTYLE objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

MLINESTYLE group codes	
Group codes	**Description**
0	Object name (MLINESTYLE)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS" (persistent reactors group appears in all dictionaries except the main dictionary)
330	Soft-pointer ID/handle to owner dictionary. For MLINESTYLE objects this is always the ACAD_MLINESTYLE entry of the named object dictionary
102	End of persistent reactors group; always "}"
100	Subclass marker (AcDbMlineStyle)
2	Mline style name

MLINESTYLE group codes (*continued*)

Group codes	Description
70	Flags (bit-coded): 1 =Fill on 2 = Display miters 16 = Start square end (line) cap 32 = Start inner arcs cap 64 = Start round (outer arcs) cap 256 = End square (line) cap 512 = End inner arcs cap 1024 = End round (outer arcs) cap
3	Style description (string, 255 characters maximum)
62	Fill color (integer, default = 256)
51	Start angle (real, default is 90 degrees)
52	End angle (real, default is 90 degrees)
71	Number of elements
49	Element offset (real, no default). Multiple entries can exist; one entry for each element
62	Element color (integer, default = 0). Multiple entries can exist; one entry for each element
6	Element linetype (string, default = BYLAYER). Multiple entries can exist; one entry for each element

The 2 group codes in mline entities and MLINESTYLE objects are redundant fields. These groups should not be modified under any circumstances, although it is safe to read them and use their values. The correct fields to modify are

Mline — The 340 group in the same object, which indicates the proper MLINESTYLE object.

Mlinestyle — The 3 group value in the MLINESTYLE dictionary, which precedes the 350 group that has the handle or entity name of the current mlinestyle.

OBJECT_PTR

The following group codes are used by OBJECT_PTR objects. In addition to the group codes described here, see “Common Group Codes for Objects” on page 120. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

OBJECT_PTR group codes

Group codes	Description
0	Object name (OBJECT_PTR)
5	Handle
102	Start of persistent reactors group; always “{ACAD_REACTORS”
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always “}”
1001	Begin ASE xdata (DC015)

PLOTSETTINGS

The following group codes are used by PLOTSETTINGS objects. In addition to the group codes described here, see “Common Group Codes for Objects” on page 120. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

PLOTSETTINGS group codes

Group codes	Description
0	Object name (PLOTSETTINGS)
5	Handle
102	Start of persistent reactors group; always “{ACAD_REACTORS”
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always “}”

PLOTSETTINGS group codes (*continued*)

Group codes	Description
330	Soft-pointer ID/handle to owner object
100	Subclass marker (AcDbPlotSettings)
1	Page Setup name
2	Name of system printer or plot configuration file
4	Paper size
6	Plot view name
40	Size, in millimeters, of unprintable margin on left side of paper
41	Size, in millimeters, of unprintable margin on bottom of paper
42	Size, in millimeters, of unprintable margin on right side of paper
43	Size, in millimeters, of unprintable margin on top of paper
44	Plot paper size: physical paper width in millimeters
45	Plot paper size: physical paper height in millimeters
46	Plot origin: *X* value of origin offset in millimeters
47	Plot origin: *Y* value of origin offset in millimeters
48	Plot window area: *X* value of lower-left window corner
49	Plot window area: *Y* value of upper-right window corner
140	Plot window area: *X* value of lower-left window corner
141	Plot window area: *Y* value of upper-right window corner
142	Numerator of custom print scale: real world (paper) units
143	Denominator of custom print scale: drawing units

PLOTSETTINGS group codes (*continued*)

Group codes	Description
70	Plot layout flag: 1 = PlotViewportBorders 2 = ShowPlotStyles 4 = PlotCentered 8 = PlotHidden 16 = UseStandardScale 32 = PlotPlotStyles 64 = ScaleLineweights 128 = PrintLineweights 512 = DrawViewportsFirst 1024 = ModelType 2048 = UpdatePaper 4096 = ZoomToPaperOnUpdate 8192 = Initializing 16384 = PrevPlotInit
72	Plot paper units: 0 = Plot in inches 1 = Plot in millimeters 2 = Plot in pixels
73	Plot rotation: 0 = No rotation 1 = 90 degrees counterclockwise 2 = Upside-down 3 = 90 degrees clockwise
74	Plot type (portion of paper space to output to the media): 0 = Last screen display 1 = Drawing extents 2 = Drawing limits 3 = View specified by code 6 4 = Window specified by codes 48, 49, 140, and 141 5 = Layout information
7	Current style sheet
75	Standard scale type: 0 = Scaled to Fit 1 = 1/128"=1'; 2 = 1/64"=1'; 3 = 1/32"=1' 4 = 1/16"=1'; 5 = 3/32"=1'; 6 = 1/8"=1' 7 = 3/16"=1'; 8 = 1/4"=1'; 9 = 3/8"=1' 10 = 1/2"=1'; 11 = 3/4"=1'; 12 = 1"=1' 13 = 3"=1'; 14 = 6"=1'; 15 = 1'=1' 16= 1:1 ; 17= 1:2; 18 = 1:4; 19 = 1:8; 20 = 1:10; 21= 1:16 22 = 1:20; 23 = 1:30; 24 = 1:40; 25 = 1:50; 26 = 1:100 27 = 2:1; 28 = 4:1; 29 = 8:1; 30 = 10:1; 31 = 100:1; 32 = 1000:1

PLOTSETTINGS group codes (*continued*)	
Group codes	**Description**
76	ShadePlot mode: 0 = As Displayed 1 = Wireframe 2 = Hidden 3 = Rendered
77	ShadePlot resolution level: 0 = Draft 1 = Preview 2 = Normal 3 = Presentation 4 = Maximum 5 = Custom
78	ShadePlot custom DPI: Valid range: 100 to 32767 Only applied when the ShadePlot resolution level is set to 5 (Custom)
147	A floating point scale factor that represents the standard scale value specified in code 75
148	Paper image origin: *X* value
149	Paper image origin: *Y* value

RASTERVARIABLES

The following group codes are used by RASTERVARIABLES objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

RASTERVARIABLES group codes	
Group codes	**Description**
0	Object name (RASTERVARIABLES)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"

RASTERVARIABLES group codes (*continued*)

Group codes	Description
330	Soft-pointer ID/handle to owner dictionary. For a RASTERVARIABLES object, this is always the ACAD_IMAGE_VARS entry of the named object dictionary
102	End of persistent reactors group; always "}"
100	Subclass marker (AcDbRasterVariables)
90	Class version 0
70	Display-image-frame flag: 0 = No frame; 1 = Display frame
71	Image display quality (screen only): 0 = Draft; 1 = High
72	AutoCAD units for inserting images. This is what one AutoCAD unit is equal to for the purpose of inserting and scaling images with an associated resolution: 0 = None; 1 = Millimeter; 2 = Centimeter 3 = Meter; 4 = Kilometer; 5 = Inch 6 = Foot; 7 = Yard; 8 = Mile

SPATIAL_INDEX

The following group codes are used by SPATIAL_INDEX objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

SPATIAL_INDEX group codes

Group codes	Description
0	Object name (SPATIAL_INDEX)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always "}"
100	Subclass marker (AcDbIndex)

SPATIAL_INDEX group codes (*continued*)

Group codes	Description
40	Timestamp (Julian date)
100	Subclass marker (AcDbSpatialIndex)

The SPATIAL_INDEX is always written out empty to a DXF file. This object can be ignored.

SPATIAL_FILTER

The following group codes are used by SPATIAL_FILTER objects. In addition to the group codes described here, see "Common Group Codes for Objects" on page 120. For information about abbreviations and formatting used in this table, see "Formatting Conventions in This Reference" on page 2.

SPATIAL_FILTER group codes

Group codes	Description
0	Object name (SPATIAL_FILTER)
5	Handle
102	Start of persistent reactors group; always "{ACAD_REACTORS"
330	Soft-pointer ID/handle to owner dictionary (SPATIAL)
102	End of persistent reactors group, always "}"
100	Subclass marker (AcDbFilter)
100	Subclass marker (AcDbSpatialFilter)
70	Number of points on the clip boundary 2 = Rectangular clip boundary (lower-left and upper-right) greater than 2 = Polyline clip boundary
10	Clip boundary definition point (in OCS) (always 2 or more) based on an xref scale of 1 DXF: *X* value; APP: 2D point
20	DXF: *Y* value of boundary definition point (always 2 or more)

SPATIAL_FILTER group codes (*continued*)

Group codes	Description
210	Normal to the plane containing the clip boundary DXF: *X* value; APP: 3D vector
220, 230	DXF: *Y* and *Z* values of extrusion direction
11	Origin used to define the local coordinate system of the clip boundary DXF: *X* value; APP: 3D point
21, 31	Origin used to define the local coordinate system of the clip boundary DXF: *Y* and *Z* values
71	Clip boundary display enabled flag 0 = Disabled; 1 = Enabled
72	Front clipping plane flag; 0 = No; 1 = Yes
40	Front clipping plane distance (if code 72 = 1)
73	Back clipping plane flag; 0 = No; 1 = Yes
41	Back clipping plane distance (if code 73 = 1)
40	4x3 transformation matrix written out in column major order. This matrix is the inverse of the original block reference (insert entity) transformation. The original block reference transformation is the one that is applied to all entities in the block when the block reference is regenerated (always 12 entries)
40	4x3 transformation matrix written out in column major order. This matrix transforms points into the coordinate system of the clip boundary (12 entries)

SORTENTSTABLE

The following group codes are used by SORTENTSTABLE objects. In addition to the group codes described here, see “Common Group Codes for Objects” on page 120. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

SORTENTSTABLE group codes

Group codes	Description
0	Object name (SORTENTSTABLE)
5	Handle
102	Start of persistent reactors group; always “{ACAD_REACTORS”
330	Soft-pointer ID/handle to owner dictionary (ACAD_SORTENTS)
102	End of persistent reactors group; always “}”
100	Subclass marker (AcDbSortentsTable)
330	Soft-pointer ID/handle to owner (currently only the *MODEL_SPACE or *PAPER_SPACE blocks)
331	Soft-pointer ID/handle to an entity (zero or more entries may exist)
5	Sort handle (zero or more entries may exist)

If the SORTENTS Regen flag (bit-code value 16) is set, AutoCAD regenerates entities in ascending handle order. When the DRAWORDER command is used, a SORTENTSTABLE object is attached to the *Model_Space or *Paper_Space block’s extension dictionary under the name ACAD_SORTENTS. The SORTENTSTABLE object related to this dictionary associates a different handle with each entity, which redefines the order in which the entities are regenerated.

VBA_PROJECT

The following group codes are used by VBA_PROJECT objects. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

VBA_PROJECT group codes

Group codes	Description
0	Object name (VBA_PROJECT)
5	Handle
102	Start of persistent reactors group; always “{ACAD_REACTORS”
330	Soft-pointer ID/handle to owner dictionary
102	End of persistent reactors group, always “}”
330	Soft-owner ID/handle to owner object
100	Subclass marker (AcDbVbaProject)
90	Number of bytes of binary chunk data (contained in the group code 310 records that follow)
310	DXF™Binary object data (multiple entries containing VBA project data)

XRECORD

The following group codes are common to all xrecord objects. In addition to the group codes described here, see “Common Group Codes for Objects” on page 120. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

Xrecord group codes

Group codes	Description
100	Subclass marker (AcDbXrecord)
280	Duplicate record cloning flag (determines how to merge duplicate entries): 0 = Not applicable 1 = Keep existing 2 = Use clone 3 = <xref>0<name> 4 = 0<name> 5 = Unmangle name
1–369 (except 5 and 105)	These values can be used by an application in any way

Xrecord objects are used to store and manage arbitrary data. They are composed of DXF group codes with “normal object” groups (that is, non-xdata group codes), ranging from 1 through 369 for supported ranges. This object is similar in concept to xdata but is not limited by size or order.

Xrecord objects are designed to work in such a way as to not offend releases R13c0 through R13c3. However, if read into a pre-R13c4 version of AutoCAD®, xrecord objects disappear.

THUMBNAILIMAGE Section

8

This chapter presents the group codes that are found in the THUMBNAILIMAGE section of a DXF™ file. This section exists only if a preview image has been saved with the DXF file.

In this chapter

THUMBNAILIMAGE Section Group Codes

The following group codes are found in the THUMBNAILIMAGE section. For information about abbreviations and formatting used in this table, see “Formatting Conventions in This Reference” on page 2.

THUMBNAILIMAGE group codes

Group codes	Description
90	The number of bytes in the image (and subsequent binary chunk records)
310	Preview image data (multiple lines; 256 characters maximum per line)

Drawing Interchange File Formats

A

This appendix describes the various file formats AutoCAD® uses to interchange drawing data with other applications. The formats presented are Drawing Interchange File (DXF™), binary DXF, Slide (SLD), and the Slide Library (SLB) file formats.

DXF files can be either ASCII or binary format. Because ASCII DXF files are more common than the binary format, the term *DXF file* is used to refer to ASCII DXF files and the term *binary DXF file* is used for the binary format.

In this appendix

ASCII DXF Files

This section describes the format of ASCII DXF files. It contains information that is needed only if you write your own programs to process DXF files or work with entity information obtained by AutoLISP®and ObjectARX™ applications.

General DXF File Structure

Essentially, a DXF file is composed of pairs of codes and associated values. The codes, known as *group codes*, indicate the type of value that follows. Using these group code and value pairs, a DXF file is organized into sections composed of records, which are composed of a group code and a data item. Each group code and value are on their own line in the DXF file.

Each section starts with a group code 0 followed by the string SECTION. This is followed by a group code 2 and a string indicating the name of the section (for example, HEADER). Each section is composed of group codes and values that define its elements. A section ends with a 0 followed by the string ENDSEC.

It may be helpful to produce a DXF file from a small drawing, print it, and refer to it while reading the information presented in this section.

The overall organization of a DXF file is as follows:

- *HEADER section.* Contains general information about the drawing. It consists of an AutoCAD database version number and a number of system variables. Each parameter contains a variable name and its associated value.
- *CLASSES section.* Holds the information for application-defined classes, whose instances appear in the BLOCKS, ENTITIES, and OBJECTS sections of the database. A class definition is permanently fixed in class hierarchy.
- *TABLES section.* Contains definitions for the following symbol tables:

  APPID (application identification table)

  BLOCK_RECORD (block reference table)

  DIMSTYLE (dimension style table)

  LAYER (layer table)

  LTYPE (linetype table)

  STYLE (text style table)

  UCS (user coordinate system table)

  VIEW (view table)

VPORT (viewport configuration table)

- *BLOCKS section*. Contains block definition and drawing entities that make up each block reference in the drawing.
- *ENTITIES section*. Contains the graphical objects (entities) in the drawing, including block references (insert entities).
- *OBJECTS section*. Contains the nongraphical objects in the drawing. All objects that are not entities or symbol table records or symbol tables are stored in this section. Examples of entries in the OBJECTS section are dictionaries that contain mline styles and groups.
- *THUMBNAILIMAGE section*. Contains the preview image data for the drawing. This section is optional.

If you use the Select Objects option of the SAVE or SAVEAS command, the ENTITIES section of the resulting DXF file contains only the entities you select.

Group Codes in DXF Files

Group codes and the associated values define a specific aspect of an object or entity. The line immediately following the group code is the associated value. This value can be a string, an integer, or a floating-point value, such as the *X* coordinate of a point. The lines following the second line of the group, if any, are determined by the group definition and the data associated with the group.

Special group codes are used as file separators, such as markers for the beginning and end of sections, tables, and the end of the file itself.

Entities, objects, classes, tables and table entries, and file separators are introduced with a 0 group code that is followed by a name describing the group.

The maximum DXF file string length is 256 characters. If your AutoCAD drawing contains strings that exceed this number, those strings are truncated during SAVE, SAVEAS, and WBLOCK. OPEN and INSERT fail if your DXF file contains strings that exceed this number.

ASCII Control Characters in DXF Files

SAVEAS handles ASCII control characters in text strings by expanding the character into a caret (^) followed by the appropriate letter. For example, an ASCII Control-G (BEL, decimal code 7) is written as ^G. If the text itself contains a caret character, it is expanded to caret, space (^). OPEN and INSERT perform the complementary conversion.

Header Group Codes in DXF Files

Applications can retrieve the values of these variables with the AutoLISP **`getvar`** function.

The following is an example of the HEADER section of a DXF™ file:

```
  0                    Beginning of HEADER section
SECTION
  2
HEADER
  9                    Repeats for each header variable
$<variable>
<group code>
<value>
  0                    End of HEADER section
ENDSEC
```

Class Group Codes in DXF Files

The following is an example of the CLASSES section of a DXF file:

```
  0                    Beginning of CLASSES section
SECTION
  2
CLASSES
  0                    Repeats for each entry
CLASS
  1
<class dxf record>
  2
<class name>
  3
<app name>
 90
<flag>
280
<flag>
281
<flag>
  0                    End of CLASSES section
ENDSEC
```

Symbol Table Group Codes in DXF Files

The following is an example of the TABLES section of a DXF file.

```
  0                    Beginning of TABLES section
SECTION
  2
TABLES
  0                    Common table group codes; repeats for
TABLE                  each entry
  2
<table type>
  5
<handle>
100
AcDbSymbolTable
 70
<max. entries>
  0                    Table entry data; repeats for each table
<table type>           record
  5
<handle>
100
AcDbSymbolTableRecord
 .
 . <data>
 .
  0                    End of table
ENDTAB
  0                    End of TABLES section
ENDSEC
```

Symbol Table Example

This DXF sequence represents three full objects: the symbol table itself plus two entries.

```
  0
TABLE                  Indicates a symbol table entry
  2
STYLE                  Text style symbol table entry. Exception
                       to rule that code 0 fully defines type
```

```
  5
1C                      STYLE table handle; same as for entities and other objects
 70
3                       Maximum number of STYLE table records to follow (pre-Release 13 field)
1001
APP_X                   APP_X has put xdata on a symbol table
1040
42.0                    Just a single floating-point number
  0
STYLE                   Beginning of first element in the STYLE symbol table
  5
3A                      The first entry's handle (DIMSTYLE entries will have 105 here)
  2
ENTRY_1                 The first entry's text name
 70
64                      Standard flag values
 40
.4                      Text height
 41
1.0                     Width scale factor
 50
0.0                     Oblique angle
 71
0                       Text generation flags
 42
0.4                     Last height used
  3
BUFONTS.TXT             Primary font file name
```

```
  0
STYLE                    Second entry begins. No xdata or
                         persistent reactors on first entry
  5
C2                       Second entry handle
  2
ENTRY_2                  Second entry text name
...
...                      Other fields down to group code 3
  3
BUFONTS.TXT              Primary font file name and last object
                         type—specific group
102
{ACAD_REACTORS           This entry has two persistent reactors
330
3C2                      Soft ID to first reactor object
330
41B                      Soft ID to first reactor object
102
}                        Indicates the end of the reactor set
1001
APP_1                    Xdata attached to this entry
1070
45
1001
APP_2
1004
18A5B3EF2C199A
  0
UCS                      Start of UCS table (and end of previous
                         record and table)
```

Blocks Group Codes in DXF Files

The following is an example of the BLOCKS section of a DXF file:

```
  0                          Beginning of BLOCKS section
SECTION
  2
BLOCKS
  0                          Begins each block entry (a block entity
BLOCK                        definition)
  5
<handle>
100
AcDbEntity
  8
<layer>
100
AcDbBlockBegin
  2
<block name>
 70
<flag>
 10
<X value>
 20
<Y value>
 30
<Z value>
  3
<block name>
  1
<xref path>
  0                          One entry for each entity definition
<entity type>                within the block
 .
 . <data>
 .
  0                          End of each block entry (an endblk
ENDBLK                       entity definition)
  5
<handle>
100
AcDbBlockEnd
  0                          End of BLOCKS section
ENDSEC
```

Entity Group Codes in DXF Files

The following is an example of the ENTITIES section of a DXF file:

```
  0                    Beginning of ENTITIES section
SECTION
  2
ENTITIES
  0                    One entry for each entity definition
<entity type>
  5
<handle>
330
<pointer to owner>
100
AcDbEntity
  8
<layer>
100
AcDb<classname>
 .
 . <data>
 .
  0                    End of ENTITIES section
ENDSEC
```

Object Group Codes in DXF Files

The following is an example of the OBJECTS section of a DXF file:

```
  0                     Beginning of OBJECTS section
SECTION
  2
OBJECTS
  0                     Beginning of named object dictionary
DICTIONARY              (root dictionary object)
  5
<handle>
100
AcDbDictionary
  3                     Repeats for each entry
<dictionary name>
350
<handle of child>
  0                     Groups of object data
<object type>
 .
 . <data>
 .
  0                     End of OBJECTS section
ENDSEC
```

Writing a DXF Interface Program

Writing a program that communicates with AutoCAD by means of the DXF file appears more difficult than it actually is. The DXF format makes it easy to ignore information you don't need, while reading the information you do need.

Reading a DXF File

The following example is a simple Visual Basic program that reads a DXF file and extracts specific codes and values from a given object in a given section.

```
' ReadDXF extracts specified code/value pairs from a DXF file.
' This function requires four string parameters, a valid DXF
' file name, a DXF section name, the name of an object in that
' section, and a comma delimited list of codes.
'
Function ReadDXF( _
        ByVal dxfFile As String, ByVal strSection As String, _
        ByVal strObject As String, ByVal strCodeList As String)
    Dim tmpCode, lastObj As String
    Open dxfFile For Input As #1
    ' Get the first code/value pair
    codes = ReadCodes
    ' Loop through the whole file until the "EOF" line
    While codes(1) <> "EOF"
        ' If the group code is '0' and the value is 'SECTION' ..
        If codes(0) = "0" And codes(1) = "SECTION" Then
            ' This must be a new section, so get the next
            ' code/value pair.
            codes = ReadCodes()
            ' If this section is the right one ..
            If codes(1) = strSection Then
                ' Get the next code/value pair and ..
                codes = ReadCodes
                ' Loop through this section until the 'ENDSEC'
                While codes(1) <> "ENDSEC"
                    ' While in a section, all '0' codes indicate
                    ' an object. If you find a '0' store the
                    ' object name for future use.
                    If codes(0) = "0" Then lastObj = codes(1)
                    ' If this object is one you're interested in
                    If lastObj = strObject Then
                        ' Surround the code with commas
                        tmpCode = "," & codes(0) & ","
                        ' If this code is in the list of codes ..
                        If InStr(strCodeList, tmpCode) Then
                            ' Append the return value.
                            ReadDXF = ReadDXF & _
                                codes(0) & "=" & codes(1) & vbCrLf
                        End If
                    End If
                    ' Read another code/value pair
                    codes = ReadCodes
                Wend
            End If
        Else
            codes = ReadCodes
        End If
    Wend
    Close #1
End Function

' ReadCodes reads two lines from an open file and returns a two item
' array, a group code and its value. As long as a DXF file is read
' two lines at a time, all should be fine. However, to make your
' code more reliable, you should add some additional error and
' other checking.
```

```
'
Function ReadCodes() As Variant
    Dim codeStr, valStr As String
    Line Input #1, codeStr
    Line Input #1, valStr
    ' Trim the leading and trailing space from the code
    ReadCodes = Array(Trim(codeStr), valStr)
End Function
```

Writing a DXF File

Writing a program that creates a DXF file can be more difficult than one that reads a DXF file, because you must maintain consistency within the drawing in order for AutoCAD to find the file acceptable. AutoCAD lets you omit many items in a DXF file and still obtain a usable drawing.

- The entire HEADER section can be omitted if you don't set header variables.
- Any of the tables in the TABLES section can be omitted if you don't need to make entries, and the entire TABLES section can be dropped if nothing in it is required.
- If you define any linetypes in the LTYPE table, this table must appear before the LAYER table.
- If no block definitions are used in the drawing, the BLOCKS section can be omitted.
- If present, the BLOCKS section must appear before the ENTITIES section.
- Within the ENTITIES section, you can reference layer names even though you haven't defined them in the LAYER table. Such layers are automatically created with color 7 and the CONTINUOUS linetype.
- The EOF item must be present at the end of file.

The following Visual Basic subroutine constructs a DXF file representing a polygon.

```
' WriteDXFPolygon creates a minimal DXF file that only contains
' the ENTITIES section. This subroutine requires five parameters,
' the DXF file name, the number of sides for the polygon, the X
' and Y coordinates for the bottom end of the right-most side
' (it starts in a vertical direction), and the length for each
' side. Note that because this only requests 2D points, it does
' not include the Z coordinates (codes 30 and 31). The lines are
' placed on the layer "Polygon."
'
Sub WriteDXFPolygon( _
        dxfFile As String, iSides As Integer, _
        dblX As Double, dblY As Double, dblLen As Double)
    Dim i As Integer
    Dim dblA1, dblA, dblPI, dblNX, dblNY As Double
    Open dxfFile For Output As #1
    Print #1, 0
    Print #1, "SECTION"
    Print #1, 2
    Print #1, "ENTITIES"
    dblPI = Atn(1) * 4
    dblA1 = (2 * dblPI) / iSides
    dblA = dblPI / 2
    For i = 1 To iSides
        Print #1, 0
        Print #1, "LINE"
        Print #1, 8
        Print #1, "Polygon"
        Print #1, 10
        Print #1, dblX
        Print #1, 20
        Print #1, dblY
        dblNX = dblLen * Cos(dblA) + dblX
        dblNY = dblLen * Sin(dblA) + dblY
        Print #1, 11
        Print #1, dblNX
        Print #1, 21
        Print #1, dblNY
        dblX = dblNX
        dblY = dblNY
        dblA = dblA + dblA1
    Next i
    Print #1, 0
    Print #1, "ENDSEC"
    Print #1, 0
    Print #1, "EOF"
    Close #1
End Sub
```

As long as a properly formatted item appears on the line on which the data is expected, DXFIN accepts it. (Of course, string items should not have leading spaces unless these are intended to be part of the string.) This BASIC program takes advantage of this flexibility in input format and does not generate a file exactly like one generated by AutoCAD.

In the case of an error in using DXFIN to load, AutoCAD reports the error with a message indicating the nature of the error and the last line processed in the DXF file before the error was detected. This may not be the line on which the error occurred, especially in the case of errors such as the omission of required groups.

Binary DXF Files

The ASCII DXF file format is a complete representation of an AutoCAD drawing in an ASCII text form, and is easily processed by other programs. In addition, AutoCAD can produce or read a binary form of the full DXF file and accept limited input in another binary file format.

The SAVE and SAVEAS commands provide a Binary option that writes binary DXF files. Such a file contains all the information present in an ASCII DXF file but in a more compact form that takes about 25 percent less file space. It can be read and written more quickly (typically, five times faster) by AutoCAD. Unlike ASCII DXF files, which entail a trade-off between size and floating-point accuracy, binary DXF files preserve the accuracy in the drawing database. (AutoCAD Release 10 was the first version to support this form of DXF file; it cannot be read by older versions.)

A binary DXF file begins with a 22-byte sentinel consisting of the following:

```
AutoCAD Binary DXF<CR><LF><SUB><NULL>
```

Following the sentinel are pairs (group, value) as in an ASCII DXF file but represented in binary form. The group code is a 2-byte binary value (1 byte in DXF files prior to AutoCAD Release 14), and the value that follows is one of the following:

- A 2-byte integer with the least significant byte first and the most significant byte last
- An 8-byte IEEE double-precision floating-point number stored with the least significant byte first and the most significant byte last
- An ASCII string terminated by a 0 (NULL) byte

The type of data following a group is determined from the group code by the same rules used in decoding ASCII DXF files. Translation of angles to degrees and dates to fractional Julian date representation is performed for binary files as well as for ASCII DXF files. The comment group, 999, is not used in binary DXF files.

Extended data group codes are represented in binary DXF as a single byte with the value 255, followed by a 2-byte integer value containing the actual group code, followed by the actual value.

Extended data long values (group code 1071) occupy 4 bytes of data. Extended data binary chunks (group code 1004) are represented as a single-byte unsigned integer length, followed by the specified number of bytes of chunk data. For example, to transfer an extended data long group, the following values would appear, occupying 1, 2, and 4 bytes respectively.

```
255          Escape group code
1071         True group code
999999       Value for the 1071 group code
```

SAVEAS writes binary DXF files with the same file type *(.dxf)* as for ASCII DXF files. The OPEN and INSERT commands automatically recognize a binary file by means of its sentinel string. You need not identify it as a binary file.

If the OPEN and INSERT commands encounter an error in a binary DXF file, AutoCAD reports the byte address within the file where the error was detected.

Slide Files

Note This information is for experienced programmers, and is subject to change without notice.

AutoCAD slide files are screen images written by the MSLIDE command and read by the VSLIDE command. This section describes the format of slide files for the benefit of developers who wish to incorporate support for slides into their programs.

A slide file consists of a header portion (31 bytes) and one or more data records of variable length. All coordinates and sizes written to the slide file reflect the drawing area of the display device from which the slide was created, with point (0,0) located at the lower-left corner of the drawing area. For AutoCAD Release 9 and later, the slide file header consists of the following fields:

Slide file header

Field	Bytes	Description
ID string	17	"AutoCAD Slide" CR LF ^Z NUL
Type indicator	1	Currently set to 56 (decimal)
Level indicator	1	Currently set to 2
High X dot	2	Width of the graphics area: 1, in pixels
High Y dot	2	Height of the graphics area: 1, in pixels
Aspect ratio	4	Drawing area aspect ratio (horizontal size/vertical size in inches), scaled by 10,000,000. This value is always written with the least significant byte first
Hardware fill	2	Either 0 or 2 (value is unimportant)
Test number	2	A number (1234 hex) used to determine whether all 2-byte values in the slide were written with the high-order byte first (Intel 8086-family CPUs) or the low-order byte first (Motorola 68000-family CPUs)

Data records follow the header. Each data record begins with a 2-byte field whose high-order byte is the record type. The remainder of the record may be composed of 1-byte or 2-byte fields as described in the following table. To

determine whether the 2-byte fields are written with the high-order byte first or the low-order byte first, examine the Test number field of the header that is described in the previous table.

Slide file data records

Record type (hex)	Bytes	Meaning	Description
00-7F	8	Vector	The from-X coordinate for an ordinary vector. From-Y, to-X, and to-Y follow, in that order, as 2-byte values. The from- point is saved as the last point
80-FA	—	Undefined	Reserved for future use
FB	5	Offset vector	The low-order byte and the following three bytes specify the endpoints (from-X, from-Y, to-X, to-Y) of a vector, in terms of offsets (–128 to +127) from the saved last point. The adjusted from- point is saved as the last point for use by subsequent vectors
FC	2	End of file	The low-order byte is 00
FD	6	Solid fill	The low-order byte is always zero. The following two 2-byte values specify the *X* and *Y* coordinates of one vertex of a polygon to be solid-filled. Three to ten such records occur in sequence. A Solid fill record with a negative Y coordinate indicates the start or end of such a flood sequence. In the start record, the X coordinate indicates the number of vertex records to follow
FE	3	Common endpoint vector	This is a vector starting at the last point. The low-order byte and the following byte specify to-X and to-Y in terms of offsets (–128 to +127) from the saved last point. The adjusted to- point is saved as the last point for use by subsequent vectors
FF	2	New color	Subsequent vectors are to be drawn using the color number indicated by the low-order byte

If a slide contains any vectors at all, a New color record will be the first data record. The order of the vectors in a slide and the order of the endpoints of those vectors may vary.

For example, the following is an annotated hex dump of a simple slide file created on an IBM PC/AT with an IBM Enhanced Graphics Adapter. The slide consists of a white diagonal line from the lower-left corner to the upper-right corner of the drawing area, a green vertical line near the lower-left corner, and a small red rectangle at the lower-left corner.

```
41 75 74 6F 43 41          ID string ("AutoCAD Slide" CR LF ^Z NUL)
44 20 53 6C 69 64
65 0D 0A 1A 00
56                         Type indicator (56)
02                         Level indicator (2)
3C 02                      High X dot (572)
24 01                      High Y dot (292)
0B 80 DF 00                Aspect ratio (14,647,307 / 10,000,000 = 1.46)
02 00                      Hardware fill (2)
34 12                      Test number (1234 hex)
07 FF                      New color (7 = white)
3C 02 24 01 00 00 00 00    Vector from 572,292 to 0,0. 572,292 becomes
"last" point
3 FF                       New color (3 = green)
0F 00 32 00 0F 00 13 00    Vector from 15,50 to 15,19. \x1115,50 becomes
                           "last" point
01 FF                      New color (1 = red)
12 FB E7 12 CE             Offset vector from 15+18,50-25 (33,25) to 15+18,
                           50-50 (33,0). 33,25 becomes "last" point
DF FE 00                   Common-endpoint vector from 33,25 to 33-33,25+0
                           (0,25). 0,25 becomes "last" point
00 FE E7                   Common-endpoint vector from (0,25) to 0+0,25-25
                           (0,0). 0,0 becomes "last" point
21 FE 00                   Common-endpoint vector from (0,0) to 0+33,0+0
                           (33,0).33,0 becomes "last" point
00 FC                      End of file
```

Old Slide Header

The slide format described in the previous section is produced by AutoCAD Release 9 and later, and is portable among all computers running AutoCAD Release 9 or later. Previous versions of AutoCAD (as well as AutoShade® 1.0 and AutoSketch® 1.02) produce slides with a somewhat different header, as shown in the following table.

Old slide file header

Field	Bytes	Description
ID string	17	"AutoCAD Slide" CR LF ^Z NUL
Type indicator	1	56 (decimal)
Level indicator	1	1 (old format)
High X dot	2	Width of the drawing area: 1, in pixels
High Y dot	2	Height of the drawing area: 1, in pixels
Aspect ratio	8	Drawing area aspect ratio (horizontal size/vertical size in inches), written as a floating-point number
Hardware fill	2	Either 0 or 2 (value is unimportant)
Filler byte	1	Unused

Note that the old-format header does not contain a test number field. The floating-point aspect ratio value and all 2-byte integers are written in the native format of the CPU that was used to create the file (for 8086-family CPUs, IEEE double-precision, and low-order byte first). Old-format slide files are not portable across machine types, but they can be read by any version of AutoCAD running on the same CPU type as the CPU with which the slide was created.

Slide Library Files

This section describes the format of AutoCAD slide libraries (Release 9 and later) for the benefit of developers who wish to incorporate support for slide libraries into their programs.

The general format of a slide library is as follows:

`"AutoCAD Slide Library 1.0" CR LF ^Z NUL NUL NUL NUL` *Header (32 bytes)*
One or more slide directory entries (36 bytes each)
One or more slides (variable length)

Slide directory entries have the following format:

Slide name (NUL terminated) (32 bytes)
Address of slide within library file (4 bytes)

The slide address is always written with the low-order byte first. Each slide to which the directory points is a complete slide file as described in the previous section. The end of the slide directory is signified by an entry with a null slide name (first byte is NUL). A slide library can contain a mixture of old-format and new-format slides.

Advanced DXF Issues

B

This appendix discusses the advanced concepts related to DXF™ group codes.

In this chapter

Database Objects

AutoCAD® drawings consist largely of structured containers for database objects. Database objects each have the following features:

- A handle whose value is unique to the drawing/DXF file, and is constant for the lifetime of the drawing. This format has existed since AutoCAD Release 10, and as of AutoCAD Release 13, handles are always enabled.
- An optional xdata table, as entities have had since AutoCAD Release 11.
- An optional persistent reactor table.
- An optional ownership pointer to an extension dictionary which, in turn, owns subobjects placed in it by an application.

Symbol tables and symbol table records are database objects and, thus, have a handle. They can also have xdata and persistent reactors in their DXF records.

Persistent Inter-Object Reference Handles

A set of group code ranges permits objects to directly specify references to other objects within the same drawing/DXF file. Four ranges are provided for the four types of reference handles that you can specify:

- Soft-pointer handle
- Hard-pointer handle
- Soft-owner handle
- Hard-owner handle

These handle types are manifested as entity names in AutoLISP®, as ads_name values in ObjectARX™, and as like-named classes derived from ObjectARX. These values are always maintained in insert, xref, and wblock operations such that references between objects in a set being copied are updated to point to the copied objects, while references to other objects remain unchanged.

Also, a group code range for “arbitrary” handles is defined to allow convenient storage of handle values that are not converted to entity names and then translated in insert, xref, or wblock.

Note If you use 1005 xdata group codes to store handles, they are treated as soft-pointer handles, which means that when groups of objects are copied or inserted into another drawing, references between the involved objects are translated. Although 1005 xdata items are always returned as handles in AutoLISP and ObjectARX, all of the reference handle group code ranges are represented as "entity names" in AutoLISP and as ads_name structures in ObjectARX.

Pointer and Ownership References

A pointer is a reference that indicates usage, but not possession or responsibility, for another object. A pointer reference means that the object uses the other object in some way, and shares access to it.

An ownership reference means that an owner object is responsible for the objects for which it has an owner handle. Ownership references direct the writing of entire DWG and DXF files in a generic manner, such as beginning from a few key root objects.

An object can have any number of pointer references associated with it, but it can have only one owner.

Hard and Soft References

Hard references, whether they are pointer or owner, protect an object from being purged. Soft references do not.

In AutoCAD, block definitions and complex entities are hard owners of their elements. A symbol table and dictionaries are soft owners of their elements. Polyline entities are hard owners of their vertex and seqend entities. Insert entities are hard owners of their attrib and seqend entities.

When establishing a reference to another object, it is recommended that you think about whether the reference should protect an object from the PURGE command.

Arbitrary Handles

Arbitrary handles are distinct in that they are not translated to session-persistent identifiers internally, or to entity names in AutoLISP, and so on. They are stored as handles. When handle values are translated in drawing-merge operations, arbitrary handles are ignored.

In all environments, arbitrary handles can be exchanged for entity names of the current drawing by means of the **handent** functions. A common usage of arbitrary handles is to refer to objects in external DXF and DWG files.

1005 Group Codes

1005 xdata group codes have the same behavior and semantics as soft pointers, which means that they are translated whenever the host object is merged into a different drawing. However, 1005 items are not translated to session-persistent identifiers or internal entity names in AutoLISP and ObjectARX. They are stored as handles.

Subclass Markers

When filing a stream of group data, a single object may be composed of several filer members, one for each level of inheritance where filing is done. Since derived classes and levels of inheritance can evolve separately, the data of each class filer member must be segregated from other members. This is achieved using subclass markers.

All class filer members are expected to precede their class-specific portion of instance data with a "subclass" marker—a 100 group code followed by a string with the actual name of the class. This does not affect the state needed to define the object's state, but it provides a means for the DXF file parsers to direct the group codes to the corresponding application software.

For example, an object that has data from different derived classes would be represented as follows:

```
999
FOOGRANDCHILD, defined by class AcDbSonOfSonOfFoo, which
999
 is derived from AcDbSonOfFoo
  0
FOOGRANDCHILD
  5
C2
100
AcDbFoo
999
Uses 10/20/30 group codes
 10
1.1
 20
2.3
 30
7.3
100
AcDbSonOfFoo
999
Also uses 10/20/30 group codes, for a different purpose
 10
1.1
 20
2.3
 30
7.3
100
AcDbSonOfSonOfFoo
999
Also uses 10/20/30 group codes, for yet another purpose
 10
13.2
 20
23.1
 30
31.2
999
Now for the Xdata
1001
APP_1
1070
45
1001
APP_2
1004
18A5B3EF2C199A
```

Extension Dictionary and Persistent Reactors

The extension dictionary is an optional sequence that stores the handle of a dictionary object that belongs to the current object, which in turn may contain entries. This facility allows attachment of arbitrary database objects to any database object. Any object or entity may have this section.

Persistent reactors are an optional sequence that stores object handles of objects registering themselves as reactors on the current object. Any object or entity may have this section.

Extended Data

Extended data (xdata) is created by AutoLISP or ObjectARX applications. If an entity contains extended data, it follows the entity's normal definition data. The group codes 1000 through 1071 describe extended data. The following is an example of an entity containing extended data in DXF format.

Normal entity definition data:

```
  0
INSERT
  5
F11
100
AcDbEntity
  8
TOP
100
AcDbBlockReference
 2
BLOCK_A
 10
0.0
 20
0.0
 30
0.0
```

Extended entity definition data:

```
1001
AME_SOL
1002
{
1070
 0
1071
 1.95059E+06
1070
 519
1010
2.54717
1020
2.122642
1030
2.049201
1005
ECD
1005
EE9
1005
0
1040
0.0
1040
1.0
1000
MILD_STEEL
```

The group code 1001 indicates the beginning of extended data. In contrast to normal entity data, with extended data the same group code can appear *multiple times*, and *order* is important.

Extended data is grouped by registered application name. Each registered application group begins with a 1001 group code, with the application name as the string value. Registered application names correspond to APPID symbol table entries.

An application can use as many APPID names as needed. APPID names are permanent, although they can be purged if they aren't currently used in the drawing. Each APPID name can have no more than one data group attached to each entity. Within an application group, the sequence of extended data groups and their meaning is defined by the application.

The extended data group codes are listed in the following table.

Extended data group codes and descriptions

Entity name	Group code	Description
String	1000	Strings in extended data can be up to 255 bytes long (with the 256th byte reserved for the null character)
Application name	1001 also a string value	Application names can be up to 31 bytes long (the 32nd byte is reserved for the null character) **NOTE** Do not add a 1001 group into your extended data because AutoCAD assumes it is the beginning of a new application extended data group
Control string	1002	An extended data control string can be either "{"or "}". These braces enable applications to organize their data by subdividing the data into lists. The left brace begins a list, and the right brace terminates the most recent list. Lists can be nested When AutoCAD reads the extended data for a particular application, it checks to ensure that braces are balanced
Layer name	1003	Name of the layer associated with the extended data
Binary data	1004	Binary data is organized into variable-length *chunks*. The maximum length of each chunk is 127 bytes. In ASCII DXF files, binary data is represented as a string of hexadecimal digits, two per binary byte
Database handle	1005	Handles of entities in the drawing database **NOTE** When a drawing with handles and extended data handles is imported into another drawing using INSERT, INSERT *, XREF Bind, XBIND, or partial OPEN, the extended data handles are translated in the same manner as their corresponding entity handles, thus maintaining their binding. This is also done in the EXPLODE block operation or for any other AutoCAD operation. When AUDIT detects an extended data handle that doesn't match the handle of an entity in the drawing file, it is considered an error. If AUDIT is fixing entities, it sets the handle to 0
3 reals	1010, 1020, 1030	Three real values, in the order *X*, *Y*, *Z*. They can be used as a point or vector record. AutoCAD never alters their value
World space position	1011, 1021, 1031	Unlike a simple 3D point, the world space coordinates are moved, scaled, rotated, and mirrored along with the parent entity to which the extended data belongs. The world space position is also stretched when the STRETCH command is applied to the parent entity and this point lies within the select window
World space displacement	1012, 1022, 1032	Also a 3D point that is scaled, rotated, and mirrored along with the parent (but is not moved or stretched)

Extended data group codes and descriptions (*continued*)

Entity name	Group code	Description
World direction	1013, 1023, 1033	Also a 3D point that is rotated and mirrored along with the parent (but is not moved, scaled, or stretched)
Real	1040	A real value
Distance	1041	A real value that is scaled along with the parent entity
Scale factor	1042	Also a real value that is scaled along with the parent. The difference between a distance and a scale factor is application-defined
Integer	1070	A 16-bit integer (signed or unsigned)
Long	1071	A 32-bit signed (long) integer

Object Coordinate Systems (OCS)

To save space in the drawing database (and in the DXF file), the points associated with each entity are expressed in terms of the entity's own object coordinate system (OCS). With OCS, the only additional information needed to describe the entity's position in 3D space are the 3D vector describing the *Z* axis of the OCS and the elevation value.

For a given *Z* axis (or extrusion) direction, there are an infinite number of coordinate systems, defined by translating the origin in 3D space and by rotating the *X* and *Y* axes around the *Z* axis. However, for the same *Z* axis direction, there is only one OCS. It has the following properties:

- Its origin coincides with the WCS origin.
- The orientation of the *X* and *Y* axes within the *XY* plane is calculated in an arbitrary but consistent manner. AutoCAD performs this calculation using the arbitrary axis algorithm (see "Arbitrary Axis Algorithm" on page 183).

For some entities, the OCS is equivalent to the WCS, and all points (DXF groups 10–37) are expressed in world coordinates. See the following table.

Coordinate systems associated with an entity type

Entities	Notes
3D entities such as line, point, 3dface, 3D polyline, 3D vertex, 3D mesh, 3D mesh vertex	These entities do not lie in a particular plane. All points are expressed in world coordinates. Of these entities, only lines and points can be extruded. Their extrusion direction can differ from the world *Z* axis
2D entities such as circle, arc, solid, trace, text, attrib, attdef, shape, insert, 2D polyline, 2D vertex, lwpolyline, hatch, image	These entities are planar in nature. All points are expressed in object coordinates. These entities can be extruded. Their extrusion direction can differ from the world *Z* axis
Dimension	Some of a dimension's points are expressed in WCS and some in OCS
Viewport	Expressed in world coordinates

Once AutoCAD has established the OCS for a given entity, the OCS works as follows: The elevation value stored with an entity indicates how far to shift the *XY* plane along the *Z* axis (from the WCS origin) to make it coincide with the plane that contains the entity. How much of this is the user-defined elevation is unimportant.

Any 2D points entered through the UCS are transformed into the corresponding 2D points in the OCS, which is shifted and rotated with respect to the UCS.

These are a few ramifications of this process:

- You cannot reliably find out what UCS was in effect when an entity was acquired.
- When you enter the *XY* coordinates of an entity in a given UCS and then do a SAVEAS, you probably won't recognize those *XY* coordinates in the DXF file. You must know the method by which AutoCAD calculates the *X* and *Y* axes in order to work with these values.
- The elevation value stored with an entity and output in DXF files is a sum of the *Z*-coordinate difference between the UCS *XY* plane and the OCS *XY* plane, and the elevation value that the user specified at the time the entity was drawn.

Arbitrary Axis Algorithm

The arbitrary axis algorithm is used by AutoCAD internally to implement the arbitrary but consistent generation of object coordinate systems for all entities that use object coordinates.

Given a unit-length vector to be used as the *Z* axis of a coordinate system, the arbitrary axis algorithm generates a corresponding *X* axis for the coordinate system. The *Y* axis follows by application of the right-hand rule.

The method is to examine the given *Z* axis (also called the *normal vector*). If it is close to the positive or negative world *Z* axis, cross the world *Y* axis with the given *Z* axis to arrive at the arbitrary *X* axis. If it is not close, cross the world *Z* axis with the given *Z* axis to arrive at the arbitrary *X* axis. The boundary at which the decision is made was chosen to be both inexpensive to calculate and completely portable across machines. This is achieved by having a sort of "square" polar cap, the bounds of which are 1/64, which is precisely specifiable in six decimal-fraction digits and in six binary-fraction bits.

The algorithm does the following (all vectors are assumed to be in 3D space and specified in the world coordinate system):

```
Let the given normal vector be called N.
Let the world Y axis be called Wy, which is always (0,1,0).
Let the world Z axis be called Wz, which is always (0,0,1).
```

Here we are looking for the arbitrary *X* and *Y* axes to go with the normal *N*. They will be called *Ax* and *Ay*. *N* could also be called *Az* (the arbitrary *Z* axis) as follows:

```
If (abs (Nx) < 1/64) and (abs (Ny) < 1/64)  then
     Ax = Wy X N  (where "X" is the cross-product operator).
Otherwise,
     Ax = Wz X N.
Scale Ax to unit length.
```

The method of getting the *Ay* vector is as follows:

```
Ay = N X Ax.  Scale Ay to unit length.
```

Index

A

B

C

D

E

F

G

H

I

J

L

M

T

U

V

W

X

Y

Z