AutoCAD®

Autodesk®

Customization Guide

2006

February 2005

1 2 3 4 5 6 7 8 9 10

Contents

Basic Customization

Your dealer can offer you independently developed applications that can further tailor AutoCAD to your needs.

In this chapter

Overview of Customization

AutoCAD can be customized in simple ways. For example, you can change the directory structure or move a button from one toolbar to another. If you want to change the interface further, you can edit the CUI file and use DIESEL code to create customizations with your own commands.

You can also use a number of powerful application programming interfaces (APIs) to add to and modify AutoCAD to suit your needs.

The list that follows is arranged from least to most complex:

- **Organize files.** You can organize program, support, and drawing files. For example, you can make a separate folder for each project that includes only the support files that project needs.
- **Customize Tool Palettes.** You can create a tool by dragging objects from your drawing onto a tool palette. You can create a tool palette by right-clicking on the Tool Palettes title bar and selecting New Palette. For information about customizing tool palettes, see "Customize Tool Palettes" in the *User's Guid*e.
- **Create custom templates.** Use templates to define common parameters when you publish a drawing using the Publish to Web wizard.
- **Run external programs and utilities from within AutoCAD.** You can, for example, copy a disk or delete a file from within AutoCAD by adding the appropriate external command to the program parameters (PGP) file, acad.pgp.
- **Define command aliases.** You can define simple abbreviations, or aliases, for frequently used commands from within AutoCAD by adding the command to the PGP file acad.pgp. For example, you might want to start the **BLOCK** command by entering **b**.
- **Create custom linetypes, hatch patterns, shapes, and text fonts.** You can create linetypes, hatch patterns, shapes, and text fonts that conform to your company standards and working methods.
- **Customize the user interface.** The CUI file controls many aspects of the user interface, including the behavior of your pointing device buttons and the functionality and appearance of pull-down, tablet, and image tile menus, toolbars, and accelerator keys. You can edit or create a CUI file to add commands or combine commands and assign them to a menu, toolbar, or other location.
- **Customize the status line.** You can use the DIESEL string expression language and the MODEMACRO system variable to provide additional

information at the status line, such as the date and time, system variable settings, or retrievable information using AutoLISP® .

- **Automate repetitive tasks by writing scripts.** A script is an ASCII text file containing commands that are processed like a batch file when you run the script. For example, if a set of drawings needs to be plotted a certain way, you can write a script that opens each drawing, hides and displays various layers, and issues PLOT commands. You can use scripts with slides to create automated presentations like those used at trade shows. A slide is a "snapshot" of the drawing area that cannot be edited. Slides can also be used in image tile menus and dialog boxes.

In addition to the methods described in the *Customization Guide*, there are application programming interfaces (APIs) available for customizing AutoCAD. "Introduction to Programming Interfaces" on page 135 briefly describes these APIs and provides cross-references to more information.

See also:

"**Organize Program and Support Files**"
"**Customize Toolbars**"
"**Customize a Publish to Web Template**" **on page 8**
"**Create Command Aliases**"
"**Custom Linetypes**"
"**Custom Hatch Patterns**"
"**Customize the User Interface**"
"**DIESEL**"
"**Customize the Status Line**"
"**Introduction to Programming Interfaces**"
"**Slides and Command Scripts**"

Organize Program and Support Files

You can change the default directory structure for the program and support files to suit your needs.

Overview of File Organization

AutoCAD uses support files for purposes such as storing customization definitions, loading AutoLISP and ObjectARX applications, and describing text fonts.

The default directory structure for the AutoCAD program and support files is designed to efficiently organize those files into logical groups. If this organization does not suit your needs, you can change it. However, some applications look for certain files in specific locations, and you should verify

that your modifications do not conflict with the requirements of those applications. Without the full path, including drive and directory, AutoCAD can locate only those files that are found in the library search path.

The location of the *support* folder changed in AutoCAD 2004. The location of local customizable files is stored in the LOCALROOTPREFIX system variable. The location of roamable customizable files is stored in the ROAMABLEROOTPREFIX system variable. If a network supports roaming, customizable files in the user's roaming profile are available on the machine the user is logged onto.

The following LISP script creates the CUSTFILES command, which launches Windows® Explorer in the correct folder.

```
(defun c:custfiles ()
(command "shell"
(strcat "explorer \"" (getvar "roamablerootprefix") "\"")
)
(princ)
)
```

Library Search Path

The library search path specifies where the program searches for files when you do not specify a full path name, as follows:

- Current directory. (This is typically determined by the "Start In" setting in your shortcut icon.)
- Directory that contains the current drawing file.
- Directories listed in the search path specified on the Files tab in . (See "Specify Search Paths, File Names, and File Locations" in the *User's Guide*.)
- Directory that contains the AutoCAD program files.

Depending on the current environment, two or more directories may be the same.

If a file is not in this search path, you must specify both its path name and file name before AutoCAD can find it. For example, if you want to insert the *part5.dwg* drawing into your current drawing and it is not in the library search path, you must specify its full path name, as shown here:

Command: **insert**
Enter block name or [?]: **/files2/olddwgs/part5**

If the drawing exists in that location, AutoCAD prompts you to finish the INSERT command in the usual manner.

Directory Structure

AutoCAD uses tree-structured directories and subdirectories. It is recommended that you keep supplemental files (such as AutoLISP applications and customization files) separate from the AutoCAD program and support files. This makes it easier to track possible conflicts and to upgrade each application without affecting the others.

The default location for AutoCAD is in the *Program Files* folder. You can create a new directory on the same level (for example, */AcadApps)* and store your custom AutoLISP and VBA macros, customization files, and other third-party applications in subdirectories on the next level. If you want to maintain multiple drawing directories (for separate job files), you can create a directory such as */AcadJobs* with subdirectories for each job.

Command Search Procedure

When you enter a command, AutoCAD goes through a series of steps to evaluate the validity of the command name. A command can be a built-in command or system variable, an external command or alias defined in the acad.pgp file, or a user-defined AutoLISP command. Commands can also be defined by ObjectARX applications or a device driver command. You can enter a command on the command line or choose a command from the appropriate menu. Commands can also be entered from a script file or by an AutoLISP or ObjectARX application.

The following list describes the search order AutoCAD uses to validate a command name.

1. If the input is a null response (SPACEBAR or ENTER), AutoCAD uses the name of the last command issued. HELP is the default.
2. AutoCAD checks the command name against the list of built-in commands. If the command is in the list and is not preceded by a period (.), AutoCAD then checks the command against a list of undefined commands. If the command is undefined, the search continues. Otherwise, the command is run, unless another reason prevents it from doing so. Running it transparently or in Perspective mode might be impossible.
3. AutoCAD checks the command name against the names of commands defined by a device driver, and then by those defined by the display driver.
4. AutoCAD checks the command name against the external commands defined in the program parameters file *(acad.pgp)*. If the command name corresponds to a defined external command, that command runs, and the search is complete.

5 AutoCAD checks the command name against the list of commands defined by AutoLISP or ObjectARX applications. At this point, an autoloaded command is loaded.

6 AutoCAD checks the command name against the list of system variables. If the command name is in the list, AutoCAD executes the SETVAR command, using the input as the variable name.

7 If the command name corresponds to a command alias defined in the program parameters file, AutoCAD uses the expanded command name and continues the search, starting a new search against the list of built-in commands.

8 If all the preceding steps fail, the search terminates with a warning message about illegal command names.

See also:

"Overview of AutoLISP Automatic Loading" on page 144
"Specify Search Paths, File Names, and File Locations" in the User's Guide

Multiple Configurations

If you use more than one pointing device or use different plotters, you can set up more than one configuration file to make it easy to switch between devices.

When you configure AutoCAD for a pointing device and plotter drivers, the information you supply is recorded in a configuration file. The default location of the *acad.cfg* configuration file is listed in the Options dialog box, Files tab, under Help and Miscellaneous File Names, but you can specify an alternative path or file name.

Typically, only a single configuration is necessary, but you may need multiple configurations. For example, if you use a mouse for most of your work but occasionally require a large digitizing tablet, you can set up your system to handle multiple configurations rather than reconfiguring each time you change a device.

The configuration file stores the values of many AutoCAD system variables and the configuration options defined in the Options dialog box. If you want different settings for these system variables and operating parameters, you can save those values to different configuration files. For a list of the system variables and where they are stored, see System Variables in the *Command Reference*.

To take advantage of multiple configurations, you must set up AutoCAD to use different configuration files. Use the **/c** switch to specify alternative configuration files at startup.

See also:

"Customize Startup" in the User's Guide

Multiple Drawing Folders

Keeping your drawing files and other associated files in separate directories makes it easier to perform basic file maintenance. The scenario described in this topic is based on the sample directory structure described in "Overview of File Organization" on page 3, but you can expand or alter it to meet your needs.

You can set up the */AcadJobs* directory to contain your drawing subdirectories. The drawing subdirectories can contain other subdirectories that hold related support files for a particular drawing type or job. The */AcadJobs/Job1/Support* directory can contain blocks and AutoLISP files specific to the drawing files in */AcadJobs/Job1*. Specifying **support** (with no path prefix) in the Support path adds the *support* directory within the current directory to the Support path. Notice that if you use the Options dialog box to specify a directory, AutoCAD creates a *hard-coded* path to that directory. To use the *relative* naming convention previously described, you must specify the Support path with the **/s** switch on the command line. See "Customize Startup" in the *User's Guide*.

To make sure that the required drawing directory is the current directory when you start AutoCAD, and that all files and subdirectories in that directory are easily accessible, you can create a program icon or a Start menu item that specifies the correct working directory for each job. This functionality works only if you set the AutoCAD system variable REMEMBERFOLDERS to 0.

You can use a batch program as an alternative to using icons or menus. With batch programs you can create new job directories automatically. The following batch program verifies that a specified directory exists, sets that directory to be current, and then runs AutoCAD.

```
@echo off
C:
if exist \AcadJobs\Jobs\%1 goto RUNACAD
echo.
echo *** Creating \AcadJobs\Jobs\%1
echo *** Press Ctrl+C to cancel.
echo.
pause
mkdir \AcadJobs\Jobs\%1
:RUNACAD
cd \AcadJobs\Jobs\%1
start C:\ AutoCAD\acad.exe
```

Using an ASCII text editor (such as Notepad), save the batch program to a file named *acad.bat*. Be sure to change the drive and directory names to match those on your system. Place this file in a directory that is on your system search path (for example, *C:\winnt*). You can run this batch program using the Run

command on the Start menu or by double-clicking the file in Explorer. If you saved the file as *acad.bat*, use the following syntax:

acad jobname

where *jobname* is the name of the job directory to make current.

Customize a Publish to Web Template

You can create customized templates to use in the Publish to Web wizard by modifying one of the Publish to Web template (PWT) files provided. Use any HTML editor or text editor.

To create a custom template, add or modify any of the following elements:

- Images
- Text
- Hyperlinks
- Color
- Title
- Video, animation, and so on

There are four default Publish to Web templates that you can customize:

- **Array of Thumbnails.** Creates a web page containing an array of thumbnail images.
- **Array Plus Summary.** Creates a web page containing an array of thumbnail images and summary information about each image.
- **List of Drawings.** Creates a web page containing a list of drawings and an image frame.
- **List Plus Summary.** Creates a web page containing a list of drawings, an image frame, and summary information about a selected image.

NOTE You must be familiar with HTML syntax to customize the Publish to Web templates.

You can make changes or additions to the look and feel of a template, but you cannot change the arrangement of images within it. For example, in the *Array of Thumbnails* template, the images are presented across the page in rows. You cannot alter the presentation of the images, but you can wrap text and graphics around the table of images.

WARNING To ensure that you do not overwrite the default Publish to Web template files, back up those files before you make any changes to them.

To create quick access to the Publish to Web templates

1. On the Tools menu, click Options.
2. In the Options dialog box, Files tab, click the plus sign (+) next to Template Settings. Then click the plus sign next to Drawing Template File Location.
3. Move the cursor to the path name that is displayed and click inside it, and press F2 , and press CTRL + C to copy it.
4. Click OK or Cancel to close the Options dialog box.
5. On the File menu, click Open.
6. In the Select File dialog box, right-click an empty area in the vertical panel on the left side, and click Add on the shortcut menu.
7. Enter a name in the Item name box (for example, **Templates**).
8. Press CTRL + V to paste the path into the Item Path box, and click OK.

   You can now access the Template folders by clicking the button in the left panel of the Select File dialog box.

To customize a Publish to Web template

1. On the File menu, click Open, and access the Publish to Web template folder.

   See "To create quick access to the Publish to Web templates" on page 9.
2. Double-click the *PTWTemplates* folder to open it. The following folders are displayed. Each contains a Publish to Web template and preview images (BMP) that you see when you run the Publish to Web wizard.

   - *Template1* . Contains the *Array of Thumbnails* template and a preview image
   - *Template2* . Contains the *Array Plus Summary* template, a preview image, and HTML frames
   - *Template3* . Contains the *List of Drawings* template, a preview image, and HTML frames
   - *Template4* . Contains the *List Plus Summary* template, a preview image, and HTML frames

3. Right-click the folder you want to use, and click Copy.
4. Press ALT + 2 , right-click the *PTWTemplates* folder, and click Paste.
5. Reopen the *PTWTemplates* folder, and right-click the new folder and rename it.
6. Right-click the new folder and click Open to display its contents.
7. Rename the Publish to Web template (PWT) file with an *.htm* or *.html* file extension.
8. Open the template file in an HTML editor or a text editor.

   The template file contains comments that help you determine which areas of the code you can modify to create your new web page.
9. Review the comments and make changes to the parts of the template you want to customize.
10. Save the template with a *.pwt* file extension. Make sure you save the file to the template folder you created in step 3.

    NOTE Each template folder can contain only one PWT file. If you create a new PWT file, make sure you delete any other PWT files that exist in the same folder.

    When you run the Publish to Web wizard, the new template is displayed in the list of templates.

Define Custom Commands

You can define external commands that run from within AutoCAD. You can also create command aliases for AutoCAD commands in the acad.pgp file, an ASCII text file that stores command definitions.

Define External Commands

External commands start other programs or utilities while AutoCAD is running.

While AutoCAD is running, you can invoke other programs or utilities, such as the following:

- Windows system commands and utilities, such as **start**, **type**, **dir**, or **copy**
- Applications such as text editors or word processors
- Database managers, spreadsheets, and communications programs
- User-supplied programs, such as batch files or VBA macros

When you enter an external command, AutoCAD looks for the command in acad.pgp. The first section of acad.pgp defines external commands. You can add command definitions by editing acad.pgp in an ASCII text editor (such as Notepad). To open the PGP file, on the Tools menu, click Customize ➤ Edit Program Parameters (acad.pgp).

NOTE Before you edit acad.pgp, create a backup file so that you can restore it later, if necessary.

When you define an external command, you specify a command name to be used at the Command prompt and an executable command string that is passed to the operating system. Each line in the external commands section has five comma-delimited fields, as follows:

command,[*executable*],*flags*[,[*]*prompt*[,*return_code*]]

command

The command that is entered at the Command prompt. If the name is an internal AutoCAD command name, it is ignored. The name is not case-sensitive.

executable

The constant string sent to the operating system when you enter a command name. It can be any command that you can execute at the operating-system prompt. The string can include switches or parameters. The case-sensitivity of this string depends on the application you are running.

flags

A required bitcoded parameter. Add these integer values in any combination to achieve the result you want.

0 Start the application and wait for it to finish.

1 Don't wait for the application to finish.

2 Run the application in Minimized mode.

4 Run the application "hidden."

8 Put the argument string in quotes.

Bit values 2 and 4 are mutually exclusive; if both are specified only the 2 bit is used. Using value 2 or 4 without value 1 should be avoided, because AutoCAD becomes unavailable until the application has completed.

Bit value 8 allows commands like **del** to work properly with file names that have embedded spaces. This eliminates the possibility of passing a

space-delimited list of file names to these commands. If you prefer multiple file support, do not use the bit value 8.

prompt

An optional field. It specifies the prompt to display on the AutoCAD command line. The response to this prompt is appended to the string supplied in the executable field. If the first character of the prompt field is an asterisk (*), the response can contain spaces and the user must press ENTER to terminate it. Otherwise, the response is terminated by either SPACEBAR or ENTER. If no prompt is specified, no input is requested; however, you must add a comma if a return code is to be supplied or if you want the prompt to have a trailing space.

return_code

An optional bitcoded parameter. You can add these integer values together in any combination to achieve the result you want. For example, if values `1` and `2` are required, you use `3` as the return code. The values are defined as follows (codes `0` and `4` are meaningless in a windowed environment and are therefore not included):

`1` Loads a DXB file. AutoCAD loads the DXB file named *$cmd.dxb* into the drawing after the command is terminated. After the DXB file is loaded, the *$cmd.dxb* file is deleted. This action produces the same result as the DXBIN command.

`2` Constructs a block definition from a DXB file. AutoCAD creates a block definition from the DXB file named *$cmd.dxb*. The response to the prompt field is used as the block name. This name must be a valid block name that does not currently exist in the drawing; therefore, this mode cannot redefine a previously defined block. After AutoCAD loads the DXB file, the *$cmd.dxb* file is deleted. The default name for the INSERT command is set to the newly defined block.

The file can also contain comment lines preceded by a semicolon (;).

Windows System Commands

The **start** and **cmd** Windows system commands are very useful when defining external commands. If you specify an executable string that does not use the **start** or **cmd** command, AutoCAD is unavailable until that window is closed.

The **start** command starts a separate window and runs a specified program or command. If **start** is used without any parameters, it opens a new command prompt window. The **start** command has many command line switches that affect the display of the new window. To launch a Windows application, use **start** without any switches. The **start** command is also very useful for starting a document that is associated with an application. For example, you can use

start to directly open a document created with a word processor or an HTML file.

The **cmd** command opens a Command prompt window that acts as a shell of AutoCAD. This window must be closed before control returns to the AutoCAD Command prompt. Two command line switches, **/c** and **/k**, are useful for external commands. The **/c** switch carries out the specified command and then stops (the window closes). The **/k** switch carries out the specified command and then continues (the window remains open). When using the **/k** switch, you must close the command window (with the **exit** command).

In general, use **start** to start a new window or application that is to be a separate process from AutoCAD. Use **cmd** to run a batch file or command script that does not create a separate window, or to create a window that must be closed before control is passed back to AutoCAD. For more information about these commands and switches, see your Windows system command documentation.

Custom-Defined Commands

The following example defines three new commands: RUN, LISTSET, and DXB2BLK.

```
RUN, cmd /c,0,*Batch file to run: ,
LISTSET,cmd /k SET,0
DXB2BLK,cmd /c DXBCOPY,0,DXB file: ,2
```

The RUN command runs a batch file or command script. The **cmd** command followed by the **/c** switch opens a command window, runs the batch file, and then closes.

The LISTSET command displays the current DOS environment variable settings. Because this example uses **cmd /k** rather than **start**, the command window must be closed before returning to AutoCAD. If you want this window to remain active, use **start /realtime**. For more information about these commands and switches, see your Windows system command documentation.

The DXB2BLK command creates a block definition from the specified DXB file. The DXB file converts all objects into lines. One beneficial by-product of this procedure is that it provides a simple method for exploding text objects into lines.

DXB2BLK passes the specified DXB file name to the *dxbcopy* batch file, which copies that file name to the file name *$cmd.dxb*. AutoCAD then creates a block from the specified DXB file. The name provided to the DXB file prompt is used as the new block name. To create the *dxbcopy.cmd* file, enter the following at the Windows Command Prompt:

```
echo copy %1.dxb $cmd.dxb > dxbcopy.cmd
```

This creates the *dxbcopy.cmd* file in the current directory. Move this file to a directory that is in your DOS path, or explicitly specify the file's location in

the acad.pgp file. For example, if the *dxbcopy.cmd* file is in *D:\cad*, enter the following in the external commands section of your *acad.pgp* file.

```
DXB2BLK, cmd /c D:\CAD\DXBCOPY,0,DXB file: ,2
```

To create a DXB file, choose AutoCAD DXB File Format as the current printer, and then plot to a file. For more information about configuring printers, see "Set Up Plotters and Printers" in the *Driver & Peripheral Guide* .

To open the program parameters file (acad.pgp)

■ On the Tools menu, click Customize ➤ Edit Program Parameters (acad.pgp).

Create Command Aliases

A command alias is an abbreviation that you enter on the command line instead of entering the entire command name.

For example, you can enter **c** instead of **circle** to start the CIRCLE command. An alias is not the same as a keyboard shortcut, which is a combination of keystrokes, such as CTRL+S for .

An alias can be defined for any AutoCAD command, device driver command, or external command. The second section of the acad.pgp file defines command aliases. You can change existing aliases or add new ones by editing acad.pgp in an ASCII text editor (such as Notepad). To open the PGP file, on the Tools menu, click Customize ➤ Edit Program Parameters (acad.pgp). The file can also contain comment lines preceded by a semicolon (;).

NOTE Before you edit *acad.pgp*, create a backup so that you can restore it later, if necessary.

To define a command alias, add a line to the command alias section of the *acad.pgp* file using the following syntax:

abbreviation,**command*

where *abbreviation* is the command alias that you enter at the Command prompt and *command* is the command being abbreviated. You must enter an asterisk (*) before the command name to identify the line as a command alias definition.

If you can enter a command transparently, you can also enter its alias transparently. When you enter the command alias, the full command name is displayed at the Command prompt and the command is executed.

You can create command aliases that include the special hyphen (–) prefix, such as those listed here, that access the command line version of certain commands.

```
BH, *-BHATCH
BD, *-BOUNDARY
```

NOTE You cannot use command aliases in command scripts. Using command aliases in customization files is not recommended.

If you edit *acad.pgp* while AutoCAD is running, enter **reinit** to use the revised file. You can also restart AutoCAD to automatically reload the file.

Custom Linetypes

2

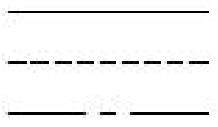
AutoCAD® provides a library of standard linetypes in the *acad.lin* and *acadiso.lin* files. You can use the linetypes as they are, modify them, or create your own custom linetypes.

In this chapter

Overview of Linetype Definitions

Linetypes are defined in one or more linetype definition files that have a *.lin* file extension.

The linetype name and definition determine the particular dash-dot sequence, the relative lengths of dashes and blank spaces, and the characteristics of any included text or shapes. You can use any of the standard linetypes that AutoCAD provides, or you can create your own linetypes.



examples of linetypes

Linetypes are defined in one or more linetype definition files that have an *.lin* file extension. An LIN file can contain definitions of many simple and complex linetypes. You can add new linetypes to an existing LIN file, or you can create your own LIN file. To create or modify linetype definitions, edit the LIN file using a text editor or word processor or use LINETYPE at the Command prompt.

When you create a linetype, you must load the linetype before you can use it.

The LIN files included in AutoCAD are *acad.lin* and *acadiso.lin*. You can display or print these text files to better understand how to construct linetypes.

Simple Custom Linetypes

Each linetype is defined on two lines in a linetype definition file. The first line contains the linetype name and an optional description. The second line is the code that defines the actual linetype pattern.

The second line must begin with the letter A (alignment), followed by a list of pattern descriptors that define pen-up lengths (spaces), pen-down lengths (dashes), and dots. You can include comments in an LIN file by beginning the line with a semicolon (;).

Linetype Definition Format

The format of the linetype definition is

```
*linetype_name,description
A,descriptor1,descriptor2, ...
```

For example, a linetype called DASHDOT is defined as

```
*DASHDOT,Dash dot __ . __ . __ . __ . __ . __ . __ . __
A,.5,-.25,0,-.25
```

This indicates a repeating pattern starting with a dash 0.5 drawing units long, a space 0.25 drawing units long, a dot, and another space 0.25 drawing units long. This pattern continues for the length of the line, ending with a dash 0.5 drawing units long. The linetype would be displayed as shown below.

__ . __ . __ . __ . __ . __ . __ . __

LIN files must be saved in ASCII format and use an *.lin* file extension. Additional information about each field in a linetype definition follows.

Linetype Name

The linetype name field begins with an asterisk (*) and should provide a unique, descriptive name for the linetype.

Description

The description of the linetype should help you visualize the linetype when you edit the LIN file. The description is also displayed in the Linetype Manager and in the Load or Reload Linetypes dialog box.

The description is optional and can include

- A simple representation of the linetype pattern using ASCII text
- An expanded description of the linetype
- A comment such as "Use this linetype for hidden lines"

If you omit the description, do not insert a comma after the linetype name. A description cannot exceed 47 characters.

Alignment Field (A)

The alignment field specifies the action for pattern alignment at the ends of individual lines, circles, and arcs. Currently, AutoCAD supports only A-type alignment, which guarantees that the endpoints of lines and arcs start and stop with a dash.

For example, suppose you create a linetype called CENTRAL that displays the repeating dash-dot sequence commonly used as a centerline. AutoCAD adjusts the dash-dot sequence on an individual line so that dashes and line endpoints coincide. The pattern fits the line so that at least half of the first dash begins and ends the line. If necessary, the first and last dashes are lengthened. If a line is too short to hold even one dash-dot sequence, AutoCAD draws a continuous line between the endpoints. For arcs also, the pattern is adjusted so that dashes are drawn at the endpoints. Circles do not have endpoints, but AutoCAD adjusts the dash-dot sequence to provide a reasonable display.

You must specify A-type alignment by entering **a** in the alignment field.

Pattern Descriptors

Each pattern descriptor field specifies the length of segments making up the linetype, separated by commas (no spaces are allowed):

- A positive decimal number denotes a pen-down (dash) segment of that length.
- A negative decimal number denotes a pen-up (space) segment of that length.
- A dash length of 0 draws a dot.

You can enter up to 12 dash-length specifications per linetype, provided they fit on one 80-character line in the LIN file. You need to include only one complete repetition of the linetype pattern defined by pattern descriptors. When the linetype is drawn, AutoCAD uses the first pattern descriptor for the starting and ending dashes. Between the starting and ending dashes, the pattern dash specifications are drawn sequentially, beginning with the second dash specification and restarting the pattern with the first dash specification when required.

A-type alignment requires that the first dash length be 0 or greater (a pen-down segment). The second dash length should be less than 0 if you need a pen-up segment and more than 0 if you are creating a continuous linetype. You must have at least two dash specifications for A-type alignment.

To create a simple linetype

1. At the Command prompt, enter **-linetype**.
2. Enter **c** (Create).
3. Enter a name for the linetype and press ENTER.

   The linetype name can include up to 255 characters. Linetype names can contain letters, digits, and the special characters dollar sign ($), hyphen (-), and underscore (_). Linetype names cannot include blank spaces.
4. In the Create or Append Linetype File dialog box, select an LIN linetype library file from the File Name box and choose Save.

   If you select an existing file, the new linetype name is added to the linetype names in the file.
5. Enter text that describes the new linetype (optional).

6 At the Enter Pattern prompt, specify the pattern of the line. Follow these guidelines:

- All linetypes must begin with a dash.
- Enter zeros for dots.
- Enter negative real numbers for spaces. The value defines the length of the space in drawing units.
- Enter positive real numbers for dashes. The value defines the length of the dash in drawing units.
- Separate each dot, dash, or space value from the next with a comma.
- Use a space between a dot and a dash.

7 Press ENTER to end the command.

NOTE When you create a linetype, it is not loaded into your drawing automatically. Use the Load option of LINETYPE.

Text in Custom Linetypes

Characters from text fonts can be included in linetypes. Linetypes with embedded characters can denote utilities, boundaries, contours, and so on. As with simple linetypes, lines are dynamically drawn as you specify the vertices. Characters embedded in lines are always displayed completely; they are never trimmed.

Embedded text characters are associated with a text style in the drawing. Any text styles associated with a linetype must exist in the drawing before you load the linetype.

The format for linetypes that include embedded characters is similar to that for simple linetypes in that it is a list of pattern descriptors separated by commas.

Character Descriptor Format

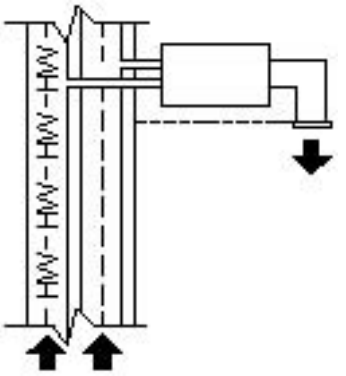
The format for adding text characters in a linetype description is as follows:

```
["text",textstylename,scale,rotation,xoffset,yoffset]
```

This format is added as a descriptor to a simple linetype. For example, a linetype called HOT_WATER_SUPPLY is defined as

```
*HOT_WATER_SUPPLY,---- HW ---- HW ---- HW ---- HW ---- HW ----
A,.5,-.2,["HW",STANDARD,S=.1,R=0.0,X=-0.1,Y=-.05],-.2
```

This indicates a repeating pattern starting with a dash 0.5 drawing units long, a space 0.2 drawing units long, the characters *HW* with some scale and placement parameters, and another space 0.2 drawing units long. The text characters come from the text font assigned to the STANDARD text style at a scale of 0.1, a relative rotation of 0 degrees, an *X* offset of -0.1, and a *Y* offset of -0.05. This pattern continues for the length of the line, ending with a dash 0.5 drawing units long. The linetype would be displayed as shown below.



Notice that the total upstroke length is 0.2 + 0.2 = 0.4 and that the text origin is offset -.01 units in the *X* direction from the end of the first upstroke. An equivalent linetype would be

```
*HOT_WATER_SUPPLY,---- HW ---- HW ---- HW ---- HW ---- HW ----
A,.5,-.1,["HW",STANDARD,S=.1,R=0.0,X=0.0,Y=-.05],-.3
```

The total upstroke is still 0.1 + 0.3 = 0.4, but the text origin is not offset in the *X* direction.

Additional information about each field in the character descriptor follows. The values to be used are signed decimal numbers such as 1, -17, and 0.01.

text

The characters to be used in the linetype.

text style name

The name of the text style to be used. If no text style is specified, AutoCAD uses the currently defined style.

scale

`S=value`. The scale factor to be used for the text style relative to the scale of the linetype. The height of the text style is multiplied by the scale factor. If the height is 0, the value for `S=value` alone is used as the height.

rotation

`R=value` or `A=value`. `R=` specifies relative or tangential rotation with respect to the line. `A=` specifies absolute rotation of the text with respect to the origin; that is, all text has the same rotation regardless of its position relative to the line. The value can be appended with a `d` for degrees (degrees is the default value), `r` for radians, or `g` for grads. If rotation is omitted, 0 relative rotation is used.

Rotation is centered between the baseline and the nominal cap height.

xoffset

`X=value`. The shift of the text on the *X* axis of the linetype, which is along the line. If `xoffset` is omitted or is 0, the text is elaborated with no offset. Use this field to control the distance between the text and the previous pen-up or pen-down stroke. This value is not scaled by the scale factor defined by `S=value`, but it is scaled to the linetype.

yoffset

`Y=value`. The shift of the text in the *Y* axis of the linetype, which is at a 90-degree angle to the line. If `yoffset` is omitted or is 0, the text is elaborated with no offset. Use this field to control the vertical alignment of the text with respect to the line. This value is not scaled by the scale factor defined by `S=value`, but it is scaled to the linetype.

To include text characters in linetypes

1. Create a simple linetype, as described in "To create a simple linetype" on page 20.
2. Add the text character descriptor within the linetype pattern, using the following format:

   [*"text",textstylename,scale,rotation,xoffset,yoffset*]
3. Press ENTER to exit LINETYPE.

Shapes in Custom Linetypes

A complex linetype can contain embedded shapes that are saved in shape files. Complex linetypes can denote utilities, boundaries, contours, and so on.

As with simple linetypes, complex lines are dynamically drawn as the user specifies vertices. Shapes and text objects embedded in lines are always displayed completely; they are never trimmed.

The syntax for complex linetypes is similar to that of simple linetypes in that it is a comma-delimited list of pattern descriptors. Complex linetypes can include shape and text objects as pattern descriptors, as well as dash-dot descriptors.

The syntax for shape object descriptors in a linetype description is as follows:

`[`*`shapename`*`,`*`shxfilename`*`] or [`*`shapename`*`,`*`shxfilename`*`,`*`transform`*`]`
where *`transform`* is optional and can be any series of the following (each preceded by a comma):

`R=##` Relative rotation

`A=##` Absolute rotation

`S=##` Scale

`X=##` *X* offset

`Y=##` *Y* offset

In this syntax, `##` is a signed decimal number (1, –17, 0.01, and so on), the rotation is in degrees, and the remaining options are in linetype-scaled drawing units. The preceding *`transform`* letters, if they are used, must be followed by an equal sign and a number.

The following linetype definition defines a linetype named CON1LINE that is composed of a repeating pattern of a line segment, a space, and the embedded shape CON1 from the *ep.shx* file. (Note that the *ep.shx* file must be in the support path for the following example to work properly.)

```
*CON1LINE, --- [CON1] --- [CON1] --- [CON1]
A,1.0,-0.25,[CON1,ep.shx],-1.0
```
Except for the code enclosed in square brackets, everything is consistent with the definition of a simple linetype.

As previously described, a total of six fields can be used to define a shape as part of a linetype. The first two are mandatory and position-dependent; the next four are optional and can be ordered arbitrarily. The following two examples demonstrate various entries in the shape definition field.

```
[CAP,ep.shx,S=2,R=10,X=0.5]
```
The code above draws the `CAP` shape defined in the *ep.shx* shape file with a scale of two times the unit scale of the linetype, a tangential rotation of 10 degrees in a counterclockwise direction, and an *X* offset of 0.5 drawing units before shape elaboration takes place.

```
[DIP8,pd.shx,X=0.5,Y=1,R=0,S=1]
```
The code above draws the `DIP8` shape defined in the *pd.shx* shape file with an *X* offset of 0.5 drawing units before shape drawing takes place, and a *Y* offset of one drawing unit above the linetype, with 0 rotation and a scale equal to the unit scale of the linetype.

The following syntax defines a shape as part of a complex linetype.

`[shapename,shapefilename,scale,rotate,xoffset,yoffset]`

The definitions of the fields in the syntax follow.

shapename

The name of the shape to be drawn. This field must be included. If it is omitted, linetype definition fails. If *shapename* does not exist in the specified shape file, continue drawing the linetype but without the embedded shape.

shapefilename

The name of a compiled shape definition file (SHX). If it is omitted, linetype definition fails. If *shapefilename* is unqualified (that is, no path is specified), search the library path for the file. If *shapefilename* is fully qualified and not found at that location, remove the prefix and search the library path for the file. If it is not found, continue drawing the linetype but without the embedded shape.

scale

`S=` *value*. The scale of the shape is used as a scale factor by which the shape's internally defined scale is multiplied. If the shape's internally defined scale is 0, the `S=` *value* alone is used as the scale.

rotate

`R=` *value* or `A=` *value*. `R=` signifies relative or tangential rotation with respect to the line's elaboration. `A=` signifies absolute rotation of the shape with respect to the origin; all shapes have the same rotation regardless of their relative position to the line. The value can be appended with a `d` for degrees (if omitted, degree is the default), `r` for radians, or `g` for grads. If rotation is omitted, 0 relative rotation is used.

xoffset

`X=` *value*. The shift of the shape in the *X* axis of the linetype computed from the end of the linetype definition vertex. If *xoffset* is omitted or is 0, the shape is elaborated with no offset. Include this field if you want a continuous line with shapes. This value is not scaled by the scale factor defined by `S=`.

yoffset

`Y=` *value*. The shift of the shape in the *Y* axis of the linetype computed from the end of the linetype definition vertex. If *yoffset* is omitted or 0, the shape is elaborated with no offset. This value is not scaled by the scale factor defined by `S=`.

See also:

"Shapes and Shape Fonts" on page 153

Custom Hatch Patterns

3

AutoCAD® provides a library of standard hatch patterns in the *acad.pat* and *acadiso.pat* files. You can use the hatch patterns as they are, modify them, or create your own custom hatch patterns.

In this chapter

Overview of Hatch Pattern Definitions

In addition to using the predefined hatch patterns that are supplied, you can design and create your own custom hatch patterns. Developing a hatch pattern definition requires knowledge, practice, and patience. Because customizing hatches requires familiarity with hatch patterns, it is not recommended for new users.

The hatch patterns supplied by AutoCAD are stored in the *acad.pat* and *acadiso.pat* text files. You can add hatch pattern definitions to this file or create your own files.

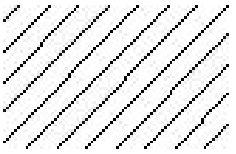
Regardless of where the definition is stored, a custom hatch pattern has the same format. It has a header line with a name, which begins with an asterisk and is no more than 31 characters long, and an optional description:

```
*pattern-name, description
```

It also has one or more line descriptors of the following form:

```
angle, x-origin,y-origin, delta-x,delta-y,dash-1,dash-2, …
```

The default hatch pattern ANSI31 shown in the Boundary Hatch and Fill dialog box looks like this:



and is defined as follows:

```
*ANSI31, ANSI Iron, Brick, Stone masonry
45, 0,0, 0,.125
```

The pattern name on the first line, `*ANSI31`, is followed by a description: `ANSI Iron, Brick, Stone masonry`. This simple pattern definition specifies a line drawn at an angle of 45 degrees, that the first line of the family of hatch lines is to pass through the drawing origin (0,0), and that the spacing between hatch lines of the family is to be 0.125 drawing units.

Hatch pattern definitions follow these rules:

- Each line in a pattern definition can contain up to 80 characters. You can include letters, numbers, and the special characters underline (_), hyphen (-), and dollar sign ($). However, you must begin a pattern definition with a letter or number, not a special character.
- AutoCAD ignores both blank lines and text to the right of a semicolon.

- Each pattern line is considered to be the first member of a line family, created by applying the delta offsets in both directions to generate an infinite family of parallel lines.
- The *delta-x* value indicates the displacement between members of the family in the direction of the line. It is used only for dashed lines.
- The *delta-y* value indicates the spacing between members of the family; that is, it is measured perpendicular to the lines.
- A line is considered to be of infinite length. A dash pattern is superimposed on the line.

The process of hatching consists of expanding each line in the pattern definition to its infinite family of parallel lines. All selected objects are checked for intersections with any of these lines; any intersections cause the hatch lines to be turned on and off as governed by the hatching style. Each family of hatch lines is generated parallel to an initial line with an absolute origin to guarantee proper alignment.

If you create a very dense hatch, AutoCAD may reject the hatch and display a message indicating that the hatch scale is too small or its dash length too short. You can change the maximum number of hatch lines by setting the MaxHatch system registry variable using (setenv MaxHatch n) where n is a number between 100 and 10000000 (ten million).

NOTE When changing the value of MaxHatch, you must enter MaxHatch with the capitalization as shown.

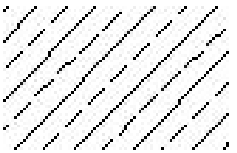
To create a simple hatch pattern

1. Open the *acad.pat* or *acadiso.pat* file in a text editor that saves in ASCII format (for example, Microsoft® Windows® Notepad).
2. Create a header line that includes an asterisk and a pattern name. The name of the hatch pattern is limited to 31 characters.
3. (Optional) To include a description in the header line, follow the pattern name with a comma and description text.
4. Create a descriptor line that includes
   - An angle at which the line is drawn
   - An *X,Y* origin point
   - A *delta-x* of 0
   - A *delta-y* of any value

Hatch Patterns with Dashed Lines

To define dashed-line patterns, you append dash-length items to the end of the line definition item. Each dash-length item specifies the length of a segment making up the line. If the length is positive, a pen-down segment is drawn. If the length is negative, the segment is pen-up, and it is not drawn. The pattern starts at the origin point with the first segment and cycles through the segments in circular fashion. A dash length of 0 draws a dot. You can specify up to six dash lengths per pattern line.

The hatch pattern ANSI33, shown in the Boundary Hatch and Fill dialog box, looks like this:



and is defined as follows:

```
*ANSI33, ANSI Bronze, Brass, Copper
45, .176776695,0, 0,.25, .125,-.0625
```

For example, to modify a pattern for 45-degree lines to draw dashed lines with a dash length of 0.5 units and a space between dashes of 0.5 units, the line definition would be

```
*DASH45, Dashed lines at 45 degrees
45, 0,0, 0,.5, .5,-.5
```

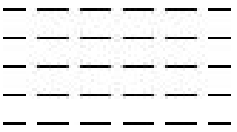
This is the same as the 45-degree pattern shown in "Overview of Hatch Pattern Definitions" on page 28, but with a dash specification added to the end. The pen-down length is 0.5 units, and the pen-up length is 0.5, meeting the stated objectives. If you wanted to draw a 0.5-unit dash, a 0.25-unit space, a dot, and a 0.25-unit space before the next dash, the definition would be

```
*DDOT45,Dash-dot-dash pattern: 45 degrees
45, 0,0, 0,.5, .5,-.25, 0,-.25
```

The following example shows the effect of *delta-x* specifications on dashed-line families. First, consider the following definition:

```
*GOSTAK
0, 0,0, 0,.5, .5,-.5
```

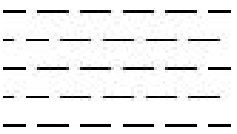
This draws a family of lines separated by 0.5, with each line broken equally into dashes and spaces. Because *delta-x* is zero, the dashes in each family member line up. An area hatched with this pattern would look like this:



Now change the pattern to

```
*SKEWED
0, 0,0, .5,.5, .5,-.5
```

It is the same, except that you have set *delta-x* to 0.5. This offsets each successive family member by 0.5 in the direction of the line (in this case, parallel to the *X* axis). Because the lines are infinite, the dash pattern slides down the specified amount. The hatched area would look like this:



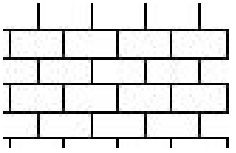
To create a hatch pattern with dashed lines

1. Open the *acad.pat* or *acadiso.pat* file in a text editor that saves in ASCII format (for example, Notepad).
2. Create a header line that includes an asterisk and a pattern name. The name of the hatch pattern is limited to 31 characters.
3. (Optional) To include a description in the header line, follow the pattern name with a comma and description text.
4. Create a descriptor line that includes
   - An angle at which the line is drawn
   - An *X,Y* origin point
   - A *delta-x* of any value if you want to offset alternating lines in the line family
   - A *delta-y* of any value
   - A value for a dash length
   - A value for a dot length
   - An optional second value for a different dash length
   - An optional second value for a different dot length

Hatch Patterns with Multiple Lines

Not all hatch patterns use origin points of 0,0. Complex hatch patterns can have an origin that passes through offsets from the origin and can have multiple members in the line family. In composing more complex patterns, you need to carefully specify the starting point, offsets, and dash pattern of each line family to form the hatch pattern correctly.

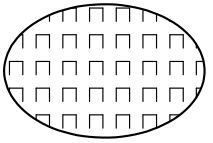
The hatch pattern AR-B816 shown in the Boundary Hatch and Fill dialog box looks like this:



and is defined as follows with multiple lines describing the pattern:

```
*AR-B816, 8x16 Block elevation stretcher bond
0, 0,0, 0,8
90, 0,0, 8,8, 8,-8
```

The following figure illustrates a squared-off, inverted-U pattern (one line up, one over, and one down). The pattern repeats every one unit, and each unit is 0.5 high and wide.



This pattern would be defined as follows:

```
*IUS,Inverted U's
90, 0,0, 0,1, .5,-.5
0, 0,.5, 0,1, .5,-.5
270, .5,.5, 0,1, .5,-.5
```

The first line (the up bar) is a simple dashed line with 0,0 origin. The second line (the top bar) should begin at the end of the up bar, so its origin is 0,.5. The third line (the down bar) must start at the end of the top bar, which is at .5,.5 for the first instance of the pattern, so its origin is at this point. The third line of the pattern could be the following:

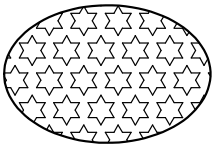
```
90, .5,0, 0,1, .5,-.5
```

or

```
270, .5,1, 0,1, -.5,.5
```

The dashed pattern starts at the origin points and continues in the vector direction given by the angle specification. Therefore, two dashed-line families that are opposed 180 degrees are not alike. Two solid-line families are alike.

The following pattern creates six-pointed stars.



This example can help you refine your skills at pattern definition. (Hint: 0.866 is the sine of 60 degrees.)

The following is the AutoCAD definition of this pattern:

```
*STARS,Star of David
0, 0,0, 0,.866, .5,-.5
60, 0,0, 0,.866, .5,-.5
120, .25,.433, 0,.866, .5,-.5
```

To create a hatch pattern with multiple lines

1. Open the *acad.pat* or *acadiso.pat* file in a text editor that saves in ASCII format (for example, Notepad).
2. Create a header line that includes an asterisk and a pattern name. The name of the hatch pattern is limited to 31 characters.
3. (Optional) To include a description in the header line, follow the pattern name with a comma and description text.
4. Create a descriptor line that includes
   - An angle at which the line is drawn
   - An *X,Y* origin point
   - A *delta-x* of any value if you want to offset alternating lines in the line family
   - A *delta-y* of any value
   - A value for a dash length
   - A value for a dot length
   - An optional second value for a different dash length

- An optional second value for a different dot length

5. Create a second line including all the parameters in the previous step.
6. (Optional) Create additional lines to complete the multiple-line hatch pattern.

Customize the User Interface

4

In this chapter

- Overview of Customizing the User Interface
- Important Customization Terms
- How Customization Has Changed
- Migrate and Transfer Customizations
- Create and Load a Partial CUI File
- Create an Enterprise CUI File
- Filter the Display of Customization Elements
- Customize Commands
- Customize Toolbars
- Create Pull-Down and Shortcut Menus
- Add Shortcut Keys and Temporary Override Keys
- Customize Workspaces
- Customize Mouse Buttons
- Create Status Line Help Messages
- Load an AutoLISP File

Overview of Customizing the User Interface

Using AutoCAD's customization tools, you can tailor your drawing environment to suit your needs.

Customization capabilities, including the CUI (Customize User Interface) file format and the Customize User Interface dialog box, help you to easily create and modify customized content. The XML-based CUI file replaces the menu files used in releases prior to AutoCAD 2006. Instead of using a text editor to customize menu files (MNU and MNS files), you customize the user interface from within AutoCAD. You can:

- Add or change toolbars and menus (including shortcut menus, image tile menus, and tablet menus)
- Create or change workspaces
- Assign commands to various user interface elements
- Create or change macros
- Define DIESEL strings
- Create or change aliases
- Add tooltips
- Provide descriptive text on the status line

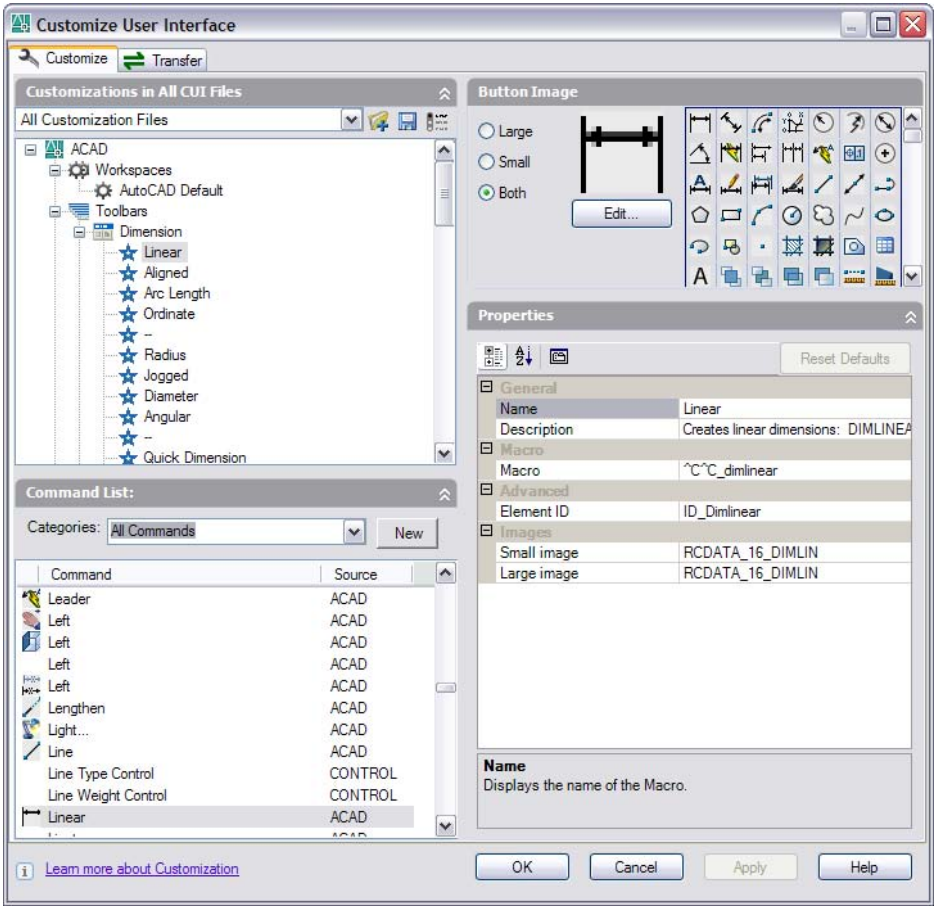
Customization Environment

Before you start customizing your own menus, toolbars, and other user interface elements, you should familiarize yourself with the customization environment. Open the Customize User Interface dialog box (click Tools menu ➤ Customize ➤ Interface) and view its contents, expand elements in the tree structure, and view element properties.

Select the Transfer tab to see how to migrate or transfer customizations; select the Customize tab to see how to create or modify user interface elements.

Once you are familiar with the environment, you can start to take advantage of the capabilities of the tools. For more information about the improved customization capabilities, see "How Customization Has Changed" on page 39.

Following is an example of the Customize User Interface dialog box, Customize tab. You use this tab to customize interface elements in CUI files.



Customize the user interface to make the drawing environment specific to certain types of tasks. For example, if you want a toolbar that contains the commands you use most often, you can create a new Favorites toolbar in the Customize User Interface dialog box and then load the new toolbar in AutoCAD.

Important Customization Terms

You should know several terms for customizing AutoCAD 2006.

Customization (CUI) file

An XML-based file that stores customization data. You modify a customization file through the Customize User Interface dialog box. CUI files replace MNU, MNS, and MNC files that were used to define menus in releases prior to AutoCAD 2006.

Main customization file

A writable CUI file that defines most of the user interface elements (including the standard menus, toolbars, keyboard accelerators, and so on). The *acad.cui* file (the default main CUI file) is automatically loaded when you start AutoCAD.

Enterprise customization file

A CUI file that is typically controlled by a CAD manager. It is often accessed by many users and is stored in a shared network location. The file is read-only to users to prevent the data in the file from being changed. A CAD manager creates an enterprise CUI file by modifying a main CUI file and then saving the file to a shared network location. Users then specify this file in the Options dialog box, Files tab.

Partial customization file

Any CUI file that is not defined as the main or enterprise CUI file. You can load and unload partial CUI files as you need them during a drawing session.

Customization group

A name that is assigned to a CUI file to identify customization content in the CUI file. A CUI file loaded into AutoCAD must have a unique customization group name to prevent conflicts between CUI files in the program. In previous releases, called a *menu group*.

Interface element

An object that can be customized, such as a toolbar, pull-down menu, shortcut key, dockable window, and so on. It is a node in the Customizations In <*file name*> pane that contains user interface items.

Interface item

The individual parts of a user interface element, such as a toolbar button, pull-down menu item, shortcut key, temporary override key, and so on.

Tree node

A hierarchical structure in the Customize User Interface dialog box that contains interface elements and items that can be imported, exported, and customized.

Workspace

A collection of user interface elements, including their contents, properties, display states, and locations.

Dockable window

An interface element that can be docked or floating in the drawing area. Dockable windows include the command window, tool palettes, Properties Palette, and so on.

Element ID

A unique identifier of an interface element. In previous releases, called a *tag*.

How Customization Has Changed

Although the basic customization techniques remain the same as in previous versions of the product, the environment in which you customize the product has changed.

All of the previous customization options are still available. You are still able to create, edit, and delete interface elements; you can create partial customization files; you use macros and advanced entries such as DIESEL expressions and AutoLISP routines.

However, you no longer perform customization tasks by creating or editing MNU or MNS text files by hand. All customizations are done through the program interface, in the Customize User Interface dialog box.

Menu Files Versus Customization Files

In releases prior to AutoCAD 2006, you customized the user interface by editing an MNU or MNS file in an ASCII text editor such as Notepad. You manually entered and verified customization data in the text file, which could be a tedious and error-prone process. As a result, a simple syntax error (such as mismatched parentheses) in the text file could invalidate the entire menu file, leading you back to the text file to investigate where you made the error.

With the Customize User Interface dialog box, you drag a command to a menu or toolbar or right-click to add, delete, or modify a user interface element. The Customize User Interface dialog box displays element properties and a list of

options from which you can choose. This prevents you from creating syntax errors or spelling mistakes that may have occurred when you manually entered text in a MNU or MNS file.

The MNU and MNS files used in the past have been replaced with just one file type, the XML-based CUI file.

The XML-based format of the CUI file allows the product to track customizations. When you upgrade to a future version of the program, all of your customizations are automatically integrated into the new release. The XML format also supports a backward-compatible customization file. This means that you can view a CUI file from a future version in the previous release while preserving the customization data from the future version. However, you cannot modify the future version's CUI file in the previous release. For more information about migrating customization data, see "Migrate and Transfer Customizations" on page 44.

The following table lists the menu files that previously shipped with the product and shows how those files are mapped to AutoCAD 2006.

Menu files mapped to CUI files

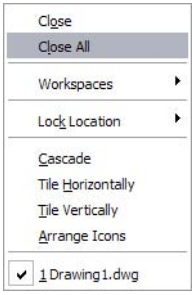
Menu file	Description	In AutoCAD 2006	Description of change
MNU	ASCII text file. In previous releases, defined most user interface elements. The main MNU file, *acad.mnu*, was automatically loaded when you started the product. Partial MNU files could be loaded or unloaded as you needed them during a drawing session.	CUI	An XML file that defines most user interface elements. The main CUI file, *acad.cui*, is automatically loaded when you start the product. Partial CUI files can be loaded or unloaded as you need them during a drawing session.
MNS	Source menu file. Was the same as the MNU ASCII text file but did not contain comments or special formatting.	CUI	An XML file that defines most user interface elements. The main CUI file, *acad.cui*, is automatically loaded when you start the product. Partial CUI files can be loaded or unloaded as you need them during a drawing session.
MNC	Compiled ASCII text file. Contained command strings and syntax that defined the functionality and appearance of user interface elements.	CUI	An XML file that defines most user interface elements. The main CUI file, *acad.cui*, is automatically loaded when you start the product.

Menu files mapped to CUI files			
Menu file	**Description**	**In AutoCAD 2006**	**Description of change**
			Partial CUI files can be loaded or unloaded as you need them during a drawing session.
MNL	Menu LISP file. Contains AutoLISP expressions that are used by the user interface elements.	MNL	No change.
MNR	Menu resource file. Contains the bitmaps that are used by the user interface elements.	MNR	No change.

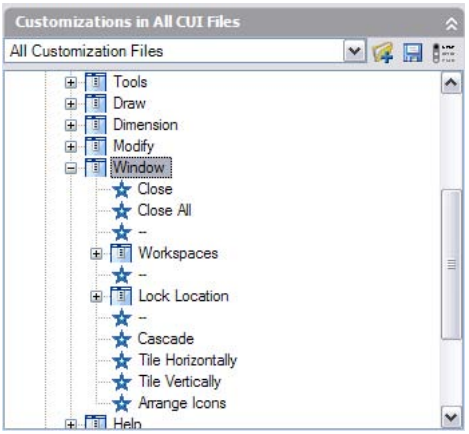
Menu Text File Structure Versus CUI Structure

In previous releases, you added, edited, and deleted menu information directly in a text file. In AutoCAD 2006, you use the Customize User Interface dialog box.

Following is an example of the Window menu in the *acad.mnu* ASCII text file:

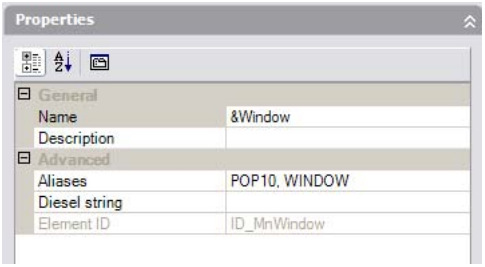
Contents of the Window menu in acad.mnu	
Window menu	**Description**
	***POP10 **WINDOW ID_MnWindow [&Window] ID_DWG_CLOSE [Cl&ose]^C^C_close ID_WINDOW_CLOSEALL [C&lose All]^C^C_closeall [--] ID_WINDOW_CASCADE [&Cascade]^C^C_syswindows;_cascade ID_WINDOW_TILE_HORZ [Tile &Horizontally]^C^C_syswindows;_hor ID_WINDOW_TILE_VERT [&Tile Vertically]^C^C_syswindows;_vert ID_WINDOW_ARRANGE [&Arrange Icons]^C^C_syswindows;_arrange

Compare the menu data above with the same menu data as it is displayed in the Customize User Interface dialog box, in the tree view.

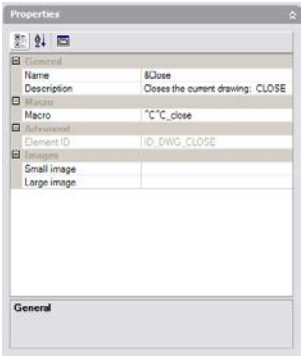


For a more detailed comparison, following are examples of the Window menu properties, Close command properties, Close All command properties, and the Window shortcut menu that is displayed with the Insert Separator option.

Window menu Properties pane

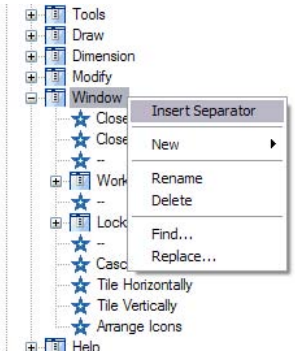


Window menu, Properties pane for the Close command



Window menu, Properties pane for the Close All command

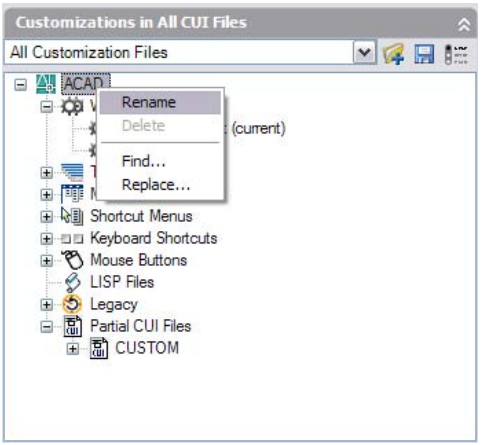
Window menu tree node, shortcut menu displayed with Insert Separator option



Menu Group Versus Customization Group

There is no difference between a *menu grou p* (the term used in previous releases) and a *customization group*. A CUI file loaded into AutoCAD must have a unique customization group name to prevent conflicts between customization files in the program. The main CUI file, *acad.cui* by default, has a customization group named ACAD. You can load as many customization files into the program, as long as they each have unique customization group name.

Following is an example of how you change the ACAD customization group name in the Customize tab of the Customize User Interface dialog box. You can change the partial CUI file (named CUSTOM in this example) using the same method.



See also:

"Migrate and Transfer Customizations" on page 44
"Create and Load a Partial CUI File" on page 46
"Create an Enterprise CUI File" on page 48

Migrate and Transfer Customizations

You can migrate custom MNU or MNS files from earlier releases using the Customize User Interface dialog box. The program transfers all of the data in the MNU or MNS file to a CUI file without modifying the original menu file. The new CUI file is an XML-based file that has the same name as your original menu file, but with a *.cui* extension.

You can also transfer customization information between files. For example, you can transfer toolbars from a partial CUI file to the main CUI file so that the program can display the toolbar information.

NOTE Button images may not appear in the program when you transfer a toolbar or menu from a partial CUI file. If the images are loaded from an image file, those images must reside in the same folder as the CUI file. If the images that are not displaying come from a third party resource DLL, contact the party who created the resource DLL.

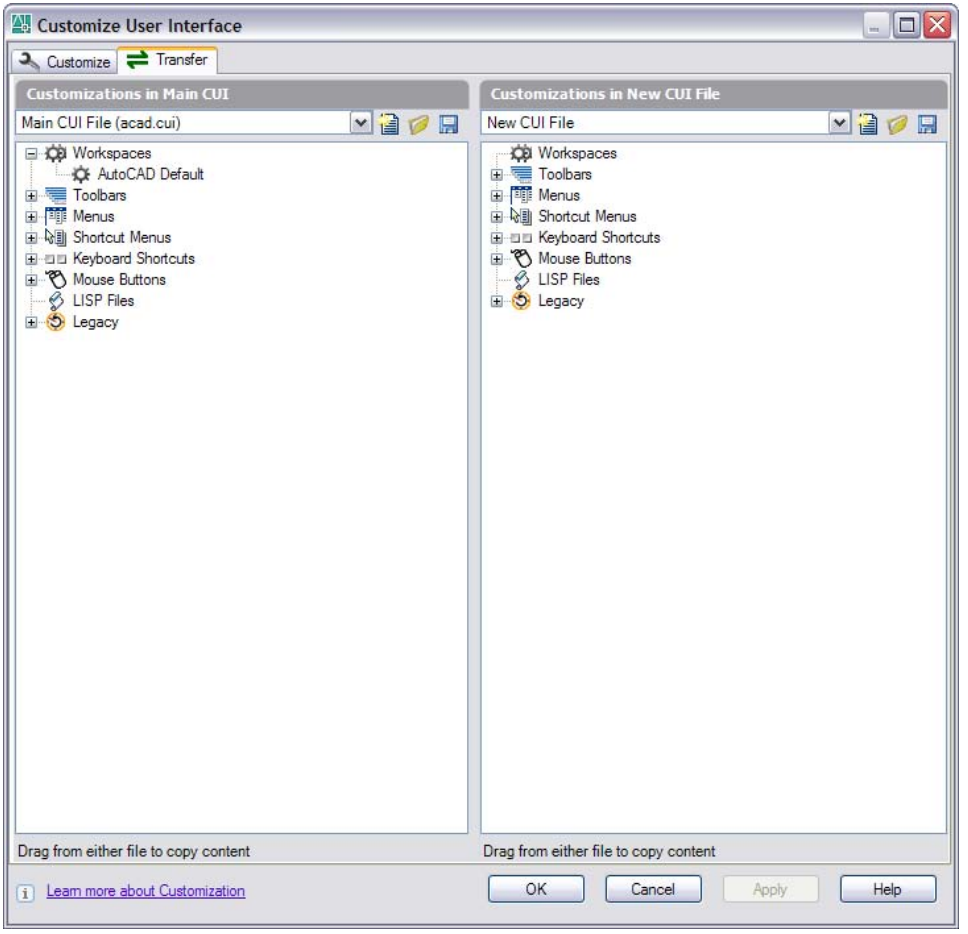
In addition, you can move customizations from the main CUI file to partial CUI files, or from a partial CUI file to another partial CUI file.

NOTE If a workspace or toolbar you are transferring contains flyout toolbars with references to another menu, toolbar, or flyout toolbar that is located in the source

CUI file, the relevant information for that interface element is also transferred. For example, if you transfer the Draw toolbar, which references the Insert toolbar, the Insert toolbar is also transferred.

A CUI file keeps track of any customizations you make. Customization data is tracked and preserved from release to release, so you can load a CUI file in another version without losing data or modifying existing CUI data.

Following is an example of the Customize User Interface dialog box, Transfer tab. You use this tab to migrate customizations.



NOTE CUI files cannot be displayed or used in versions prior to AutoCAD 2006.

To transfer customizations

1. Click Tools menu ➤ Customize ➤ Import Customizations.
2. In the Customize User Interface dialog box, on the Transfer tab, in the left pane, click the Open Customization File button.
3. In the Open dialog box, locate the customization file (MNU, MNS, or CUI) from which you want to export customizations, and select it. In the right pane, click the Open Customization File button.
4. In the Open dialog box, locate the customization file (MNU, MNS, or CUI) to which you want to import customizations, and select it.
5. In the left pane, click the plus sign (+) next to an interface element node to expand it. Expand the corresponding node in the right pane.
6. Drag an interface element from the left pane to the appropriate location in the right pane. Menus can be dragged to menus, toolbars to toolbars, and so on.
7. When you finish transferring customizations, click OK.

Create and Load a Partial CUI File

Create, load, or unload partial customization files as you need them. Loading and using a partial CUI file allows you to create and modify most interface elements (toolbars, menus, and so on) in a separate CUI file without having to import the customizations to your main CUI file.

The order of the partial CUI files in the Partial CUI Files tree determines the order they are loaded in the program. You can rearrange the tree hierarchy to change the load order. Use the Transfer tab of the Customize User Interface dialog box to create a partial CUI file. See "To transfer customizations " on page 46 for more information. To load or unload a CUI file, you can use the CUILOAD or CUIUNLOAD command in the program or you can use the Customize tab in the Customize User Interface dialog box.

NOTE When you load a partial CUI file, its workspace information (menus, toolbars, and dockable windows) is ignored by the main CUI file. To add workspace content from a partial CUI file to a main CUI file, you must transfer the workspace. For more information, see "Import a Workspace to a Main CUI File" in the "Customize Workspaces" on page 85 topic.

To load a partial CUI file using the CUILOAD command

1. On the command line, enter **cuiload** .

2. In the Load/Unload Customizations dialog box, in the File Name box, enter a path to the CUI file you want to load, or click Browse to locate the file.
3. Click Load, and then click Close.

To load a partial CUI file using the Customize tab

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, select Main CUI File from the drop-down list. To the right of the drop-down list, click the Load Partial Customization File button.
3. In the Open dialog box, locate and click the partial CUI file you want to open, and click Open.

   NOTE If the partial CUI file you are attempting to load has the same customization group name as the main CUI file, you need to change the customization group name. Open the CUI file in the Customize dialog box, select the file name, and right-click to rename it.

4. To verify that the file has been loaded into the main CUI file, in the Customizations In pane, select the main CUI file from the drop-down list.
5. In the tree view of the main customization file, click the plus sign (+) next to the Partial CUI Files node to expand it.

   Any partial menus loaded in the main CUI file are displayed.
6. Click OK to save the changes and view them in the program.

To unload a partial CUI file using the CUIUNLOAD command

1. On the command line, enter **cuiunload**.
2. In the Load/Unload Customizations dialog box, in the Loaded Customizations Group box, select a CUI file.
3. Click Unload, and then click Close.

To unload a partial CUI file using the Customize tab

1. Click Tools menu ➤ Customize ➤ Interface.

2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, select Main CUI File from the drop-down list.
3. In the tree view of the main customization file, click the plus sign (+) next to the Partial CUI Files node to expand it.

   Any partial menus loaded in the main CUI file are displayed.
4. Right-click the partial menu that you want to unload. Click Unload CUI File.

   The file is removed from the list.
5. Click OK to save the changes and view them in the program.

Create an Enterprise CUI File

An enterprise CUI file typically stores customization information that is shared by many users but is controlled by a CAD manager. Enterprise CUI files make maintaining and modifying customization data easier for the individual responsible for controlling users' drawing environments.

Create an enterprise CUI file by performing the following tasks:

- **Create an enterprise CUI file from an existing CUI file.** By copying the main customization file (*acad.cui*), you start with a file that contains all of the interface elements you need.
- **Designate the new file as your main CUI file.** Using the Options dialog box, you can make the enterprise file you create the main customization file.
- **Modify the contents of the enterprise CUI file.** Once the enterprise file you created is designated as the main CUI file, you change the customization group name and modify the CUI file contents as needed. Changing the customization group name allows you to load more than one CUI file in the program at one time. CUI files with the same customization group name cannot be loaded into the program.
- **Save the enterprise file to a shared network location.** When you save the new enterprise file to a shared network location, all of your users can access the file but they cannot modify the file.

NOTE The folder where you place the enterprise CUI file must be a shared location that your users can access. To learn more about creating a network share, see "How to Create a Network Share" in the *Network Administrator's Guide*.

■ **Specify the enterprise file location.** The program automatically designates an enterprise file as read-only when you specify its file location in the Options dialog box. Specifying the enterprise file location can be done on individual workstations or in the Deployment wizard. For more information about setting the location in the Deployment wizard, see Select Search Paths and File Locations (optional) in the *Network Administrator's Guide*.

To create a CUI file from an existing CUI file

1. In Windows Explorer, navigate to the following location:

   C:\Documents and Settings\<user profile name>\Application Data\Autodesk\<product name>\<release number>\enu\support\<customization file name>.cui

   NOTE In some operating systems, the folders that are located under your profile are hidden by default. To display these files, you may need to change your display settings. Click Start menu ➤ Control Panel ➤ Folder Options. In the Folder Options dialog box, View tab, click Show Hidden Files and Folders.

2. Copy the selected CUI file to a new file name (such as *enterprise.cui*) or location (such as the shared network location where users will access the file) so that you preserve the original CUI file (in case you want to modify or use it again later).

To designate a CUI file as the main CUI file

1. Click Tools ➤ Options.
2. In the Options dialog box, Files tab, click the plus sign (+) next to Customization Files to expand the list.
3. Click the plus sign next to Main Customization File to open it.
4. Click Browse. In the Select a File dialog box, browse to the location of the main customization file. Click Open.

   The file you created is now designated as the main CUI file in the program.

To change a customization group name and modify a CUI file

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize In *<file name>* pane, select a CUI file from the drop-down list.

3 In the Customize In <*file name*> pane, click the file name. Right-click the file name, and click Rename. Enter a new customization group name.

NOTE The customization group name cannot contain spaces.

4 Modify any elements necessary.

5 When you are finished modifying the CUI file, click OK.

To define an enterprise CUI file

1 On each user's workstation, in AutoCAD, click Tools ➤ Options.

2 In the Options dialog box, Files tab, click the plus sign (+) next to Customization Files to expand the list.

3 Click the plus sign next to Enterprise Customization File to open it.

4 Click Browse. In the Select a File dialog box, browse to the location of the enterprise customization file. Click Open.

The CUI file must be saved in a shared network location that users can access.

5 In the Options dialog box, click OK.

NOTE In the Deployment wizard, you designate the enterprise CUI file in the wizard's Specify Settings page. For more information about designating an enterprise CUI file in the Deployment wizard, see Select Search Paths and File Locations (optional) in the Network Administrator's Guide.

Filter the Display of Customization Elements

You can display all elements that you want to customize or selected elements only. You filter the display of customization elements on the Customize tab in the Customize User Interface dialog box.

To filter the display of customization elements

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, in the Customize In <*file name*> pane, select a CUI file from the drop-down list.

3 In the Customize In <*file name*> pane, click the Display Filters button.

4 In the Display Filters dialog box, click the check box next to the following elements to display or hide an element. Checked items are displayed in

the tree view in the Customizations In <*file name*> pane. Items without a check mark are hidden.

- Toolbars
- Menus
- Shortcut menus
- Keyboard shortcuts
- Mouse buttons
- Legacy
- LISP Files

5 Click OK to Close the Display Filters dialog box.

6 When you are finished customizing, click OK.

Customize Commands

You can easily create, edit, and reuse commands. The Customize tab of the Customize User Interface dialog box displays a master list of commands that are loaded in the product. You can add any commands from this list to toolbars, menus, and other user interface elements.

When you change properties of a command in the master list or on the tree view, the properties of the command are changed everywhere that command is used.

The following table shows the Scale command properties as they appear in the Properties pane.

Properties for the Scale Command in the Modify Menu

Properties pane item	Description	Example
Name	The string is displayed as a menu name or as a tooltip when you click a toolbar button. The string must include alphanumeric characters with no punctuation other than a hyphen (-) or an underscore (_).	Sca&le
Description	The status line text. This string is displayed on the status bar when the cursor hovers over a toolbar button or menu item.	Enlarges or reduces objects proportionally in the X, Y, and Z directions: SCALE

Properties for the Scale Command in the Modify Menu		
Properties pane item	**Description**	**Example**
Macro	The command macro. It follows the standard macro syntax.	$M=$(if,$(eq,$(substr,$(getvar,cmdnames),1,6),'SCALE'),^C^C_scale)
	When you change the name of a macro, the name of its corresponding menu item or toolbar button does not change. You must change a menu item or toolbar button name by selecting it in the tree view.	
Element ID	The tag that uniquely identifies a command.	ID_Scale
Small Image	The ID string of the small-image resource (16 × 16 bitmap). The string must include alphanumeric characters with no punctuation other than a hyphen (-) or an underscore (_). It can also specify a user-defined bitmap.	RCDATA_16_ZOOSCA
Large Image	The ID string of the large-image resource (32 × 32 bitmap). If the specified bitmap is not 32 × 32, the program scales it to that size. The string must include alphanumeric characters with no punctuation other than a hyphen (-) or an underscore (_). This can also specify a user-defined bitmap.	RCDATA_16_ZOOSCA

Create, Edit, and Reuse Commands

You can create a new command from scratch or you can edit the properties of an existing command. When you create or edit a command, the properties you can define are the command name, description, macro, element ID (for new commands only), and large or small image.

When you change any properties of a command in the Command List pane, the command is updated for all interface items that reference that command.

To create a command

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, Command List pane, click New.

A new command (named Command1) is displayed in both the Command List pane and the Properties pane.

3 In the Properties pane, do the following:

- In the Name box, enter a name for the command. The name will be displayed as a tooltip or menu name when you select this command.
- In the Description box, enter a description for the command. The description will be displayed on the status bar when the cursor hovers over the menu item or toolbar button.
- In the Macro box, enter a macro for the command.
- In the Element ID box, enter an element ID for the command. (For new commands only. You cannot modify the element ID of an existing command.)

For information about adding a button image to a command, see "Create and Edit Toolbar Buttons" on page 65.

To edit a command

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, do one of the following:

- In the Command List pane, click the command you want to edit.
- In the tree view pane, locate and then click the command you want to edit.

3 In the Properties pane, do any of the following to edit the command:

- In the Name box, enter a new name for the command. In the program, the name is displayed on the menu where you assign this command.
- In the Description box, enter a new description for the command. The description is displayed on the status bar when the command is selected in the program.
- In the Macro box, enter a new macro for the command.
- In the Element ID box, enter a new element ID for the command. (For new commands only. You cannot modify the element ID of an existing command).

For information about adding a button image to a command, see "Create and Edit Toolbar Buttons" on page 65.

To reuse a command

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, Command List pane, locate the command you want to reuse and drag it to an interface element.

   For information about adding a button image to a command, see "Create and Edit Toolbar Buttons" on page 65.

Find Command Names and Search Strings

You can search one or more CUI files for commands or search strings (including command names, descriptions, and macros). You can also replace commands or search strings one at a time or all at once.

You can limit or expand your search depending on the search results you want to achieve.

- **Limit the search to commands located in the Command List pane.** This search does not include command properties such as the command description or the assigned macro. For example, if you limit the search for the LINE command in the command list only, a message similar to the following is displayed when you start your search: "Command found in tree node 'Line' (1/3)."
- **Expand the search to include all properties in all tree view nodes in the Customizations In pane.** This type of search finds all instances or a search string. For example, if you search for the search string "line" and start in the tree view, a message similar to the following is displayed: "Search string found in tree node 'Linear' property 'Name' at position 0 (1/358)."

To find a search string

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, right-click anywhere in the tree view of the Customizations In *<file name>* pane.
3. In the Find and Replace dialog box, Find tab, do the following:
   - In the Find What box, enter the search string.
   - In the Ignore Case option, clear the check box if you want the search to find every instance of the search string regardless of its case.

- In the Restrict Search To option, select the check box if you want to restrict the search to just one CUI file. Then, under this option, select a CUI file from the drop-down list.
- Click Find Next to locate all instances of the search string.

A message is displayed that details the location of the search string and the number of results generated from the search.

4. Click Find Next to continue your search.
5. When you finish, click Close.
6. In the Customize User Interface dialog box, click Close.

To find a command in the Command List pane

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Command List pane, right-click the command name you want to find.
3. In the Find and Replace dialog box, Find tab, do the following:
   - In the Find What box, enter the command name.
   - In the Ignore Case option, clear the check box if you want the search to find every instance of the search string regardless of its case.
   - In the Restrict Search To option, select the check box if you want to restrict the search to just one CUI file. Then, under this option, select a CUI file from the drop-down list.
   - Click Find Selected Command to locate all instances of the command.
4. In the text that is displayed, view each location of the command or search string, its exact position in the tree node or Properties pane, and the number of instances in which the command or search string occurs.
5. Click Find Selected Command to continue your search.
6. When you finish, click Close.
7. In the Customize User Interface dialog box, click Close.

To replace a search string

1. Click Tools menu ➤ Customize ➤ Interface.

2. In the Customize User Interface dialog box, Customize tab, Customizations In *<file name>* pane, right-click anywhere in the tree view. Click Replace.
3. In the Find and Replace dialog box, Replace tab, do the following:
   - In the Find What box, enter the search string.
   - In the Replace With box, specify the text string you want to use to replace the found string.
   - In the Ignore Case option, clear the check box if you want the search to find every instance of the search string, regardless of its case.
   - In the Restrict Search To option, select the check box if you want to restrict the search to just one CUI file. Then, under this option, select a CUI file from the drop-down list.
   - To step through each instance of a found string before replacing it, click Replace. In the text that is displayed, view each location of the search string, its exact position in the tree node or Properties pane, and the number of instances in which the search string occurs. You cannot undo this action.
   - To replace all instances of the search string, click Replace All. You cannot undo this action.
4. When you finish, click Close.
5. In the Customize User Interface dialog box, click Close.

To replace a command

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, Command List pane, right-click the command name you want to replace.
3. In the Find and Replace dialog box, Replace tab, in the Find What box, the command name you selected in the previous step is displayed. To complete the dialog box, do the following:
   - In the Replace With box, specify the command name you want to use to replace the found command.
   - In the Ignore Case option, clear the check box if you want the search to find every instance of the command, regardless of its case.
   - In the Restrict Search To option, select the check box if you want to restrict the search to just one CUI file. Then, under this option, select a CUI file from the drop-down list.

- To step through each instance of a command name before replacing it, click Replace. In the text that is displayed, view each location of the command, its exact position in the tree node or Properties pane, and the number of instances in which the command occurs. By renaming the command in the command list, you rename the command *everywhere* that command is used in the CUI file. You cannot undo this action.
- To replace all instances of the command, click Replace All. You cannot undo this action.

4. When you finish, click Close.
5. In the Customize User Interface dialog box, click Close.

Control the Display of Command Labels

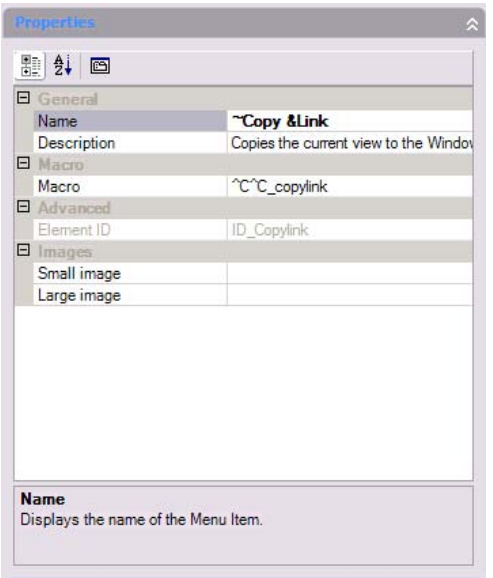
You can control the way that menu labels indicate a command's availability in the program. Display menu commands as grayed out (disabled), mark them with a check mark or border, or use a combination of indicators.

Menu commands can also contain DIESEL string expressions that gray out, mark, or interactively change the text of the displayed label. For more information about using DIESEL expressions, see "DIESEL Expressions in Macros" on page 113.

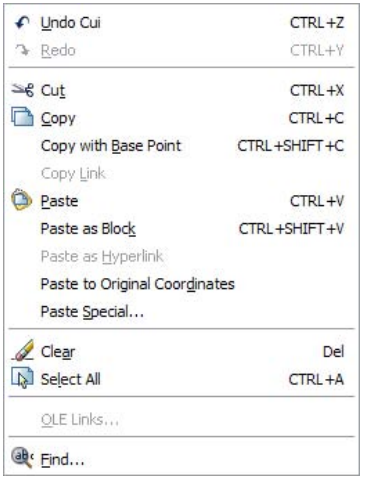
Gray Out (Disable) Menu Labels

To gray out a label in a menu, you begin the command name with a tilde (~). Any commands associated with the item are not issued, and submenus are inaccessible.

In the following example, the tilde (~) is placed at the beginning of the Copy Link command label in the Name cell of the Properties pane.



Following is the resulting Copy Link command grayed out in the Edit menu.



Command labels can contain DIESEL string expressions that conditionally disable or enable command labels each time they are displayed. For example, the DIESEL string expression in the Macros cell of the Properties pane disables the MOVE command while any other command is active.

```
$(if,$(getvar,cmdactive),~)MOVE^C^C_move
```

The AutoLISP **menucmd** function can also be used to disable and enable items from a macro or application. For examples, see "Reference Pull-Down or Shortcut Menus" on page 74.

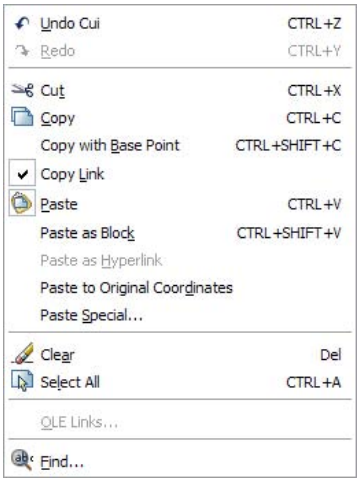
Mark Menu Labels

You can mark a menu label by including an exclamation point and a period (`!.`) in the Name cell of the Properties pane for the command. A menu item is marked in one of two ways:

- **A check mark.** Displayed when a menu item does not have an image associated with it.

- **A border.** Displayed when a menu item has an image associated with it; a border is displayed around the image.

Following is an example of the Edit menu with the Copy Link command marked with a check mark and the Paste command's image marked with a border:



Command labels can also contain DIESEL string expressions that conditionally mark command labels each time they are displayed. When the following DIESEL string is added to the Macros cell for the applicable command in the Properties pane, a check mark is placed to the left of the menu label whose related system variable is currently enabled.

```
$(if,$(getvar,orthomode),!.)Ortho^O
$(if,$(getvar,snapmode),!.)Snap^B
$(if,$(getvar,gridmode),!.)Grid^G
```

The AutoLISP **menucmd** function can be used to mark labels from a macro or application. For examples, see "Reference Pull-Down or Shortcut Menus" on page 74.

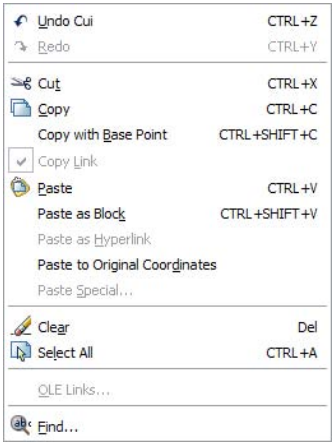
Simultaneously Disable and Mark Command Labels

You can mark and disable commands at the same time using either of the following formats:

```
~!. labeltext
!.~ labeltext
```

The tilde (~) is the special character code to disable a command and an exclamation point and period (!.) is the special character code to mark a command.

The tilde (~), exclamation point, and period (`!.`) are placed at the beginning of the Copy Link command label in the Name cell of the Properties pane. Following is the resulting Copy Link marked and grayed out in the Edit menu.



As with the previous examples, a DIESEL expression can be used to simultaneously disable and mark a command label.

See also:

"DIESEL Expressions in Macros" on page 113

To gray out (disable) a command's menu label

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to the menu that contains the command you want to disable.

3 Click the command you want to gray out.

4 In the Properties pane, in the Name cell, add a tilde (~) at the beginning of the command

5 Click OK.

To mark command's menu label

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to the menu that contains the command you want to disable.

3 Click the command you want to gray out.

4 In the Properties pane, in the Name cell, add an exclamation point and a period (!.) at the beginning of the command.

5 Click OK.

To simultaneously gray out (disable) and mark a command's menu label

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to the menu that contains the command you want to disable.

3 Click the command you want to gray out.

4 In the Properties pane, in the Name cell, add a tilde, an exclamation point, and a period (~!. or !.~) at the beginning of the command.

5 Click OK.

Customize Toolbars

Toolbar customization can be as easy as placing or resizing a toolbar in a drawing area to gain the most drawing efficiency or space. You can also create and modify toolbars and flyout toolbars, adding commands and control elements, and creating and editing toolbar buttons.

Create and Edit Toolbars

Some of the simplest toolbar customizations can make your daily drawing tasks more efficient. For example, you can consolidate frequently used buttons

onto one toolbar, remove or hide toolbar buttons that you never use, or change some simple toolbar properties.

You can also specify information to be displayed when the cursor passes over a button.

You can add buttons to toolbars, remove buttons you use infrequently, and rearrange buttons and toolbars. You can also create your own toolbars and flyout toolbars, and create or change the button image associated with a toolbar command.

NOTE When you create a toolbar, you should determine in which workspaces you want to display the toolbar. By default, a new toolbar is displayed in all workspaces.

A flyout is a set of buttons nested under a single button on a toolbar. Flyout buttons have a black triangle in the lower-right corner. To create a flyout, you can start from scratch or drag an existing toolbar onto another toolbar.

When you create a new toolbar or flyout toolbar, the first task you need to do is assign a name to it. A new toolbar has no commands or buttons assigned to it. It is ignored by the program if you do not add at least one command to it. You can drag commands and add buttons onto the new toolbar from existing toolbars or from commands listed on the Command List pane in the Customize User Interface dialog box.

To create a toolbar

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, right-click Toolbars. Click New ➤ Toolbar.

   A new toolbar (named Toolbar1) is placed at the bottom of the Toolbars tree.
3. Do one of the following:
   - Enter a new name over the Toolbar1 text.
   - Right-click Toolbar1. Click Rename. Enter a new toolbar name.
4. Select the new toolbar in the tree view, and update the Properties pane:
   - In the Description box, enter a description for the toolbar.
   - In the On By Default box, click Hide or Show. If you choose Show, this toolbar will be displayed in all workspaces.
   - In the Orientation box, click Floating, Top, Bottom, Left, or Right.

- In the Default X Location box, enter a number.
- In the Default Y Location box, enter a number.
- In the Rows box, enter the number of rows for an undocked toolbar.
- In the Aliases box, enter an alias for the toolbar.

5. In the Command List pane, drag the command you want to add to a location just below the name of the toolbar in the Customizations In <*file name*> pane.
6. When you finish adding commands to the new toolbar, click OK or continue customizing.

To create a flyout toolbar from scratch

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to the Toolbars tree node to expand it.
3. Right-click the toolbar to which you want to add a flyout toolbar. Click New ➤ Flyout.

   A new flyout toolbar (named Toolbar1) is placed below the toolbar you selected.
4. Right-click Toolbar1. Click Rename. Enter a new toolbar name.
5. In the Command List pane, drag the command you want to add to a location just below the toolbar flyout name in the Customizations In <*file name*> pane.
6. When you finish adding commands to the new flyout, click OK.

To create a flyout toolbar from another toolbar

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to the Toolbars tree node to expand it.
3. Click the plus sign (+) next to the toolbar to which you want to add a flyout toolbar.
4. Locate the toolbar you want to add as a flyout. Drag that toolbar to a location in the expanded toolbar.

5 Click OK.

To add a command to a toolbar

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, Command List pane, drag the command you want to add to a location just below the toolbar name in the Customizations In <*file name*> pane.

Click the plus sign (+) to the left of the toolbar to display the command you just added.

3 When you finish adding commands to the toolbar, click OK.

To reposition a button on a toolbar

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the toolbar whose buttons you want to reposition.

3 Click the plus sign (+) next to the toolbar to expand it.

4 Drag the name of the button you want to reposition to the new location in the list of tools.

When the splitter bar is displayed, you can place the button between two buttons. When the left arrow appears, you can place the button below another button.

5 When you finish repositioning buttons, click OK.

To delete a button from a toolbar

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the toolbar whose buttons you want to delete.

3 Click the plus sign (+) to the left of the toolbar to expand it.

4 Right-click the name of the button you want to remove. Click Delete.

5 When you finish deleting buttons, click OK.

To change properties of a toolbar

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the toolbar whose properties you want to change.
3. In the Properties pane, make your changes.

   NOTE Before you attempt to change an alias in a toolbar, you need to understand how aliases function. For more information about aliases, see "Create Command Aliases" on page 14.

4. When you finish changing properties, click OK.

Create and Edit Toolbar Buttons

After you create a toolbar, you can add buttons that are supplied by Autodesk or you can edit or create buttons.

Autodesk provides standard button images for buttons that start commands. You can create custom button images to run custom macros. You can either modify an existing button image or create your own. Button images are saved as BMP files. The BMP files must be saved in the same folder as the CUI file that it references.

User-defined bitmaps can be used in place of the `small image` and `large image` resource names in button and flyout commands.

Small images should be 16 x 16 pixels. Large images should be 32 x 32 pixels. Images that do not match these sizes are scaled to fit.

See also:

"Create Macros" on page 99
"Overview of File Organization" on page 3

To change the command associated with a button

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, Command List pane, click the command you want to change.

   In the Button Images pane, the image associated with the command (if any) is displayed.

3 In the Properties pane, in the Macro section, edit the macro by changing the command information.

TIP Most commands start with ^C^C to cancel a command that may be running. An underscore (_) enables commands to work on international versions of the product. An apostrophe (') enables the command to work transparently (if possible). If you associate a series of commands with a button, separate them with semicolons or spaces. Semicolons are easier to see than spaces.

For more information about entering commands in macros, see "Create Macros" on page 99.

4 Click OK.

To edit or create a button image

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Command List pane, click any command to display the Button Images pane (in the upper-right corner).

3 In the Button Images pane, click a button that is closest in appearance to the button you want to create. Click Edit.

4 In the Button Editor, use the Pencil, Line, Circle, and Erase buttons to create or edit the button image. To use color, select a color from the color palette, or click More to open the "True Color Tab (Select Color Dialog Box)".

- **Pencil button**. Edits one pixel at a time in the selected color. You can drag the pointing device to edit several pixels at once.
- **Line button**. Creates lines in the selected color. Click and hold to set the first endpoint of the line. Drag to draw the line. Release to complete the line.
- **Circle button**. Creates circles in the selected color. Click and hold to set the center of the circle. Drag to set the radius. Release to complete the circle.
- **Erase button.** Sets the pixels to white.

NOTE You cannot edit flyout buttons.

5 To save the customized button as a BMP file, click Save. Use Save As to save it under a different name. Save the new button image to the following location:

C:\Documents and Settings\<user profile name>\Application Data\Autodesk\<product name>\<release number>\enu\support

NOTE You can save buttons in BMP (**.bmp*, **.rle*, or **.dib*) format only.

Add or Switch Toolbar Controls

Toolbar controls are drop-down lists of toolbar-specific options that you can choose from a toolbar. For example, the Layers toolbar contains controls that allow you to define layer settings. In the Customize User Interface dialog box, you can add, remove, and relocate controls within toolbars.

The following table lists the toolbar controls found in the Customize User Interface dialog box and their definitions. The control elements in the left column of this table are not always the text that is displayed as a tooltip in the program (for example, Undo Skinny Button is displayed as Undo in the program's tooltip). Refer to this table when you want to change a control in a toolbar.

Control elements for toolbars

Control element	Description
Dim Style Control	Drop-down list that provides specification of the current dimension style.
Layer Control	Drop-down list that provides control of the current layers in the drawing.
Line Type Control	Drop-down list that provides specification of the current linetype.
Line Weight Control	Drop-down list that provides specification of the current lineweight.
OPT Color Control	Drop-down list that provides specification of the current color.
Plot Style Control	Drop-down list that provides specification of the current plot style.
Reference Block Name Control	Displays the current xref name in edit mode.
UCS Control	Drop-down list that provides specification of the current UCS.
View Control	Drop-down list that provides specification of the current standard 3D views.

Control elements for toolbars	
Control element	**Description**
Viewport Scale Control	Drop-down list that provides specification of viewport scaling in layouts.
Undo Skinny Button Control	Standard toolbar button that cancels the previous action.
Redo Skinny Button Control	Standard toolbar button that repeats the previous action.
Text Style Control	Drop-down list that sets the current text style.
Table Style Control	Drop-down list that sets the current table style.
Named View Control	Drop-down list that displays the named view.
Workspaces Control	Drop-down list that sets the current workspace.

See also:

"Customize Toolbars" on page 61

To add a control to a toolbar

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, Customizations In <*file name*> pane, click the plus sign (+) next to the toolbar to which you want to add a control.
3. In the Command List pane, in the Categories list, click Control Elements.

   The Command List pane displays control elements only.
4. In the Command list, drag the control to the Customizations In <*file name*> pane to the position where you want to add it in the toolbar.
5. Click OK.

To switch a control in a toolbar

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, Customizations In <*file name*> pane, click the plus sign (+) next to the toolbar that contains the control element you want to switch.
3. Click the control element.

4. In the Properties pane, in the Control box, click the arrow to display a list of controls.
5. Click a control to replace the original control with the one you selected.
6. Click OK.

Create Pull-Down and Shortcut Menus

Pull-down menus are displayed as a list under a menu bar. Shortcut menus (also called context menus) are displayed at or near the crosshairs or cursor when you right-click in the drawing window, text window, command window, or in toolbar areas.

A pull-down menu can contain up to 999 commands. A shortcut menu can contain up to 499 commands. The command limit includes all menus in a hierarchy. If commands in the menu file exceed these limits (which is unlikely), the program ignores the extra commands. If a pull-down or shortcut menu is longer than the available display space, it is truncated to fit.

Pull-Down Menu Aliases

Pull-down menus should have one alias in the range of POP1 through POP499. Menus with an alias of POP1 through POP16 are loaded by default when a menu loads. All other menus must be added to a workspace to be displayed.

NOTE When you create a pull-down or shortcut menu, you must also add a command to the menu. Otherwise, the menu will not be saved to the file.

Create a Pull-Down Menu

You can add commands to the menu, and create or add images to each menu command.

NOTE When you create a menu, you should determine in which workspaces you want to display the menu. By default, a new menu is displayed in all workspaces.

To create a pull-down menu

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, right-click Menus. Click New ➤ Menu.

A new menu (named Menu1) is placed at the bottom of the Menus tree.

3 Do one of the following:

- Enter a new name over the Menu1 text.
- Right-click Menu1. Click Rename. Enter a new menu name.

4 Select the new menu in the tree view, and update the Properties pane as follows:

- In the Description box, enter a description for the menu.
- In the Aliases box, an alias is automatically assigned to the new menu, based on the number of menus already loaded. For example, if the alias assignment is POP12, eleven menus are already loaded. View or edit the alias.
- (Optional) If the name change is based upon a DIESEL expression, the DIESEL expression should be included in the Name box.

5 In the Command List pane, drag the command to a location just below the menu in the Customizations In <*file name*> pane.

6 When you finish adding commands, click OK.

To add a command to pull-down menu

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the menu to which you want to add a command.

3 In the Command List pane, drag the command you want to add to a location just below the menu in the Customizations In <*file name*> pane.

4 When you finish adding commands, Click OK.

For information about creating a command, see "Create, Edit, and Reuse Commands" on page 52.

Create a Shortcut Menu

Shortcut menus are displayed at your cursor location when you right-click a pointing device. The shortcut menu and the options it provides depend on the pointer location and other conditions, such as whether an object is selected or a command is in progress. You can also use scripts to display shortcut menus.

Context-sensitive shortcut menus display menu options when you right-click that are relative to the current command or the selected object.

Shortcut Menu Aliases

Shortcut menus are referenced by their aliases and are used in specific situations. In the Customize User Interface dialog box, the alias names must follow the proper naming conventions. For example, the shortcut menu named "Default Menu" displays the following information in the Aliases section of the Properties pane:

POP501, CMDEFAULT

The Object Snap shortcut menu must have an alias of `POP0`. Aliases for context-sensitive shortcut menus must be numbered between `POP500` and `POP999`. The following aliases are reserved for use by the program:

Program aliases for shortcut menus

Alias	Description
GRIPS	Defines the Hot Grip shortcut menu. (Right-click the drawing area while a grip on an object is selected.)
CMDEFAULT	Defines the Default mode shortcut menu. (Right-click the drawing area while no command is active and no objects are selected.)
CMEDIT	Defines the Edit mode shortcut menu. (Right-click the drawing area while one or more objects are selected, no grips are selected, and no command is active.)
CMCOMMAND	Defines the Command mode menu. (Right-click the drawing area while a command is active.) In addition to the content of the `CMCOMMAND` menu, the command line options (keywords within the square brackets) are inserted into this menu.
SNAP	Defines the Object Snap menu. (SHIFT+right-click the drawing area.)

The `CMEDIT` and `CMCOMMAND` shortcut menus can be made context-sensitive. In addition to the content of the `CMEDIT` menu, the appropriate object menu (if it exists) is inserted into this menu when one or more of a specific object type are selected. Object menus use either of the following naming conventions:

`OBJECT_` *`objectname`*

`OBJECTS_` *`objectname`*

If a single object is selected, the `OBJECT_` *`objectname`* menu is used, and if more than one of the same object is selected, the `OBJECTS_` *`objectname`* menu is used. If no `OBJECT_` *`objectname`* is available, the program uses the `OBJECTS_` *`objectname`* menu (if it exists).

The object name is the drawing interchange format (DXF™) name of the object in all cases except the inserted object. To differentiate between a block insertion and an xref, use the names BLOCKREF and XREF.

For example, to support an object-specific shortcut command for one or more selected block references, you would add the following properties on the Customize tab, Properties pane of the Customize User Interface dialog box:

Properties for the Block Objects shortcut menu	
Properties pane item	**Entry**
Name	Block Objects Menu
Description	Shortcut menu for block objects
Alias	POP512,OBJECTS_BLOCKREF
Element ID	ID_BLOCK [Explode]

Like the CMEDIT menu, the CMCOMMAND menu can contain context-sensitive information. Any menu named COMMAND_ *commandname* is appended to the CMCOMMAND menu. The text of *commandname* can be any valid AutoCAD command, including custom-defined or third-party commands.

In many cases, you can enter a hyphen before a command to suppress the dialog box and display prompts on the command line. To create a context-sensitive menu that displays prompts on the command line (such as -INSERT), you need to name the menu COMMAND_-INSERT.

To create a shortcut menu

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, right-click Shortcut Menus. Click New ➤ Shortcut Menu.

   The new shortcut menu (named "ShortcutMenu1") is placed at the bottom of the Menus tree.
3. Do one of the following:
   - Enter a new name over the ShortcutMenu1 text.
   - Right-click ShortcutMenu1. Click Rename. Enter a new shortcut menu name.
4. In the Properties pane, do the following:
   - In the Description box, enter a description for the shortcut menu.

- In the Aliases box, enter additional aliases for this menu. An alias is automatically assigned, and defaults to the next available POP number, based on the number of shortcut menus already loaded in the program.

5. In the Command List pane, drag the command you want to add to the location just below the shortcut menu in the Customizations In <*file name*> pane.
6. Continue adding commands until the new shortcut menu is complete.

   Click OK.

Create Submenus

You create submenus much the same way that you create a menu.

The following table describes the non-alphanumeric characters that can be used in the Customize User Interface dialog box. Non-alphanumeric characters not listed are reserved for future use as special menu characters.

Special characters for submenus

Character	Description	Example
`$(`	Enables the pull-down or shortcut command label to evaluate a DIESEL string macro if `$(` are the first characters.	
`~`	Makes a command unavailable.	
`!.`	Marks a command with a check mark.	
`&`	Placed directly before a character, specifies the character as the menu access key in a pull-down or shortcut menu label.	`S&ample` displays Sample (with the letter *a* underlined).
`\t`	Pushes all label text entered after these characters to the right side of the menu.	Help\tF1 displays Help on the left side of the pull-down menu and F1 on the right side.

To create a submenu

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to Menus. Select the menu to which you want to add a submenu.

3. Right-click the menu. Click New ➤ Sub-menu.

   The new submenu name "Menu1" is placed at the bottom of the Menu you selected in the Submenu tree.

4. Do one of the following:

   - Enter a new name over the Menu1 text.
   - Right-click Menu1. Click Rename. Enter a new submenu name.

5. In the Properties pane, do the following:

   - In the Description box, enter a description for the submenu.
   - In the Aliases box, enter an alias for this submenu.

6. In the Command List pane, drag the command you want to add to a location just below the name of the submenu in the Customizations In <*file name*> pane.

7. Continue adding commands until the new submenu is complete.

   Click OK.

Reference Pull-Down or Shortcut Menus

Using a method similar to that used to activate submenus, you can activate or deactivate another pull-down or shortcut menu. This is called *referencing* a menu.

The two methods for referencing a pull-down or shortcut menu are *relative* and *absolute*. Relative referencing uses the customization group and element ID. Absolute referencing uses the absolute position of a menu item in the menu hierarchy. Relative referencing is recommended because of its dynamic nature, which allows it to function regardless of the current state of a menu.

Relative Referencing of Pull-Down and Shortcut Commands

To reference a pull-down or shortcut menu item based on its customization group and element ID, use the AutoLISP **`menucmd`** function. The following syntax references a menu item based on its element ID.

```
(menucmd "Gmenugroup.element_id=value")
```

The following example uses the relative referencing syntax to disable the menu item `ID_Line` that is stored in the `sample` customization group. It works regardless of the menu item's location in the menu.

```
(menucmd "Gsample.ID_Line=~")
```

If you know what is contained in the main CUI file, you can create a partial CUI file with an additional menu item that references the main file. In this manner, partial CUI files and specific base files can work together.

Absolute Referencing of Pull-Down and Shortcut Menu Items

In addition to referencing a menu item, you can activate or deactivate a menu item with the `$P n = xxx` syntax. This is the format:

`$Pn.i=xxx`
The `$` loads a menu section; `P n` specifies the active menu section (0 through 16 are valid values); `i` specifies the menu item number; and `xxx` (if present), specifies a string of grayed out or marked characters.

Using the `$P n = xxx` syntax, the following example adds a check mark to item 1 in the `POP7` section.

```
$P7.1=!.
```
The following example uses the AutoLISP **menucmd** function to reference a pull-down or shortcut menu item. Because customization files are dynamic (through the loading of partial CUI files), the following syntax won't work in all cases.

```
(menucmd "P1.2=~")
```
This syntax relies on the location of the menu item and does not work if a new item is inserted before `POP1` by the CUILOAD command.

Menu item numbering is consecutive regardless of the hierarchy of the menu.

To make it easy for an item to address itself regardless of its location in the menu hierarchy, use these forms:

`$P@.@= xxx`

References the current or most recently chosen command.

`$P@. n = xxx`

References item `n` in the current or most recently chosen menu.

AutoLISP Access to Label Status

The AutoLISP **menucmd** function accepts `$P n = xxx` command strings but without the leading `$`. For these functions, the `xxx` portion of the command string can have special values.

`P n . i =?`

Returns the current disabled and marked status for the specified item as a string (for example, `~` for a disabled item, `!.` for an item with a check mark, and `""` for an item that is neither grayed out nor marked).

P *n* . *i* =#?

Returns the same type of string as described for P *n* .i=?, but with the P *n* . *i* = prefix. This is useful in conjunction with the @ forms, because the actual menu and item number are returned.

For example, if the fifth item in the POP6 section is disabled, the following **menucmd** code returns the following string values.

```
(menucmd "P6.5=?") returns "~"
(menucmd "P6.5=#?") returns "P6.5=~"
```

See "Use of AutoLISP in macros" in the *AutoLISP Developer's Guide*.

Swap and Insert Pull-Down Menus

Using the Customize User Interface dialog box, you can use workspaces to control the swapping of pull-down menus. However, you can also swap a pull-down menu programatically (for example, when a user loads an application that requires an additional menu). Menu swapping activates one menu directly from another menu.

Swap Pull-Down Menus

Because the program has cascading pull-down menus, there is little need to swap menus. Also, swapping menus can detract from the consistency of the user interface. However, using $ commands, you can swap pull-down menus and submenus. An alternative to menu swapping involves relative (or global) referencing. Using this method, you can insert the new menu in front of a known menu and then remove the known menu.

For menu-swapping purposes, the pull-down menu areas are named P1 through P16. You can change the title that appears in the menu bar by replacing that line of the menu with a $Pn= command. You can use the special command $Pn=* from within any command to force the menu currently assigned to area POP *n* to pull down for greater flexibility in movement of the pointing device.

The following macro example replaces a menu at position P3 with the menu named BudsMenu in the customization group named MYMENU.

```
$P3=MyMenu.BudsMenu
```

The same thing can be done with the AutoLISP **menucmd** function as follows:

```
(menucmd "P3=MyMenu.BudsMenu")
```

You can use the $P *n* =* special command from within any macro to force the menu currently assigned to area POP *n* to be displayed.

NOTE The swapping of pull-down menus does not conform to the Microsoft® user interface guidelines and is not guaranteed to be available in future releases of the program.

Insert and Remove Pull-Down Menus

Menu swapping is activating one menu directly from another menu. Menu swapping is supported for the following interface elements:

- Buttons
- Pull-down menus
- Mouse buttons
- Image tile menus
- Tablet menus

The syntax for the swapping of partial menus is as follows:

`$section=menugroup.menuname`

section

B1-4, A1-4, P0-16, T1-4

menugroup

Customization group name in the desired CUI file

menuname

Main label or alias.

You can use the AutoLISP **menucmd** function to insert or remove a pull-down menu. The syntax is similar to that used to swap pull-down menus except that the left side of the assignment is the pull-down menu before which the new menu will be inserted. The right side of the assignment is a plus sign (+) followed by the name of the menu group, a period, and the menu's alias, as shown in the following syntax:

```
(menucmd "Gmenugroup1.menuname1=+menugroup2.menuname2")
```

You can also insert a menu with the `P n =` syntax. The following macro inserts a menu after the `P5` menu. (You can also use the **menucmd** function with this format.)

```
(menucmd "P5=+mymenu.new3")
```

If you use this method to insert a menu, remember that you cannot rely on its being inserted at the `P6` menu location as you might expect. There are two reasons that this may not be the case:

- If the current menu bar has only three menus, inserting a menu after menu `P5` results in the new menu's location being `P4`.
- If the user inserts or removes a customization file with the command or when another application inserts or removes customization files, menu numbering can get out of sync.

This is the syntax for removing a menu:

`(menucmd "G*menugroup.menuname*=-")`

The following example removes the menu `NEW3` that is a member of the `MyMenu` group.

```
(menucmd "Gmymenu.new3=-")
```

As you might expect, the preceding format is preferable to the `P n =` format because it removes only the specified menu. The following example removes the menu at the `P4` location (whatever it is).

```
$P4=-
```

NOTE Use the `P n` syntax as part of the syntax for a **`menucmd`** statement only. Use the $Pn syntax for macro–specific statements.

Control Toolbars Across Partial CUI Files

To control toolbars across partial CUI files, use the following syntax at the Toolbar Name prompt of the - "TOOLBAR Command Line" on the command line.

`menugroup.subsection-name`

This syntax accesses the toolbar identified by `menugroup.menuname` and allows you to use the full spectrum of -TOOLBAR command options on that toolbar.

If the menu group is left out of any of these commands and functions, the program defaults to the main CUI file

You should be aware of the following:

- Image tile menus cannot be swapped from external customization files.
- You can swap customization elements of the same type only; that is, one shortcut menu for another, one toolbar for another, and so on. Trying to swap between types may result in unpredictable behavior.

Add Shortcut Keys and Temporary Override Keys

You can assign shortcut keys (sometimes called accelerator keys) to commands you use frequently, and temporary override keys to execute a command or change a setting when a key is pressed.

Shortcut keys are keys and key combinations that start commands. For example, you can press CTRL + O to open a file and CTRL + S to save a file, which is the same result as choosing Open and Save from the File menu.

Temporary override keys are keys that temporarily turn on or turn off one of the drawing aids that are set in the Drafting Settings dialog box (for example, Ortho mode, object snaps, or Polar mode).

Shortcut keys can be associated with any command in the command list. You can create new shortcut keys or modify existing shortcut keys.

The following table lists the default actions for shortcut keys.

Shortcut key assignments	
Shortcut key	**Shortcut**
CTRL+0	Toggles Clean Screen
CTRL+1	Toggles Properties palette
CTRL+2	Toggles DesignCenter
CTRL+3	Toggles the Tool palettes window
CTRL+4	Toggles Sheet Set Manager
CTRL+5	Toggles Info Palette
CTRL+6	Toggles dbConnect Manager
CTRL+7	Toggles Markup Set Manager
CTRL+8	Toggles the QuickCalc calculator
CTRL+9	Toggles the command window
CTRL+A	Selects objects in drawing
CTRL+B	Toggles Snap
CTRL+C	Copies objects to Clipboard
CTRL+D	Toggles coordinate display
CTRL+E	Cycles through isometric planes
CTRL+F	Toggles running object snaps
CTRL+G	Toggles Grid
CTRL+H	Toggles PICKSTYLE
CTRL+J	Repeats last command
CTRL+L	Toggles Ortho mode
CTRL+M	Repeats last command

Shortcut key assignments	
Shortcut key	**Shortcut**
CTRL+N	Creates a new drawing
CTRL+O	Opens existing drawing
CTRL+P	Prints current drawing
CTRL+R	Cycles layout viewports
CTRL+S	Saves current drawing
CTRL+T	Toggles Tablet mode
CTRL+V	Pastes data from Clipboard
CTRL+X	Cuts objects to Clipboard
CTRL+Y	Cancels the preceding Undo action
CTRL+Z	Reverses last action
CTRL+[	Cancels current command
CTRL+\	Cancels current command
F1	Displays Help
F2	Toggles text window on/off
F3	Toggles OSNAP
F4	Toggles TABMODE
F5	Toggles ISOPLANE
F6	Toggles COORDS
F7	Toggles GRIDMODE
F8	Toggles ORTHOMODE
F9	Toggles SNAPMODE
F10	Toggles Polar Tracking
F11	Toggles Object Snap Tracking
F12	Toggles Dynamic Input

In the Customize User Interface dialog box, you can view, print, or copy a list of shortcut keys, temporary override keys, or both. The shortcut keys and

temporary override keys in the list are those keys used by the CUI files that are loaded in the program.

See also:

"Customize the User Interface" on page 35

To create a shortcut key

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, click the plus sign (+) next to Keyboard Shortcuts to expand it.
3. Click the plus sign (+) next to Shortcut Keys to expand it.
4. In the Command List pane, drag the command you want to add to a location in the Shortcut Keys node of the Customizations In <*file name*> pane.

   In the Properties pane, the properties for the new shortcut key you created are displayed.
5. In the Key(s) box, click the shortcut key. Click the [...] button to open the Shortcut Keys dialog box.
6. In the Shortcut Keys dialog box, in the Press New Shortcut Key box, hold a modifier key (CTRL or SHIFT) and press a letter, number, or function key. Valid modifier keys include the following:

   - Function (F*n*) keys containing no modifiers
   - CTRL+letter, CTRL+number, CTRL+function key
   - CTRL+ALT+letter, CTRL+ALT+number, CTRL+ALT+function key
   - SHIFT+CTRL+letter, SHIFT+CTRL+number, SHIFT+CTRL+function key
   - SHIFT+CTRL+ALT+letter, SHIFT+CTRL+ALT+number, SHIFT+CTRL+ALT+function key

   Under the Press New Shortcut Key box, Currently Assigned To displays any current assignments for the shortcut key.
7. If you do not want to replace the current assignment, use a different shortcut key. Otherwise, click Assign.

   NOTE More than one command can share the same shortcut, but only the last command assigned will be active.

8 Click OK to assign the shortcut key and close the Shortcut Keys dialog box.

9 In the Customize User Interface dialog box, click OK.

To modify a shortcut key

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, click the plus sign (+) next to Keyboard Shortcuts to expand it.

3 Click the plus sign (+) next to Shortcut Keys to expand it.

4 Click a shortcut key.

In the Properties pane, the properties for the shortcut key you selected are displayed.

5 In the Key(s) box, click the shortcut key. Click the [...] button to open the Shortcut Keys dialog box.

6 In the Shortcut Keys dialog box, in the Press New Shortcut Key box, hold down a modifier key (CTRL or SHIFT) and press a letter, number, or function key. Valid modifier keys include the following:

- Function (F*n*) keys containing no modifiers
- CTRL+letter, CTRL+number, CTRL+function key
- CTRL+ALT+letter, CTRL+ALT+number, CTRL+ALT+function key
- SHIFT+CTRL+letter, SHIFT+CTRL+number, SHIFT+CTRL+function key
- SHIFT+CTRL+ALT+letter, SHIFT+CTRL+ALT+number, SHIFT+CTRL+ALT+function key

Below the Press New Shortcut Key box, Currently Assigned To displays any current assignments for the shortcut key.

7 If you do not want to replace the current assignment, use a different shortcut key. Otherwise, click Assign.

NOTE More than one command can share the same shortcut, but only the last command assigned will be active.

8 Click OK to assign the shortcut key and close the Shortcut Keys dialog box.

9 In the Customize User Interface dialog box, click OK.

To create a temporary override key

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, click the plus sign (+) next to Keyboard Shortcuts to expand it.
3. In the Customizations In <*file name*> pane, right-click Temporary Override Keys. Click New ➤ Temporary Override.

   A new temporary override (named TemporaryOverride1) is placed at the bottom of the Temporary Override Keys tree.
4. Do one of the following:

   - Enter a new name over the TemporaryOverride1 text.
   - Right-click TemporaryOverride1. Click Rename. Enter a new temporary override name.
5. Select the new temporary override in the tree view, and update the Properties pane:

   - In the Description box, enter a description for the temporary override.
   - In the Macro 1 (Key Down) box, enter a macro to be executed when the temporary override key is pressed. When no value is assigned, the default macro is c^c^.
   - In the Key(s) box, click the [...] button to open the Shortcut Keys dialog box. In the Shortcut Keys dialog box, in the Press New Shortcut Key box, press a key. Valid modifier keys include function (F*n* keys) with no modifiers, SHIFT+letter, SHIFT+number, or SHIFT+function key.
   - Under the Press New Shortcut Key box, Currently Assigned To displays any current assignments for the key. If a key you select is not already assigned, click Assign, and then click OK.
   - In the Macro 2 (Key Up) box, enter a macro to be executed when the temporary override key is released. When no value is defined, key up restores the application to its previous state (before the temporary override was executed).

     NOTE For information about creating a macro, see "Create Macros" on page 99.

To modify a temporary override key

1. Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, Customize tab, click the plus sign (+) next to Keyboard Shortcuts to expand it.

3 In the Customizations In <*file name*> pane, click the temporary override key you want to modify.

4 Update the Properties pane as necessary:

- In the Description box, enter a description for the temporary override.
- In the Macro 1 (Key Down) box, enter a macro to be executed when the temporary override key is pressed. When no value is assigned, the default macro is c^c^.
- In the Key(s) box, click the [...] button to open the Shortcut Keys dialog box. In the Shortcut Keys dialog box, in the Press New Shortcut Key box, press a key. Under the Press New Shortcut Key box, Currently Assigned To displays any current assignments for the key. If a key you select is not already assigned, click Assign, and then click OK.
- In the Macro 2 (Key Up) box, enter a macro to be executed when the temporary override key is released. When no value is defined, key up restores the application to its previous state (before the temporary override was executed).

NOTE For information about creating a macro, see "Create Macros" on page 99.

To print a list of shortcut keys or temporary override keys

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customizations In <*file name*> pane, click the plus sign (+) next to Keyboard Shortcuts to expand it.

3 In the Shortcuts pane, filter the type and status of keyboard shortcuts to print.

- In the Type list, select the type of keyboard shortcuts displayed in the list. Choices include All Keys, Accelerator Keys, or Temporary Override Keys.
- In the Status list, select the status of keyboard shortcuts displayed in the list. Choices include All, Active, Inactive, and Unassigned.

4 In the Shortcuts pane, click Print.

Customize Workspaces

You can customize workspaces to create a drawing environment that displays only those toolbars, menus, and dockable windows that you select. Customization options for workspaces include creating a workspace using the Customize User Interface dialog box, changing the properties of a workspace, and displaying a toolbar in all workspaces.

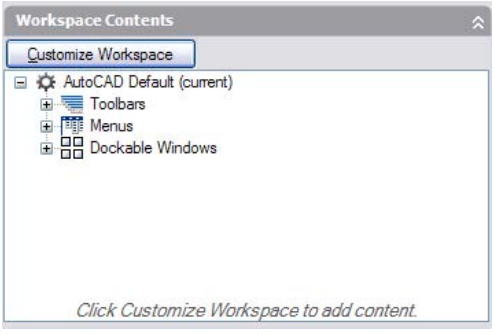
Create or Modify a Workspace Using the Customize User Interface Dialog Box

The easiest way for users to create or modify a workspace is to set up the toolbars and dockable windows that best suit a drawing task, and then save that setup as a workspace in the program. That workspace can be accessed any time the user needs to draw within that workspace environment.

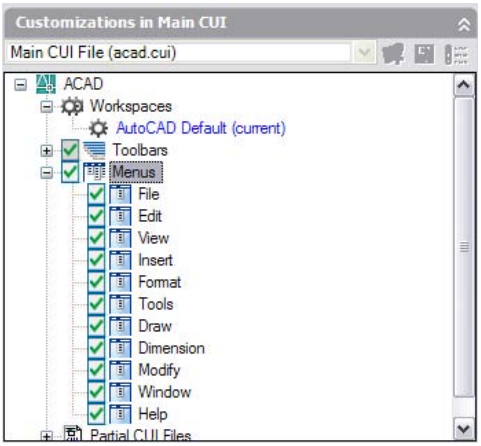
You can also set up a workspace using the Customize User Interface dialog box. In this dialog box, you can create or modify workspaces with precise properties and elements (toolbars, menus, and dockable windows) that you want your users to access for certain tasks. You can specify the CUI file containing this workspace as an enterprise CUI file, so that you can share the workspace with your users.

For more information about creating an enterprise CUI file see "Create an Enterprise CUI File" on page 48.

Following is an example of the Customize User Interface dialog box, Workspace Contents pane. You click Customize Workspaces to create or modify the selected workspace.



Following is an example of the Customizations In pane after you click Customize Workspaces in the Workspace Contents pane. Check boxes are displayed next to each element that you can add to a workspace. You click a check box to add the element to the workspace.



Change the Properties of a Workspace

In the Customize User Interface dialog box, you can define workspace properties, such as the workspace name, description, whether it is displayed on the Model or Layout tab, and so on.

Change the Properties of a Dockable Window

Dockable windows are windows that you can dock or undock in a drawing area. You can define the size, location, or appearance of a dockable window by changing their properties in the Workspace Contents pane of the Customize User Interface dialog box. Dockable windows include:

- Command window
- Properties palette
- DesignCenter
- Tool Palettes window
- Info palette
- dbConnect Manager
- Markup Set Manager
- QuickCalc calculator

Import a Workspace to a Main CUI File

Workspaces in partial CUI files are ignored by the main CUI file, even when the partial CUI file is loaded in the main CUI file. You can import a workspace

to the main CUI file using the Transfer tab of the Customize User Interface dialog box.

Display a Toolbar in All Workspaces

When you create a toolbar, you can add it to all workspaces by choosing Show (the default) in the On By Default box in the Properties pane. The Show setting indicates that the toolbar is displayed in all workspaces that have already been created.

To create a workspace using the Customize User Interface dialog box

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, right-click the Workspaces tree node, and select New ➤ Workspace.

   The new workspace is placed at the bottom of the Workspaces tree with a default name "Workspace1."
3. Do one of the following:

   - Enter a new name over the Workspace1 text.
   - Right-click Workspace1. Click Rename. Then, enter a new workspace name.
4. In the Workspace Contents pane, click Customize Workspace.
5. In the Customizations In <*file name*> pane, click the plus sign (+) next to the Toolbars tree node, Menus tree node, or Partial CUI files tree node to expand it.

   NOTE The menu, toolbar, and partial CUI file nodes now display check boxes so that you can easily add elements to the workspace.
6. Click the check box next to each menu, toolbar, or partial CUI file that you want to add to the workspace.

   In the Workspace Contents pane, the selected elements are added to the workspace.
7. In the Workspace Contents pane, click Done.

To change the properties of a workspace

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, click the Customize tab.

3. On the Customize tab, in the Customizations In <*file name*> pane, click the workspace whose properties you want to change.
4. In the Properties pane, do any of the following:
   - In the Name box, enter a new name for the workspace.
   - In the Description box, enter a description.
   - In the Display Model/Layout Tab box, select an option (On, Off, Do Not Change).
   - In the Display Screen Menus box, select an option (On, Off, Do Not Change).
   - In the Display Scroll Bars box, select an option (On, Off, Do Not Change).
5. When you are finished changing the workspace properties, click OK.

To import a workspace to a main CUI file

1. Click Tools menu ➤ Customize ➤ Import Customizations.

   The Transfer tab is displayed, with the main CUI file displayed in the Customizations In pane (right side).
2. On the Transfer tab, in the Customizations In <*file name*> pane (left side), click the Open Customization File button.
3. In the Open dialog box, locate and select the partial customization file that contains the workspace you want to add.
4. In the left pane, drag the workspace from the partial CUI file to the Workspace node in the main CUI file.
5. Click OK.

To change the properties of a dockable window

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the workspace that contains the dockable window you want to modify.
3. In the Workspace Contents pane, click Customize Workspace.
4. In the Workspace Contents pane, click the plus sign (+) next to Dockable Windows to expand the list.

5 Click the dockable window that you want to modify.

6 In the Properties pane, do any of the following:

- In the Show box, select an option (No, Yes, or Do Not Change).
- In the Orientation box, select an option (Floating, Top, Bottom, Left, or Right).
- In the Allow Docking box, select an option (No, Yes, or Do Not Change).
- In the Auto Hide box, select an option (On, Off, or Do Not Change).
- In the Use Transparency box, select an option (No, Yes, or Do Not Change).
- In the Transparency Amount box, enter a number (if applicable).
- (Tool Palettes only) In the Default Group box, select a Tool Palette group.
- In the Height box, enter a number. A value of 0 is equivalent to Do Not Change.
- In the Width box, enter a number. A value of 0 is equivalent to Do Not Change.

7 In the Workspace Contents pane, click Done.

8 When you finish, click OK.

To duplicate a workspace

1 Click Tools menu ➤ Customize ➤ Interface.

2 In the Customize User Interface dialog box, in the Customizations In <*file name*> pane (left side), click the Open Customization File button.

3 In the Open dialog box, locate and select the main customization file that contains the workspace you want to duplicate.

4 Right-click the workspace. Click Duplicate Workspace.

The duplicated workspace is displayed at the bottom of the list of workspaces (and is named Copy of <*workspace name*>).

5 Right-click the duplicated workspace. Click Rename.

6 Enter the name of the new workspace.

7 Modify the workspace as necessary.

8 Click OK.

To set a workspace to current

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, in the Customizations In <*file name*> pane (left side), click the Open Customization File button.
3. In the Open dialog box, locate and select the main customization file that contains the workspace you want to set current.
4. Right-click the workspace. Click Set Workspace Current.
5. Click Apply.

Customize Mouse Buttons

You can change the standard behavior of pointing devices in the program. Mouse buttons define how a Windows system pointing device functions. You can customize the behavior of a mouse or other pointing device in the Customize User Interface dialog box. If a pointing device has more than two buttons, you can change the behavior of the second and third buttons. The first button on any pointing device cannot be changed in the Customize User Interface dialog box.

By using the SHIFT and CTRL keys, you can create a number of combinations to suit your needs. Your pointing device can recognize as many commands as it has assignable buttons. The Mouse Buttons section of the tree node is organized by keyboard combination such as Click, SHIFT+Click, CTRL+Click, and CTRL+SHIFT+Click. The tablet buttons are numbered sequentially. Drag a command to assign the command to a mouse button. Create additional buttons by dragging commands to a Click node.

Accept Coordinate Entry in Button Menus

When you click one of the buttons on a multibutton pointing device, the program reads not only the button number but also the coordinate of the crosshairs at the time you click. By carefully constructing macros, you can choose to either ignore the coordinate or use it with the command activated by the button.

As described in "Pause for User Input in Macros" on page 103, you can include a backslash (\) in a command to pause for user input. For the Mouse and Digitize Buttons menus, the coordinate of the crosshairs is supplied as user input when the button is clicked. This occurs only for the first backslash in the command; if the item contains no backslashes, the crosshairs coordinate is not used. Consider the following commands:

```
line
line \
```
The first button starts the command and displays the Specify First Point prompt in the normal fashion. The second button also starts the LINE command, but the program uses the current crosshairs location at the Specify First Point prompt and displays the Specify Next Point prompt.

To add a button to a tablet

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, click the Customize tab.
3. In the Customizations In < *file name* > pane, click the plus sign (+) next to Mouse Buttons to expand the list.
4. Right-click a mouse button section. Click New Button.

   A new mouse button (named Button*n*) is placed at the bottom of the selected list.
5. In the Command List pane, drag the command you want to add to the mouse button in the Customizations In <*file name*> pane.
6. When you finish, click OK.

Create Status Line Help Messages

Status line Help messages are the simple, descriptive messages that are displayed on the status line (at the bottom of the drawing area) when the pointing device hovers over a menu option or toolbar button. You can change or add descriptions for menus and buttons by updating the Description property for the related command.

To create a status line Help message

1. Click Tools menu ➤ Customize ➤ Interface
2. In the Customize User Interface dialog box, click the Customize tab.
3. In the Command List pane, click the command to which you want to add a Help message.

   The Properties pane is displayed on the right side of the dialog box.
4. In the Properties pane, Description box, enter the descriptive text for the selected command.
5. Click the Apply button.

The next time you use the command, the descriptive text you added is displayed in the status line when you hover your mouse button over a toolbar button or menu item.

Load an AutoLISP File

AutoLISP (LSP or MNL) files contain scripts that add customization actions and behaviors to the interface. You can load AutoLISP files into a CUI file using the Customize tab of the Customize User Interface dialog box.

For more information about using AutoLISP, see "AutoLISP and Visual LISP" on page 141.

NOTE MNL files with the same name and location as your main, enterprise, or partial CUI files are loaded automatically. These files cannot be removed.

To load an AutoLISP file in the Customize User Interface dialog box

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In *<file name>* pane, right-click LISP Files. Click Load LISP.
3. In the Load LISP dialog box, located and select the AutoLISP file you want to load. Files that you can load include MNL and LSP files.
4. Click Open.

Customize Legacy Interface Elements

The term "legacy" refers to those user interface elements that are not commonly used with the current version of the program, but are still supported because some users prefer them to alternative user interface elements that are now provided.

Legacy interface elements include:

- Tablet menus
- Tablet buttons
- Screen menus
- Image tile menus

Create Tablet Menus

You can configure up to four areas of your digitizing tablet as menu areas for command input.

The nodes in the Customize User Interface dialog box are labeled Tablet Menu 1 through Tablet Menu 4 and define the macros associated with tablet selections.

The tablet menu areas that you define with the Cfg option of the command are divided into equal-sized menu selection boxes, which are determined by the number of columns and rows you specify in each area. These tablet menu selection boxes correspond directly with the lines that follow the Tablet section labels from left to right and top to bottom (whether or not they contain text).

For example, if you configure a menu area for five columns and four rows, the command on the line immediately following the Row label corresponds to the left-most selection box in the top row. The program can recognize up to 32,766 commands in each tablet section, which should be more than enough for any tablet menu.

You can add your own macros to the Macros cell in the Properties pane. The command labels in this area correspond to the 225 boxes at the top of your tablet template (rows A through I and columns 1 through 25). You can add a macro using standard command syntax.

See also:

"Create Macros" on page 99

To define rows and columns in a tablet menu

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to Legacy to expand the list.
3. Click the plus sign (+) next to Tablet Menus to expand the list.
4. Click the plus sign (+) next to a tablet menu to expand the list.
5. Click the row that you want to define.
6. In the Command List pane, locate the command you want to add.
7. Drag the command to a column.
8. When you finish adding commands, click OK.

To clear a tablet menu assignment

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to Legacy to expand the list.
3. Click the plus sign (+) next to Tablet Menus to expand the list.
4. Click the plus sign (+) next to a tablet menu to expand the list.
5. Right-click the row or column that you want to clear. Click Clear Assignment.
6. When you finish adding commands, click OK.

Customize Tablet Buttons

To customize tablet buttons, you follow the same procedures as for customizing mouse buttons.

For more information about customizing tablet buttons, see "Customize Mouse Buttons" on page 90.

Create Screen Menus

Screen menus provide a legacy interface for displaying menus in a dockable window. You create and edit screen menus in the Customize User Interface dialog box.

By default, the screen menu is disabled. You turn on the screen menu display in the Options dialog box, Display tab. In addition, the MENUCTL system variable controls whether the screen menu is updated as you enter commands at the command line.

NOTE Future releases of the product will not support screen menus.

In the Customize User Interface dialog box, each screen menu consists of several menu lines, which define the screen submenus. You assign a submenu to a screen menu by dragging it to the screen menu in the Customize In pane. You assign a command to a menu by dragging it from the Command List pane to the numbered line in the menu. Unassigned lines are left empty in the menu.

Edit Screen Menu Properties

You can modify screen menu properties, as shown in the following table:

Properties for screen menus		
Properties pane item	**Description**	**Example**
Name	Sets the name of the menu.	AutoCAD
Description	Describes the user interface element.	Usually empty for screen menus
Start line	Sets the start line of the screen menu submenu.	1
Aliases	Specifies the alias for the screen menu. "Collection" is displayed if multiple definitions are assigned to this alias. Click the ellipses button [...] to open the Aliases dialog box.	(Collection)

For the AutoCAD screen menu, which is the root menu, the aliases in the Aliases box are Screen (which represents the beginning of the screen menu) and S (which represents the submenu section label). Line assignments for other menus define the order of the options on the menu. For example, the File menu on Line 3 in the tree view of the AutoCAD screen menu is in the third position on the AutoCAD screen menu.

The submenu names in the tree view correspond to the name of the first submenu item. For example, the New submenu contains commands such as OPEN, QSAVE, and SAVEAS—in addition to NEW. The Aliases box for these submenus defines which menu contains them and the Start Line box specifies their position on that menu. The New submenu is displayed in position 3 on the File screen menu. Therefore, in the Properties pane, its start line is 3. When you double-click Aliases to display the Aliases dialog box, you can see that its menu assignment is 01_FILE.

You can designate which menu items are always displayed by controlling the start line settings. For example, since the New submenu is set to start on line 3, the menu items on lines 1 and 2 of the AutoCAD screen menu (the AutoCAD and **** menu options) continue to display when the New submenu is displayed.

Similarly, you can set a menu to mask or show menu options on other menus by using blank lines. For example, there are only 22 lines (including blank lines) defined for the New submenu. Therefore, the Assist and Last submenus on lines 25 and 26 of the AutoCAD screen menu continue to be displayed when the New submenu is selected. An option on line 22, however, would be hidden.

To display the screen menu

1. Click Tools menu ➤ Options.
2. In the Options dialog box, Display tab, under Window Elements, select Display Screen Menu.
3. Click OK.

To set screen menus to reflect the current command

1. At the Command prompt, enter **menuctl**.
2. Do one of the following:
   - Enter **1** to set screen menus to reflect the current command.
   - Enter **0** to set screen menus to ignore the current command.

To add commands to the screen menu

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to Legacy to expand the list.
3. In the Legacy list, click the plus sign (+) next to a screen menu to expand the list.
4. In the Command List pane, locate the command you want to add. Drag the command to the screen menu. An arrow is displayed next to the cursor when the command can be dropped.
5. When you finish, click OK.

To create a submenu on a screen menu

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to Legacy to expand the list.
3. In the Legacy list, right-click Screen Menu. Click New Screen Menu.

   A new screen menu (named ScreenMenu1) is placed at the bottom of the Screen Menus tree.

4 Do one of the following:

- Enter a new name over the ScreenMenu1 text.
- Right-click ScreenMenu1. Click Rename. Then, enter a new menu name.

5 Select the new screen menu in the tree view, and update the Properties pane as follows:

- In the Description box, enter a description for the screen menu.
- In the Start Line box, enter the line number for the first option in the menu.
- In the Aliases box, enter an alias.

6 In the Command List pane, drag the command to a location just below the screen menu in the Customizations In <*file name*> pane.

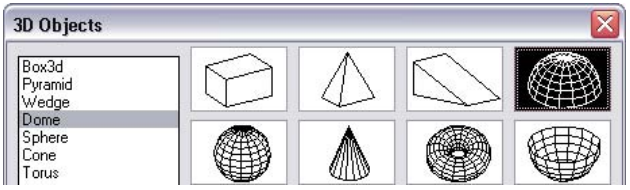
7 When you finish, click OK.

Create Image Tile Menus

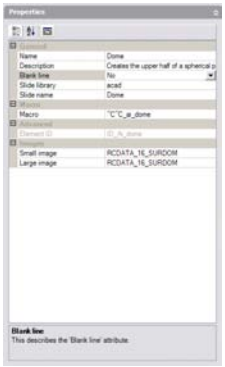
The purpose of an image tile menu is to provide an image that can be selected instead of text. You can create, edit, or add image tiles and image tile slides.

An image tile dialog box displays images in groups of 20, along with a scrolling list box on the left that displays the associated slide file names or related text. If an image tile dialog box contains more than 20 slides, the additional slides are added to a new page. Next and Previous buttons are activated so that you can browse the pages of images.

Following is an example of the 3D Objects image tile dialog box with the Dome image tile slide selected.



You define an image tile menu in the Customize User Interface dialog box. Following is an example of the Properties pane display for the Dome image tile.



You can use any slide generated by AutoCAD as an image. Keep the following suggestions in mind as you prepare slides for an image tile menu.

- **Keep the image simple.** When an image tile menu is displayed, you must wait for all images to be drawn before making a selection. If you show numerous complex symbols, use simple, recognizable images rather than full renditions.
- **Fill the box.** When making a slide for an image, be sure to fill the screen with the image before starting . If the image is very wide and short, or long and thin, the image tile menu will look best if you use PAN to center the image on the screen before making the slide.

  Images are displayed with an aspect ratio of 3:2 (3 units wide by 2 units high). If your drawing area has a different aspect ratio, it can be difficult to produce image slides that are centered in the image tile menu. If you work within a layout viewport that has an aspect ratio of 3:2, you can position the image and be assured that it will look the same when it is displayed in the image tile menu.
- **Remember the purpose of the images.** Do not use images to encode abstract concepts into symbols. Image tiles are useful primarily for selecting a graphic symbol.

To create an image tile menu and assign an image tile slide

1. Click Tools menu ➤ Customize ➤ Interface.
2. In the Customize User Interface dialog box, Customize tab, in the Customizations In <*file name*> pane, click the plus sign (+) next to Legacy to expand the list.
3. In the Legacy list, right-click Image Tile Menu. Click New Image Tile Menu.

A new image tile menu (named ImageTileMenu1) is placed at the bottom of the Image Tile Menus tree.

4. Do one of the following:
   - Enter a new name over the ImageTileMenu1 text.
   - Right-click ImageTileMenu1. Click Rename. Then, enter a new image tile name.
5. In the Command List pane, drag a command to the new image tile menu in the Customizations In <*file name*> pane.
6. In the Properties pane, enter properties for the new image tile slide.
7. When you finish, click OK.

To create an image tile slide

1. In AutoCAD, draw a symbol or block.
2. At the Command prompt, enter **mslide**.
3. Click File menu ➤ ZOOM ➤ CENTER.
4. At the Command prompt, enter **mslide**.
5. In the Create a Slide File dialog box, specify the file name.
6. Save the file, and add it to the slide library file. You can associate this image slide to a new image tile.

Create Macros

A macro defines the action that results when an interface element is selected. A macro accomplishes a drawing task that would otherwise take a series of actions by a user. A macro can contain commands, special characters, DIESEL (Direct Interpretively Evaluated String Expression Language) or AutoLISP programming code.

NOTE As AutoCAD is revised and enhanced, the sequence of prompts for various commands (and sometimes command names) might change. Therefore, your custom macros might require minor changes when you upgrade to a new release of AutoCAD.

You add macros to interface elements by using the Customize User Interface dialog box. Select an existing command or create a new command in the Command List pane. Enter macros in the Macros section of the Properties pane. There are no length limitations for macros. However, you do need to

know how specific characters are used in macros and be aware of other considerations or limitations.

Macro Basics

A macro in a user interface element can be as simple as a command (such as **circle**) and some special characters (such as ^C^C).

For example, the macro `^C^C_circle \1`, draws a circle with a radius of 1 unit. The components that define this macro are explained in the table below:

Components in CIRCLE macro

Component	Component type	Result
^C^C	Special control character	Cancels any running commands
_	Special control character	Automatically translates the command that follows into other languages
CIRCLE	Command	Starts the CIRCLE command
\	Special control character	Creates a pause for the user to specify the center point
1	Special control character	Responds to the prompt for the circle's radius (1)

For a list of special control characters that you can use in macros, see "Use Special Control Characters in Macros" on page 101.

Cancel Running Commands

Make sure that you have no AutoCAD commands in progress before you execute a macro. To automatically cancel a command before executing a macro, enter `^C^C` at the beginning of the macro (which is the same as pressing ESC twice). Although a single `^C` cancels most commands, `^C^C` is required to return to the Command prompt from a dimensioning command; therefore, it is good practice to use `^C^C`.

Verify Macro Characters

Every character in a macro is significant, even a blank space.

When you place a space at the end of the macro, AutoCAD processes the macro as though you had entered a command (**circle**, for example) and then pressed the SPACEBAR to complete the command.

Terminate Macros

Some macros require special terminators. Some commands (, for example) require you to press ENTER rather than SPACEBAR to terminate the command. Some commands require more than one space (or ENTER) to complete, but some text editors cannot create a line with trailing blanks.

Two special conventions resolve these problems.

- A semicolon (;) in a macro automatically issues ENTER on the command line.
- If a line ends with a control character, a backslash (\), a plus sign (+), or a semicolon (;), AutoCAD does not add a blank space after it.

An item that ends with a backslash (\) pauses a macro for user input.

Compare the following macros:

```
ucs
ucs ;
```

The first example enters **ucs** on the command line and presses SPACEBAR. The following prompt is displayed.

Origin/ZAxis/3point/Entity/View/X/Y/Z/Prev/Restore/Save/Del/?/ <World>:

The second example enters **ucs**, presses SPACEBAR, and presses ENTER, which accepts the default value (World).

Suppress Echoes and Prompts in Macros

Characters in a macro appear in the command window as though you had typed the characters on the keyboard. They are also displayed in the user interface element. This display duplication is called “echoing”. You can suppress the “echoed” displays with the system variable. If echoes and prompts from item input are turned off, a `^P` in the item turns them off.

Create Long Macros

You can create a macro of any length, without requiring any special characters at the end of a line. The Properties pane in the Customize User Interface dialog box accepts a macro of any length.

Use Special Control Characters in Macros

You can use special characters, including control characters, in macros. In a macro, the caret (^) is equivalent to pressing the CTRL key on the keyboard. You can combine the caret with another character to construct macros that do such things as turn the grid on and off (`^G`) or cancel a command (`^C`).

The macro for the Address command below uses the backslash (\) to pause for user input and the semicolon (;) for ENTER.

```
text \.4 0 DRAFT Inc;;;Main St.;;;City, State;
```

The macro starts the command, pauses for the user to specify a start point, and then enters the address on three lines. In the triple semicolon (;;;), the first semicolon ends the text string, the second repeats TEXT, and the third accepts the default placement below the previous line.

Macros use the special characters listed in the following table.

Special characters used in macros

Character	Description
;	Issues ENTER
^M	Issues ENTER
^I	Issues TAB
[blank space]	Enters a space; a blank space between command sequences in a command is equivalent to pressing the SPACEBAR
\	Pauses for user input (cannot be used with accelerators)
_	Translates AutoCAD commands and options that follow
=*	Displays the current top-level pull-down, shortcut, or image menu
*^C^C	Repeats a command until another command is chosen
$	Introduces a conditional DIESEL macro expression ($M=)
^B	Turns Snap on or off (equivalent to CTRL+B)
^C	Cancels a command (equivalent to ESC)
^D	Turns Coords on or off (equivalent to CTRL+D)
^E	Sets the next isometric plane (equivalent to CTRL+E)
^G	Turns Grid on or off (equivalent to CTRL+G)
^H	Issues BACKSPACE
^O	Turns Ortho on or off
^P	Turns MENUECHO on or off
^Q	Echoes all prompts, status listings, and input to the printer (equivalent to CTRL+Q)
^T	Turns tablet on or off (equivalent to CTRL+T)

Special characters used in macros	
Character	**Description**
^V	Changes the current viewport
^Z	Null character that suppresses the automatic addition of SPACEBAR at the end of a command

Pause for User Input in Macros

To accept input from the keyboard or pointing device in the middle of a command, place a backslash (\) in the macro at the point where you want input.

```
circle \1
```

In the circle example, \1 pauses for the user to specify the center point and then reads a radius of 1. Note that there is no space after the backslash.

```
-layer off \;
```

In this example, the macro starts on the command line (-layer), enters the Off option (off), and then pauses for the user to enter a layer name (\). the macro then turns that layer off and exits the LAYER command (;).

NOTE LAYER normally prompts for another operation and exits only if you press SPACEBAR or ENTER. In the macro, the semicolon (;) is the equivalent of pressing ENTER.

A macro typically resumes after one user input, such as a single point location. Therefore, you cannot construct a macro that accepts a variable number of inputs (as in object selection) and then continues. However, an exception is made for : a backslash (\) suspends the SELECT command until object selection has been completed. Consider the following example:

```
select \change previous ;properties color red ;
```

In this macro, SELECT creates a selection set of one or more objects (select \). The macro then starts CHANGE (change), references the selection set using the Previous option (previous;), and changes the color of all selected objects to red (properties color red ;).

NOTE The backslash character (\) causes a macro to pause for user input. You cannot use a backslash for any other purpose in a macro. When you need to specify a file directory path, use a forward slash (/) as the path delimiter: for example, /direct/file.

The following circumstances delay resumption of a macro after a pause:

- If input of a point location is expected, object snap modes may be used before the point is specified.
- If X/Y/Z point filters are used, the command remains suspended until the entire point has been accumulated.
- For SELECT only, the macro does not resume until object selection has been completed.
- If the user responds with a transparent command, the suspended macro remains suspended until the transparent command is completed and the originally requested input is received.
- If the user responds by choosing another command (to supply options or to execute a transparent command), the original macro is suspended, and the newly selected item is processed to completion. Then, the suspended macro is resumed.

NOTE When command input comes from a command, the settings of the and system variables are assumed to be 1 and 0, respectively. This preserves compatibility with previous releases of AutoCAD and makes customization easier because you are not required to check the settings of these variables.

Provide International Support in Macros

To develop menus that can be used with a non-English-language version of AutoCAD, precede each command or option with the underscore character (_). The underscore character allows the standard commands and options to be translated automatically.

Repeat Commands in Macros

You can use a leading asterisk (*) to repeat a command in a macro until you choose another command.

Once you have selected a command, you might want to use it several times before moving on to another command. In a macro, you can repeat a command until you choose another command. You cannot use this feature to choose options.

If a macro begins with `*^C^C`, the command is repeated until you terminate by pressing ESC on the keyboard or by selecting another command.

NOTE Do not use ^C (Cancel) within a macro that begins with the string *^C^C; this cancels the repetition.

The macros in the following examples repeat the commands:

```
*^C^Cmove Single
*^C^Ccopy Single
*^C^Cerase Single
*^C^Cstretch Single Crossing
*^C^Crotate Single
*^C^Cscale Single
```

Each macro in the example starts a command and then prompts you to select an object. Any other prompts necessary to complete the command are displayed, and then the command ends and starts again.

NOTE Command repetition cannot be used in macros for image tile menus.

Use Single Object Selection Mode in Macros

Single Object Selection mode cancels the normal repetition of the Select Objects prompt in editing commands. After you select one object and respond to any other prompts, the command ends.

Consider the macro in the following example:

```
*^C^Cerase single
```

This macro terminates the current command and starts in Single Object Selection mode. After you choose this command, you either select a single object to be erased or click a blank area in the drawing and specify window selection. Any objects selected in this way are erased, and the command is repeated (due to the leading asterisk) so that you can erase additional objects. Press ESC to exit this mode.

Use Macros to Swap User Interface Elements

You can replace the contents of active menus, mouse buttons, tablet buttons, tablet menus or screen menus. The swapped content can be that of another user interface element of the same type in the main CUI file, or it can come from a partial CUI file.

You cannot swap interface elements that are of different types (menus and mouse buttons, for example). However, within a given type, you can swap any user interface element for any other element.

NOTE Swapping can lead to some strange behavior for tablet menus, because they typically have a different number of macros.

Use the following syntax in a macro to swap elements:

```
$section=menugroup.menuname
```

The following describes each section of the macro syntax for swapping elements:

Macro syntax for swapping elements

$

Loads an interface element

section

Specifies the element type. Valid names are:

`A1-A4` for Aux menus 1 through 4

`B1-B4` for mouse buttons 1 through 4

`P0-P16` for pull-down menus 0 through 16

`I` for the image tile menu

`S` for the screen menu

`T1-T4` for tablet menus 1 through 4

infogroup

Specifies the information group that *menuname* is a member of (not necessary if *menuname* is in the main CUI file).

menuname

Specifies which section or submenu to insert. It is the main label or alias for the section to load

The following commands illustrate submenu referencing:

```
$S=PARTS
$T1=EDITCMDS
```

You can activate the submenu mechanism in the middle of a command without interrupting the command. For example, the following command strings are equivalent:

```
$S=ARCSTUFF ARC
ARC $S=ARCSTUFF
```

Each command starts the command, switches to the `ARCSTUFF` screen submenu, and awaits the entry of arc parameters. A space must follow the submenu reference to separate it from subsequent commands in the command.

A pull-down menu can be present either in the menu bar or on the active shortcut menu but not both.

Use Conditional Expressions in Macros

You can add conditional expressions to a macro by using a command that introduces macro expressions written in DIESEL (Direct Interpretively Evaluated String Expression Language).

The format is:

`$M=`*`expression`*
Introducing the macro with `$M=` tells AutoCAD to evaluate a string as a DIESEL expression, and that *`expression`* is the DIESEL expression. The following example defines a conditional expression in a macro:

```
FILLMODE $M=$(-,1,$(getvar,fillmode))
```
The macro switches the system variable on and off by subtracting the current value of FILLMODE from 1 and returning the resulting value to the FILLMODE system variable. You can use this method to toggle system variables whose valid values are 1 or 0.

Termination of Macros That Contain Conditional Expressions

If you use the DIESEL string language to perform "if-then" tests, conditions might exist where you do not want the normal terminating space or semicolon (resulting in ENTER). If you add `^Z` to the end of the macro, AutoCAD does not automatically add a space (ENTER) to the end of the macro expression.

As with other control characters in commands, the `^Z` used here is a string composed of `^` (a caret) and `Z` and is not equivalent to pressing CTRL+Z.

In the following examples, `^Z` is used as a macro terminator.

```
^C^C$M=$(if,$(=,$(getvar,tilemode),0),$S=mview _mspace )^Z
^C^C$M=$(if,$(=,$(getvar,tilemode),0),$S=mview _pspace )^Z
```
If these macros did not end with `^Z`, AutoCAD would automatically add a space (ENTER), repeating the last command entered.

See also:

"Use Special Control Characters in Macros" on page 101
"DIESEL" on page 109

Use AutoLISP in Macros

Creating commands that use AutoLISP is a more advanced way to use the AutoCAD customization feature.

You can use AutoLISP variables and expressions to create macros that perform complex tasks. To use AutoLISP efficiently in macros, place AutoLISP code in a separate MNL file. AutoCAD loads the MNL file when it loads a CUI file with the same name and in the same location.

You can specify additional AutoLISP files to load in the Customize User Interface dialog box. Creating commands that use AutoLISP is a more advanced way to use the AutoCAD customization feature. Carefully study the following examples and the information in the *AutoLISP Reference* and the *AutoLISP Developer's Guide* (on the Help menu, click Additional Resources ➤ Developer Help). Experimentation and practice will help you use this feature effectively.

Call a Macro

To programmatically execute a pull-down menu macro, use the following syntax:

```
(menucmd "Gmenugroup.element_ID=|")
```

The previous syntax works only if the menu macro is part of a menu that is on the AutoCAD menu bar and is available for use. For more information about this syntax, see the *AutoLISP Reference*.

Preset Values

An application that uses block insertion presets could provide commands like these: [Set WINWID][Set WALLTHK][Insert Window]

```
^C^C^P(setq WWID (getreal"Enter window width: ")) ^P
^C^C^P(setq WTHK (getreal"Enter wall thickness: ")) ^P
^C^C_INSERT window XScale !WWID YScale !WTHK
```

This code inserts the block named "window," scaling its *X* axis to the current window width and its *Y* axis to the current wall thickness. In this example, the actual values come from the user-defined AutoLISP symbols WINWID and WALLTHK. The rotation is up to the user to decide so that the window can be rotated in the wall.

Resize Grips

With the following commands, grip size adjustment can be done on the fly:

```
^P(setvar"gripsize"(1+(getvar"gripsize")))(redraw)(princ)
^P(setvar"gripsize"(1-(getvar"gripsize")))(redraw)(princ)
```

To add validity checking to these commands, values less than 0 and greater than 255 cannot be used for the GRIPSIZE system variable.

Prompt for User Input

The following item prompts for two points and draws a rectangular polyline with the specified points as its corners.

```
^P(setq a (getpoint "Enter first corner: "));\+
(setq b (getpoint "Enter opposite corner: "));\+
pline !a (list (car a)(cadr b)) !b (list (car b)(cadr a)) c;^P
```

DIESEL

5

You can use DIESEL (Direct Interpretively Evaluated String Expression Language) to alter the AutoCAD® status line through the MODEMACRO system variable. You can also use DIESEL in menu items as a macro language instead of AutoLISP®. DIESEL expressions accept strings and generate string results.

Because DIESEL expressions handle strings exclusively, the USERS1-5 system variables are useful for passing information from an AutoLISP routine to a DIESEL expression. DIESEL expressions are evaluated by AutoLISP routines through the use of the AutoLISP **menucmd** function.

In this chapter

Customize the Status Line

You can use the MODEMACRO system variable to display information on the status line.

Overview of the MODEMACRO System Variable

The status line can provide the user with important information without interrupting the work flow. The MODEMACRO system variable controls the user-defined area on the status line. The calculated value of the MODEMACRO system variable is displayed in a left-aligned panel in the status bar at the bottom of the AutoCAD® window. This variable is set to the null string when you start AutoCAD. Its value is not saved in the drawing, the configuration file, or anywhere else.

The number of characters displayed on the status line is limited only by the size of the AutoCAD window (and your monitor). The default panels move to the right as the content of the MODEMACRO panel grows. It is possible to push the default panels completely off the screen (if you want to).

You can use the MODEMACRO system variable to display, in the status line, most data known to AutoCAD. With its calculation, decision, and editing facilities you can compose the status line to your precise specifications.

MODEMACRO is a user-string variable. It can be set to any string value. The maximum string value is 4095 characters. You can set MODEMACRO with or by entering **modemacro** at the Command prompt. If you modify the MODEMACRO setting, you can experiment with various status line formats; however, the maximum number of characters you can enter in this manner is 255.

If you set MODEMACRO to the null string by entering a period (.), AutoCAD displays the standard status line.

Set MODEMACRO Values

You can use text strings and DIESEL to display messages in the user-defined section of the status line.

The value of MODEMACRO determines what is displayed in the mode status line. The simplest (and least useful) MODEMACRO consists of constant text. For example, to display a company name in the status line, you enter the following:

Command: **modemacro**
New value for MODEMACRO, or . for none <"">: **Greg's Bank and Grill**

This MODEMACRO value always displays the same text; the status line does not reflect changes to the AutoCAD internal state. It doesn't change until you change MODEMACRO.

To make the status line reflect the AutoCAD current state, enter macro expressions using the DIESEL language in the following format:

`$(somefun, arg1, arg2, ...)`
In the macro expression, ***somefun*** is the name of the DIESEL function (similar to an AutoLISP function name) and *arg1*, *arg2*, and so on, are arguments to the function, interpreted according to the function's definition. Unlike AutoLISP, DIESEL macro expressions have only one data type: strings. Macros that operate on numbers express the numbers as strings and convert back and forth as required.

For descriptions of the DIESEL functions, see "Catalog of DIESEL Functions" on page 116.

Now define a more interesting status line (for example, one that shows the current text style name):

Command: **modemacro**
New value for MODEMACRO, or . for none <"">: **Style: $(getvar, textstyle)**

- **Style:** is a text string to be displayed on the status line.
- **$(getvar,textstyle)** is a DIESEL function (**getvar**) and argument that retrieves the current value of the system variable.

NOTE The examples in this topic may show the MODEMACRO string as more than one line of text. You enter it as one long string at the prompt.

You can retrieve any system variable by entering **$(getvar, *varname*)**. The current setting of the system variable replaces the macro expression on the status line. Then, when you switch text styles, for example, MODEMACRO is reevaluated. If it changes, the new text style name is displayed on the status line.

Expressions can be nested, and they can be as complex as you want. The example that follows displays the current snap value and angle (in degrees) in the status line. It uses nested expressions to convert the snap angle from radians to degrees and truncates the value to an integer.

Command: **modemacro**
New value for MODEMACRO, or . for none <"">: **Snap: $(getvar, snapunit) $(fix,$(*,$(getvar,snapang),$(/,180,3.14159)))**
You can also display the values in the current linear and angular units modes.

Command: **modemacro**
New value for MODEMACRO, or . for none <"">: **Snap: $(rtos,$(index,0,**

$(getvar,snapunit))),$(rtos,$(index,1,$(getvar,snapunit))) $(angtos, $(getvar,snapang))
DIESEL copies its input directly to the output until it comes to the dollar sign character ($) or a quoted string. You can use quoted strings to suppress evaluation of character sequences that would otherwise be interpreted as DIESEL functions. You can include quotation marks in quoted strings by using two adjacent quotation marks. In the following example, the current layer is set to LAYOUT, and MODEMACRO is set to the string.

Command: **modemacro**
New value for MODEMACRO, or . for none <"">: "**$(getvar,clayer)=**
""**$(getvar,clayer)**""""
The status line displays the following:

$(getvar,clayer)="LAYOUT"

Set MODEMACRO with AutoLISP

You can save the code samples shown here as ASCII format text files and load them with the AutoLISP **load** function.

The following AutoLISP command defines a MODEMACRO string that provides similar information to that in the built-in status line. Because AutoLISP cannot continue strings from line to line, you use the AutoLISP **strcat** function to assemble the complete MODEMACRO string from shorter component strings.

```
(defun C:ACADMODE ( )
(setvar "modemacro"
(strcat
"Layer $(substr,$(getvar,clayer),1,8)"
"$(if,$(getvar,orthomode), Ortho)"
"$(if,$(getvar,snapmode), Snap)"
"$(if,$(getvar,tabmode), Tablet)"
"$(if,$(=,$(getvar,tilemode),0),"
"$(if,$(=,$(getvar,cvport),1), P)"
")"
)
)
)
```

Save this AutoLISP routine in a file called *acadmode.lsp*. When you load the routine and execute it, it displays information on the status line. This is not the most useful application of this feature; it is provided only as an example.

The following sample *acad.lsp* file uses the **S::STARTUP** function to set the MODEMACRO variable to a string defined by the AutoLISP file *mode1.lsp*.

```
;;; Sample acad.lsp file that uses S::STARTUP to load the
;;; file MODE1.LSP which defines a MODEMACRO string
(defun S::STARTUP ( )
(load "mode1")
(princ)
```

```
)
;;; Additional AutoLISP files can also be defined or
;;; loaded here
```

When the AutoLISP file *(mode1.lsp)* is loaded, it uses the MODEMACRO system variable to define a status line that displays *L:* followed by the first eight characters of the layer name, the drawing name and a portion of the path, and the first letter of each name of the currently active modes. The position of the drawing name remains constant, regardless of the length of the layer name.

```
;;; MODE1.LSP
;;;
(setvar "modemacro"
(strcat
"L:$(substr,$(getvar,clayer),1,30)"
"$(substr, ,1,$(-,30,$(strlen,$(getvar,clayer)))) "
;; ^^^^^^^^ Note the 8 spaces here
"<.."
"$(if,$(eq,$(getvar,dwgname),UNNAMED),UNNAMED,"
"$(substr,$(getvar,dwgname),"
"$(if,$(>,$(strlen,$(getvar,dwgprefix)),29),"
"$(-,$(strlen,$(getvar,dwgprefix)),29),1"
"),"
"$(strlen,$(getvar,dwgname))"
")"
")"
">"
"$(if,$(getvar,orthomode), O, )"
"$(if,$(getvar,snapmode), S, )"
"$(if,$(getvar,tabmode), T, )"
"$(if,$(and,"
"$(=,$(getvar,tilemode),0),$(=,$(getvar,cvport),1)),P)"
)
)
```

Indenting code improves the readability of AutoLISP files and DIESEL strings.

DIESEL Expressions in Macros

These expressions can return string values (text strings) in response to standard AutoCAD commands, AutoLISP and ObjectARX® routines, and other macros. They can also return string values to the menu itself, thereby altering the appearance or content of a menu label.

This string provides a way to toggle between paper space and model space if TILEMODE is set to 0. This expression is evaluated transparently. If the special character ^P (which toggles MENUECHO on and off) is omitted, the expression displays only the issued command.

A DIESEL expression that you use in a menu item must follow the `$section=submenu` format where the section name is `M` and the submenu is

the DIESEL expression you want. Frequently, you can implement a macro more easily with AutoLISP.

The following examples show two menu items that produce the same result; one uses DIESEL, and the other uses AutoLISP.

This menu item uses the DIESEL expression:

```
^C^C^P$M=$(if,$(=,$(getvar,cvport),1),mspace,pspace)
```

This menu item uses the AutoLISP expression:

```
^C^C^P(if (= (getvar "cvport") 1)(command "mspace")+
(command "pspace"))(princ) ^P
```

Both menu items provide a way to switch between paper space and model space (if TILEMODE is set to 0), but the DIESEL expression is shorter and is evaluated transparently, not requiring the call to the AutoLISP **princ** function. If the special character `^P` (which switches MENUECHO on and off) is omitted in both cases, the DIESEL expression displays only the issued command, whereas the AutoLISP expression displays the entire line of code.

Because the value returned by a DIESEL expression is a text string, it can be used in response to an AutoLISP **get *xxx*** function call. This functionality enables menu items to evaluate current drawing conditions and to return a value to an AutoLISP routine.

The next example is based on these assumptions:

- The AutoLISP routine is loaded into memory.
- The CUI excerpt is included in the current customization file.
- The symbols to insert are one unit high by one unit wide.
- The DIMSCALE variable is set to the drawing's scale factor (that is, a drawing to be plotted at a scale of 1" = 10' would have a scale factor of 120, or a 1/4" = 1' scale drawing would have a scale factor of 48).

If you load and execute the sample AutoLISP routine, AutoCAD inserts the symbol at the size and location you have specified. When plotted, the symbols are the specified size (if the drawing is plotted at the same scale as that specified by DIMSCALE).

The following is a sample AutoLISP routine.

```
(defun C:SYMIN ( )
(setq sym
(getstring
"\nEnter symbol name: ") ; Prompts for a symbol name
)
(menucmd "s=symsize") ; Switches the screen menu
; to the symsize submenu
(setq
siz (getreal
```

```
"\nSelect symbol size: ") ; Prompts for a symbol size
p1 (getpoint
"\nInsertion point: ") ; Prompts for insertion point
)
(command "insert" ; Issues the INSERT command
sym ; using the desired symbol
p1 siz siz 0) ; insertion point, and size
(menucmd "s=") ; Switches to the previous
; screen menu
(princ) ; Exits quietly
)
```

NOTE An AutoLISP routine that you use regularly should include error checking to verify the validity of user input.

The DIESEL expressions in the following example multiply the current value of DIMSCALE by the specified value, and return an appropriate scale factor.

This cannot be done with similar AutoLISP code; a value returned by an AutoLISP expression cannot typically be used as a response to a **`get xxx`** function call (such as, the **`getreal`** function in the preceding sample).

```
$M=$(*,$(getvar,dimscale),0.375)
$M=$(*,$(getvar,dimscale),0.5)
$M=$(*,$(getvar,dimscale),0.625)
```

DIESEL expressions can also return string values to pull-down menu item labels, so that you can make menus unavailable or otherwise alter the way they are displayed. To use a DIESEL expression in a pull-down menu label, make sure that the first character is the `$` character.

In the next example, the current layer is set to BASE and the following DIESEL expression is used as the label.

```
$(eval,"Current layer: " $(getvar,clayer))
```

The result is that the appropriate pull-down menu is displayed and updated whenever the current layer changes.

Current Layer: BASE

You can also use this method to interactively change the text displayed in a pull-down menu. You use an AutoLISP routine that sets the USERS1-5 system variables to the selected text, which can be retrieved by a DIESEL macro in a menu label.

NOTE The width of pull-down and shortcut menus is determined when the customization file is being loaded. Menu labels generated or changed by DIESEL expressions after a menu is loaded are truncated to fit within the existing menu width.

If you anticipate that a DIESEL-generated menu label will be too wide, you can use the following example to ensure that the menu width will

accommodate your labels. This example displays the first 10 characters of the current value of the USERS3 () system variable.

```
$(eval,"Current value: " $(getvar,users3))+
$(if, $(eq,$(getvar,users3),""), 10 spaces )^C^Cusers3
```

You cannot use trailing spaces in a menu label to increase the menu width, because trailing spaces are ignored while the menu is being loaded. Any spaces you use to increase the width of a menu label must be within a DIESEL expression.

The next example uses the same DIESEL expression as the label and a portion of the menu item. It provides a practical way to enter the current day and date into a drawing.

```
$(edtime,$(getvar,date),DDD", "D MON YYYY)^C^Ctext +
\\\ $M=$(edtime,$(getvar,date),DDD", "D MON YYYY);
```

Also, you can use a DIESEL macro to mark pull-down menu labels or make them unavailable. The following pull-down menu label displays an unavailable ERASE while a command is active. The text is displayed normally when a command is not active.

```
$(if,$(getvar,cmdactive),~)ERASEerase
```

You can use a similar approach to place a mark beside a pull-down menu item or to interactively change the character used for the mark.

You can use the CLIPBOARD system variable to enable or disable Clipboard commands on your menu. For example, the following DIESEL expression disables the Paste Special menu item when the Clipboard is empty:

```
$(If,$(getvar, clipboard),,~)/Paste &Special...
^C^C_pastespec
```

Catalog of DIESEL Functions

Status retrieval, computation, and display are performed by DIESEL functions. All functions have a limit of 10 parameters, including the function name itself. If this limit is exceeded, you get a DIESEL error message.

+ (addition)

Returns the sum of the numbers *val1*, *val2*, ..., *val9*.

$(+, *val1* **[,** *val2* **, …,** *val9* **])**

If the current thickness is set to 5, the following DIESEL string returns 15.

```
$(+, $(getvar,thickness),10)
```

- (subtraction)

Returns the result of subtracting the numbers *val2* through *val9* from *val1*.

$(-, *val1* **[,** *val2* **, …,** *val9* **])**

* (multiplication)

Returns the result of multiplying the numbers *val1*, *val2*, ..., *val9*.

$(*, *val1* **[,** *val2* **, …,** *val9* **])**

/ (division)

Returns the result of dividing the number *val1* by *val2*, ..., *val9*.

$(/, *val1* **[,** *val2* **, …,** *val9* **])**

= (equal to)

If the numbers *val1* and *val2* are equal, the string returns 1; otherwise, it returns 0.

$(=, *val1* **,** *val2* **)**

< (less than)

If the number *val1* is less than *val2*, the string returns 1; otherwise, it returns 0.

$(< , *val1*, *val2*)
The following expression gets the current value of ; if the value is less than the value stored in the system variable USERR1, it returns 1. If the value 10.0 is stored in USERR1 and the current setting of HPANG is 15.5, the following string returns 0.

```
$(<, $(getvar,hpang),$(getvar,userr1))
```

> (greater than)

If the number *val1* is greater than *val2*, the string returns 1; otherwise, it returns 0.

$(>, *val1* **,** *val2* **)**

!= (not equal to)

If the numbers *val1* and *val2* are not equal, the string returns 1; otherwise, it returns 0.

$(!=, *val1* **,** *val2* **)**

<= (less than or equal to)

If the number *val1* is less than or equal to *val2*, the string returns 1; otherwise, it returns 0.

$(<=, ***val1*** **,** ***val2*** **)**

>= (greater than or equal to)

If the number *val1* is greater than or equal to *val2*, the string returns 1; otherwise, it returns 0.

$(>=, *val1* **,** *val2* **)**

and

Returns the bitwise logical AND of the integers *val1* through *val9*.

$(and, *val1* **[,** *val2* **,…,** *val9* **])**

angtos

Returns the angular value in the format and precision specified.

$(angtos, *value* [, *mode*, *precision*])
Edits the given *value* as an angle in the format specified by the *mode* and *precision* as defined for the analogous AutoLISPfunction. (The values for *mode* are shown in the following table.) If *mode* and *precision* are omitted, it uses the current values chosen by the command.

Angular units values	
Mode value	**String format**
0	Degrees
1	Degrees/minutes/seconds
2	Grads

Angular units values	
Mode value	**String format**
3	Radians
4	Surveyor's units

edtime

Returns a formatted date and time based on a given picture.

$(edtime, *time* **,** *picture* **)**
Edits the AutoCAD Julian date given by *time* (obtained, for example, from **$(getvar,date)** according to the given *picture*). The *picture* consists of format phrases replaced by specific representations of the date and time. Characters not interpretable as format phrases are copied literally into the result of **$(edtime)**. Format phrases are defined as shown in the following table. Assume that the date and time are Saturday, 5 September 1998 4:53:17.506.

edtime format phrases			
Format	**Output**	**Format**	**Output**
D	5	H	4
DD	05	HH	04
DDD	Sat	MM	53
DDDD	Saturday	SS	17
M	9	MSEC	506
MO	09	AM/PM	AM
MON	Sep	am/pm	am
MONTH	September	A/P	A
YY	98	a/p	a
YYYY	1998		

Enter the entire AM/PM phrase as shown in the preceding table; if AM is used alone, the A will be read literally and the M will return the current month.

If any AM/PM phrases appear in the picture, the H and HH phrases edit the time according to the 12-hour civil clock (12:00–12:59 1:00–11:59) instead of the 24-hour clock (00:00–23:59).

The following example uses the date and time from the preceding table. Notice that the comma must be enclosed in quotation marks because it is read as an argument separator.

```
$(edtime, $(getvar,date),DDD"," DD MON YYYY - H:MMam/pm)
```

It returns the following:

Sat, 5 Sep 1998 – 4:53am

If *time* is 0, the time and date at the moment that the outermost macro was executed is used. This avoids lengthy and time-consuming multiple calls on `$(getvar,date)` and guarantees that strings composed with multiple `$(edtime)` macros all use the same time.

eq

If the strings *val1* and *val2* are identical, the string returns 1; otherwise, it returns 0.

$(eq, *val1* **,** *val2* **)**

The following expression gets the name of the current layer; if the name matches the string value stored in the USERS1 () system variable, it returns 1. Assume the string `"PART12"` is stored in USERS1 and the current layer is the same.

```
$(eq, $(getvar,users1),$(getvar,clayer))Returns 1
```

eval

Passes the string *str* to the DIESEL evaluator and returns the result of evaluating it.

$(eval, *str* **)**

fix

Truncates the real number *value* to an integer by discarding any fractional part.

$(fix, *value* **)**

getenv

Returns the value of the environment variable *varname*.

$(getenv, *varname* **)**

If no variable with that name is defined, it returns the null string.

getvar

Returns the value of the system variable with the given *varname*.

$(getvar, *varname* **)**

if

Conditionally evaluates expressions.

$(if, *expr* **,** *dotrue* **[,** *dofalse* **])**
If *expr* is nonzero, it evaluates and returns *dotrue*. Otherwise, it evaluates and returns *dofalse*. Note that the branch not chosen by *expr* is not evaluated.

index

Returns the specified member of a comma-delimited string.

$(index, *which* **,** *string* **)**
Assumes that the *string* argument contains one or more values delimited by the macro argument separator character, the comma. The *which* argument selects one of these values to be extracted, with the first item numbered 0. This function is most frequently used to extract *X*, *Y*, or *Z* coordinate values from point coordinates returned by `$(getvar)`.

Applications can use this function to retrieve values stored as comma-delimited strings from the system variables.

nth

Evaluates and returns the argument selected by *which*.

$(nth, *which* **,** *arg0* **[,** *arg1* **,…,** *arg7* **])**
If *which* is 0, `nth` returns *arg0*, and so on. Note the difference between **$(nth)** and **$(index)**; **$(nth)** returns one of a series of arguments to the function, while **$(index)** extracts a value from a comma-delimited string passed as a single argument. Arguments not selected by *which* are not evaluated.

or

Returns the bitwise logical OR of the integers *val1* through *val9*.

$(or, *val1* **[,** *val2* **,…,** *val9* **])**

rtos

Returns the real value in the format and precision specified.

$(rtos, *value* **[**, *mode* , *precision* **])**
Edits the given *value* as a real number in the format specified by the *mode* and *precision* as defined by the analogous AutoLISP function. If *mode* and *precision* are omitted, it uses the current values selected with the command.

Edits the given *value* as a real number in the format specified by *mode* and *precision*. If *mode* and *precision* are omitted, it uses the current values selected with the command.

strlen

Returns the length of *string* in characters.

$(strlen, *string* **)**

substr

Returns the substring of *string*, starting at character *start* and extending for *length* characters.

$(substr, *string* **,** *start* **[,** *length* **])**
Characters in the string are numbered from 1. If *length* is omitted, it returns the entire remaining length of the string.

upper

Returns the *string* converted to uppercase according to the rules of the current locale.

$(upper, *string* **)**

xor

Returns the bitwise logical XOR of the integers *val1* through *val9*.

$(xor, *val1* **[,** *val2* **,…,** *val9* **])**

DIESEL Error Messages

Generally, if you make a mistake in a DIESEL expression, what went wrong will be obvious. Depending on the nature of the error, DIESEL embeds an error indication in the output stream.

DIESEL error messages

Error message	Description
$?	Syntax error (usually a missing right parenthesis or a runaway string)
$*(func,??)*	Incorrect arguments to *func*
$*(func)??*	Unknown function *func*
$(++)	Output string too long—evaluation truncated

Slides and Command Scripts

6

Slides are snapshots of drawing files that can be used for giving presentations, for creating image tile menus, and for viewing another drawing while you work.

A script reads and executes commands from a text file. You can run a script when you start AutoCAD®, or you can run a script from within AutoCAD using the SCRIPT command. A script provides an easy way to create continuously running displays for product demonstrations and trade shows.

In this chapter

Create Slides

Slides are snapshots of drawing files. You can use slides for giving presentations, creating custom image tile menus, and viewing an image of another drawing while you work.

Overview of Slides

A slide is a snapshot of a drawing. Although it contains a picture of the drawing at a given instant, it is not a drawing file. You cannot import a slide file into the current drawing, nor can you edit or print a slide. You can only view it.

You can use slide files in the following ways:

- For making presentations within AutoCAD®
- For viewing a snapshot of a drawing while working on a different drawing
- For creating menus of image tiles within a dialog box

You create a slide by saving the current view in slide format. A slide created in model space shows only the current viewport. A slide created in paper space shows all visible viewports and their contents. Slides show only what was visible. They do not show objects on layers that were turned off or frozen or objects in viewports that were turned off.

When you view a slide file, it temporarily replaces objects on the screen. You can draw on top of it, but when you change the view (by redrawing, panning, or zooming), the slide file disappears, and AutoCAD redisplays only what you drew and any preexisting objects.

You can display slides one by one or use a script to display slides in sequence. Slides also can be used in custom menus. For example, if you create scripts that insert blocks containing mechanical parts you use frequently, you can design a custom image tile menu that displays a slide of each part. When you click the slide image on the menu, AutoCAD inserts the block into the drawing.

A slide library is a file containing one or more slides. Slide library files are used for creating custom image tile menus and for combining several slide files for convenient file management.

You cannot edit a slide. You must change the original drawing and remake the slide. If you use a low-resolution graphics monitor when creating a slide file and later upgrade to a high-resolution monitor, you can still view the slide. AutoCAD adjusts the image accordingly; however, the slide does not take full advantage of the new monitor until you remake the slide file from the original drawing.

To make a slide

1. Display the view you want to use for the slide.
2. At the Command prompt, enter **mslide**.
3. In the Create Slide File dialog box, enter a name and select a location for the slide.

   AutoCAD offers the current name of the drawing as a default name for the slide and automatically appends the *.sld* file extension.
4. Click Save.

   The current drawing remains on the screen, and the slide file is saved in the folder that you specified.

View Slides

You can view slides individually using VSLIDE. To view a series of slides for a presentation, use a script file.

Be careful about using editing commands while you view a slide, which looks like an ordinary drawing. Editing commands affect the current drawing underneath the slide but not the slide itself.

Some commands may force redrawing, which removes the slide from display.

To view a slide

1. At the Command prompt, enter **vslide**.
2. In the Select Slide File dialog box, select a slide to view and click OK.

   The slide image is displayed in the drawing area.
3. On the View menu, click Redraw.

   The slide image disappears.

Create and View Slide Libraries

A slide library is a file containing one or more slides. Slide library files are used for creating custom image tile menus and for combining several slide files for convenient file management.

You can create slide libraries from slide files using the SLIDELIB utility. After you have set up a slide library, you can view slides by specifying the name of the slide library and the slide.

Do not delete the original slides after creating the slide library. The SLIDELIB utility cannot update a slide library once it is created. If you want to add or delete a slide, update the slide list file and remake the library with SLIDELIB. When you remake the slide library, all the slide files that you intend to include must be available.

To create a slide library

1. Use a Windows ASCII text editor to create a list of slide files to include in the library. The file would look similar to this example:

   ```
   entrance.sld
   hall.sld
   stairs.sld
   study.sld
   balcony.sld
   ```

2. Name and save the file as a text file with a *.txt* file extension.
3. On the Start menu (Windows), click All Programs (or Programs) ➤ Accessories ➤ Command Prompt.
4. Change to the AutoCAD directory.
5. In the AutoCAD Command Prompt window, use the following syntax to create the slide library:

   slidelib *library name*<*list* .txt

   For example, if you named your text file *areas.txt*, you could create a library called *house.slb* by entering **slidelib house<areas.txt**. The SLIDELIB utility appends the file extension *.slb* to the slide library file.

To view a slide in a slide library

1. At the Command prompt, set the FILEDIA system variable to 0.
2. At the Command prompt, enter **vslide**.
3. Enter ***library* (*slidename*)** to specify the slide.

   For example, enter **house** (**balcony**) to open the *balcony* slide, which is stored in the *house* slide library file.
4. On the View menu, click Redraw to remove the slide from the display.

Create Command Scripts

A script is a text file that contains a series of commands. Common uses for scripts are to customize startup and to run slide shows.

Overview of Command Scripts

A script is a text file with one command on each line.

You can invoke a script at startup, or you can run a script during a work session by using the command. A script also provides an easy way to create continuously running displays for product demonstrations and trade shows.

The BACKGROUNDPLOT system variable must be set to 0 before a script can plot multiple jobs.

You create script files outside the program using a text editor (such as Microsoft® Windows® Notepad) or a word processor (such as Microsoft Word) that can save the file in ASCII format. The file extension must be *.scr*.

Each line of the script file contains a command. Each blank space in a script file is significant because SPACEBAR is accepted as a command or data field terminator. You must be very familiar with the sequence of prompts to provide an appropriate sequence of responses in the script file.

NOTE Keep in mind that prompts and command names may change in future releases, so you may need to revise your scripts when you upgrade to a later version of this program. For similar reasons, avoid the use of abbreviations; future command additions might create ambiguities.

A script can execute any command at the Command prompt except a command that displays a dialog box. Command line versions are provided for many dialog box commands.

Script files can contain comments. Any line that begins with a semicolon (;) is considered a comment, and it is ignored while the script file is being processed. The last line of the file must be blank.

All references to long file names that contain embedded spaces must be enclosed in double quotes. For example, to open the drawing *my house.dwg* from a script, you must use the following syntax:

```
open "my house"
```

The following commands are useful in scripts:

'DELAY

Provides a timed pause within a script (in milliseconds)

'GRAPHSCR

Switches from the text window to the drawing area

RESUME

Continues an interrupted script

RSCRIPT

Repeats a script file

'TEXTSCR

Switches to the text window

When command input comes from a script, it is assumed that the settings of the PICKADD and PICKAUTO system variables are 1 and 0, respectively; therefore, you do not have to check the settings of these variables.

A script is treated as a group, a unit of commands, reversible by a single U command. However, each command in the script causes an entry in the undo log, which can slow script processing. If you like, you can use UNDO Control None to turn off the undo feature before running the script, or you can write it at the beginning of the script itself. Remember to turn it back on (UNDO Control All) when the script is finished.

The script that is running stops when another script command is invoked.

To create a script that changes settings in a drawing

This script turns on the grid, sets the global linetype scale to 3.0, and sets layer 0 as the current layer with red as the color.

1. In a text editor, enter **grid on**.
2. On the next line, enter **ltscale 3.0**.
3. On the next line, enter **layer set 0 color red 0**.
4. Add a blank line.
5. Save the file as ASCII text (TXT file), with a file extension of *.scr*.

   The script file may contain comments, as follows:

```
; Turn grid on
grid on
; Set scale for linetypes
ltscale 3.0
; Set current layer and its color
layer set 0 color red 0
; Blank line above to end LAYER command
```

Run Scripts at Startup

A script that runs at startup can open a drawing and change its settings.

Suppose that every time you begin a new drawing, you turn on the grid, set the global linetype scale to 3.0, and set layer 0 as your current layer, with red as the color. You can do this using a drawing template, but you could do it instead with the following script and store it in a text file called *setup.scr*.

```
grid on
ltscale 3.0
layer set 0 color red 0
```

The first line turns on the grid. The second line sets the global scale for linetypes. The third line sets the current layer to layer 0 and sets its default color to red. AutoCAD assumes that in a script you want to use the command line version of LAYER rather than the dialog box version. The result is equivalent to entering **-layer** on the command line. The fourth line is blank, ending LAYER.

NOTE VBA and AutoLISP® scripts that run at startup should check for whether the AutoCAD process is visible or invisible. If the process is invisible, the script should not execute, because the process may be performing background plotting or publishing operations. To check for whether the AutoCAD process is visible or invisible, you can use the Visible property of the Application object in the AutoCAD Object Model.

You could run a script at startup to open a drawing by using the following syntax in the Run dialog box:

```
ACAD drawing_name /b setup
```

All file names that contain embedded spaces must be enclosed in double quotes, for example, `"guest house"`. You can also specify the view that is displayed when the drawing opens by using the /v switch and the view name. The /b switch and the script file must be the last parameter listed.

Including the file extensions *.exe*, *.dwg*, *.dwt*, and *.scr* is optional. If AutoCAD cannot find the script file, AutoCAD reports that it cannot open the file.

To run the same script at startup but create a new drawing using the *MyTemplate.dwt* file as the template, enter the following in the Run dialog box:

```
ACAD /t MyTemplate /b setup
```

This command creates a new drawing and issues a sequence of setup commands from the *setup.scr* file. When the script has finished running, the Command prompt is displayed. If you want to use the default template for the new drawing, you can omit the /t switch and the template file name.

NOTE You can no longer use this method to start a new drawing and give it a name. Name the drawing when you save it.

To run a script at startup

1. On the Start menu (Windows), click Run.
2. In the Run dialog box, enter **acad** ***drawing_name*** **/b** ***script_name*** .

   To start a new file, instead of a drawing file name, enter the /t switch and the name of a template file: **/t** ***template_drawing***.

   To open a drawing file to a particular view, follow the drawing name with the /v switch and the name of the view: **/v** ***view_name***.

   The name of the script file must be the last parameter listed. The file extensions are optional.
3. Click OK.

   AutoCAD opens the drawing and executes the commands in the script file. When the script has been completed, the Command prompt is displayed.

Run Slide Shows from Scripts

Scripts are useful for creating slide shows. Ordinarily, the speed with which you can display slides is limited by the number of times AutoCAD must access the disk to read the slide file. You can, however, preload the next slide from disk into memory while your audience is viewing the current slide and then quickly display the new slide from memory.

To preload a slide, place an asterisk before the file name in VSLIDE. The next VSLIDE command detects that a slide has been preloaded and displays it without asking for a file name.

The disk-access time to load the next slide overlaps with the viewing time for the current slide. You can specify additional delays with the DELAY command. Each delay unit is equal to one millisecond.

To stop a repeating script press ESC. You can resume the script with RESUME.

If the script will run for a long time, it is recommended that you use UNDO Control None to turn off the Undo log file.

To run slide shows from scripts

1. Create the slide library file as described in "To create a slide library " on page 128.
2. Create a script file using an ASCII text editor, as shown in "To create a script that preloads slides" on page 133.
3. On the command line, enter **script**.
4. In the Select Script File dialog box, select a script file and click Open.

To create a script that preloads slides

In this example of a script that displays three slides (files *slide1.sld*, *slide2.sld,* and *slide3.sld*), the time it takes to access the disk drive and load the next slide into memory overlaps with the viewing time for the current slide.

1. On the first line of the script, enter **vslide slide1**.

   The first line begins the slide show and loads *slide1*.
2. On the second line, enter **vslide *slide2**.

   The asterisk (*) preceding the slide name on the second line preloads *slide2*.
3. On the third line, enter **delay 2000**.

   The third line specifies a delay of 2000 milliseconds to allow the audience to view *slide1*.
4. On the fourth line, enter **vslide**. On the fifth line, enter **vslide *slide3**. On the sixth line, enter **delay 2000**.

   The fourth, fifth, and sixth lines display *slide2*, preload *slide3*, and specify a delay for viewing *slide2*.
5. On the seventh line, enter **vslide**. On the eighth line, enter **delay 3000**.

   The seventh and eighth lines display *slide3* and specify a delay for viewing *slide3*.
6. On the last line, enter **rscript** to repeat the script.
7. To stop a repeating script press ESC. To continue the script, enter **resume**.

   The script may contain comments, as follows:

```
; Begin slide show, load SLIDE1
VSLIDE SLIDE1
; Preload SLIDE2
VSLIDE *SLIDE2
; Let audience view SLIDE1
DELAY 2000
```

```
; Display SLIDE2
VSLIDE
; Preload SLIDE3
VSLIDE *SLIDE3
; Let audience view SLIDE2
DELAY 2000
; Display SLIDE3
VSLIDE
; Let audience view SLIDE3
DELAY 3000
; Cycle
RSCRIPT
```

Introduction to Programming Interfaces

7

The programming interfaces introduced here are ActiveX® Automation, VBA (Visual Basic® for Applications), AutoLISP®, Visual LISP™ , ObjectARX™ , and .NET. The type of interface you use depends on your application needs and programming experience.

In this chapter

ActiveX Automation

ActiveX Automation is a technology developed by Microsoft® and is based on the COM (component object model) architecture. You can use it to customize AutoCAD, share your drawing data with other applications, and automate tasks.

Overview of ActiveX

You can create and manipulate AutoCAD objects from any application that serves as an Automation controller. Thus, Automation enables macro programming across applications, a capability that does not exist in AutoLISP.

Through Automation, AutoCAD exposes programmable objects, described by the AutoCAD Object Model, that can be created, edited, and manipulated by other applications. Any application that can access the AutoCAD Object Model is an Automation controller, and the most common tool used for manipulating another application using Automation is Visual Basic for Applications (VBA). VBA is found as a component in many Microsoft Office applications. You can use these applications, or other Automation controllers, such as Visual Basic, .NET, and Delphi, to drive AutoCAD.

The advantage of implementing an ActiveX interface for AutoCAD is twofold:

- Programmatic access to AutoCAD drawings is opened up to many more programming environments. Before ActiveX Automation, developers were limited to an AutoLISP or C++ interface.
- Sharing data with other Windows applications, such as Microsoft Excel and Microsoft Word, is made dramatically easier.

For detailed information about using VBA to control AutoCAD ActiveX Automation, see the *ActiveX and VBA Developer's Guide* and *ActiveX and VBA Reference* in the Help system. On the Help menu, click Additional Resources ➤ Developer Help.

For example, you might want to prompt for input, set preferences, make a selection set, or retrieve drawing data. You can decide on the controller to use, depending on the type of manipulation.

Using Automation, you can create and manipulate AutoCAD objects from any application that serves as an Automation controller. Thus, Automation enables macro programming across applications, a capability that does not exist in AutoLISP. With Automation you can combine the features of many applications into a single application.

The displayed objects are called *Automation objects*. Automation objects make methods, properties, and events available. *Methods* are functions that perform an action on an object. *Properties* are functions that set or return information about the state of an object. *Events* are user-initiated actions or occurrences to which a program responds.

Virtually any type of application can access the displayed Automation objects within AutoCAD. These applications can be stand-alone executables, dynamic linked library (DLL) files, and macros within applications such as Microsoft Word or Microsoft Excel. The most common of these is most likely the stand-alone executable. If you are using applications from application developers, follow their instructions for installation and use of their product.

See also:

ActiveX and VBA Developer's Guide
ActiveX and VBA Reference

Define a Command to Start Your Application

You can use the acad.pgp file to define a new AutoCAD command that runs an external command to start your application. The following example defines the RUNAPP1 command, which runs the application *app1.exe* in the *c:\vbapps* directory. (Add this code to the external commands section of your acad.pgp file.)

```
RUNAPP1, start c:\vbapps\app1, 0
```

If your application requires command line parameters, you can use the following code:

```
RUNAPP2, start c:\vbapps\app2, 0, *Parameters: ,
```

This example defines the RUNAPP2 command, which prompts you for parameters and then passes them to your application.

You can also use the AutoLISP **`startapp`** function to start an application that makes use of Automation. Once AutoLISP starts the external application, it has no control over its actions. You can, however, use AutoLISP to locate and run different applications based on certain parameters.

Start an Application from a Menu or Toolbar

After defining a new command to start your application, you can make that command available from a menu or toolbar.

The macro can be called from an interface element in the customization (CUI) file. If you use only one or two applications, you can add them to one of the standard pull-down menus. If you have a group of applications, you can add your own pull-down menu or toolbar that is specifically dedicated to those

applications. For information about creating, editing, and loading customization files, see “Customize the User Interface” on page 35.

AutoCAD VBA

Microsoft Visual Basic for Applications (VBA) is an object-based programming environment designed to provide rich development capabilities. The main difference between VBA and VB (Visual Basic 6) is that VBA runs in the same process space as AutoCAD, providing an AutoCAD-intelligent and very fast programming environment.

Overview of AutoCAD VBA

VBA provides application integration with other VBA-enabled applications. This means that AutoCAD, using other application object libraries, can be an Automation controller for other applications such as Microsoft Word or Excel.

The stand-alone development editions of Visual Basic 6, which must be purchased separately, complement AutoCAD VBA with additional components such as an external database engine and report-writing capabilities.

Develop with AutoCAD VBA

VBA sends messages to AutoCAD by the AutoCAD ActiveX Automation Interface. AutoCAD VBA permits the Visual Basic environment to run simultaneously with AutoCAD and provides programmatic control of AutoCAD through the ActiveX Automation Interface. This linking of AutoCAD, ActiveX Automation, and VBA provides an extremely powerful interface. It not only controls AutoCAD objects, but it also sends data to or retrieves data from other applications.

The integration of VBA into AutoCAD provides an easy-to-use visual tool for customizing AutoCAD. For example, you can create an application that extracts attribute information automatically, inserts the results directly into an Excel spreadsheet, and performs any data transformations you need.

Three fundamental elements define VBA programming in AutoCAD. The first is AutoCAD itself, which has a rich set of objects that include AutoCAD entities, data, and commands. AutoCAD is an open-architecture application with multiple levels of interface. To use VBA effectively, familiarity with AutoCAD programmability is highly desirable. However, you will find that the VBA object-based approach is quite different from that of AutoLISP.

The second element is the AutoCAD ActiveX Automation Interface, which establishes messages (communication) with AutoCAD objects. Programming in VBA requires a fundamental understanding of ActiveX Automation. A description of the AutoCAD ActiveX Automation Interface can be found in

the *ActiveX and VBA Developer's Guide* (on the Help menu in AutoCAD, click Additional Resources ➤ Developer Help).

The third element that defines VBA programming is VBA itself. It has its own set of objects, keywords, constants, and so forth, that provide program flow, control, debugging, and execution. The Microsoft extensive Help system for VBA is included with AutoCAD VBA.

The AutoCAD ActiveX/VBA interface provides several advantages over other AutoCAD API environments:

- *Speed.* Running in-process with VBA, ActiveX applications are faster than AutoLISP applications.
- *Ease of use.* The programming language and development environment are easy-to-use and come installed with AutoCAD.
- *Windows interoperability.* ActiveX and VBA are designed to be used with other Windows applications and provide an excellent path for communication of information across applications.
- *Rapid prototyping.* The rapid interface development of VBA provides the perfect environment for prototyping applications, even if those applications will be developed eventually in another language.
- *Programmer base.* Programmers already use Visual Basic 6. AutoCAD ActiveX/VBA opens up AutoCAD customization and application development to these programmers as well as those who will learn Visual Basic 6 in the future.

Use AutoCAD VBA Applications

You load a VBA project with the command. Once loaded, its modules and macros are available in the Macros dialog box.

Although Microsoft applications store VBA projects, macros, and programs inside a specific document, AutoCAD uses a separate file with the *.dvb* extension. In this way, VBA interfaces with AutoCAD in much the same way that AutoLISP and ObjectARX do. Because VBA projects are stored in a separate file, a VBA project can open and close different AutoCAD drawings during an AutoCAD session.

NOTE AutoCAD VBA projects are not binary compatible with stand-alone Visual Basic 6 projects (VBP files). However, forms, modules, and classes can be exchanged between dissimilar projects using the IMPORT and EXPORT VBA commands in the VBA integrated development environment (IDE).

You load a VBA project with the command. Once loaded, its modules and macros are available in the Macros dialog box. To run the VBA module you use the command. If no VBA project is loaded, the options are unavailable. Procedures listed in the Macro Name box use the following syntax:

module.macro

In the Macros dialog box you choose the Macro Scope and select from the listed modules.

Use the Command Line to Run a VBA Macro

AutoCAD command line prompt equivalents are available using the –VBARUN command (signified by a hyphen in front of the VBARUN command). You can run VBA macros from the command line, scripts, and other AutoCAD programming environments. The only argument for the command is the module name using the *module.macro* syntax. The syntax looks like this:

```
-vbarun <module.macro>
```

Because macros with the same name can be duplicated in modules, the *module.macro* syntax differentiates the macro and allows for unique selection.

Automatically Load and Execute VBA Projects

As you build up a number of VBA projects, you can load them automatically each time you run AutoCAD. The macros they contain are immediately available. Additionally, the command provides a Startup Suite option that automatically loads the specified applications.

acvba.arx — Automatically Load VBA

You cannot load VBA until an AutoCAD VBA command is issued. If you want to load VBA automatically every time you start AutoCAD include the following line in the *acad.rx* file:

```
acvba.arx
```

You can automatically run a macro in the *acad.dvb* file by naming the macro AcadStartup. Any macro in your *acad.dvb* file called AcadStartup automatically executes when VBA loads.

acad.dvb — Automatically Load a VBA Project

The *acad.dvb* file is useful if you want to load a specific VBA project that contains macros you want each time you start AutoCAD. Each time you start a new AutoCAD drawing session, AutoCAD searches for the *acad.dvb* file and loads it.

If you want a macro in your *acad.dvb* file to run each time you start a new drawing or open an existing one, add the following code to your *acaddoc.lsp* file:

```
(defun S::STARTUP()
(command "_-vbarun" "updatetitleblock")
)
```

The project name in the example is `updatetitleblock`.

AutoLISP and Visual LISP

AutoLISP is based on the LISP programming language, which is simple to learn and very powerful. Because AutoCAD has a built-in LISP interpreter, you can enter AutoLISP code on the command line or load AutoLISP code from external files. Visual LISP (VLISP) is a software tool designed to expedite AutoLISP program development.

Overview of AutoLISP and Visual LISP

AutoLISP has been enhanced with Visual LISP (VLISP), which offers an integrated development environment (IDE) that includes a compiler, debugger, and other development tools to increase productivity. VLISP adds more capabilities and extends the language to interact with objects using ActiveX. VLISP also enables AutoLISP to respond to events through object reactors.

Unlike in ObjectARX, or VBA, each document opened in the Multiple Design Environment (MDE) has its own Visual LISP *namespace* and environment. A namespace is an insulated environment keeping AutoLISP routines that are specific to one document from having symbol or variable name and value conflicts with those in another document. For example, the following line of code sets a different value to the symbol `a` for different documents.

```
(setq a (getvar "DWGNAME"))
```

Visual LISP provides mechanisms for loading symbols and variables from one namespace to another. More information about namespaces can be found in the *AutoLISP Developer's Guide* (on the Help menu in AutoCAD, click Additional Resources ➤ Developer Help).

AutoLISP applications or routines can interact with AutoCAD in many ways. These routines can prompt the user for input, access built-in AutoCAD commands directly, and modify or create objects in the drawing database. By creating AutoLISP routines you can add discipline-specific commands to AutoCAD. Some of the standard AutoCAD commands are actually AutoLISP applications.

Visual LISP provides three file format options for AutoLISP applications:

- Reading an LSP file (*.lsp*)—an ASCII text file that contains AutoLISP program code.
- Reading an FAS file (*.fas*)—a binary, compiled version of a single LSP program file.
- Reading a VLX file (*.vlx*)—a compiled set of one or more LSP and/or dialog control language (DCL) files.

NOTE Like-named AutoLISP application files are loaded based on their Modified time stamp; the LSP, FAS, or VLX file with the most recent time stamp is loaded unless you specify the full file name (including the file name extension).

Because AutoCAD can read AutoLISP code directly, no compiling is required. While Visual LISP provides an IDE, you may choose to experiment by entering code at the Command prompt, which allows you to see the results immediately. This makes AutoLISP an easy language to experiment with, regardless of your programming experience.

Even if you are not interested in writing AutoLISP applications, your AutoCAD package includes many useful routines. Routines are also available as shareware through third-party developers. Knowing how to load and use these routines can enhance your productivity.

NOTE When command input comes from the AutoLISP `command` function, the settings of the and system variables are assumed to be 1 and 0, respectively. This preserves compatibility with previous releases of AutoCAD and makes customization easier (because you don't have to check the settings of these variables).

For information about AutoLISP programming, see the *AutoLISP Developer's Guide*, and for information about AutoLISP and Visual LISP functions, see the *AutoLISP Reference* (on the Help menu in AutoCAD, click Additional Resources ➤ Developer Help). AutoLISP programs can use dialog boxes with their applications. Programmable dialog boxes are described only in the *AutoLISP Developer's Guide*.

Use AutoLISP Applications

AutoLISP applications are stored in ASCII text files with the *.lsp* extension. These files generally have a header portion that describes a routine, its use, and any specific instructions. This header might also include comments that document the author and the legal information regarding the use of the routine. Comments are preceded by a semicolon (;). You can view and edit

these files with a text editor or word processor that can produce an ASCII text file.

Before you can use an AutoLISP application, it must first be loaded. You can use the command or the AutoLISP **load** function to load an application. Loading an AutoLISP application loads the AutoLISP code from the LSP file into your system's memory.

Loading an application with the **load** function involves entering AutoLISP code at the Command prompt. If the **load** function is successful, it displays the value of the last expression in the file on the command line. This is usually the name of the last function defined in the file or instructions on using the newly loaded function. If **load** fails, it returns an AutoLISP error message. A **load** failure can be caused by incorrect coding in the file or by entering the wrong file name on the command line. The syntax for the **load** function is

(load *filename* [*onfailure*])

This syntax shows that the load function has two arguments: *filename*, which is required, and *onfailure*, which is optional. When loading an AutoLISP file on the command line, you typically supply only the *filename* argument. The following example loads the AutoLISP file *newfile.lsp*.

Command: **(load "newfile")**

The *.lsp* extension is not required. This format works for any LSP file in the current library path.

To load an AutoLISP file that is not in the library path, you must provide the full path and file name as the *filename* argument.

Command: **(load "d:/files/morelisp/newfile")**

NOTE When specifying a directory path, you must use a slash (/) or two backslashes (\\) as the separator, because a single backslash has a special meaning in AutoLISP.

See also:

"Overview of File Organization" on page 3

Automatically Load and Run AutoLISP Routines

You can load AutoLISP routines each time you run AutoCAD. You can also execute certain commands or functions at specific times during a drawing session.

Overview of AutoLISP Automatic Loading

AutoCAD loads the contents of three user-definable files automatically: *acad.lsp*, *acaddoc.lsp*, and the MNL file that accompanies your current customization file. By default, the *acad.lsp* file is loaded only once, when AutoCAD starts, whereas *acaddoc.lsp* is loaded with each individual document (or drawing). This lets you associate the loading of the *acad.lsp* file with application startup, and the *acaddoc.lsp* file with document (or drawing) startup. The default method for loading these startup files can be modified by changing the setting of the system variable.

If one of these files defines a function of the special type **S::STARTUP**, this routine runs immediately after the drawing is fully initialized. The **S::STARTUP** function is described in "S::STARTUP Function: Postinitialization Execution" on page 148. As an alternative, the command provides a Startup Suite option that loads the specified applications without the need to edit any files.

The *acad.lsp* and *acaddoc.lsp* startup files are not provided with AutoCAD. It is up to the user to create and maintain these files.

Command Autoloader

When you automatically load a command using the **load** or **command** functions, the command's definition takes up memory whether or not you actually use the command. The AutoLISP **autoload** function makes a command available without loading the entire routine into memory. Adding the following code to your *acaddoc.lsp* file automatically loads the commands CMD1, CMD2, and CMD3 from the *cmds.lsp* file and the NEWCMD command from the *newcmd.lsp* file.

```
(autoload "CMDS" '("CMD1" "CMD2" "CMD3"))
(autoload "NEWCMD" '("NEWCMD"))
```

The first time you enter an automatically loaded command at the Command prompt, AutoLISP loads the entire command definition from the associated file. AutoLISP also provides the **autoarxload** function for ObjectARX applications. See **autoload** and **autoarxload** in the *AutoLISP Reference* (on the Help menu in AutoCAD, click Additional Resources ➤ Developer Help).

NOTE Like-named AutoLISP startup files are loaded based on their Modified time stamp; the LSP file with the most recent time stamp is loaded unless you specify the full file name (including the file name extension).

See also:

"Load an AutoLISP File" on page 92
"S::STARTUP Function: Postinitialization Execution" on page 148

The ACAD.LSP File

You can create an *acad.lsp* file if you regularly use specific AutoLISP routines. When you start AutoCAD, it searches the support file search path for an *acad.lsp* file. If an *acad.lsp* file is found, it is loaded into memory.

The *acad.lsp* file is loaded at each drawing session startup when AutoCAD is launched. Because the *acad.lsp* file is intended to be used for application-specific startup routines, all functions and variables defined in an *acad.lsp* file are only available in the first drawing. You will probably want to move routines that should be available in all documents from your *acad.lsp* file into the *acaddoc.lsp* file.

The recommended functionality of *acad.lsp* and *acaddoc.lsp* can be overridden with the system variable. If the ACADLSPASDOC system variable is set to 0 (the default setting), the *acad.lsp* file is loaded just once: upon application startup. If ACADLSPASDOC is set to 1, the *acad.lsp* file is reloaded with each new drawing.

The ACADLSPASDOC system variable is ignored in SDI (single document interface) mode. When the system variable is set to 1, the system variable controls reinitialization of AutoLISP between drawings. When LISPINIT is set to 1, AutoLISP functions and variables are valid in the current drawing only; each time you start a new drawing or open an existing one, all functions and variables are cleared from memory and the *acad.lsp* file is reloaded. Changing the value of LISPINIT when the SDI system variable is set to 0 has no effect.

The *acad.lsp* file can contain AutoLISP code for one or more routines, or just a series of **load** function calls. The latter method is preferable, because modification is easier. If you save the following code as an *acad.lsp* file, the files *mysessionapp1.lsp*, *databasesynch.lsp*, and *drawingmanager.lsp* are loaded every time you start AutoCAD.

```
(load "mysessionapp1")
(load "databasesynch")
(load "drawingmanager")
```

WARNING Do not modify the reserved *acad2006.lsp* file. Autodesk provides the *acad2006.lsp* file, which contains AutoLISP defined functions that are required by AutoCAD. This file is loaded into memory immediately before the *acad.lsp* file is loaded.

See also:

"Overview of File Organization" on page 3
"Prevent AutoLISP Errors When Loading Startup Files" on page 147

The ACADDOC.LSP File

The *acaddoc.lsp* file is intended to be associated with each document (or drawing) initialization. This file is useful if you want to load a library of AutoLISP routines to be available every time you start a new drawing (or open an existing drawing).

Each time a drawing opens, AutoCAD searches the library path for an *acaddoc.lsp* file. If it finds one, it loads the file into memory. The *acaddoc.lsp* file is always loaded with each drawing regardless of the settings of and .

Most users will have a single *acaddoc.lsp* file for all document-based AutoLISP routines. AutoCAD searches for an *acaddoc.lsp* file in the order defined by the library path; therefore, with this feature, you can have a different *acaddoc.lsp* file in each drawing directory, which would load specific AutoLISP routines for certain types of drawings or jobs.

The *acaddoc.lsp* file can contain AutoLISP code for one or more routines, or just a series of **load** function calls. The latter method is preferable, because modification is easier. If you save the following code as an *acaddoc.lsp* file, the files *mydocumentapp1.lsp*, *build.lsp*, and *counter.lsp* are loaded every time a new document is opened.

```
(load "mydocumentapp1")
(load "build")
(load "counter")
```

WARNING Do not modify the reserved *acad2006doc.lsp* file. Autodesk provides the *acad2006doc.lsp* file, which contains AutoLISP-defined functions that are required by AutoCAD. This file is loaded into memory immediately before the *acaddoc.lsp* file is loaded.

See also:

"Overview of File Organization" on page 3
"Prevent AutoLISP Errors When Loading Startup Files" on page 147

The MNL File for an AutoLISP Menu

When AutoCAD loads a customization file, it searches for an MNL file with a matching file name. If it finds the file, it loads the file into memory. This function ensures that AutoCAD loads the AutoLISP functions that are needed for proper operation of a menu.

This function ensures that AutoCAD loads the AutoLISP functions that are needed for proper operation of a menu. For example, the default AutoCAD customization file, *acad.cui*, relies on the file *acad.mnl*. This file defines

numerous AutoLISP functions used by the menu. The MNL file is loaded after the *acaddoc.lsp* file.

NOTE If a customization file is loaded with the AutoLISP **command** function—with syntax similar to `(command "menu" "newmenu")`—the associated MNL file is not loaded until the entire AutoLISP routine has run.

In this example, calls to the **princ** function can be used to display status messages. The first use of **princ** displays the following on the command line:

Newmenu utilities... Loaded.

The second call to **princ** exits the AutoLISP function. Without this second call to **princ**, the message would be displayed twice. As mentioned previously, you can include the *onfailure* argument with calls to the **load** function as an extra precaution.

Prevent AutoLISP Errors When Loading Startup Files

If an AutoLISP error occurs while you are loading a startup file, the remainder of the file is ignored and is not loaded.

Files specified in a startup file that do not exist or that are not in the AutoCAD library path generally cause errors. Therefore, you may want to use the *onfailure* argument with the **load** function. The following example uses the *onfailure* argument:

```
(princ (load "mydocapp1" "\nMYDOCAPP1.LSP file not loaded."))
(princ (load "build" "\nBUILD.LSP file not loaded."))
(princ (load "counter" "\nCOUNTER.LSP file not loaded."))
(princ)
```

If a call to the **load** function is successful, it returns the value of the last expression in the file (usually the name of the last defined function or a message regarding the use of the function). If the call fails, it returns the value of the *onfailure* argument. In the preceding example, the value returned by the **load** function is passed to the **princ** function, causing that value to be displayed on the command line.

For example, if an error occurs while AutoCAD loads the *mydocapp1.lsp* file, the **princ** function displays the following message and AutoCAD continues to load the two remaining files:

MYDOCAPP1.LSP file not loaded.

If you use the **command** function in an *acad.lsp*, *acaddoc.lsp,* or MNL file, it should be called only from within a **defun** statement. Use the **S::STARTUP** function to define commands that need to be issued immediately when you begin a drawing session.

See also:

"S::STARTUP Function: Postinitialization Execution" on page 148

S::STARTUP Function: Postinitialization Execution

You can define an S::STARTUP function to perform any needed setup operations after the drawing is initialized.

The startup LISP files (*acad.lsp*, *acaddoc.lsp*, and MNL) are all loaded into memory before the drawing is completely initialized. Typically, this does not pose a problem, unless you want to use the **command** function, which is not guaranteed to work until after a drawing is initialized.

If the user-defined function **S::STARTUP** is included in an *acad.lsp*, acaddoc.lsp, or MNL file, it is called when you enter a new drawing or open an existing drawing. Thus, you can include a definition of **S::STARTUP** in the LISP startup file to perform any setup operations.

For example, if you want to override the standard command by adding a message and then switching to the command, use an acaddoc.lsp file that contains the following:

```
(defun C:HATCH ( )
(alert "Using the BHATCH command!")
(princ "\nEnter OLDHATCH to get to real HATCH command.\n")
(command "BHATCH")
(princ)
)
(defun C:OLDHATCH ( )
(command ".HATCH")
(princ)
)
(defun-q S::STARTUP ( )
(command "undefine" "hatch")
(princ "\nRedefined HATCH to BHATCH!\n")
)
```

Before the drawing is initialized, new definitions for HATCH and OLDHATCH are defined with the **defun** function. After the drawing is initialized, the **S::STARTUP** function is called and the standard definition of HATCH is undefined.

NOTE To be appended, the **S::STARTUP** function must have been defined with the **defun-q** function rather than **defun**.

Because an **S::STARTUP** function can be defined in many places (an *acad.lsp*, *acaddoc.lsp*, or MNL file or any other AutoLISP file loaded from any of these), it's possible to overwrite a previously defined **S::STARTUP** function. The following example shows one method of ensuring that your startup function works with other functions.

```
(defun-q MYSTARTUP ( )
... your startup function ...
)
(setq S::STARTUP (append S::STARTUP MYSTARTUP))
```

The previous code appends your startup function to that of an existing **S::STARTUP** function and then redefines the **S::STARTUP** function to include your startup code. This works properly regardless of the prior existence of an **S::STARTUP** function.

ObjectARX

ObjectARX technology provides the foundation for design software applications to share intelligent object data. You can run third-party ObjectARX application programs or write your own.

Overview of ObjectARX

ObjectARX® (AutoCAD Runtime Extension) is a compiled-language programming environment for developing AutoCAD applications. The ObjectARX programming environment includes a number of dynamic link libraries (DLLs) that run in the same address space as AutoCAD and operate directly with core AutoCAD data structures and code. These libraries take advantage of the AutoCAD open architecture, providing direct access to the AutoCAD database structures, graphics system, and AutoCAD geometry engine to extend AutoCAD classes and capabilities at runtime. Additionally, you can use DLLs to create new commands that operate exactly the same way as native AutoCAD commands.

You can use ObjectARX libraries in conjunction with other AutoCAD programming interfaces, such as AutoLISP or VBA, enabling cross-API integration.

The ObjectARX programming environment is described in the *ObjectARX Developer's Guide*. The documentation is part of the ObjectARX Software Development Kit, which can be downloaded from the Development Tools section of the Autodesk website. For more information, click Additional Resources ➤ Developer Help on the Help menu and then click ObjectARX.

Use ObjectARX Applications

To load an ObjectARX application, you use the Load option of the command. After loading, all commands defined by this application are available at the Command prompt.

Some ObjectARX applications use large amounts of system memory. If you are finished using an application and want to remove it from memory, use the Unload option of ARX.

You can also load an ObjectARX application with the **arxload** AutoLISP function. The syntax for the **arxload** function is almost identical to that of the **load** function used with AutoLISP files. If the **arxload** function loads the ObjectARX program successfully, it returns the program name. The syntax for the **arxload** function is as follows:

```
(arxload filename [onfailure])
```

The two arguments for the **arxload** function are *filename* and *onfailure*. As with the **load** function, the *filename* argument is required and must be the complete path name description of the ObjectARX program file to load. The *onfailure* argument is optional and typically not used when you load ObjectARX programs from the command line. The following example loads the ObjectARX application *myapp.arx*.

```
(arxload "myapp")
```

As with AutoLISP files, AutoCAD searches the library path for the specified file. If you need to load a file that is not in the library path, you must provide the full path name description of the file.

NOTE When specifying a directory path, you must use a slash (/) or two backslashes (\\) as the separator, because a single backslash has a special meaning in AutoLISP.

Attempting to load an application that has previously been loaded results in an error. Before using **arxload** you should use the **arx** function to check the currently loaded applications.

To unload an application with AutoLISP, use the **arxunload** function. The following example unloads the *myapp* application.

```
(arxunload "myapp")
```

Using the **arxunload** function not only removes the application from memory but also removes the command definitions associated with that application.

See also:

"Overview of File Organization" on page 3

Automatically Load ObjectARX Applications

Some ObjectARX samples contain an *acad.rx* file, which lists ObjectARX program files that are loaded automatically when you start AutoCAD.

You can create or edit this file with a text editor or word processor that produces files in ASCII text format, adding to or deleting from its contents to

make the appropriate ObjectARX programs available for use. As an alternative, the command provides a Startup Suite option that loads the specified applications without the need to edit any files.

Because AutoCAD searches for the *acad.rx* file in the order specified by the library path, you can have a different *acad.rx* file in each drawing directory. This makes specific ObjectARX programs available for certain types of drawings. For example, you might keep 3D drawings in a directory called *AcadJobs/3d_dwgs*. If that directory is set up as the current directory, you could copy the *acad.rx* file into that directory and modify it in the following manner:

```
myapp1
otherapp
```

If you place this new *acad.rx* file in the *AcadJobs/3d_dwgs* directory and you start AutoCAD with that as the current directory, these new ObjectARX programs are then loaded and are available from the AutoCAD command line. Because the original *acad.rx* file is still in the directory with the AutoCAD program files, the default *acad.rx* file will be loaded if you start AutoCAD from another directory that does not contain an *acad.rx* file.

You can load ObjectARX programs from an MNL file using the **arxload** function. This ensures that an ObjectARX program, required for proper operation of a menu, will be loaded when the menu file is loaded.

You can also autoload many ObjectARX-defined AutoCAD commands. See "Overview of AutoLISP Automatic Loading" on page 144 and autoarxload in the *AutoLISP Reference* (on the Help menu in AutoCAD, click Additional Resources ➤ Developer Help).

See also:

"**Overview of AutoLISP Automatic Loading**" **on page 144**

.NET

With the Microsoft .NET Framework, you can create applications that interoperate with AutoCAD using programming languages like VB .NET and C#.

Overview of .NET

The .NET Framework is a language-neutral programming environment developed by Microsoft. In addition to the run-time environment, the Framework provides class libraries that facilitate development of Windows- and Web-based applications that are interoperable and secure.

AutoCAD supports .NET application development with ObjectARX managed wrapper classes. See the "AutoCAD Managed Class Reference" and the

"ObjectARX Managed Wrapper Classes" sections of the *ObjectARX Developer's Guide*, both in the ObjectARX SDK, for a complete list of the managed wrapper classes that are available. For more information about the .NET Framework, see the Microsoft documentation.

Managed wrapper classes are provided for most of the ObjectARX SDK, enabling you to write applications in any language that is supported by the .NET Framework, including VB .NET and C#. The managed classes implement database functionality and enable you to write applications that read and write drawing format (DWG) files. They also provide access to AutoCAD user interface elements, including the command line and feature dialog boxes, the AutoCAD editor, and the publishing and plotting components.

Loading Managed Applications in AutoCAD

To load a managed application, enter NETLOAD at the AutoCAD Command prompt and browse to the desired DLL file. Managed applications are unloaded only when AutoCAD exits.

Shapes and Shape Fonts

8

With AutoCAD®, you can define shapes to use as drawing symbols and text fonts. This chapter explains how to create and compile your own shape and font files.

In this chapter

Overview of Shape Files

Shapes are objects that you use like blocks. First you use the LOAD command to load the compiled shape file containing the shape definition. Then you use the SHAPE command to insert shapes from the file into your drawing. You can specify the scale and rotation to use for each shape as you add it. AutoCAD SHP fonts are a special type of shape file, and are defined in the same way as shape files.

Blocks are more versatile and easier to use and apply than shapes. However, shapes are more efficient for AutoCAD to store and draw. User-defined shapes are helpful when you must insert a simple part many times and when speed is important.

Compile Shape/Font Files

You enter the description of shapes in a specially formatted text file with a file extension of *.shp*. To create the file, use a text editor or word processor that enables you to save in ASCII format, and then compile the ASCII file. Compiling a shape definition file (SHP) generates a compiled shape file (SHX).

The compiled file has the same name as the shape definition file but with a file type of SHX. If the shape definition file defines a font, you use the STYLE command to define a text style. Then, you use one of the text placement commands (TEXT or MTEXT) to place the characters in the drawing. If the shape definition file defines shapes, you use the LOAD command to load the shape file into the drawing. Then, you use the SHAPE command to place the individual shapes in the drawing (similar in concept to the INSERT command).

Compile PostScript Fonts

To use a Type 1 PostScript font in AutoCAD, you must first compile it into an AutoCAD shape file. The COMPILE command accepts both SHP and PFB files as input and generates an SHX file. Compiled versions of PostScript fonts can take a lot of disk space, so compile only those fonts you use frequently.

AutoCAD cannot compile and load every Type 1 font. The PostScript font facilities in AutoCAD are intended to process a subset of Adobe fonts. If you receive an error while compiling a PostScript font, the resulting SHX file (if one is generated) may not load into AutoCAD.

For more information on the Adobe Type 1 font format, refer to *Adobe Type1 Font Format Version 1.1*. When you've purchased and installed these fonts, you can begin using them with AutoCAD.

NOTE Make sure you understand any copyright that accompanies the PostScript fonts you use. The same copyright restrictions generally apply to the SHX form of fonts you've compiled.

To compile a shape or font file

■ On the Command line, enter **compile**.

In the Select Shape File dialog box, you can select a shape definition file (SHP) or PostScript font file (PFB). After you select the file name, compiling begins. If AutoCAD finds an error in the shape descriptions, a message is displayed telling you the type of error and the line number. When compiling is complete, the following messages are displayed:

Compilation successful.
Output file *name.shx* contains *nnn* bytes.

Create Shape Definition Files

AutoCAD font and shape files (SHX) are compiled from shape definition files (SHP). You can create or modify shape definition files with a text editor or word processor that saves files in ASCII format.

Shape Descriptions

AutoCAD font and shape files (SHX) are compiled from shape definition files (SHP). You can create or modify shape definition files with a text editor or word processor that saves files in ASCII format.

The syntax of the shape description for each shape or character is the same regardless of the final use (shape or font) for that shape description. If a shape definition file is to be used as a font file, the first entry in the file describes the font itself rather than a shape within the file. If this initial entry describes a shape, the file is used as a shape file.

Being able to create your own shape definitions is a valuable skill. Keep in mind, however, that this is a very complex subject to learn and requires patience.

Each line in a shape definition file can contain up to 128 characters. Longer lines cannot be compiled. Because AutoCAD ignores blank lines and text to the right of a semicolon, you can embed comments in shape definition files.

Each shape description has a header line of the following form and is followed by one or more lines containing specification bytes, separated by commas and terminated by a 0.

*shapenumber,defbytes,shapename
specbyte1,specbyte2,specbyte3,...,0

The following list describes the fields of a shape description:

shapenumber

A number, unique to the file, between 1 and 258 (and up to 32768 for Unicode fonts), and preceded by an asterisk (*). Non-Unicode font files use the shape numbers 256, 257, and 258 for the symbolic identifiers Degree_Sign, Plus_Or_Minus_Sign, and Diameter_Symbol. For Unicode fonts these glyphs appear at the U+00B0, U+00B1, and U+2205 shape numbers and are part of the "Latin Extended-A" subset.

Text fonts (files containing shape definitions for each character) require specific numbers corresponding to the value of each character in the ASCII code; other shapes can be assigned any numbers.

defbytes

The number of data bytes (*specbytes*) required to describe the shape, including the terminating 0. The limit is 2,000 bytes per shape.

shapename

The shape name. Shape names must be uppercase to be recognized. Names with lowercase characters are ignored and are usually used to label font shape definitions.

specbyte

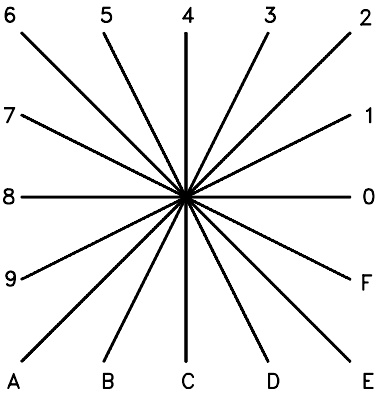
A shape specification byte. Each specification byte is a code that defines either a vector length and direction or one of a number of special codes. A specification byte can be expressed in the shape definition file as either a decimal or hexadecimal value. This section uses both decimal and hexadecimal specification byte values for its examples (as do many of the shape definition files). If the first character of a specification byte is a 0 (zero), the two characters that follow are interpreted as hexadecimal values.

Vector Length and Direction Code

A simple shape specification byte contains vector length and direction encoded into one specification byte.

A simple shape specification byte contains vector length and direction encoded into one specification byte (one *specbyte* field). Each vector length and direction code is a string of three characters. The first character must be a 0, which indicates to AutoCAD that the next two characters are interpreted as hexadecimal values. The second character specifies the length of the vector in units. Valid hexadecimal values range from 1 (one unit long) through F (15

units long). The third character specifies the direction of the vector. The following figure illustrates the direction codes.



Vector direction codes

All the vectors in the preceding figure were drawn with the same length specification. Diagonal vectors stretch to match the *X* or *Y* displacement of the closest orthogonal vector. This is similar to the action of the snap grid in AutoCAD.

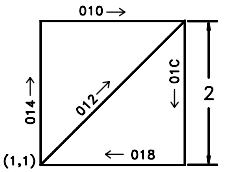
The following example constructs a shape named DBOX with an arbitrarily assigned shape number of 230.

```
*230,6,DBOX
014,010,01C,018,012,0
```

The preceding sequence of specification bytes defines a box one unit high byone unit wide, with a diagonal line running from the lower left to the upper right. After saving the file as *dbox.shp*, use the COMPILE command to generate the *dbox.shx* file. Use the LOAD command to load the shape file containing this definition, and then use the SHAPE command as follows:

Command: **shape**
Enter shape name or [?]: **dbox**
Specify insertion point: **1,1**
Specify height <*current*>: **2**
Specify rotation angle <*current*>: **0**

The resulting shape is shown in the following illustration.



Special Codes

In addition to defining vectors, a specification byte can use special codes to create additional forms and specify certain actions.

In addition to defining vectors, a specification byte can use special codes to create additional forms and specify certain actions. To use a special code, the second character of the three-character string (the vector length specification) must be 0, or you can specify only the code number. For example, 008 and 8 are both valid specifications.

Specification byte codes	
Code	**Description**
000	End of shape definition
001	Activate Draw mode (pen down)
002	Deactivate Draw mode (pen up)
003	Divide vector lengths by next byte
004	Multiply vector lengths by next byte
005	Push current location onto stack
006	Pop current location from stack
007	Draw subshape number given by next byte
008	*X-Y* displacement given by next two bytes
009	Multiple *X-Y* displacements, terminated (0,0)
00A	Octant arc defined by next two bytes
00B	Fractional arc defined by next five bytes
00C	Arc defined by *X-Y* displacement and bulge
00D	Multiple bulge-specified arcs

Specification byte codes	
Code	**Description**
00E	Process next command only if vertical text

Codes 0, 1, and 2: End of Shape and Draw Mode Control

Code 0 marks the end of the shape definition.

Codes 1 and 2 control Draw mode. Draw is activated at the start of each shape. When Draw mode is turned on (code 1), the vectors cause lines to be drawn. When Draw mode is turned off (code 2), the vectors move to a new location without drawing.

Codes 3 and 4: Size Control

Codes 3 and 4 control the relative size of each vector. The height specified with the SHAPE command is initially considered the length of a single orthogonal vector (direction 0, 4, 8, or C). Code 3 divides vector lengths by the next byte. Code 4 multiplies vector lengths by the next byte. Codes 3 and 4 are followed by a specification byte containing an integer scale factor (1 through 255). If you want the shape height to specify the size of the entire shape, and you use 10 vector lengths to draw it, you can use 3,10 to scale the height specification. The scale factor is cumulative within a shape; that is, multiplying by 2 and again by 6 results in a scale factor of 12. Usually, you should reverse the effect of your scale factors at the end of the shape, especially for subshapes and text font shapes. AutoCAD does not reset the scale factor for you.

Codes 5 and 6: Location Save/Restore

Code 5 pushes (saves) and code 6 pops (restores) the current coordinate position while drawing a shape so that you can return to it from a later point in the shape. You must pop everything you push. The position stack is only four locations deep. If the stack overflows because of too many pushes or too many missing pops, the following message is displayed when the shape is drawn.

Position stack overflow in shape *nnn*
Similarly, if you try to pop more locations than have been pushed onto the stack, the following message is displayed when the shape is drawn.

Position stack underflow in shape *nnn*

Code 7: Subshape

Code 7 draws the subshape number given by the next byte. For a non-Unicode font the specification byte following code 7 is a shape number from 1 to 255. For a Unicode font, code 7 is followed by a Unicode shape number from 1 to 65535. Unicode shape numbers should be counted as two bytes (for specific information about the differences between Unicode and non-Unicode fonts, see "Unicode Font Descriptions" on page 208). The shape with that number (in the same shape file) is drawn at this time. Draw mode is not reset for the new shape. When the subshape is complete, drawing the current shape resumes.

Codes 8 and 9: X-Y Displacements

Normal vector specification bytes draw only in the 16 predefined directions, and the longest length is 15. These restrictions help make shape definitions efficient but are sometimes limiting. With codes 8 and 9 you can draw nonstandard vectors using *X-Y* displacements. Code 8 specifies the *X-Y* displacement given by the next two bytes. Code 8 must be followed by two specification bytes in the format:

`8,`*`X-displacement`*`,`*`Y-displacement`*

The *X-Y* displacements can range from –128 to +127. A leading + is optional, and you can use parentheses to improve readability. The following example results in a vector that draws (or moves) 10 units to the left and three units up.

```
8,(-10,3)
```

Following the two displacement specification bytes, the shape returns to Normal Vector mode.

You can use code 9 to draw a sequence of nonstandard vectors. Code 9 specifies any number of *X-Y* displacement pairs. The code sequence is terminated by a (0,0) pair. The following example draws three nonstandard vectors and returns to Normal Vector mode.

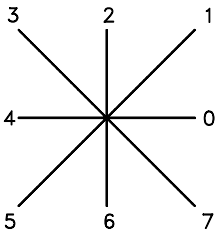
```
9,(3,1),(3,2),(2,-3),(0,0)
```

You must terminate the sequence of *X-Y* displacement pairs with a (0,0) pair in order for AutoCAD to recognize any Normal Vectors or special codes that follow.

Code 00A: Octant Arc

Special code 00A (or 10) uses the next two specification bytes to define an arc. This is called an *octant arc* because it spans one or more 45-degree *octants*, starting and ending on an octant boundary. Octants are numbered

counterclockwise from the 3 o'clock position, as shown in the following illustration.



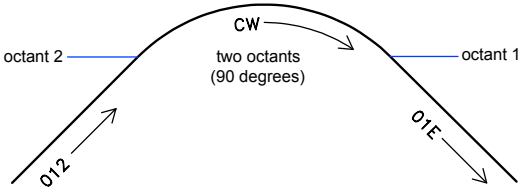
The arc specification is

`10,`*`radius`*`,(-)0`*`SC`*

The radius can be any value from 1 through 255. The second specification byte indicates the direction of the arc (counterclockwise if positive, and clockwise if negative), its starting octant (*s*, a value from 0 through 7), and the number of octants it spans (*c*, a value from 0 through 7, in which 0 equals eight octants, or a full circle). You can use parentheses to improve readability. For example, consider the following fragment of a shape definition:

```
...012,10,(1,-032),01E,...
```

This code draws a one-unit vector up and to the right, a clockwise arc from octant 3 (with a radius of one unit for two octants), and then a one-unit vector down and to the right, as shown in the following illustration.



Code 00B: Fractional Arc

Special code 00B (11) draws an arc that doesn't necessarily start and end on an octant boundary. The definition uses five specification bytes.

`11,`*`start_offset`*`,`*`end_offset`*`,`*`high_radius`*`,`*`radius`*`,(-)0`*`SC`*

The *`start_offset`* and *`end_offset`* represent how far from an octant boundary the arc begins or ends. The *`high_radius`* represents the most significant eight bits of the radius; the high radius will be 0 unless the *`radius`* is greater than

255 units. Multiply the *high_radius* value by 256 and add that value to the *radius* value to generate an arc radius greater than 255. The *radius* and ending specification byte are the same as for the octant arc specification (code 00A, described previously).

You determine the `start offset` by calculating the difference in degrees between the starting octant's boundary (a multiple of 45 degrees) and the start of the arc. Then, you multiply this difference by 256 and divide by 45. If the arc starts on an octant boundary, its `start offset` is 0.

The `end offset` is calculated in a similar fashion, but you use the number of degrees from the last octant boundary crossed to the end of the arc. If the arc ends on an octant boundary, its `end offset` is 0.

For example, a fractional arc from 55 degrees to 95 degrees with a 3 unit radius would be coded as follows:

```
11,(56,28,0,3,012)
```

Here is the explanation:

```
start_offset = 56 because ((55 - 45) * 256 / 45) = 56
end_offset = 28 because ((95 - 90) * 256 / 45) = 28
high_radius = 0 because (radius < 255)
radius = 3
starting octant = 1 because arc starts in the 45 degree octant
ending octant = 2 because arc ends in the 90 degree octant
```

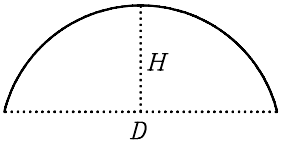
Codes 00C and 00D: Bulge-Specified Arcs

Special codes 00C and 00D (12 and 13) provide another mechanism for including arc segments in shape descriptions. They are similar to codes 8 and 9 in that you can use them to specify *X-Y* displacements. However, codes 00C and 00D draw arcs by applying a *bulge factor* to the displacement vector. Code 00C draws one arc segment, while code 00D draws multiple arc segments *(polyarcs)* until it is terminated by a (0,0) displacement.

Code 00C must be followed by three bytes describing the arc:

```
0C,X-displacement,Y-displacement,Bulge
```

Both the *X* and *Y* displacement and the bulge, which specifies the curvature of the arc, can range from –127 to +127. If the line segment specified by the displacement has length *D*, and the perpendicular distance from the midpoint of that segment has height *H*, the magnitude of the bulge is ((2* *H* / *D*) * 127). The sign is negative if the arc from the current location to the new location is clockwise.



A semicircle has bulge 127 (or –127) and is the greatest arc that can be represented as a single-arc segment using these codes (use two consecutive arc segments for larger arcs). A bulge specification of 0 is valid and represents a straight-line segment. Note, however, that using code 8 for a straight-line segment saves a byte in the shape description.

The polyarc code (00D, or 13) is followed by 0 or by more arc segment triples, and is terminated by a (0,0) displacement. Note that no bulge is specified after the final displacement. For example, the letter *S* might be defined by the following sequence:

```
13,(0,5,127),(0,5,-127),(0,0)
```

Zero bulge segments are useful within polyarcs to represent straight segments; they are more efficient than terminating the polyarc, inserting one straight segment, and then starting another polyarc.

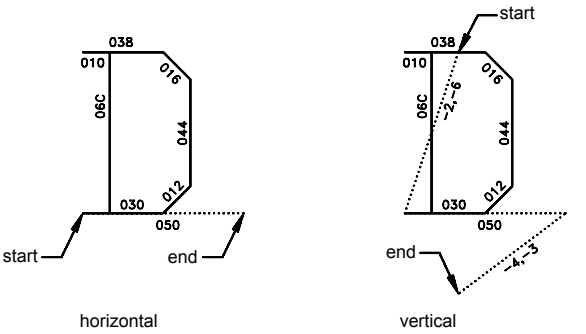
The number –128 cannot be used in arc segment and polyarc definitions.

Code 00E: Flag Vertical Text Command

Special code 00E (14) is used only in dual-orientation text font descriptions, where the font is used in both horizontal and vertical orientations. When this special code is encountered in a character definition, the next code is either processed or skipped, depending on orientation. If the orientation is vertical, the next code is processed; if it is horizontal, the next code is skipped.

In horizontal text, the start point for each character is the left end of the baseline. In vertical text, the start point is assumed to be the top center of the character. At the end of each character, a pen-up segment is normally drawn to position to the next character's start point. For horizontal text, it is to the right; for vertical text, it is downward. The special 00E (14) code is used primarily to adjust for differences in start points and endpoints, permitting the same character shape definition to be used both horizontally and vertically. For instance, the following definition of an uppercase D could be used in either horizontal or vertical text.

```
*68,22,ucd
2,14,8,(-2, 6),1,030,012,044,016,038,2,010,1,06C,2,050,
14,8,(-4,-3),0
```



Text Font Descriptions

Text fonts must include a special shape number 0 that conveys information about the font itself.

AutoCAD is packaged with numerous text fonts. You can use the STYLE command to apply expansion, compression, or obliquing to any of these fonts, thereby tailoring the characters to your needs. You can draw text of any height, at any baseline angle, and with either horizontal or vertical orientation using these fonts.

AutoCAD text fonts are files of shape definitions with shape numbers corresponding to the ASCII code for each character. Codes 1 through 31 are for control characters, only one of which is used in AutoCAD text fonts:

10 (LF)

The line feed (LF) must drop down one line without drawing. This is used for repeated commands, to place succeeding lines below the first one.

```
*10,5,lf

2,8,(0,-10),0
```

You can modify the spacing of lines by adjusting the downward movement specified by the LF shape definition.

Text fonts must include a special shape number 0 that conveys information about the font itself. The format has the following syntax:

```
*0,4,font-name
above,below,modes,0
```

The *above* value specifies the number of vector lengths above the baseline that the uppercase letters extend, and *below* indicates how far the lowercase

letters descend below the baseline. The baseline is similar in concept to the lines on writing paper. These values define the basic character size and are used as scale factors for the height specified in the TEXT command.

The *modes* byte should be 0 for a horizontally oriented font and 2 for a dual-orientation (horizontal or vertical) font. The special 00E (14) command code is honored only when *modes* is set to 2.

The standard fonts supplied with AutoCAD include a few additional characters required for the AutoCAD dimensioning feature.

%%d Degree symbol ( )

%%p Plus/minus tolerance symbol ( )

%%c Circle diameter dimensioning symbol

You can use these and the %% *nnn* control sequences, as described under in the *Command Reference*.

NOTE AutoCAD draws text characters by their ASCII codes (shape numbers) and not by name. To save memory, specify the shape name portion of each text shape definition in lowercase as shown in the following example. (Lowercase names are not saved in memory.)

```
*65,11,uca
024,043,04d,02c,2,047,1,040,2,02e,0
```

Because the shape name *uca* contains lowercase letters, AutoCAD doesn't save the name in memory. However, you can use the name for reference when editing the font definition file. In this example, *uca* stands for uppercase A.

Sample Files

This topic contains sample files that help extend the font characters provided with AutoCAD.

Extended Simplex Roman

```
;;
;; romans.shp - Extended Simplex Roman
;;
;; Copyright 1997 by Autodesk, Inc.
;;
;; Permission to use, copy, modify, and distribute this software for
;; any purpose and without fee is hereby granted, provided that the
;; above copyright notice appears in all copies and that the
restricted
;; rights notice below appear in all supporting documentation.
;;
;; Use, duplication, or disclosure by the U.S. Government is subject
```

```
;; to restrictions set forth in FAR 52.227-19 (Commercial Computer
;; Software - Restricted Rights) and DFAR 252.227-7013(c)(1)(ii)
;; (Rights in Technical Data and Computer Software), as applicable.
;;
*UNIFONT,6,ROMANS Copyright 1997 by Autodesk, Inc.
21,7,2,0,0,0
*0000A,9,lf
2,8,(0,-34),14,8,(30,34),0
*00020,9,spc
2,8,(21,0),14,8,(-21,-30),0
*00021,30,kexc
2,14,8,(-5,-21),14,5,8,(5,21),1,0EC,2,05C,1,01A,01E,012,016,2,
8,(5,-2),14,6,14,8,(5,-9),0
*00022,41,kdblqt
2,14,8,(-8,-25),14,5,8,(6,24),1,01A,016,012,01E,02C,02B,01A,2,
8,(8,5),1,01A,016,012,01E,02C,02B,01A,2,8,(4,-19),14,6,
14,8,(8,-9),0
*00023,57,kns
2,14,3,2,14,8,(-21,-50),14,4,2,14,5,8,(11,25),1,8,(-7,-32),2,
8,(13,32),1,8,(-7,-32),2,8,(-6,19),1,0E0,2,8,(-15,-6),1,0E0,2,
8,(4,-6),14,6,14,3,2,14,8,(21,-32),14,4,2,0
*00024,67,kds
2,14,8,(-10,-25),14,5,8,(8,25),1,8,(0,-29),2,8,(4,29),1,
8,(0,-29),2,8,(5,22),1,026,8,(-3,1),048,8,(-3,-1),02A,02C,02D,
01E,02F,8,(6,-2),02F,01E,02D,03C,02A,8,(-3,-1),048,8,(-3,1),026,
2,8,(17,-3),14,6,14,8,(10,-13),0
*00025,64,kpc
2,14,8,(-12,-21),14,5,8,(21,21),1,8,(-18,-21),2,8,(5,21),1,02E,
02C,02B,029,028,026,024,023,021,020,02F,8,(3,-1),030,8,(3,1),021,
2,8,(-4,-14),1,029,02B,02C,02E,020,021,023,024,026,028,2,
8,(7,-7),14,6,14,8,(12,-9),0
*00026,67,kand
2,14,8,(-13,-21),14,5,8,(23,12),1,014,016,018,01A,02B,8,(-2,-5),
8,(-2,-3),02A,029,048,027,016,025,024,023,012,8,(7,4),012,023,
024,025,027,029,02B,02C,8,(1,-3),8,(2,-3),8,(5,-7),02E,02F,020,
012,014,2,8,(3,-2),14,6,14,8,(13,-9),0
*00027,29,kapos
2,14,8,(-5,-25),14,5,8,(6,24),1,01A,016,012,01E,02C,02B,01A,2,
8,(6,-19),14,6,14,8,(5,-9),0
*00028,39,klp
2,14,8,(-7,-25),14,5,8,(11,25),1,02A,8,(-2,-3),04B,8,(-1,-5),04C,
8,(1,-5),04D,8,(2,-3),02E,2,8,(3,7),14,6,14,8,(7,-16),0
*00029,39,krp
2,14,8,(-7,-25),14,5,8,(3,25),1,02E,8,(2,-3),04D,8,(1,-5),04C,
8,(-1,-5),04B,8,(-2,-3),02A,2,8,(11,7),14,6,14,8,(7,-16),0
*0002A,37,kas
2,14,8,(-8,-21),14,5,8,(8,21),1,0CC,2,8,(-5,9),1,8,(10,-6),2,064,
1,8,(-10,-6),2,8,(13,-12),14,6,14,8,(8,-9),0
*0002B,31,kpls
2,14,8,(-13,-18),14,5,8,(13,18),1,8,(0,-18),2,096,1,8,(18,0),2,
8,(4,-9),14,6,14,8,(13,-9),0
*0002C,29,kcma
2,14,8,(-5,-2),14,5,8,(6,1),1,01A,016,012,01E,02C,02B,01A,2,
8,(6,4),14,6,14,8,(5,-13),0
*0002D,25,ksub
2,14,8,(-13,-9),14,5,8,(4,9),1,8,(18,0),2,8,(4,-9),14,6,
```

```
14,8,(13,-9),0
*0002E,26,kper
2,14,8,(-5,-2),14,5,8,(5,2),1,01A,01E,012,016,2,8,(5,-2),14,6,
14,8,(5,-9),0
*0002F,25,kdiv
2,14,8,(-11,-25),14,5,8,(20,25),1,8,(-18,-32),2,8,(20,7),14,6,
14,8,(11,-16),0
*00030,62,n0
2,14,8,(-10,-21),14,5,8,(9,21),1,8,(-3,-1),8,(-2,-3),8,(-1,-5),
03C,8,(1,-5),8,(2,-3),8,(3,-1),020,8,(3,1),8,(2,3),8,(1,5),034,
8,(-1,5),8,(-2,3),8,(-3,1),028,2,8,(11,-21),14,6,14,8,(10,-9),0
*00031,27,n1
2,14,8,(-10,-21),14,5,8,(6,17),1,021,032,8,(0,-21),2,8,(9,0),
14,6,14,8,(10,-9),0
*00032,37,n2
2,14,8,(-10,-21),14,5,8,(4,16),1,014,023,012,021,040,02F,01E,02D,
02C,02B,8,(-2,-3),0AA,0E0,2,8,(3,0),14,6,14,8,(10,-9),0
*00033,46,n3
2,14,8,(-10,-21),14,5,8,(5,21),1,0B0,8,(-6,-8),030,02F,01E,
8,(1,-3),02C,8,(-1,-3),02A,8,(-3,-1),038,8,(-3,1),016,025,2,
8,(17,-4),14,6,14,8,(10,-9),0
*00034,34,n4
2,14,8,(-10,-21),14,5,8,(13,21),1,8,(-10,-14),0F0,2,8,(-5,14),1,
8,(0,-21),2,8,(7,0),14,6,14,8,(10,-9),0
*00035,52,n5
2,14,8,(-10,-21),14,5,8,(15,21),1,0A8,8,(-1,-9),012,8,(3,1),030,
8,(3,-1),02E,8,(1,-3),02C,8,(-1,-3),02A,8,(-3,-1),038,8,(-3,1),
016,025,2,8,(17,-4),14,6,14,8,(10,-9),0
*00036,68,n6
2,14,8,(-10,-21),14,5,8,(16,18),1,025,8,(-3,1),028,8,(-3,-1),
8,(-2,-3),8,(-1,-5),05C,8,(1,-4),02E,8,(3,-1),010,8,(3,1),022,
8,(1,3),014,8,(-1,3),026,8,(-3,1),018,8,(-3,-1),02A,8,(-1,-3),2,
8,(16,-7),14,6,14,8,(10,-9),0
*00037,31,n7
2,14,8,(-10,-21),14,5,8,(17,21),1,8,(-10,-21),2,8,(-4,21),1,0E0,
2,8,(3,-21),14,6,14,8,(10,-9),0
*00038,66,n8
2,14,8,(-10,-21),14,5,8,(8,21),1,8,(-3,-1),02B,02C,02D,02F,
8,(4,-1),8,(3,-1),02E,02D,03C,02B,01A,8,(-3,-1),048,8,(-3,1),016,
025,034,023,022,8,(3,1),8,(4,1),021,023,024,025,8,(-3,1),048,2,
8,(12,-21),14,6,14,8,(10,-9),0
*00039,68,n9
2,14,8,(-10,-21),14,5,8,(16,14),1,8,(-1,-3),02A,8,(-3,-1),018,
8,(-3,1),026,8,(-1,3),014,8,(1,3),022,8,(3,1),010,8,(3,-1),02E,
8,(1,-4),05C,8,(-1,-5),8,(-2,-3),8,(-3,-1),028,8,(-3,1),025,2,
8,(16,-3),14,6,14,8,(10,-9),0
*0003A,33,kcol
2,14,8,(-5,-14),14,5,8,(5,14),1,01A,01E,012,016,2,0CC,1,01A,01E,
012,016,2,8,(5,-2),14,6,14,8,(5,-9),0
*0003B,38,ksmc
2,14,8,(-5,-14),14,5,8,(5,14),1,01A,01E,012,016,2,8,(1,-13),1,
01A,016,012,01E,02C,02B,01A,2,8,(6,4),14,6,14,8,(5,-13),0
*0003C,28,klt
2,14,8,(-12,-18),14,5,8,(20,18),1,8,(-16,-9),8,(16,-9),2,8,(4,0),
14,6,14,8,(12,-9),0
*0003D,33,keq
```

```
2,14,8,(-13,-12),14,5,8,(4,12),1,8,(18,0),2,8,(-18,-6),1,
8,(18,0),2,8,(4,-6),14,6,14,8,(13,-9),0
*0003E,28,kgt
2,14,8,(-12,-18),14,5,8,(4,18),1,8,(16,-9),8,(-16,-9),2,8,(20,0),
14,6,14,8,(12,-9),0
*0003F,42,kqm
2,14,8,(-9,-21),14,5,8,(3,16),1,014,023,012,021,040,02F,01E,02D,
02C,02B,01A,049,03C,2,05C,1,01A,01E,012,016,2,8,(9,-2),14,6,
14,8,(9,-9),0
*00040,93,kea
2,14,3,2,14,8,(-27,-42),14,4,2,14,5,8,(18,13),1,025,027,038,029,
01A,02B,03C,02D,01E,02F,030,021,023,2,084,1,0AC,01E,020,022,
8,(1,3),024,8,(-1,3),025,026,027,8,(-3,1),038,8,(-3,-1),029,02A,
02B,8,(-1,-3),03C,8,(1,-3),02D,02E,02F,8,(3,-1),030,8,(3,1),021,
012,2,8,(6,-3),14,6,14,3,2,14,8,(27,-18),14,4,2,0
*00041,39,uca
2,14,8,(-9,-21),14,5,8,(9,21),1,8,(-8,-21),2,8,(8,21),1,
8,(8,-21),2,8,(-13,7),1,0A0,2,8,(4,-7),14,6,14,8,(9,-9),0
*00042,70,ucb
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(0,21),1,090,8,(3,-1),01E,02D,02C,02B,01A,8,(-3,-1),2,098,1,
090,8,(3,-1),01E,02D,03C,02B,01A,8,(-3,-1),098,2,8,(17,0),14,6,
14,3,2,14,8,(21,-18),14,4,2,0
*00043,55,ucc
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,8,(18,16),1,025,026,027,048,
029,02A,02B,8,(-1,-3),05C,8,(1,-3),02D,02E,02F,040,021,022,023,2,
8,(3,-5),14,6,14,3,2,14,8,(21,-18),14,4,2,0
*00044,61,ucd
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(0,21),1,070,8,(3,-1),02E,02D,8,(1,-3),05C,8,(-1,-3),02B,02A,
8,(-3,-1),078,2,8,(17,0),14,6,14,3,2,14,8,(21,-18),14,4,2,0
*00045,55,uce
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(0,21),1,0D0,2,8,(-13,-10),1,080,2,8,(-8,-11),1,0D0,2,8,(2,0),
14,6,14,3,2,14,8,(19,-18),14,4,2,0
*00046,37,ucf
2,14,8,(-9,-21),14,5,8,(4,21),1,8,(0,-21),2,8,(0,21),1,0D0,2,
8,(-13,-10),1,080,2,8,(6,-11),14,6,14,8,(9,-9),0
*00047,60,ucg
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,8,(18,16),1,025,026,027,048,
029,02A,02B,8,(-1,-3),05C,8,(1,-3),02D,02E,02F,040,021,022,023,
034,2,058,1,050,2,8,(3,-8),14,6,14,3,2,14,8,(21,-18),14,4,2,0
*00048,39,uch
2,14,8,(-11,-21),14,5,8,(4,21),1,8,(0,-21),2,8,(14,21),1,
8,(0,-21),2,8,(-14,11),1,0E0,2,8,(4,-11),14,6,14,8,(11,-9),0
*00049,25,uci
2,14,8,(-4,-21),14,5,8,(4,21),1,8,(0,-21),2,8,(4,0),14,6,
14,8,(4,-9),0
*0004A,37,ucj
2,14,8,(-8,-21),14,5,8,(12,21),1,8,(0,-16),8,(-1,-3),01A,029,028,
027,016,8,(-1,3),024,2,8,(14,-7),14,6,14,8,(8,-9),0
*0004B,49,uck
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(14,21),1,0EA,2,052,1,8,(9,-12),2,8,(3,0),14,6,14,3,2,
14,8,(21,-18),14,4,2,0
*0004C,43,ucl
```

```
2,14,3,2,14,8,(-17,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(0,0),1,0C0,2,8,(1,0),14,6,14,3,2,14,8,(17,-18),14,4,2,0
*0004D,49,ucm
2,14,8,(-12,-21),14,5,8,(4,21),1,8,(0,-21),2,8,(0,21),1,
8,(8,-21),2,8,(8,21),1,8,(-8,-21),2,8,(8,21),1,8,(0,-21),2,
8,(4,0),14,6,14,8,(12,-9),0
*0004E,41,ucn
2,14,8,(-11,-21),14,5,8,(4,21),1,8,(0,-21),2,8,(0,21),1,
8,(14,-21),2,8,(0,21),1,8,(0,-21),2,8,(4,0),14,6,14,8,(11,-9),0
*0004F,50,uco
2,14,8,(-11,-21),14,5,8,(9,21),1,029,02A,02B,8,(-1,-3),05C,
8,(1,-3),02D,02E,02F,040,021,022,023,8,(1,3),054,8,(-1,3),025,
026,027,048,2,8,(13,-21),14,6,14,8,(11,-9),0
*00050,55,ucp
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(0,21),1,090,8,(3,-1),01E,02D,03C,02B,01A,8,(-3,-1),098,2,
8,(17,-10),14,6,14,3,2,14,8,(21,-18),14,4,2,0
*00051,56,ucq
2,14,8,(-11,-21),14,5,8,(9,21),1,029,02A,02B,8,(-1,-3),05C,
8,(1,-3),02D,02E,02F,040,021,022,023,8,(1,3),054,8,(-1,3),025,
026,027,048,2,8,(3,-17),1,06E,2,8,(4,2),14,6,14,8,(11,-11),0
*00052,61,ucr
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(0,21),1,090,8,(3,-1),01E,02D,02C,02B,01A,8,(-3,-1),098,2,070,
1,8,(7,-11),2,8,(3,0),14,6,14,3,2,14,8,(21,-18),14,4,2,0
*00053,51,ucs
2,14,8,(-10,-21),14,5,8,(17,18),1,026,8,(-3,1),048,8,(-3,-1),02A,
02C,02D,01E,02F,8,(6,-2),02F,01E,02D,03C,02A,8,(-3,-1),048,
8,(-3,1),026,2,8,(17,-3),14,6,14,8,(10,-9),0
*00054,31,uct
2,14,8,(-8,-21),14,5,8,(8,21),1,8,(0,-21),2,8,(-7,21),1,0E0,2,
8,(1,-21),14,6,14,8,(8,-9),0
*00055,39,ucu
2,14,8,(-11,-21),14,5,8,(4,21),1,0FC,8,(1,-3),02E,8,(3,-1),020,
8,(3,1),022,8,(1,3),0F4,2,8,(4,-21),14,6,14,8,(11,-9),0
*00056,33,ucv
2,14,8,(-9,-21),14,5,8,(1,21),1,8,(8,-21),2,8,(8,21),1,
8,(-8,-21),2,8,(9,0),14,6,14,8,(9,-9),0
*00057,49,ucw
2,14,8,(-12,-21),14,5,8,(2,21),1,8,(5,-21),2,8,(5,21),1,
8,(-5,-21),2,8,(5,21),1,8,(5,-21),2,8,(5,21),1,8,(-5,-21),2,
8,(7,0),14,6,14,8,(12,-9),0
*00058,33,ucx
2,14,8,(-10,-21),14,5,8,(3,21),1,8,(14,-21),2,8,(0,21),1,
8,(-14,-21),2,8,(17,0),14,6,14,8,(10,-9),0
*00059,34,ucy
2,14,8,(-9,-21),14,5,8,(1,21),1,8,(8,-10),0BC,2,8,(8,21),1,
8,(-8,-10),2,8,(9,-11),14,6,14,8,(9,-9),0
*0005A,37,ucz
2,14,8,(-10,-21),14,5,8,(17,21),1,8,(-14,-21),2,8,(0,21),1,0E0,2,
8,(-14,-21),1,0E0,2,8,(3,0),14,6,14,8,(10,-9),0
*0005B,37,klb
2,14,8,(-7,-25),14,5,8,(4,25),1,8,(0,-32),2,8,(0,32),1,070,2,
8,(-7,-32),1,070,2,8,(3,7),14,6,14,8,(7,-16),0
*0005C,25,kbkslsh
2,14,8,(-11,-25),14,5,8,(2,25),1,8,(18,-32),2,8,(2,7),14,6,
```

```
14,8,(11,-16),0
*0005D,37,krb
2,14,8,(-7,-25),14,5,8,(9,25),1,8,(0,-32),2,8,(-7,32),1,070,2,
8,(-7,-32),1,070,2,8,(4,7),14,6,14,8,(7,-16),0
*0005E,28,kcaret
2,14,8,(-11,-25),14,5,8,(3,20),1,8,(8,5),8,(8,-5),2,8,(3,-20),
14,6,14,8,(11,-9),0
*0005F,21,kundrl
2,14,8,(-12,-14),14,5,02E,1,8,(20,0),2,022,14,6,14,8,(12,-11),0
*00060,29,krvap
2,14,8,(-5,-25),14,5,8,(4,24),1,01E,012,016,01A,02C,02D,01E,2,
8,(6,-19),14,6,14,8,(5,-9),0
*00061,55,lca
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(15,14),1,0EC,2,0B4,1,026,
027,038,029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,
8,(4,-3),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*00062,57,lcb
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,0B4,1,
022,021,030,02F,02E,8,(1,-3),02C,8,(-1,-3),02A,029,038,027,026,2,
8,(15,-3),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*00063,39,lcc
2,14,8,(-9,-14),14,5,8,(15,11),1,026,027,038,029,02A,8,(-1,-3),
02C,8,(1,-3),02E,02F,030,021,022,2,8,(3,-3),14,6,14,8,(9,-9),0
*00064,57,lcd
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(15,21),1,8,(0,-21),2,0B4,
1,026,027,038,029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,
2,8,(4,-3),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*00065,42,lce
2,14,8,(-9,-14),14,5,8,(3,8),1,0C0,024,025,016,027,038,029,02A,
8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,8,(3,-3),14,6,
14,8,(9,-9),0
*00066,36,lcf
2,14,8,(-6,-21),14,5,8,(10,21),1,028,029,8,(-1,-3),8,(0,-17),2,
8,(-3,14),1,070,2,8,(3,-14),14,6,14,8,(6,-9),0
*00067,66,lcg
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(15,14),1,8,(0,-16),
8,(-1,-3),01A,029,038,027,2,8,(9,17),1,026,027,038,029,02A,
8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,8,(4,-3),14,6,14,3,
2,14,8,(19,-32),14,4,2,0
*00068,48,lch
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,0A4,1,
032,021,030,02F,8,(1,-3),0AC,2,8,(4,0),14,6,14,3,2,14,8,(19,-18),
14,4,2,0
*00069,32,lci
2,14,8,(-4,-21),14,5,8,(3,20),1,01E,012,016,01A,2,8,(1,-7),1,0DC,
2,8,(4,0),14,6,14,8,(4,-9),0
*0006A,39,lcj
2,14,8,(-5,-21),14,5,8,(5,20),1,01E,012,016,01A,2,8,(1,-7),1,
8,(0,-16),8,(-1,-3),029,028,2,8,(9,7),14,6,14,8,(5,-16),0
*0006B,49,lck
2,14,3,2,14,8,(-17,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(10,14),1,0AA,2,042,1,8,(7,-8),2,8,(2,0),14,6,14,3,2,
14,8,(17,-18),14,4,2,0
*0006C,25,lcl
2,14,8,(-4,-21),14,5,8,(4,21),1,8,(0,-21),2,8,(4,0),14,6,
14,8,(4,-9),0
```

```
*0006D,45,lcm
2,14,8,(-15,-14),14,5,8,(4,14),1,0EC,2,0A4,1,032,021,030,02F,
8,(1,-3),0AC,2,0A4,1,032,021,030,02F,8,(1,-3),0AC,2,8,(4,0),14,6,
14,8,(15,-9),0
*0006E,46,lcn
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(4,14),1,0EC,2,0A4,1,032,
021,030,02F,8,(1,-3),0AC,2,8,(4,0),14,6,14,3,2,14,8,(19,-18),
14,4,2,0
*0006F,58,lco
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(8,14),1,029,02A,8,(-1,-3),
02C,8,(1,-3),02E,02F,030,021,022,8,(1,3),024,8,(-1,3),026,027,
038,2,8,(11,-14),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*00070,59,lcp
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(4,14),1,8,(0,-21),2,
8,(0,18),1,022,021,030,02F,02E,8,(1,-3),02C,8,(-1,-3),02A,029,
038,027,026,2,8,(15,-3),14,6,14,3,2,14,8,(19,-32),14,4,2,0
*00071,59,lcq
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(15,14),1,8,(0,-21),2,
8,(0,18),1,026,027,038,029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,
030,021,022,2,8,(4,-3),14,6,14,3,2,14,8,(19,-32),14,4,2,0
*00072,44,lcr
2,14,3,2,14,8,(-13,-28),14,4,2,14,5,8,(4,14),1,0EC,2,084,1,
8,(1,3),022,021,030,2,8,(1,-14),14,6,14,3,2,14,8,(13,-18),14,4,2,
0
*00073,60,lcs
2,14,3,2,14,8,(-17,-28),14,4,2,14,5,8,(14,11),1,025,8,(-3,1),038,
8,(-3,-1),02B,02D,02F,8,(5,-1),02F,02D,01C,02B,8,(-3,-1),038,
8,(-3,1),025,2,8,(14,-3),14,6,14,3,2,14,8,(17,-18),14,4,2,0
*00074,36,lct
2,14,8,(-6,-21),14,5,8,(5,21),1,8,(0,-17),8,(1,-3),02F,020,2,
8,(-8,14),1,070,2,8,(3,-14),14,6,14,8,(6,-9),0
*00075,46,lcu
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(4,14),1,0AC,8,(1,-3),02F,
030,021,032,2,0A4,1,0EC,2,8,(4,0),14,6,14,3,2,14,8,(19,-18),14,4,
2,0
*00076,33,lcv
2,14,8,(-8,-14),14,5,8,(2,14),1,8,(6,-14),2,8,(6,14),1,
8,(-6,-14),2,8,(8,0),14,6,14,8,(8,-9),0
*00077,49,lcw
2,14,8,(-11,-14),14,5,8,(3,14),1,8,(4,-14),2,8,(4,14),1,
8,(-4,-14),2,8,(4,14),1,8,(4,-14),2,8,(4,14),1,8,(-4,-14),2,
8,(7,0),14,6,14,8,(11,-9),0
*00078,43,lcx
2,14,3,2,14,8,(-17,-28),14,4,2,14,5,8,(3,14),1,8,(11,-14),2,0E4,
1,8,(-11,-14),2,8,(14,0),14,6,14,3,2,14,8,(17,-18),14,4,2,0
*00079,37,lcy
2,14,8,(-8,-14),14,5,8,(2,14),1,8,(6,-14),2,8,(6,14),1,
8,(-6,-14),04B,02A,029,018,2,8,(15,7),14,6,14,8,(8,-16),0
*0007A,47,lcz
2,14,3,2,14,8,(-17,-28),14,4,2,14,5,8,(14,14),1,8,(-11,-14),2,
0E4,1,0B0,2,8,(-11,-14),1,0B0,2,8,(3,0),14,6,14,3,2,
14,8,(17,-18),14,4,2,0
*0007B,54,klbr
2,14,3,2,14,8,(-13,-50),14,4,2,14,5,8,(9,25),1,029,01A,02B,02C,
02D,01E,02D,02C,02A,029,02F,02E,02C,02B,01A,02B,02C,02D,01E,02F,
2,8,(5,7),14,6,14,3,2,14,8,(13,-32),14,4,2,0
```

```
*0007C,25,kvbar
2,14,8,(-4,-25),14,5,8,(4,25),1,8,(0,-32),2,8,(4,7),14,6,
14,8,(4,-16),0
*0007D,54,krbr
2,14,3,2,14,8,(-15,-50),14,4,2,14,5,8,(5,25),1,02F,01E,02D,02C,
02B,01A,02B,02C,02E,02F,029,02A,02C,02D,01E,02D,02C,02B,01A,029,
2,8,(9,7),14,6,14,3,2,14,8,(15,-32),14,4,2,0
*0007E,37,ktlde
2,14,8,(-13,-14),14,5,8,(4,6),1,024,8,(1,3),021,020,02F,8,(4,-3),
02F,020,021,023,024,2,8,(4,-12),14,6,14,8,(13,-9),0
*00080,4,keuroRef
7,020AC,0
*000A0,9,spc
2,8,(21,0),14,8,(-21,-30),0
*000A1,28,kiexc
2,14,8,(-5,-21),14,5,050,1,0E4,2,054,1,012,016,01A,01E,2,
8,(8,-19),14,6,14,8,(5,-9),0
*000A2,43,kcent
2,14,8,(-9,-17),14,5,03E,1,8,(12,20),2,06C,1,026,027,038,029,02A,
8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,8,(3,-3),14,6,
14,8,(9,-9),0
*000A3,37,kpound
2,14,8,(-10,-21),14,5,8,(12,18),1,025,027,029,02B,0FC,03A,0E0,2,
8,(-8,10),1,068,014,060,2,8,(9,-11),14,6,14,8,(10,-9),0
*000A5,44,kyen
2,14,8,(-12,-21),14,5,8,(1,21),1,8,(8,-10),0BC,2,8,(8,21),1,
8,(-8,-10),2,078,1,0E0,2,8,(-14,-3),1,0E0,2,8,(6,-8),14,6,
14,8,(12,-9),0
*000A7,78,kpar
2,14,8,(-10,-25),14,5,060,1,012,016,01A,01C,02D,01E,02F,020,021,
012,023,014,025,016,8,(-8,4),016,025,014,023,012,021,010,
8,(8,-4),2,094,028,1,01A,01E,012,014,025,016,027,028,029,01A,02B,
01C,02D,01E,8,(8,-4),01E,02D,01C,02B,01A,029,018,8,(-8,4),2,
8,(16,-9),14,6,14,8,(10,-13),0
*000AA,51,lcau
2,14,8,-7,-21,14,5,8,4,14,3,2,1,0A0,2,054,1,02A,029,028,027,016,
8,-1,3,024,8,1,3,012,021,020,02F,02E,2,034,1,0CC,2,4,2,8,4,-15,
14,6,14,8,7,-9,0
*000AB,25,kfrew
2,14,8,(-9,-14),14,5,0A0,1,076,072,2,050,1,07A,07E,2,030,14,6,
14,8,(9,-9),0
*000B0,25,kdeg
2,14,8,(-3,-21),14,5,8,(1,19),1,10,(2,64),2,8,(8,-19),14,6,
14,8,(3,-9),0
*000B1,39,kpls-min
2,14,8,(-13,-21),14,5,8,(13,21),1,8,(0,-18),2,096,1,8,(18,0),2,
8,(-18,-11),1,8,(18,0),2,8,(4,-1),14,6,14,8,(13,-9),0
*000B5,48,kmicro
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,07C,1,022,8,(3,19),0AC,
8,(1,-3),02F,030,021,032,2,0A4,1,0EC,2,8,(4,0),14,6,14,3,2,
14,8,(19,-32),14,4,2,0
*000BA,56,lcou
2,14,8,-7,-21,14,5,8,4,14,3,2,1,0A0,2,8,-4,14,1,028,029,01A,8,-1,
-3,02C,8,1,-3,01E,02F,020,021,012,8,1,3,024,8,-1,3,016,027,2,4,2,
8,6,-21,14,6,14,8,7,-9,0
*000BB,25,kffrw
```

```
2,14,8,(-9,-14),14,5,030,1,072,076,2,050,1,07E,07A,2,0A0,14,6,
14,8,(9,-9),0
*000BC,43,kquart
2,14,8,(-14,-25),14,5,8,(4,21),1,021,022,0EC,2,8,(-2,-14),1,
8,(16,29),2,8,(2,-23),1,0A8,8,(7,9),0EC,2,8,(7,3),14,6,
14,8,(14,-13),0
*000BD,50,khalf
2,14,8,(-14,-25),14,5,8,(4,21),1,021,022,0EC,2,8,(-2,-14),1,
8,(16,29),2,8,(-6,-18),1,014,023,021,020,02F,02D,01C,02B,
8,(-7,-8),080,2,8,(4,3),14,6,14,8,(14,-13),0
*000BF,47,kiqm
2,14,8,(-9,-21),14,5,8,(13,4),1,016,012,01E,01C,02B,01A,029,038,
8,(-3,1),025,024,023,012,021,022,034,2,054,1,012,016,01A,01E,2,
8,(8,-19),14,6,14,8,(9,-9),0
*000C0,43,uc^
2,14,8,(-9,-25),14,5,8,(9,23),1,047,2,04E,1,8,(-8,-21),2,
8,(8,21),1,8,(8,-21),2,8,(-13,7),1,0A0,2,8,(4,-7),14,6,
14,8,(9,-9),0
*000C1,43,uc^
2,14,8,(-9,-25),14,5,8,(9,23),1,041,2,04A,1,8,(-8,-21),2,
8,(8,21),1,8,(8,-21),2,8,(-13,7),1,0A0,2,8,(4,-7),14,6,
14,8,(9,-9),0
*000C2,44,uc
2,14,8,(-9,-25),14,5,8,(5,23),1,041,04F,2,049,1,8,(-8,-21),2,
8,(8,21),1,8,(8,-21),2,8,(-13,7),1,0A0,2,8,(4,-7),14,6,
14,8,(9,-9),0
*000C3,55,uc^
2,14,8,(-9,-25),14,5,8,(4,22),5,1,023,10,(2,-50),01E,10,(2,82),
023,2,6,8,(5,-1),1,8,(-8,-21),2,8,(8,21),1,8,(8,-21),2,8,(-13,7),
1,0A0,2,8,(4,-7),14,6,14,8,(9,-9),0
*000C4,53,uc„
2,14,8,(-9,-25),14,5,8,(4,24),1,01E,012,016,01A,2,080,1,01E,012,
016,01A,2,03A,1,8,(-8,-21),2,8,(8,21),1,8,(8,-21),2,8,(-13,7),1,
0A0,2,8,(4,-7),14,6,14,8,(9,-9),0
*000C5,45,uc^
2,14,8,(-9,-25),14,5,8,(7,23),1,10,(2,64),2,02E,1,8,(-8,-21),2,
8,(8,21),1,8,(8,-21),2,8,(-13,7),1,0A0,2,8,(4,-7),14,6,
14,8,(9,-9),0
*000C6,45,uc^
2,14,8,(-9,-21),14,5,010,1,8,(8,21),8,(0,-21),080,2,8,(-8,7),1,
058,2,8,(5,4),1,050,2,8,(-5,10),1,080,2,8,(2,-21),14,6,
14,8,(9,-9),0
*000C7,65,uc‡
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,8,(18,16),1,025,026,027,048,
029,02A,02B,8,(-1,-3),05C,8,(1,-3),02D,02E,02F,040,021,022,023,2,
8,(-9,-11),1,01E,030,012,024,016,028,034,2,0A0,14,6,14,3,2,
14,8,(21,-32),14,4,2,0
*000C8,53,uc^
2,14,3,2,14,8,(-19,-50),14,4,2,14,5,8,(6,25),1,8,(9,-4),2,
8,(2,-2),1,0D8,8,(0,-19),0D0,2,8,(-13,10),1,080,2,8,(7,-10),14,6,
14,3,2,14,8,(19,-18),14,4,2,0
*000C9,53,uc^
2,14,3,2,14,8,(-19,-50),14,4,2,14,5,8,(6,21),1,8,(9,4),2,
8,(2,-6),1,0D8,8,(0,-19),0D0,2,8,(-13,10),1,080,2,8,(7,-10),14,6,
14,3,2,14,8,(19,-18),14,4,2,0
*000CA,53,uc^
```

```
2,14,3,2,14,8,(-19,-50),14,4,2,14,5,8,(6,23),1,041,010,04F,2,
8,(2,-2),1,0D8,8,(0,-21),0D0,2,8,(-13,11),1,080,2,8,(7,-11),14,6,
14,3,2,14,8,(19,-18),14,4,2,0
*000CB,61,uc^
2,14,3,2,14,8,(-19,-50),14,4,2,14,5,8,(6,24),1,01E,012,016,01A,2,
070,1,01E,012,016,01A,2,8,(4,-3),1,0D8,8,(0,-21),0D0,2,
8,(-13,11),1,080,2,8,(7,-11),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000CC,29,uc^
2,14,8,(-4,-25),14,5,8,(4,23),1,026,2,04D,1,8,(0,-21),2,8,(4,0),
14,6,14,8,(4,-9),0
*000CD,29,uc^
2,14,8,(-4,-25),14,5,8,(4,23),1,022,2,04B,1,8,(0,-21),2,8,(4,0),
14,6,14,8,(4,-9),0
*000CE,30,uc^
2,14,8,(-4,-25),14,5,8,(2,23),1,022,02E,2,02A,1,8,(0,-21),2,
8,(4,0),14,6,14,8,(4,-9),0
*000CF,41,uc^
2,14,8,(-4,-25),14,5,8,(1,24),1,01E,012,016,01A,2,040,1,01E,012,
016,01A,2,8,(-1,-3),1,8,(0,-21),2,8,(4,0),14,6,14,8,(4,-9),0
*000D1,41,uc
2,14,8,(-11,-25),14,5,040,1,8,(0,19),8,(14,-19),8,(0,19),2,
8,(-13,3),1,032,010,8,(4,-3),010,032,2,8,(5,-25),14,6,
14,8,(11,-9),0
*000D2,44,uc^
2,14,8,(-11,-25),14,5,8,(6,25),1,08F,2,8,(-6,-2),1,029,02A,04B,
05C,04D,02E,02F,040,021,022,043,054,045,026,027,048,2,8,(13,-19),
14,6,14,8,(11,-9),0
*000D3,42,uc^
2,14,8,(-11,-25),14,5,8,(6,21),1,081,2,06A,1,029,02A,04B,05C,04D,
02E,02F,040,021,022,043,054,045,026,027,048,2,8,(13,-19),14,6,
14,8,(11,-9),0
*000D4,57,uc^
2,14,8,(-11,-25),14,5,8,(6,23),1,041,04F,2,8,(-6,-2),1,029,02A,
02B,8,(-1,-3),05C,8,(1,-3),02D,02E,02F,040,021,022,023,8,(1,3),
054,8,(-1,3),025,026,027,048,2,8,(13,-21),14,6,14,8,(11,-9),0
*000D5,66,uc^
2,14,8,(-11,-25),14,5,8,(6,22),5,1,023,10,(2,-50),01E,10,(2,82),
023,2,6,8,(3,-1),1,029,02A,02B,8,(-1,-3),05C,8,(1,-3),02D,02E,
02F,040,021,022,023,8,(1,3),054,8,(-1,3),025,026,027,048,2,
8,(13,-21),14,6,14,8,(11,-9),0
*000D6,66,uc^
2,14,8,(-11,-25),14,5,8,(6,24),1,01E,012,016,01A,2,080,1,01E,012,
016,01A,2,8,(-5,-3),1,029,02A,02B,8,(-1,-3),05C,8,(1,-3),02D,02E,
02F,040,021,022,023,8,(1,3),054,8,(-1,3),025,026,027,048,2,
8,(13,-21),14,6,14,8,(11,-9),0
*000D8,54,uc›
2,14,8,(-11,-21),8,(9,21),1,029,02A,02B,8,(-1,-3),05C,8,(1,-3),
02D,02E,02F,040,021,022,023,8,(1,3),054,8,(-1,3),025,026,027,048,
2,8,(-6,-21),1,8,(16,21),2,8,(3,-21),14,8,(-11,-9),0
*000D9,43,uc^
2,14,8,(-11,-25),14,5,8,(15,21),1,087,2,06B,1,0DC,8,(1,-3),02E,
8,(3,-1),020,8,(3,1),022,8,(1,3),0D4,2,8,(4,-19),14,6,
14,8,(11,-9),0
*000DA,45,uc^
2,14,8,(-11,-25),14,5,8,(15,25),1,089,2,8,(-3,-2),1,0DC,8,(1,-3),
02E,8,(3,-1),020,8,(3,1),022,8,(1,3),0D4,2,8,(4,-19),14,6,
```

```
14,8,(11,-9),0
*000DB,46,uc^
2,14,8,(-11,-25),14,5,8,(15,23),1,047,049,2,8,(-3,-2),1,0FC,
8,(1,-3),02E,8,(3,-1),020,8,(3,1),022,8,(1,3),0F4,2,8,(4,-21),
14,6,14,8,(11,-9),0
*000DC,55,uc^
2,14,8,(-11,-25),14,5,8,(14,24),1,01E,012,016,01A,2,088,1,01E,
012,016,01A,2,8,(-2,-3),1,0FC,8,(1,-3),02E,8,(3,-1),020,8,(3,1),
022,8,(1,3),0F4,2,8,(4,-21),14,6,14,8,(11,-9),0
*000DD,38,uc^
2,14,8,(-9,-25),14,5,8,(13,25),1,089,2,049,1,8,(8,-9),0AC,2,
8,(8,19),1,8,(-8,-9),2,8,(9,-10),14,6,14,8,(9,-9),0
*000DF,53,kgers
2,14,8,(-9,-21),14,5,030,1,012,8,(0,16),023,012,021,020,02F,01E,
02D,02C,02B,01A,029,028,2,020,1,8,(3,-1),01E,02D,03C,02B,01A,029,
028,027,016,012,01E,2,8,(10,-2),14,6,14,8,(9,-9),0
*000E0,63,lc…
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,21),1,8,(8,-4),2,
8,(2,-3),1,0EC,2,0B4,1,026,027,038,029,02A,8,(-1,-3),02C,
8,(1,-3),02E,02F,030,021,022,2,8,(4,-3),14,6,14,3,2,
14,8,(19,-18),14,4,2,0
*000E1,63,lc
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,17),1,8,(8,4),2,
8,(2,-7),1,0EC,2,0B4,1,026,027,038,029,02A,8,(-1,-3),02C,
8,(1,-3),02E,02F,030,021,022,2,8,(4,-3),14,6,14,3,2,
14,8,(19,-18),14,4,2,0
*000E2,64,lc^
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,18),1,8,(4,3),8,(4,-3),
2,04D,1,0EC,2,0B4,1,026,027,038,029,02A,8,(-1,-3),02C,8,(1,-3),
02E,02F,030,021,022,2,8,(4,-3),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000E3,63,lcƒ
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,18),1,032,010,03E,010,
032,2,07C,1,0EC,2,0B4,1,026,027,038,029,02A,8,(-1,-3),02C,
8,(1,-3),02E,02F,030,021,022,2,8,(4,-3),14,6,14,3,2,
14,8,(19,-18),14,4,2,0
*000E4,71,lc„
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,20),1,01E,012,016,01A,2,
090,1,01E,012,016,01A,2,8,(2,-6),1,0EC,2,0B4,1,026,027,038,029,
02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,8,(4,-3),14,6,
14,3,2,14,8,(19,-18),14,4,2,0
*000E5,63,lc†
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(7,19),1,10,(2,64),2,
8,(8,-5),1,0EC,2,0B4,1,026,027,038,029,02A,8,(-1,-3),02C,
8,(1,-3),02E,02F,030,021,022,2,8,(4,-3),14,6,14,3,2,
14,8,(19,-18),14,4,2,0
*000E6,51,lc
2,14,8,(-10,-14),14,5,8,(10,8),1,070,014,8,(-1,3),026,028,02A,
026,028,02A,8,(-1,-3),04C,8,(1,-3),02E,020,022,02E,020,021,023,2,
8,(-7,11),1,0EC,2,0A0,14,6,14,8,(10,-9),0
*000E7,49,lc‡
2,14,8,(-9,-14),14,5,8,(15,11),1,026,027,038,029,02A,8,(-1,-3),
02C,8,(1,-3),02E,02F,030,021,022,2,8,(-8,-9),1,01E,030,012,024,
016,028,034,2,090,14,6,14,8,(9,-16),0
*000E8,48,lcŠ
2,14,8,(-9,-21),14,5,8,(5,21),1,08F,2,8,(-10,-9),1,0C0,024,025,
016,027,038,029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,
```

```
8,(3,-3),14,6,14,8,(9,-9),0
*000E9,48,lc,
2,14,8,(-9,-21),14,5,8,(5,17),1,081,2,8,(-10,-13),1,0C0,024,025,
016,027,038,029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,
8,(3,-3),14,6,14,8,(9,-9),0
*000EA,51,lc^
2,14,8,(-9,-21),14,5,8,(5,18),1,8,(4,3),8,(4,-3),2,0AA,1,0C0,024,
025,016,027,038,029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,
022,2,8,(3,-3),14,6,14,8,(9,-9),0
*000EB,58,lc%
2,14,8,(-9,-21),14,5,8,(4,20),1,01E,012,016,01A,2,080,1,01E,012,
016,01A,2,8,(-9,-12),1,0C0,024,025,016,027,038,029,02A,8,(-1,-3),
02C,8,(1,-3),02E,02F,030,021,022,2,8,(3,-3),14,6,14,8,(9,-9),0
*000EC,27,lc_
2,14,8,(-7,-21),14,5,8,(3,21),1,08F,2,04A,1,0DC,2,8,(4,0),14,6,
14,8,(7,-9),0
*000ED,27,lc
2,14,8,(-7,-21),14,5,8,(3,17),1,081,2,08B,1,0DC,2,8,(4,0),14,6,
14,8,(7,-9),0
*000EE,34,lcŒ
2,14,8,(-7,-21),14,5,8,(3,18),1,8,(4,3),8,(4,-3),2,8,(-4,-5),1,
0DC,2,8,(4,0),14,6,14,8,(7,-9),0
*000EF,39,lc‹
2,14,8,(-7,-21),14,5,8,(3,20),1,01E,012,016,01A,2,060,1,01E,012,
016,01A,2,8,(-2,-7),1,0DC,2,8,(4,0),14,6,14,8,(7,-9),0
*000F1,56,lc
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,18),1,032,010,03E,010,
032,2,8,(-11,-7),1,0EC,2,0A4,1,032,021,030,02F,8,(1,-3),0AC,2,
8,(4,0),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000F2,64,lc•
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,21),1,8,(9,-4),2,069,1,
029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,8,(1,3),024,
8,(-1,3),026,027,038,2,8,(11,-14),14,6,14,3,2,14,8,(19,-18),14,4,
2,0
*000F3,66,lc
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,17),1,8,(9,4),2,
8,(-6,-7),1,029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,
8,(1,3),024,8,(-1,3),026,027,038,2,8,(11,-14),14,6,14,3,2,
14,8,(19,-18),14,4,2,0
*000F4,73,lc‟
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,18),3,2,1,8,(9,6),
8,(9,-6),2,4,2,8,(-6,-4),1,029,02A,8,(-1,-3),02C,8,(1,-3),02E,
02F,030,021,022,8,(1,3),024,8,(-1,3),026,027,038,2,8,(11,-14),
14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000F5,68,lc^
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,18),1,032,010,03E,010,
032,2,8,(-7,-7),1,029,02A,8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,
022,8,(1,3),024,8,(-1,3),026,027,038,2,8,(11,-14),14,6,14,3,2,
14,8,(19,-18),14,4,2,0
*000F6,74,lc”
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,20),1,01E,012,016,01A,2,
090,1,01E,012,016,01A,2,8,(-5,-6),1,029,02A,8,(-1,-3),02C,
8,(1,-3),02E,02F,030,021,022,8,(1,3),024,8,(-1,3),026,027,038,2,
8,(11,-14),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000F7,41,kto
2,14,8,(-9,-14),14,5,8,(8,13),1,01E,012,016,01A,2,8,(-5,-6),1,
```

```
0C0,2,8,(-7,-6),1,01E,012,016,01A,2,8,(10,-1),14,6,14,8,(9,-9),0
*000F8,24,lc›
7,06F,2,8,(-3,14),14,8,(9,9),1,8,(-13,-14),2,8,(17,0),
14,8,(-10,-9),0
*000F9,54,lc—
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,21),1,8,(9,-4),2,
8,(-10,-3),1,0AC,8,(1,-3),02F,030,021,032,2,0A4,1,0EC,2,8,(4,0),
14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000FA,54,lc
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,17),1,8,(9,4),2,
8,(-10,-7),1,0AC,8,(1,-3),02F,030,021,032,2,0A4,1,0EC,2,8,(4,0),
14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000FB,61,lc–
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(5,18),3,2,1,8,(9,6),
8,(9,-6),4,2,2,8,(-10,-4),1,0AC,8,(1,-3),02F,030,021,032,2,0A4,1,
0EC,2,8,(4,0),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000FC,62,lc_
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,20),1,01E,012,016,01A,2,
090,1,01E,012,016,01A,2,8,(-9,-6),1,0AC,8,(1,-3),02F,030,021,032,
2,0A4,1,0EC,2,8,(4,0),14,6,14,3,2,14,8,(19,-18),14,4,2,0
*000FD,43,lc^
2,14,8,(-8,-14),14,5,8,(2,14),5,032,1,8,(9,4),6,8,(6,-14),2,
8,(6,14),1,8,(-6,-14),04B,02A,029,018,2,8,(15,7),14,6,
14,8,(8,-16),0
*000FF,53,lc˜
2,14,8,(-8,-21),14,5,8,(3,20),1,01E,012,016,01A,2,080,1,01E,012,
016,01A,2,8,(-9,-6),1,8,(6,-14),2,8,(6,14),1,8,(-6,-14),04B,02A,
029,018,2,8,(15,7),14,6,14,8,(8,-16),0
*00104,50,c164
2,14,8,(-9,-21),14,5,8,(9,21),1,8,(-8,-21),2,8,(8,21),1,
8,(8,-21),3,2,10,(5,36),1,10,(5,-100),4,2,2,8,(-13,7),1,0A0,2,
8,(4,-7),14,6,14,8,(9,-9),0
*00105,66,c165
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(15,14),1,0EC,3,2,10,
(5,36),2,10,(5,-100),4,2,2,0B4,1,026,027,038,029,02A,8,(-1,-3),
02C,8,(1,-3),02E,02F,030,021,022,2,8,(4,-3),14,6,14,3,2,
14,8,(19,-18),14,4,2,0
*00106,71,c143
2,14,3,2,14,8,(-21,-42),14,4,2,14,5,3,2,2,8,(18,48),1,8,(8,6),2,
8,(-26,-54),4,2,8,(18,16),1,025,026,027,048,029,02A,02B,
8,(-1,-3),05C,8,(1,-3),02D,02E,02F,040,021,022,023,2,8,(3,-5),
14,6,14,3,2,14,8,(21,-18),14,4,2,0
*00107,54,c134
2,14,8,(-9,-14),14,5,8,(15,11),3,2,8,(-15,10),1,8,(6,4),2,
8,(9,-14),4,2,1,026,027,038,029,02A,8,(-1,-3),02C,8,(1,-3),02E,
02F,030,021,022,2,8,(3,-3),14,6,14,8,(9,-9),0
*0010C,49,c172
2,14,8,(-11,-27),8,(18,16),1,025,026,027,048,029,02A,02B,
8,(-1,-3),05C,8,(1,-3),02D,02E,02F,040,021,022,023,2,8,(-7,18),1,
042,2,04A,1,046,2,8,(14,-27),14,8,(-10,-9),0
*0010D,41,c159
2,14,8,(-9,-20),0F0,0B4,1,026,027,038,029,02A,8,(-1,-3),02C,
8,(1,-3),02E,02F,030,021,022,2,068,0D4,1,042,2,04A,1,046,2,
8,(13,-20),14,09A,0
*0010E,53,c210
2,14,8,(-11,-27),8,(4,21),1,0FC,06C,2,0F4,064,1,070,8,(3,-1),02E,
```

```
02D,8,(1,-3),05C,8,(-1,-3),02B,02A,8,(-3,-1),078,2,8,(7,23),1,
042,2,04A,1,046,2,8,(14,-27),14,8,(-10,-9),0
*0010F,46,c212
2,14,8,(-10,-21),8,(15,21),1,0FC,06C,2,0B4,1,026,027,038,029,02A,
8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,8,(4,18),1,
8,(-3,-4),2,8,(3,-17),14,09A,0
*00118,66,c168
2,14,3,2,14,8,(-19,-42),14,4,2,14,5,8,(4,21),1,8,(0,-21),2,
8,(0,21),1,0D0,2,8,(-13,-10),1,080,2,8,(-8,-11),1,0D0,3,2,10,
(5,36),1,10,(5,-100),4,2,2,8,(2,0),14,6,14,3,2,14,8,(19,-18),
14,4,2,0
*00119,66,c169
2,14,8,(-9,-14),14,5,8,(3,8),1,0C0,024,025,016,027,038,029,02A,
8,(-1,-3),02C,8,(1,-3),02E,02F,030,3,2,10,(5,36),2,10,(5,-100),1,
4,2,021,022,2,3,2,10,(5,36),2,10,(5,-100),4,2,2,8,(3,-3),14,6,
14,8,(9,-9),0
*0011A,43,c183
2,14,8,(-10,-27),8,(4,21),1,0FC,06C,2,0F4,064,1,0D0,2,0D8,0AC,1,
080,2,088,0BC,1,0D0,2,8,(-7,23),1,042,2,04A,1,046,2,8,(13,-27),
14,09A,0
*0011B,44,c216
2,14,8,(-9,-20),030,084,1,0C0,024,025,016,027,038,029,02A,
8,(-1,-3),02C,8,(1,-3),02E,02F,030,021,022,2,068,0D4,1,042,2,04A,
1,046,2,8,(13,-20),14,09A,0
*00141,51,c157
2,14,3,2,14,8,(-17,-42),14,4,2,14,5,8,(3,21),1,8,(0,-21),2,
8,(-1,11),1,8,(7,8),2,8,(-6,-19),1,0C0,2,8,(3,0),14,6,14,3,2,
14,8,(17,-18),14,4,2,0
*00142,33,c136
2,14,8,(-4,-21),14,5,8,(5,21),1,8,(0,-21),2,8,(-1,10),1,8,(4,6),
2,8,(3,-16),14,6,14,8,(4,-9),0
*00143,52,c227
2,14,8,(-11,-21),14,5,8,(4,21),8,(5,3),1,8,(4,3),2,8,(-9,-6),1,
8,(0,-21),2,8,(0,21),1,8,(14,-21),2,8,(0,21),1,8,(0,-21),2,
8,(4,0),14,6,14,8,(11,-9),0
*00144,57,c228
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(4,14),8,(4,2),1,8,(3,2),2,
8,(-7,-4),1,0EC,2,0A4,1,032,021,030,02F,8,(1,-3),0AC,2,8,(4,0),
14,6,14,3,2,14,8,(19,-18),14,4,2,0
*00147,38,c213
2,14,8,(-11,-27),8,(4,21),1,0FC,06C,2,0F4,064,1,8,(14,-21),0F4,
064,2,078,024,1,042,2,04A,1,046,2,8,(15,-27),14,8,(-11,-9),0
*00148,37,c229
2,14,8,(-10,-20),040,0E4,1,0EC,2,0A4,1,032,021,030,02F,8,(1,-3),
0AC,2,8,(-5,16),1,042,2,04A,1,046,2,8,(13,-20),14,09A,0
*00150,58,c138
2,14,8,(-11,-21),14,5,8,(9,21),1,029,02A,02B,8,(-1,-3),05C,
8,(1,-3),02D,02E,02F,040,021,022,023,8,(1,3),054,8,(-1,3),025,
026,027,048,2,034,1,044,2,040,1,04C,2,8,(9,-24),14,6,
14,8,(11,-9),0
*00151,68,c139
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(8,14),1,029,02A,8,(-1,-3),
02C,8,(1,-3),02E,02F,030,021,022,8,(1,3),024,8,(-1,3),026,027,
038,2,8,(4,4),1,044,2,058,1,04C,2,8,(12,-18),14,6,14,3,2,
14,8,(19,-18),14,4,2,0
*00158,53,c252
```

```
2,14,8,(-11,-27),8,(4,21),1,0FC,06C,2,0F4,064,1,090,8,(3,-1),01E,
02D,02C,02B,01A,8,(-3,-1),098,2,070,1,8,(7,-11),2,8,(-7,23),1,
042,2,04A,1,046,2,8,(14,-27),14,8,(-10,-9),0
*00159,35,c253
2,14,8,(-7,-20),040,0E4,1,0EC,2,084,1,8,(1,3),022,021,030,2,058,
024,1,042,2,04A,1,046,2,0FD,05D,14,8,(-6,-9),0
*0015A,62,c151
2,14,8,(-10,-21),14,5,8,(17,18),8,(-9,6),1,8,(4,3),2,8,(5,-9),1,
026,8,(-3,1),048,8,(-3,-1),02A,02C,02D,01E,02F,8,(6,-2),02F,01E,
02D,03C,02A,8,(-3,-1),048,8,(-3,1),026,2,8,(17,-3),14,6,
14,8,(10,-9),0
*0015B,71,c152
2,14,3,2,14,8,(-17,-28),14,4,2,14,5,8,(14,11),8,(-7,5),1,8,(3,2),
2,8,(4,-7),1,025,8,(-3,1),038,8,(-3,-1),02B,02D,02F,8,(5,-1),02F,
02D,01C,02B,8,(-3,-1),038,8,(-3,1),025,2,8,(14,-3),14,6,14,3,2,
14,8,(17,-18),14,4,2,0
*00160,57,c230
2,14,8,(-10,-27),8,(17,18),1,026,8,(-3,1),048,8,(-3,-1),02A,02C,
02D,01E,02F,8,(6,-2),02F,01E,02D,03C,02A,8,(-3,-1),048,8,(-3,1),
026,2,8,(7,20),1,042,2,04A,1,046,2,8,(14,-27),14,8,(-10,-9),0
*00161,52,c231
2,14,8,(-9,-20),0E0,0B4,1,025,8,(-3,1),038,8,(-3,-1),02B,02D,02F,
8,(5,-1),02F,02D,01C,02B,8,(-3,-1),038,8,(-3,1),025,2,060,0D4,1,
042,2,04A,1,046,2,8,(12,-20),14,8,(-8,-9),0
*00164,35,c155
2,14,8,(-8,-27),8,(8,21),1,0FC,06C,2,8,(-7,21),1,0E0,2,078,024,1,
042,2,04A,1,046,2,8,(12,-27),14,8,(-8,-9),0
*00165,36,c156
2,14,8,(-6,-21),8,(5,21),1,0FC,02C,8,(1,-3),02F,020,2,088,0E4,1,
070,2,074,1,8,(-3,-4),2,8,(6,-17),14,8,(-6,-9),0
*0016E,45,c222
2,14,8,(-11,-27),8,(4,21),1,0FC,8,(1,-3),02E,8,(3,-1),020,
8,(3,1),022,8,(1,3),0F4,2,078,024,1,021,024,027,029,02C,02F,2,
8,(11,-23),14,8,(-11,-9),0
*0016F,38,c133
2,14,8,(-10,-20),040,0E4,1,0AC,8,(1,-3),02F,030,021,032,2,0A4,1,
0EC,2,8,(-5,16),1,021,024,027,029,02C,02F,2,8,(9,-16),14,09A,0
*00170,52,c235
2,14,8,(-11,-21),14,5,8,(4,21),1,0FC,8,(1,-3),02E,8,(3,-1),020,
8,(3,1),022,8,(1,3),0F4,2,058,034,1,044,2,048,1,04C,2,8,(9,-2),2,
8,(4,-22),14,6,14,8,(11,-9),0
*00171,60,c251
2,14,3,2,14,8,(-19,-28),14,4,2,14,5,8,(4,14),1,0AC,8,(1,-3),02F,
030,021,032,2,0A4,1,0EC,2,8,(-8,18),1,044,2,050,1,04C,2,
8,(3,-18),2,8,(4,0),14,6,14,3,2,14,8,(19,-20),14,4,2,0
*00179,45,c141
2,14,8,(-10,-21),14,5,8,(17,21),1,8,(-14,-21),2,8,(5,24),1,
8,(4,3),2,8,(-9,-6),1,0E0,2,8,(-14,-21),1,0E0,2,8,(3,0),14,6,
14,8,(10,-9),0
*0017A,58,c171
2,14,3,2,14,8,(-17,-28),14,4,2,14,5,8,(14,14),8,(-7,2),1,8,(3,2),
2,8,(4,-4),1,8,(-11,-14),2,0E4,1,0B0,2,8,(-11,-14),1,0B0,2,
8,(3,0),14,6,14,3,2,14,8,(17,-18),14,4,2,0
*0017B,45,c189
2,14,8,(-10,-21),14,5,8,(17,21),5,3,2,8,(-13,6),1,10,(2,96),4,2,
6,1,8,(-14,-21),0E0,2,8,(-14,21),1,0E0,2,8,(3,-21),14,6,
```

```
14,8,(10,-9),0
*0017C,59,c190
2,14,3,2,14,8,(-17,-28),14,4,2,14,5,8,(14,14),5,3,2,8,(-11,5),1,
10,(1,96),4,2,6,8,(-11,-14),2,0E4,1,0B0,2,8,(-11,-14),1,0B0,2,
8,(3,0),14,6,14,3,2,14,8,(17,-18),14,4,2,0
*0017D,42,c166
2,14,8,(-10,-27),8,(17,21),1,8,(-14,-21),2,0F4,064,1,0E0,2,
8,(-14,-21),1,0E0,2,8,(-7,23),1,042,2,04A,1,046,2,8,(14,-27),
14,8,(-10,-9),0
*0017E,38,c167
2,14,8,(-9,-20),0E2,1,8,(-11,-14),2,0E4,1,0B0,2,0B8,0EC,1,0B0,2,
8,(-5,16),1,042,2,04A,1,046,2,8,(12,-20),14,8,(-8,-9),0
*00410,38,_
2,14,8,(-9,-21),2,8,(4,7),1,9,(10,0),(0,0),2,8,(-13,-7),1,9,
(8,21),(8,-21),(0,0),2,8,(1,0),1,2,14,8,(-9,-9),0
*00411,46,_
2,14,8,(-10,-21),2,8,(15,21),1,9,(-11,0),(0,-21),(9,0),(3,1),
(1,1),(1,2),(0,3),(-1,2),(-1,1),(-3,1),(-9,0),(0,0),2,8,(17,-11),
1,2,14,8,(-11,-9),0
*00412,68,,
2,14,8,(-11,-21),2,8,(13,21),1,9,(-9,0),(0,-21),(9,0),(3,1),
(1,1),(1,2),(0,3),(-1,2),(-1,1),(-3,1),(-9,0),(0,0),2,8,(9,10),1,
9,(3,-1),(1,-1),(1,-2),(0,-2),(-1,-2),(-1,-1),(-3,-1),(0,0),2,
8,(9,-11),1,2,14,8,(-11,-9),0
*00413,28,ƒ
2,14,8,(-8,-21),2,8,(16,21),1,9,(-12,0),(0,-21),(0,0),2,8,(13,0),
1,2,14,8,(-9,-9),0
*00414,50,„
2,14,8,(-12,-21),2,8,(22,-4),1,9,(0,4),(-20,0),(0,-4),(0,0),2,
8,(2,4),1,9,(3,3),(1,2),(1,4),(0,12),(11,0),(0,-21),(0,0),2,
8,(4,0),1,2,14,8,(-12,-13),0
*00415,40,…
2,14,8,(-9,-21),2,8,(17,21),1,9,(-13,0),(0,-21),(13,0),(0,0),2,
8,(-1,11),1,9,(-12,0),(0,0),2,8,(15,-11),1,2,14,8,(-10,-9),0
*00416,66,†
2,14,8,(-12,-21),2,8,(1,0),1,9,(9,12),(0,0),2,8,(2,9),1,9,
(0,-21),(0,0),2,8,(11,21),1,9,(-11,-11),(0,0),2,8,(-11,11),1,9,
(11,-11),(0,0),2,8,(11,-10),1,9,(-9,12),(0,0),2,8,(10,-12),1,2,
14,8,(-12,-9),0
*00417,68,‡
2,14,8,(-9,-21),2,8,(3,20),1,9,(4,1),(3,0),(3,-1),(1,-2),(0,-2),
(-1,-2),(-3,-2),(3,-1),(2,-2),(1,-2),(0,-2),(-1,-2),(-2,-2),
(-3,-1),(-3,0),(-3,1),(-3,2),(0,0),2,8,(9,9),1,9,(-4,0),(0,0),2,
8,(12,-12),1,2,14,8,(-9,-9),0
*00418,30,ˆ
2,14,8,(-11,-21),2,8,(4,21),1,9,(0,-21),(14,21),(0,-21),(0,0),2,
8,(4,0),1,2,14,8,(-11,-9),0
*00419,48,‰
2,14,8,(-11,-21),2,8,(4,21),1,9,(0,-21),(14,21),(0,-21),(0,0),2,
8,(-11,26),1,9,(1,-1),(2,-1),(2,0),(2,1),(1,1),(0,0),2,8,(7,-26),
1,2,14,8,(-11,-9),0
*0041A,46,Š
2,14,8,(-10,-21),2,8,(18,0),1,9,(-9,12),(0,0),2,8,(9,9),1,9,
(-14,-14),(0,0),2,8,(0,14),1,9,(0,-21),(0,0),2,8,(16,0),1,2,
14,8,(-10,-9),0
*0041B,36,‹
```

```
2,14,8,(-10,-21),2,8,(16,0),1,9,(0,21),(-11,0),(0,-17),(-1,-2),
(-1,-1),(-2,-1),(0,0),2,8,(19,0),1,2,14,8,(-10,-9),0
*0041C,32,Œ
2,14,8,(-12,-21),2,8,(20,0),1,9,(0,21),(-8,-15),(-8,15),(0,-21),
(0,0),2,8,(20,0),1,2,14,8,(-12,-9),0
*0041D,46,_
2,14,8,(-11,-21),2,8,(4,21),1,9,(0,-21),(0,0),2,8,(0,11),1,9,
(14,0),(0,0),2,8,(0,-11),1,9,(0,21),(0,0),2,8,(4,-21),1,2,
14,8,(-11,-9),0
*0041E,64,_
2,14,8,(-11,-21),2,8,(9,21),1,9,(4,0),(2,-1),(2,-2),(1,-2),
(1,-3),(0,-5),(-1,-3),(-1,-2),(-2,-2),(-2,-1),(-4,0),(-2,1),
(-2,2),(-1,2),(-1,3),(0,5),(1,3),(1,2),(2,2),(2,1),(0,0),2,
8,(13,-21),1,2,14,8,(-11,-9),0
*0041F,30,_
2,14,8,(-11,-21),2,8,(4,0),1,9,(0,21),(14,0),(0,-21),(0,0),2,
8,(4,0),1,2,14,8,(-11,-9),0
*00420,44,_
2,14,8,(-10,-21),2,8,(4,0),1,9,(0,21),(9,0),(3,-1),(1,-1),(1,-2),
(0,-3),(-1,-2),(-1,-1),(-3,-1),(-9,0),(0,0),2,8,(16,-10),1,2,
14,8,(-10,-9),0
*00421,62,`
2,14,8,(-10,-21),2,8,(18,16),1,9,(-1,2),(-1,1),(-1,1),(-2,1),
(-4,0),(-2,-1),(-1,-1),(-1,-1),(-1,-2),(-1,-3),(0,-5),(1,-3),
(1,-2),(2,-2),(2,-1),(4,0),(2,1),(2,2),(1,2),(0,0),2,8,(2,-5),1,
2,14,8,(-10,-9),0
*00422,36,'
2,14,8,(-8,-21),2,8,(8,21),1,9,(0,-21),(0,0),2,8,(-7,21),1,9,
(14,0),(0,0),2,8,(1,-21),1,2,14,8,(-8,-9),0
*00423,44,"
2,14,8,(-8,-21),2,8,(15,21),1,9,(-7,-17),(-1,-2),(-1,-1),(-2,-1),
(-1,0),(0,0),2,8,(-2,21),1,9,(7,-17),(0,0),2,8,(8,-4),1,2,
14,8,(-8,-9),0
*00424,74,"
2,14,8,(-13,-21),2,8,(11,19),1,9,(4,0),(3,-1),(2,-1),(2,-2),
(1,-2),(0,-4),(-1,-2),(-2,-2),(-2,-1),(-3,-1),(-4,0),(-3,1),
(-2,1),(-2,2),(-1,2),(0,4),(1,2),(2,2),(2,1),(3,1),(0,0),2,
8,(2,2),1,9,(0,-21),(0,0),2,8,(13,0),1,2,14,8,(-13,-9),0
*00425,36,•
2,14,8,(-8,-21),2,8,(15,21),1,9,(-14,-21),(0,0),2,8,(0,21),1,9,
(14,-21),(0,0),2,8,(1,0),1,2,14,8,(-8,-9),0
*00426,40,-
2,14,8,(-11,-21),2,8,(4,21),1,9,(0,-21),(16,0),(0,-4),(0,0),2,
8,(-2,25),1,9,(0,-21),(0,0),2,8,(4,0),1,2,14,8,(-11,-13),0
*00427,44,—
2,14,8,(-10,-21),2,8,(3,21),1,9,(0,-8),(1,-3),(1,-1),(3,-1),
(9,0),(0,0),2,8,(0,13),1,9,(0,-21),(0,0),2,8,(4,0),1,2,
14,8,(-11,-9),0
*00428,40,˜
2,14,8,(-14,-21),2,8,(4,21),1,9,(0,-21),(21,0),(0,21),(0,0),2,
8,(-10,0),1,9,(0,-21),(0,0),2,8,(14,0),1,2,14,8,(-15,-9),0
*00429,50,™
2,14,8,(-14,-21),2,8,(4,21),1,9,(0,-21),(23,0),(0,-4),(0,0),2,
8,(-12,25),1,9,(0,-21),(0,0),2,8,(10,21),1,9,(0,-21),(0,0),2,
8,(4,0),1,2,14,8,(-15,-13),0
```

```
*0042A,48,š
2,14,8,(-10,-21),2,8,(1,21),1,9,(4,0),(0,-21),(8,0),(2,0),(2,1),
(1,1),(1,2),(0,4),(-1,2),(-1,1),(-2,1),(-10,0),(0,0),2,
8,(16,-12),1,2,14,8,(-11,-9),0
*0042B,54,›
2,14,8,(-12,-21),2,8,(4,21),1,9,(0,-21),(9,0),(2,1),(1,1),(1,2),
(0,4),(-1,2),(-1,1),(-2,1),(-9,0),(0,0),2,8,(16,9),1,9,(0,-21),
(0,0),2,8,(4,0),1,2,14,8,(-12,-9),0
*0042C,44,œ
2,14,8,(-10,-21),2,8,(4,21),1,9,(0,-21),(10,0),(2,1),(1,1),(1,2),
(0,4),(-1,2),(-1,1),(-2,1),(-10,0),(0,0),2,8,(16,-12),1,2,
14,8,(-10,-9),0
*0042D,64,_
2,14,8,(-9,-21),2,8,(6,11),1,9,(10,0),(0,0),2,8,(-14,7),1,9,
(2,2),(2,1),(4,0),(2,-1),(2,-2),(1,-2),(1,-3),(0,-5),(-1,-3),
(-1,-2),(-2,-2),(-2,-1),(-4,0),(-2,1),(-2,2),(0,0),2,8,(17,-3),1,
2,14,8,(-10,-9),0
*0042E,76,_
2,14,8,(-13,-21),2,8,(14,21),1,9,(4,0),(2,-1),(2,-3),(1,-4),
(0,-5),(-1,-4),(-2,-3),(-2,-1),(-4,0),(-2,1),(-2,3),(-1,4),(0,5),
(1,4),(2,3),(2,1),(0,0),2,8,(-5,-10),1,9,(-5,0),(0,0),2,8,(0,10),
1,9,(0,-21),(0,0),2,8,(22,0),1,2,14,8,(-13,-9),0
*0042F,54,Ÿ
2,14,8,(-10,-21),2,8,(2,0),1,9,(7,11),(0,0),2,8,(7,-11),1,9,
(0,21),(-9,0),(-3,-1),(-1,-1),(-1,-2),(0,-2),(1,-2),(1,-1),
(3,-1),(9,0),(0,0),2,8,(4,-11),1,2,14,8,(-10,-9),0
*00430,62,
2,14,8,(-9,-14),2,8,(15,3),1,9,(-2,-2),(-2,-1),(-3,0),(-2,1),
(-1,1),(-1,1),(-1,3),(0,2),(1,3),(2,2),(2,1),(3,0),(2,-1),(2,-2),
(0,0),2,8,(0,3),1,9,(0,-14),(0,0),2,8,(4,0),1,2,14,8,(-10,-9),0
*00431,64,
2,14,8,(-9,-21),2,8,(14,21),1,9,(-2,-1),(-5,-1),(-2,-1),(-1,-2),
(0,-12),(1,-2),(1,-1),(2,-1),(3,0),(2,1),(2,2),(1,3),(0,2),
(-1,3),(-1,1),(-1,1),(-2,1),(-3,0),(-2,-1),(-2,-2),(0,0),2,
8,(15,-11),1,2,14,8,(-10,-9),0
*00432,68,
2,14,8,(-9,-21),2,8,(4,11),1,9,(2,2),(2,1),(3,2),(1,2),(-1,2),
(-1,1),(-3,0),(-2,-1),(-1,-1),(0,-15),(1,-2),(1,-1),(2,-1),(3,0),
(2,1),(2,2),(1,3),(0,2),(-1,3),(-2,2),(-2,1),(-3,0),(0,0),2,
8,(11,-14),1,2,14,8,(-10,-9),0
*00433,56,
2,14,8,(-8,-14),2,8,(3,11),1,9,(1,2),(3,1),(3,0),(3,-1),(1,-2),
(-1,-2),(-2,-1),(-5,-1),(-2,-1),(-1,-2),(0,-1),(1,-2),(3,-1),
(3,0),(3,1),(1,2),(0,0),2,8,(3,-3),1,2,14,8,(-9,-9),0
*00434,72,
2,14,8,(-9,-14),2,8,(15,11),1,9,(-1,1),(-1,1),(-2,1),(-3,0),
(-2,-1),(-2,-2),(-1,-3),(0,-2),(1,-3),(2,-2),(2,-1),(3,0),(2,1),
(2,2),(0,0),2,8,(0,11),1,9,(0,-16),(-1,-3),(-1,-1),(-2,-1),
(-3,0),(-2,1),(0,0),2,8,(13,6),1,2,14,8,(-10,-16),0
*00435,56,
2,14,8,(-9,-14),2,8,(3,8),1,9,(12,0),(0,2),(-1,2),(-1,1),(-2,1),
(-3,0),(-2,-1),(-2,-2),(-1,-3),(0,-2),(1,-3),(2,-2),(2,-1),(3,0),
(2,1),(2,2),(0,0),2,8,(3,-3),1,2,14,8,(-9,-9),0
*00436,58,
2,14,8,(-11,-14),2,8,(1,0),1,9,(7,8),(0,0),2,8,(12,6),1,9,
(-9,-9),(-9,9),(0,0),2,8,(9,0),1,9,(0,-14),(0,0),2,8,(3,8),1,9,
```

```
(7,-8),(0,0),2,8,(1,0),1,2,14,8,(-11,-9),0
*00437,62,
2,14,8,(-7,-14),2,8,(8,8),1,9,(2,-1),(1,-1),(1,-2),(-1,-2),
(-1,-1),(-2,-1),(-4,0),(-3,1),(0,0),2,8,(0,12),1,9,(3,1),(3,0),
(3,-1),(1,-2),(-1,-2),(-2,-1),(-3,0),(0,0),2,8,(9,-8),1,2,
14,8,(-7,-9),0
*00438,46,
2,14,8,(-9,-14),2,8,(4,14),1,9,(0,-10),(1,-3),(2,-1),(3,0),(2,1),
(3,3),(0,0),2,8,(0,10),1,9,(0,-14),(0,0),2,8,(4,0),1,2,
14,8,(-10,-9),0
*00439,64,
2,14,8,(-9,-14),2,8,(4,14),1,9,(0,-10),(1,-3),(2,-1),(3,0),(2,1),
(3,3),(0,0),2,8,(0,10),1,9,(0,-14),(0,0),2,8,(-9,18),1,9,(1,-1),
(2,-1),(2,0),(2,1),(1,1),(0,0),2,8,(5,-18),1,2,14,8,(-10,-9),0
*0043A,46,
2,14,8,(-8,-14),2,8,(4,14),1,9,(0,-14),(0,0),2,8,(4,8),1,9,
(7,-8),(0,0),2,8,(-1,14),1,9,(-10,-10),(0,0),2,8,(12,-4),1,2,
14,8,(-8,-9),0
*0043B,44,
2,14,8,(-9,-14),2,8,(14,14),1,9,(-6,0),(0,-10),(-1,-3),(-2,-1),
(-2,0),(0,0),2,8,(11,14),1,9,(0,-14),(0,0),2,8,(4,0),1,2,
14,8,(-9,-9),0
*0043C,32,
2,14,8,(-11,-14),2,8,(18,0),1,9,(0,14),(-7,-11),(-7,11),(0,-14),
(0,0),2,8,(18,0),1,2,14,8,(-11,-9),0
*0043D,46,-
2,14,8,(-9,-14),2,8,(4,7),1,9,(11,0),(0,0),2,8,(0,7),1,9,(0,-14),
(0,0),2,8,(-11,14),1,9,(0,-14),(0,0),2,8,(15,0),1,2,
14,8,(-10,-9),0
*0043E,56,
2,14,8,(-9,-14),2,8,(8,14),1,9,(3,0),(2,-1),(2,-2),(1,-3),(0,-2),
(-1,-3),(-2,-2),(-2,-1),(-3,0),(-2,1),(-2,2),(-1,3),(0,2),(1,3),
(2,2),(2,1),(0,0),2,8,(11,-14),1,2,14,8,(-10,-9),0
*0043F,46,
2,14,8,(-9,-14),2,8,(4,14),1,9,(0,-14),(0,0),2,8,(0,10),1,9,
(3,3),(2,1),(3,0),(2,-1),(1,-2),(0,-11),(0,0),2,8,(4,0),1,2,
14,8,(-10,-9),0
*00440,60,
2,14,8,(-9,-14),2,8,(4,14),1,9,(0,-21),(0,0),2,8,(0,18),1,9,
(2,2),(2,1),(3,0),(2,-1),(2,-2),(1,-3),(0,-2),(-1,-3),(-2,-2),
(-2,-1),(-3,0),(-2,1),(-2,2),(0,0),2,8,(15,-3),1,2,
14,8,(-10,-16),0
*00441,50,
2,14,8,(-8,-14),2,8,(15,11),1,9,(-2,2),(-2,1),(-3,0),(-2,-1),
(-2,-2),(-1,-3),(0,-2),(1,-3),(2,-2),(2,-1),(3,0),(2,1),(2,2),
(0,0),2,8,(2,-3),1,2,14,8,(-9,-9),0
*00442,66,
2,14,8,(-14,-14),2,8,(4,14),1,9,(0,-14),(0,0),2,8,(0,10),1,9,
(3,3),(2,1),(2,0),(2,-1),(1,-3),(0,-10),(0,0),2,8,(0,10),1,9,
(3,3),(2,1),(2,0),(2,-1),(1,-3),(0,-10),(0,0),2,8,(4,0),1,2,
14,8,(-14,-9),0
*00443,44,
2,14,8,(-7,-14),2,8,(1,14),1,9,(6,-14),(0,0),2,8,(6,14),1,9,
(-6,-14),(-2,-4),(-2,-2),(-2,-1),(-1,0),(0,0),2,8,(14,7),1,2,
14,8,(-7,-16),0
*00444,66,
```

```
2,14,8,(-11,-14),2,8,(14,14),1,9,(-6,0),(-2,-1),(-2,-2),(-1,-3),
(0,-2),(1,-3),(2,-2),(2,-1),(6,0),(2,1),(2,2),(1,3),(0,2),(-1,3),
(-2,2),(-2,1),(0,0),2,8,(-3,0),1,9,(0,-21),(0,0),2,8,(11,7),1,2,
14,8,(-11,-16),0
*00445,36,
2,14,8,(-6,-14),2,8,(1,14),1,9,(11,-14),(0,0),2,8,(0,14),1,9,
(-11,-14),(0,0),2,8,(12,0),1,2,14,8,(-7,-9),0
*00446,50,
2,14,8,(-9,-14),2,8,(4,14),1,9,(0,-10),(1,-3),(2,-1),(3,0),(2,1),
(3,3),(0,0),2,8,(0,10),1,9,(0,-14),(2,0),(0,-3),(0,0),2,8,(2,3),
1,2,14,8,(-10,-12),0
*00447,46,
2,14,8,(-9,-14),2,8,(3,14),1,9,(0,-5),(1,-3),(2,-1),(3,0),(2,1),
(3,3),(0,0),2,8,(0,5),1,9,(0,-14),(0,0),2,8,(4,0),1,2,
14,8,(-9,-9),0
*00448,66,
2,14,8,(-14,-14),2,8,(24,0),1,9,(0,14),(0,0),2,8,(0,-10),1,9,
(-3,-3),(-2,-1),(-2,0),(-2,1),(-1,3),(0,10),(0,0),2,8,(0,-10),1,
9,(-3,-3),(-2,-1),(-2,0),(-2,1),(-1,3),(0,10),(0,0),2,8,(24,-14),
1,2,14,8,(-14,-9),0
*00449,70,
2,14,8,(-14,-14),2,8,(14,4),1,9,(-3,-3),(-2,-1),(-2,0),(-2,1),
(-1,3),(0,10),(0,0),2,8,(20,-10),1,9,(-3,-3),(-2,-1),(-2,0),
(-2,1),(-1,3),(0,10),(0,0),2,8,(10,0),1,9,(0,-14),(2,0),(0,-3),
(0,0),2,8,(2,3),1,2,14,8,(-14,-12),0
*0044A,64,
2,14,8,(-9,-14),2,8,(4,8),1,9,(1,1),(2,1),(3,0),(1,0),(2,-1),
(1,-1),(1,-2),(0,-2),(-1,-2),(-1,-1),(-2,-1),(-4,0),(-2,1),
(-1,1),(0,0),2,8,(-3,12),1,9,(3,0),(0,-14),(0,0),2,8,(14,0),1,2,
14,8,(-9,-9),0
*0044B,72,
2,14,8,(-10,-14),2,8,(4,8),1,9,(1,1),(2,1),(3,0),(1,0),(2,-1),
(1,-1),(1,-2),(0,-2),(-1,-2),(-1,-1),(-2,-1),(-4,0),(-2,1),
(-1,1),(0,0),2,8,(0,12),1,9,(0,-14),(0,0),2,8,(13,14),1,9,
(0,-14),(0,0),2,8,(4,0),1,2,14,8,(-11,-9),0
*0044C,62,
2,14,8,(-9,-14),2,8,(4,8),1,9,(1,1),(2,1),(3,0),(1,0),(2,-1),
(1,-1),(1,-2),(0,-2),(-1,-2),(-1,-1),(-2,-1),(-4,0),(-2,1),
(-1,1),(0,0),2,8,(0,12),1,9,(0,-14),(0,0),2,8,(14,0),1,2,
14,8,(-9,-9),0
*0044D,60,
2,14,8,(-8,-14),2,8,(2,11),1,9,(2,2),(2,1),(3,0),(2,-1),(2,-2),
(1,-3),(0,-2),(-1,-3),(-2,-2),(-2,-1),(-3,0),(-2,1),(-2,2),(0,0),
2,8,(12,4),1,9,(-6,0),(0,0),2,8,(9,-7),1,2,14,8,(-9,-9),0
*0044E,70,
2,14,8,(-12,-14),2,8,(4,14),1,9,(0,-14),(0,0),2,8,(0,7),1,9,
(5,0),(0,1),(1,3),(2,2),(2,1),(2,0),(2,-1),(2,-2),(1,-3),(0,-2),
(-1,-3),(-2,-2),(-2,-1),(-2,0),(-2,1),(-2,2),(-1,3),(0,1),(0,0),
2,8,(15,-7),1,2,14,8,(-12,-9),0
*0044F,54,
2,14,8,(-8,-14),2,8,(3,0),1,9,(4,5),(0,0),2,8,(6,-5),1,9,(0,14),
(-6,0),(-2,-1),(-1,-1),(-1,-2),(0,-1),(1,-2),(1,-1),(2,-1),(6,0),
(0,0),2,8,(4,-5),1,2,14,8,(-9,-9),0
*020A0,4,keuroRef2
7,0020AC,0
*020A7,49,kpes
```

```
2,14,8,(-11,-21),14,5,030,1,8,(0,21),050,02F,01E,02D,05C,02B,01A,
029,058,2,8,(12,13),1,8,(0,-18),02D,02F,010,2,8,(-6,14),1,060,2,
8,(3,-14),14,6,14,8,(11,-9),0
*020AC,45,keuro
2,14,8,(-10,-21),8,(2,10),5,1,014,00A,(9,-043),2,6,5,1,01C,00A,(9,043),
2,6,8,(-1,2),5,1,8,(13,0),2,6,04C,1,0B0,2,08C,080,14,8,(-10,-9),0
*02126,53,komega
2,14,8,(-11,-21),14,5,8,(3,1),1,01E,040,014,036,025,8,(-1,3),044,
8,(1,3),023,022,021,040,02F,02E,02D,8,(1,-3),04C,8,(-1,-3),02B,
03A,01C,040,012,2,8,(3,-1),14,6,14,8,(11,-9),0
*02205,64,kdiam
2,14,3,2,14,8,(-19,-40),14,4,2,14,5,8,(8,17),1,029,02A,8,(-1,-3),
02C,8,(1,-3),02E,02F,030,021,022,8,(1,3),024,8,(-1,3),026,027,
038,2,061,1,8,(-9,-20),2,8,(14,0),14,6,14,3,2,14,8,(19,-18),14,4,
2,0
*0221E,45,kinfin
2,14,8,(-11,-14),14,5,8,(11,11),1,025,027,028,029,02B,01C,02D,
02F,020,021,023,014,023,021,020,02F,02D,01C,02B,029,028,027,025,
2,8,(11,-10),14,6,14,8,(11,-9),0
*02264,36,kleq
2,14,8,(-13,-21),14,5,8,(22,21),1,8,(-18,-9),8,(18,-9),2,
8,(-18,-2),1,8,(18,0),2,8,(4,-1),14,6,14,8,(13,-9),0
*02302,25,ktria
2,14,8,(-10,-21),14,5,040,1,0C0,084,066,06A,08C,2,8,(16,0),14,6,
14,8,(10,-9),0
```

Extended Standard Font for UNICODE

```
;;
;; txt.shp - Extended Standard Font for UNICODE
;;
;; Copyright 1997 by Autodesk, Inc.
;;
;; Permission to use, copy, modify, and distribute this software for
;; any purpose and without fee is hereby granted, provided that the
;; above copyright notice appears in all copies and that the
restricted
;; rights notice below appear in all supporting documentation.
;;
;; Use, duplication, or disclosure by the U.S. Government is subject
;; to restrictions set forth in FAR 52.227-19 (Commercial Computer
;; Software - Restricted Rights) and DFAR 252.227-7013(c)(1)(ii)
;; (Rights in Technical Data and Computer Software), as applicable.
;;
*UNIFONT,6,TXT Copyright 1997 by Autodesk, Inc.
6,2,2,0,0,0
*0000A,7,lf
2,0AC,14,8,(9,10),0
*00020,7,spc
2,060,14,8,(-6,-8),0
*00021,17,kexc
2,14,06C,1,014,2,014,1,044,2,020,06C,14,8,(-2,-3),0
*00022,20,kdblqt
2,14,8,(-1,-6),044,1,023,2,010,1,02B,2,04C,030,14,8,(-3,1),0
```

```
*00023,27,kns
2,14,8,(-2,-6),024,1,040,2,024,1,048,2,023,1,06C,2,020,1,064,2,
06D,14,8,(-4,-3),0
*00024,25,kds
2,14,8,(-2,-6),014,1,030,012,016,028,016,012,030,2,027,1,06C,2,
040,14,8,(-4,-3),0
*00025,31,kpc
2,14,8,(-2,-6),064,1,01C,010,014,018,2,040,1,8,(-4,-6),2,040,1,
018,014,010,01C,2,020,14,8,(-4,-3),0
*00026,24,kand
2,14,8,(-2,-6),041,1,02A,018,016,014,022,014,016,01A,01C,04E,2,
020,14,8,(-4,-3),0
*00027,28,kapos
2,14,3,2,14,8,(-1,-12),14,4,2,044,1,023,2,06C,020,14,3,2,
14,8,(-5,2),14,4,2,0
*00028,16,klp
2,14,8,(-1,-6),064,020,1,02A,02C,02E,2,020,14,03A,0
*00029,15,krp
2,14,8,(-1,-6),064,1,02E,02C,02A,2,040,14,03A,0
*0002A,27,kas
2,14,8,(-2,-5),021,1,044,2,02E,1,048,2,041,1,04A,2,044,1,04E,2,
02F,14,8,(-4,-3),0
*0002B,19,kpls
2,14,8,(-2,-5),021,1,044,2,02E,1,048,2,06F,14,8,(-4,-3),0
*0002C,28,kcma
2,14,3,2,14,02B,14,4,2,014,010,1,01C,01A,2,012,020,14,3,2,
14,8,(-3,-8),14,4,2,0
*0002D,14,ksub
2,14,8,(-2,-3),034,1,040,2,020,03C,14,048,0
*0002E,12,kper
2,14,01C,1,014,2,02F,14,8,(-2,-3),0
*0002F,17,kdiv
2,14,8,(-2,-6),1,8,(4,6),2,020,06C,14,8,(-4,-3),0
*00030,34,n0
2,14,3,2,14,8,(-3,-12),14,4,2,010,1,016,044,012,010,01E,04C,01A,
018,2,040,14,3,2,14,8,(-7,-6),14,4,2,0
*00031,18,n1
2,14,8,(-1,-6),054,1,012,06C,2,018,1,020,2,020,14,03A,0
*00032,23,n2
2,14,8,(-2,-6),054,1,012,020,01E,01C,01A,028,01A,02C,040,2,020,
14,8,(-4,-3),0
*00033,29,n3
2,14,8,(-2,-6),054,1,012,020,01E,01C,01A,018,2,010,1,01E,01C,01A,
028,016,2,01C,060,14,8,(-4,-3),0
*00034,19,n4
2,14,8,(-2,-6),041,1,048,8,(3,4),06C,2,030,14,8,(-4,-3),0
*00035,23,n5
2,14,8,(-2,-6),014,1,01E,020,012,024,016,038,024,040,2,020,06C,
14,8,(-4,-3),0
*00036,24,n6
2,14,8,(-2,-6),034,1,030,01E,01C,01A,028,016,034,022,010,2,030,
06C,14,8,(-4,-3),0
*00037,16,n7
2,14,8,(-2,-6),064,1,040,06B,2,050,14,8,(-4,-3),0
*00038,32,n8
2,14,8,(-2,-6),010,1,016,014,012,020,012,014,016,028,01A,01C,01E,
```

```
2,020,1,01E,01C,01A,028,2,050,14,8,(-4,-3),0
*00039,24,n9
2,14,8,(-2,-6),010,1,010,022,034,016,028,01A,01C,01E,030,2,020,
03C,14,8,(-4,-3),0
*0003A,17,kcol
2,14,04C,044,1,01C,2,01C,1,01C,2,02F,14,8,(-2,-3),0
*0003B,34,ksmc
2,14,3,2,14,8,(-1,-8),14,4,2,010,044,1,01C,2,01C,1,02C,01A,2,012,
020,14,3,2,14,8,(-5,-8),14,4,2,0
*0003C,29,klt
2,14,3,2,14,8,(-3,-12),14,4,2,064,030,1,03A,03E,2,020,14,3,2,
14,8,(-5,-6),14,4,2,0
*0003D,18,keq
2,14,04B,044,1,040,2,02C,1,048,2,060,02C,14,8,(-4,-1),0
*0003E,28,kgt
2,14,3,2,14,8,(-3,-12),14,4,2,064,1,03E,03A,2,050,14,3,2,
14,8,(-7,-6),14,4,2,0
*0003F,36,kqm
2,14,3,2,14,8,(-3,-12),14,4,2,054,1,012,010,01E,01C,01A,01C,2,
01C,1,01C,2,030,14,3,2,14,8,(-7,-6),14,4,2,0
*00040,28,kea
2,14,8,(-2,-6),032,1,01A,018,014,012,010,02C,012,024,016,028,01A,
04C,01E,030,2,020,14,8,(-4,-3),0
*00041,21,uca
2,14,8,(-2,-6),1,024,043,04D,02C,2,047,1,040,2,02E,14,8,(-4,-3),0
*00042,29,ucb
2,14,8,(-2,-6),1,030,012,014,016,028,2,020,1,012,014,016,038,2,
010,1,06C,2,050,14,8,(-4,-3),0
*00043,23,ucc
2,14,8,(-2,-6),040,014,1,01A,028,016,044,012,020,01E,2,02E,03C,
14,8,(-4,-3),0
*00044,22,ucd
2,14,8,(-2,-6),1,030,012,044,016,038,2,010,1,06C,2,050,
14,8,(-4,-3),0
*00045,25,uce
2,14,8,(-2,-6),1,064,040,2,048,03C,1,020,2,028,03C,1,040,2,020,
14,8,(-4,-3),0
*00046,21,ucf
2,14,8,(-2,-6),1,064,040,2,048,03C,1,020,2,03C,040,14,8,(-4,-3),0
*00047,22,ucg
2,14,8,(-2,-6),032,1,010,03C,038,016,044,012,030,2,020,06C,
14,8,(-4,-3),0
*00048,22,uch
2,14,8,(-2,-6),1,064,2,03C,1,040,2,034,1,06C,2,020,14,8,(-4,-3),0
*00049,21,uci
2,14,8,(-1,-6),064,1,020,2,018,1,06C,2,018,1,020,2,020,14,03A,0
*0004A,19,ucj
2,14,8,(-2,-6),014,1,01E,020,012,054,2,020,06C,14,8,(-4,-3),0
*0004B,23,uck
2,14,8,(-2,-6),1,064,2,040,1,03A,018,2,010,1,03E,2,020,
14,8,(-4,-3),0
*0004C,16,ucl
2,14,8,(-2,-6),064,1,06C,040,2,020,14,8,(-4,-3),0
*0004D,17,ucm
2,14,8,(-2,-6),1,064,04D,043,06C,2,020,14,8,(-4,-3),0
*0004E,19,ucn
```

```
2,14,8,(-2,-6),1,064,8,(4,-6),064,2,06C,020,14,8,(-4,-3),0
*0004F,17,uco
2,14,8,(-2,-6),1,064,040,06C,048,2,060,14,8,(-4,-3),0
*00050,19,ucp
2,14,8,(-2,-6),1,064,030,01E,01C,01A,038,2,06F,14,8,(-4,-3),0
*00051,25,ucq
2,14,8,(-2,-6),022,1,01E,01A,018,016,044,012,020,01E,03C,01A,01E,
2,020,14,8,(-4,-3),0
*00052,23,ucr
2,14,8,(-2,-6),1,064,030,01E,01C,01A,038,2,010,1,03E,2,020,
14,8,(-4,-3),0
*00053,22,ucs
2,14,8,(-2,-6),014,1,01E,020,012,046,012,020,01E,2,020,05C,
14,8,(-4,-3),0
*00054,19,uct
2,14,8,(-2,-6),064,1,040,2,028,1,06C,2,040,14,8,(-4,-3),0
*00055,20,ucu
2,14,8,(-2,-6),064,1,05C,01E,020,012,054,2,020,06C,14,8,(-4,-3),0
*00056,15,ucv
2,14,06B,064,1,06D,063,2,020,06C,14,8,(-5,-3),0
*00057,24,ucw
2,14,06B,064,1,9,(2,-6),(1,3),(1,-3),(2,6),(0,0),2,020,06C,
14,8,(-5,-3),0
*00058,22,ucx
2,14,8,(-2,-6),1,8,(4,6),2,048,1,8,(4,-6),2,020,14,8,(-4,-3),0
*00059,25,ucy
2,14,8,(-2,-6),064,1,8,(2,-3),03C,2,034,1,8,(2,3),2,020,06C,
14,8,(-4,-3),0
*0005A,19,ucz
2,14,8,(-2,-6),064,1,040,8,(-4,-6),040,2,020,14,8,(-4,-3),0
*0005B,17,klb
2,14,8,(-1,-6),1,064,020,2,06C,1,028,2,040,14,03A,0
*0005C,17,kbkslsh
2,14,8,(-2,-6),064,1,8,(4,-6),2,020,14,8,(-4,-3),0
*0005D,15,krb
2,14,8,(-1,-6),064,1,020,06C,028,2,040,14,03A,0
*0005E,16,kcaret
2,14,8,(-2,-6),044,1,022,02E,2,04D,14,8,(-4,1),0
*0005F,11,kundrl
2,14,028,01C,1,040,2,021,14,04A,0
*00060,27,krvap
2,14,3,2,14,8,(-1,-12),14,4,2,064,1,02D,2,04D,14,3,2,14,8,(-5,2),
14,4,2,0
*00061,24,lca
2,14,04B,020,1,018,016,024,012,010,01E,02C,01A,2,012,1,01E,2,020,
14,8,(-4,-3),0
*00062,25,lcb
2,14,8,(-2,-6),1,064,2,04C,1,022,010,01E,02C,01A,018,026,2,02C,
060,14,8,(-4,-3),0
*00063,17,lcc
2,14,04B,042,1,038,01A,02C,01E,030,2,020,14,8,(-4,-3),0
*00064,25,lcd
2,14,8,(-2,-6),041,1,02A,018,016,024,012,010,02E,2,044,1,06C,2,
020,14,8,(-4,-3),0
*00065,20,lce
2,14,04B,024,1,030,012,016,028,01A,02C,01E,020,2,030,
```

```
14,8,(-4,-3),0
*00066,22,lcf
2,14,8,(-2,-6),034,1,030,2,023,1,016,018,01A,05C,2,050,
14,8,(-4,-3),0
*00067,22,lcg
2,14,04B,01C,1,01E,020,012,044,016,028,01A,02C,01E,030,2,020,
14,8,(-4,-5),0
*00068,21,lch
2,14,8,(-2,-6),1,064,2,04C,1,022,010,01E,03C,2,020,14,8,(-4,-3),0
*00069,17,lci
2,14,06C,1,044,2,014,1,014,2,020,06C,14,8,(-2,-3),0
*0006A,35,lcj
2,14,3,2,14,8,(-3,-12),14,4,2,01C,1,01E,010,012,054,2,014,1,014,
2,020,06C,14,3,2,14,8,(-7,-10),14,4,2,0
*0006B,23,lck
2,14,8,(-2,-6),1,064,2,04C,1,020,022,2,02A,1,02E,2,020,
14,8,(-4,-3),0
*0006C,28,lcl
2,14,3,2,14,8,(-1,-12),14,4,2,064,1,05C,01E,2,020,14,3,2,
14,8,(-5,-6),14,4,2,0
*0006D,24,lcm
2,14,04B,1,044,2,01C,1,012,01E,01C,2,014,1,012,01E,03C,2,020,
14,8,(-4,-3),0
*0006E,21,lcn
2,14,8,(-2,-6),1,044,2,02C,1,022,010,01E,03C,2,020,14,8,(-4,-3),0
*0006F,20,lco
2,14,04B,030,1,028,016,024,012,020,01E,02C,01A,2,030,
14,8,(-4,-3),0
*00070,22,lcp
2,14,04B,02C,1,064,2,01C,1,012,020,01E,02C,01A,038,2,060,
14,8,(-4,-5),0,
*00071,22,lcq
2,14,04B,04F,1,064,2,01C,1,016,028,01A,02C,01E,030,2,020,
14,8,(-4,-5),0,
*00072,19,lcr
2,14,04B,1,044,2,02C,1,022,010,01E,2,020,03C,14,8,(-4,-3),0
*00073,18,lcs
2,14,04B,1,030,012,016,028,016,012,030,2,04D,14,8,(-4,-3),0
*00074,21,lct
2,14,8,(-2,-6),044,1,040,2,026,1,05C,01E,012,2,02F,14,8,(-4,-3),0
*00075,20,lcu
2,14,04B,044,1,03C,01E,010,022,2,024,1,04C,2,020,14,8,(-4,-3),0
*00076,14,lcv
2,14,04B,044,1,04D,043,2,04D,14,8,(-4,-3),0
*00077,23,lcw
2,14,04B,044,1,9,(1,-4),(1,4),(1,-4),(1,4),(0,0),2,04D,
14,8,(-4,-3),0
*00078,16,lcx
2,14,04B,1,042,2,048,1,04E,2,020,14,8,(-4,-3),0
*00079,19,lcy
2,14,04B,044,1,04D,2,043,1,06B,018,2,024,060,14,8,(-4,-5),0
*0007A,15,lcz
2,14,04B,044,1,040,04A,040,2,020,14,8,(-4,-3),0
*0007B,19,klbr
2,14,8,(-1,-6),064,020,1,01A,01C,01A,01E,01C,01E,2,020,14,03A,0
*0007C,13,kvbar
```

```
2,14,06C,1,064,2,06C,020,14,8,(-2,-3),0
*0007D,18,krbr
2,14,8,(-1,-6),1,012,014,012,016,014,016,2,06C,040,14,03A,0
*0007E,15,ktlde
2,14,04B,034,1,012,02F,012,2,04D,14,8,(-4,-2),0
*00080,4,keuroRef
7,020AC,0
*000A0,7,NoBrkSpc
2,060,14,8,(-6,-8),0
*000A1,18,kiexc
2,14,06C,1,044,2,014,1,014,2,8,(2,-6),14,8,(-2,-3),0
*000A2,23,kcent
2,14,8,(-2,-5),01E,1,8,(2,6),2,01E,1,038,01A,02C,01E,030,2,020,
14,04A,0,
*000A3,23,kpound
2,14,8,(-2,-6),040,1,048,012,044,012,01E,2,02B,1,028,2,06F,
14,8,(-4,-3),0
*000A5,34,kyen
2,14,8,(-2,-6),064,1,8,(2,-3),03C,2,025,1,020,2,027,1,020,2,018,
1,8,(2,3),2,8,(2,-6),14,8,(-4,-3),0
*000A7,45,kpar
2,14,3,2,14,8,(-3,-12),14,4,2,014,1,01E,010,012,016,018,016,012,
2,021,1,016,018,01A,01E,010,01E,01A,2,8,(3,-2),14,3,2,
14,8,(-7,-6),14,4,2,0
*000AA,36,lcau
2,14,8,(-1,-6),2,3,2,8,3,9,1,01A,018,016,024,012,010,01E,02C,01E,
2,049,1,040,2,4,2,8,2,-3,14,8,(-3,1),0
*000AB,21,kfrew
2,14,8,(-2,-5),021,1,026,022,2,020,1,02A,02E,2,02F,14,8,(-4,-2),0
*000B0,19,kdeg
2,14,8,(-1,-6),054,1,012,01E,01A,016,2,05C,040,14,8,(-3,2),0
*000B1,23,kpls-min
2,14,8,(-2,-6),014,1,040,2,027,1,044,2,02A,1,040,2,04D,
14,8,(-4,-2),0
*000B5,24,kmicro
2,14,04B,02C,1,8,(1,6),1,03C,01E,010,023,2,024,1,04C,2,020,
14,8,(-4,-5),0
*000BA,35,lcou
2,14,8,(-1,-6),3,2,2,8,3,12,1,028,01A,02C,01E,020,012,024,016,2,
06B,1,040,2,4,2,8,2,-3,14,8,(-3,1),0
*000BB,22,kffrw
2,14,8,(-2,-5),014,1,022,026,2,020,1,02E,02A,2,040,01C,
14,8,(-4,-2),0
*000BC,41,kquart
2,3,2,14,8,(-5,-12),1,8,(10,12),2,8,(-8,-6),1,064,01A,2,05C,1,
020,2,8,(6,-4),1,048,8,(3,4),06C,2,060,14,8,(-9,-6),4,2,0
*000BD,45,khalf
2,3,2,14,8,(-5,-12),1,8,(10,12),2,8,(-8,-6),1,064,01A,2,05C,1,
020,2,8,(3,-1),1,012,020,01E,01C,01A,028,01A,02C,040,2,040,
14,8,(-9,-6),4,2,0
*000BF,32,kiqm
2,3,2,14,8,(-3,-12),8,(6,2),1,02A,028,026,024,022,024,2,024,1,
024,2,8,(8,-12),14,8,(-7,-6),4,2,0
*000C0,31,uc^
2,14,8,(-2,-6),1,024,022,02E,02C,2,8,(-4,1),1,040,2,8,(-2,4),1,
027,2,8,(6,-6),14,8,(-4,-3),0
```

```
*000C1,31,uc^
2,14,8,(-2,-6),1,024,022,02E,02C,2,8,(-4,1),1,040,2,8,(-2,4),1,
021,2,8,(2,-6),14,8,(-4,-3),0
*000C2,32,uc^
2,14,8,(-2,-6),1,024,022,02E,02C,2,8,(-4,1),1,040,2,8,(-4,3),1,
022,02E,2,8,(2,-4),14,8,(-4,-3),0
*000C3,33,uc^
2,14,8,(-2,-6),1,024,022,02E,02C,2,8,(-4,1),1,040,2,8,(-4,4),1,
012,02F,012,2,8,(2,-6),14,8,(-4,-3),0
*000C4,32,uc„
2,14,8,(-2,-6),1,024,043,2,029,1,014,2,040,1,01C,2,027,1,04D,02C,
2,047,1,040,2,02E,14,8,(-4,-3),0
*000C5,25,uc^
2,14,8,(-2,-6),1,024,032,016,01A,03E,02C,2,8,(-4,1),1,040,2,02F,
14,8,(-4,-3),0
*000C6,33,uc^
2,14,8,(-2,-6),1,034,8,(2,3),020,2,8,(-2,-3),1,020,2,03C,1,028,
064,2,04B,1,020,2,04F,14,8,(-4,-3),0
*000C7,29,uc‡
2,14,8,(-2,-6),02E,1,010,014,018,014,2,021,1,01A,028,016,044,012,
020,01E,2,02E,03C,14,8,(-4,-5),0
*000C8,28,uc^
2,14,8,(-2,-6),1,044,040,2,016,1,027,2,04C,018,1,020,2,02A,1,040,
2,020,14,8,(-4,-3),0
*000C9,28,uc^
2,14,8,(-2,-6),1,044,040,2,025,1,029,2,03C,018,1,020,2,02A,1,040,
2,020,14,8,(-4,-3),0
*000CA,29,uc^
2,14,8,(-2,-6),1,044,040,2,016,1,016,01A,2,03C,018,1,020,2,02A,1,
040,2,020,14,8,(-4,-3),0
*000CB,32,uc^
2,14,8,(-2,-6),1,044,040,2,016,1,014,2,028,1,01C,2,03C,018,1,020,
2,02A,1,040,2,020,14,8,(-4,-3),0
*000CC,25,uc^
2,14,8,(-1,-6),044,1,020,2,014,1,027,2,02D,1,04C,2,018,1,020,2,
020,14,03A,0
*000CD,25,uc^
2,14,8,(-1,-6),044,1,020,2,024,1,029,2,01E,1,04C,2,018,1,020,2,
020,14,03A,0
*000CE,26,uc^
2,14,8,(-1,-6),044,1,020,2,014,1,016,01A,2,01E,1,04C,2,018,1,020,
2,020,14,03A,0
*000CF,29,uc^
2,14,8,(-1,-6),044,1,020,2,014,1,014,2,028,1,01C,2,01E,1,04C,2,
018,1,020,2,020,14,03A,0
*000D0,25,uc
2,14,8,(-2,-6),1,064,030,01E,04C,01A,038,2,024,015,1,020,2,01D,
04F,14,8,(-4,-3),0
*000D1,25,uc
2,14,8,(-2,-6),1,044,04E,044,2,048,014,1,012,02F,012,2,8,(2,-6),
14,8,(-4,-3),0
*000D2,25,uc^
2,14,8,(-2,-6),1,044,040,2,016,1,027,2,02C,030,1,04C,048,2,060,
14,8,(-4,-3),0
*000D3,25,uc^
2,14,8,(-2,-6),1,044,040,2,025,1,029,2,01C,030,1,04C,048,2,060,
```

```
14,8,(-4,-3),0
*000D4,26,uc^
2,14,8,(-2,-6),1,044,040,2,016,1,016,01A,2,01C,030,1,04C,048,2,
060,14,8,(-4,-3),0
*000D5,27,uc^
2,14,8,(-2,-6),1,044,040,2,024,1,01A,027,01A,2,01C,040,1,04C,048,
2,060,14,8,(-4,-3),0
*000D6,31,uc^
2,14,8,(-2,-6),1,044,2,012,1,014,2,02B,1,040,2,016,1,014,2,02D,1,
04C,048,2,060,14,8,(-4,-3),0
*000D8,29,ucd”
2,14,8,(-2,-6),1,8,(4,6),2,018,1,01E,04C,01A,028,016,044,012,020,
2,8,(3,-6),14,8,(-4,-3),0
*000D9,24,uc^
2,14,8,(-2,-6),064,1,05C,01E,020,012,054,2,01A,1,027,2,050,06C,
14,8,(-4,-3),0
*000DA,24,uc^
2,14,8,(-2,-6),064,1,05C,01E,020,012,054,2,018,1,029,2,050,05C,
14,8,(-4,-3),0
*000DB,25,uc^
2,14,8,(-2,-6),064,1,05C,01E,020,012,054,2,01A,1,016,01A,2,050,
05C,14,8,(-4,-3),0
*000DC,27,uc^
2,14,8,(-2,-6),064,1,05C,01E,020,012,054,2,01A,1,014,2,028,1,01C,
2,05E,14,8,(-4,-3),0
*000DD,25,uc^
2,14,8,(-2,-6),044,1,02E,02C,2,024,1,022,2,025,1,029,2,050,05C,
14,8,(-4,-3),0
*000DE,27,lc
2,14,8,(-2,-6),02C,1,084,2,03C,1,012,020,01E,02C,01A,028,016,2,
8,(6,-1),14,8,(-4,-5),0
*000DF,24,kgers
2,14,8,(-2,-6),1,012,044,012,010,01E,01C,01A,01E,01C,01A,018,2,
040,14,8,(-4,-3),0
*000E0,39,lc…
2,14,8,(-2,-6),020,1,018,016,024,012,010,01E,02C,01A,2,012,1,01E,
2,2,3,2,8,(-3,10),1,047,2,8,(11,-12),4,2,14,8,(-4,-3),0
*000E1,39,lc
2,14,8,(-2,-6),020,1,018,016,024,012,010,01E,02C,01A,2,012,1,01E,
2,2,3,2,8,(-3,12),1,049,2,8,(11,-10),4,2,14,8,(-4,-3),0
*000E2,40,lcƒ
2,14,8,(-2,-6),020,1,018,016,024,012,010,01E,02C,01A,2,012,1,01E,
2,2,3,2,8,(-3,10),1,026,02A,2,8,(11,-10),4,2,14,8,(-4,-3),0
*000E3,40,lcƒ
2,14,8,(-2,-6),020,1,018,016,024,012,010,01E,02C,01A,2,012,1,01E,
2,3,2,8,(-7,10),1,012,02F,012,2,8,(7,-11),4,2,14,8,(-4,-3),0
*000E4,37,lc„
2,14,8,(-2,-6),020,1,018,016,024,2,034,1,01C,2,030,1,014,2,03A,1,
012,010,01E,02C,01A,2,012,1,01E,2,020,14,8,(-4,-3),0
*000E5,38,lc†
2,14,8,(-2,-6),3,2,8,(3,8),1,022,026,02A,02E,2,8,(3,-6),4,2,1,
01A,018,016,024,012,010,01E,02C,01E,2,020,14,8,(-4,-3),0
*000E6,30,lc
2,14,04B,021,1,01A,016,024,012,01E,03C,2,034,1,012,01E,01C,028,2,
01C,1,01E,010,2,020,14,8,(-4,-3),0
*000E7,24,lc‡
```

```
2,14,04B,042,1,038,01A,02C,01E,030,2,028,1,01C,010,01C,018,2,041,
14,8,(-4,-5),0
*000E8,27,lcŠ
2,14,8,(-2,-6),024,1,030,012,016,028,01A,02C,01E,020,2,054,1,027,
2,050,06C,14,8,(-4,-3),0
*000E9,26,lc,
2,14,8,(-2,-6),024,1,030,012,016,028,01A,02C,01E,020,2,064,1,029,
2,05E,14,8,(-4,-3),0
*000EA,27,lcˆ
2,14,8,(-2,-6),024,1,030,012,016,028,01A,02C,01E,020,2,054,1,016,
01A,2,05E,14,8,(-4,-3),0
*000EB,31,lc‰
2,14,8,(-2,-6),024,1,030,012,016,028,01A,02C,01E,020,2,064,010,1,
01C,2,048,1,014,2,06E,14,8,(-4,-3),0
*000EC,18,lc_
2,14,8,(-1,-6),010,1,044,2,025,1,02F,2,020,05C,14,03A,0
*000ED,18,lc
2,14,8,(-1,-6),010,1,044,2,016,1,021,2,020,06C,14,03A,0
*000EE,19,lcŒ
2,14,8,(-1,-6),010,1,044,2,016,1,012,01E,2,020,05C,14,03A,0
*000EF,22,lc‹
2,14,8,(-1,-6),010,1,044,2,016,1,014,2,020,1,01C,2,020,05C,
14,03A,0
*000F0,30,lc
2,14,8,(-2,-6),8,(3,4),1,028,01A,02C,01E,020,012,024,036,2,01C,1,
021,2,8,(3,-6),14,8,(-4,-3),0
*000F1,27,lc
2,14,8,(-2,-6),1,044,2,014,1,012,02F,012,2,04A,1,022,010,01E,03C,
2,020,14,8,(-4,-3),0
*000F2,27,lc•
2,14,8,(-2,-6),030,1,028,016,024,012,020,01E,02C,01A,2,054,1,027,
2,050,06C,14,8,(-4,-3),0
*000F3,26,lc
2,14,8,(-2,-6),030,1,028,016,024,012,020,01E,02C,01A,2,064,1,029,
2,05E,14,8,(-4,-3),0
*000F4,27,lc‟
2,14,8,(-2,-6),030,1,028,016,024,012,020,01E,02C,01A,2,054,1,016,
01A,2,05E,14,8,(-4,-3),0
*000F5,32,lc^
2,14,8,(-2,-6),030,1,028,016,024,012,020,01E,02C,01A,2,8,(-3,5),
1,012,02F,012,2,8,(2,-6),14,8,(-4,-3),0
*000F6,35,lc”
2,14,8,(-2,-6),030,1,028,016,024,2,034,1,01C,2,040,1,014,2,
8,(-4,-3),1,012,020,01E,02C,01A,2,030,14,8,(-4,-3),0
*000F7,23,kto
2,14,8,(-2,-5),021,1,014,2,021,1,048,2,022,1,01C,2,04E,
14,8,(-4,-2),0
*000F8,24,lcd”
2,14,04B,010,1,020,012,024,016,028,01A,02C,01E,2,018,1,042,2,04D,
14,8,(-4,-3),0
*000F9,27,lc—
2,14,8,(-2,-6),044,1,03C,01E,010,022,2,038,044,1,02F,2,01E,1,04C,
2,020,14,8,(-4,-3),0
*000FA,26,lc
2,14,8,(-2,-6),044,1,03C,01E,010,022,2,036,1,021,2,02D,1,04C,2,
020,14,8,(-4,-3),0
```

```
*000FB,27,lc-
2,14,8,(-2,-6),044,1,03C,01E,010,022,2,036,1,012,01E,2,01E,1,04C,
2,020,14,8,(-4,-3),0
*000FC,32,lc_
2,14,8,(-2,-6),064,010,1,01C,2,01A,1,03C,01E,010,022,2,044,018,1,
01C,2,01E,1,04C,2,020,14,8,(-4,-3),0
*000FD,27,lc^
2,14,8,(-2,-6),044,1,04D,2,8,(-1,5),1,021,2,02D,1,06B,018,2,024,
060,14,8,(-4,-3),0
*000FE,25,uc
2,14,8,(-2,-6),1,064,2,01E,019,1,030,01E,01C,01A,038,2,01F,01C,
050,14,8,(-4,-3),0
*000FF,30,lc~
2,14,8,(-2,-6),044,1,04D,2,054,018,1,014,2,020,1,01C,2,01E,1,06B,
018,2,024,060,14,8,(-4,-5),0
*00104,26,c164
2,14,8,(-2,-6),1,024,043,04D,02C,2,047,1,040,2,02C,1,01A,01E,2,
022,14,8,(-4,-3),0
*00105,30,c165
2,14,04B,020,1,018,016,024,012,010,01E,02C,01A,2,012,1,01E,3,2,
01A,01E,4,2,2,021,14,8,(-4,-3),0
*00106,28,c143
2,14,8,(-2,-6),040,014,1,01A,028,016,044,012,020,01E,2,038,024,1,
021,2,08C,030,14,8,(-4,-3),0
*00107,30,c134
2,14,04B,042,1,038,01A,02C,01E,030,2,3,2,8,(-5,10),1,021,2,029,
8,(9,-10),4,2,14,8,(-4,-3),0
*0010C,31,c172
2,14,8,(-2,-8),040,014,1,01A,028,016,044,012,020,01E,2,026,1,012,
2,01A,1,016,2,050,08C,14,8,(-4,-3),0
*0010D,29,c159
2,14,8,(-2,-6),042,1,038,01A,02C,01E,030,2,028,054,1,012,2,01A,1,
016,2,050,06C,14,8,(-4,-3),0
*0010E,32,c210
2,14,8,(-2,-8),1,030,012,044,016,038,2,010,1,06C,2,010,074,1,012,
2,01A,1,016,2,050,08C,14,8,(-4,-3),0
*0010F,31,c212
2,14,8,(-2,-6),041,1,02A,018,016,024,012,010,02E,2,044,1,06C,2,
020,064,1,01A,2,010,05C,14,8,(-4,-3),0
*00118,28,c168
2,14,8,(-2,-6),1,064,040,2,048,03C,1,020,2,028,03C,1,040,1,01A,
01E,2,022,14,8,(-4,-3),0
*00119,27,c169
2,14,04B,024,1,030,012,016,028,01A,02C,01E,020,3,2,01A,01E,4,2,2,
014,030,14,8,(-4,-3),0
*0011A,35,c183
2,14,8,(-2,-8),1,064,040,2,048,03C,1,020,2,028,03C,1,040,2,028,
074,1,012,2,01A,1,016,2,050,08C,14,8,(-4,-3),0
*0011B,32,c216
2,14,8,(-2,-6),024,1,030,012,016,028,01A,02C,01E,020,2,018,054,1,
012,2,01A,1,016,2,050,06C,14,8,(-4,-3),0
*00141,29,c157
2,14,8,(-2,-6),064,1,06C,040,2,048,034,1,3,2,8,(4,5),2,8,(8,-11),
4,2,14,8,(-4,-3),0
*00142,47,c136
2,14,3,2,14,8,(-1,-12),14,4,2,064,3,2,010,4,2,1,05C,01E,2,3,2,
```

```
8,(-3,5),1,8,(3,4),2,8,(4,-9),4,2,14,3,2,14,8,(-5,-6),14,4,2,0
*00143,24,c227
2,14,8,(-2,-6),1,064,8,(4,-6),064,2,038,014,1,021,2,08C,030,
14,8,(-4,-3),0
*00144,42,c228
2,14,3,2,14,8,(-3,-8),14,4,2,1,044,2,01C,1,012,010,01E,03C,2,
8,(-2,5),1,011,2,019,8,(4,-5),14,3,2,14,8,(-7,-6),14,4,2,0
*00147,27,c213
2,14,8,(-2,-8),1,064,8,(4,-6),064,2,027,1,012,2,01A,1,016,2,050,
08C,14,8,(-4,-3),0
*00148,29,c229
2,14,8,(-2,-6),1,044,2,01C,1,012,010,01E,03C,2,018,054,1,012,2,
01A,1,016,2,040,06C,14,03A,0
*00150,27,c138
2,14,8,(-2,-6),1,064,040,06C,048,2,074,010,1,03C,2,020,1,034,2,
07C,030,14,8,(-4,-3),0
*00151,29,c139
2,14,04B,030,1,028,016,024,012,020,01E,02C,01A,2,074,1,02C,2,028,
1,024,2,07C,050,14,8,(-4,-3),0
*00158,33,c252
2,14,8,(-2,-8),1,064,030,01E,01C,01A,038,2,010,1,03E,2,028,074,1,
012,2,01A,1,016,2,050,08C,14,8,(-4,-3),0
*00159,29,c253
2,14,8,(-2,-6),1,044,2,02C,1,022,010,01E,2,026,1,012,2,01A,1,016,
2,050,06C,14,8,(-4,-3),0
*0015A,27,c151
2,14,8,(-2,-6),014,1,01E,020,012,046,012,020,01E,2,024,038,1,021,
2,08C,030,14,8,(-4,-3),0
*0015B,31,c152
2,14,04B,1,030,012,016,028,016,012,030,2,3,2,8,(-5,2),1,021,2,
029,8,(9,-10),4,2,14,8,(-4,-3),0
*00160,30,c230
2,14,8,(-2,-8),014,1,01E,020,012,046,012,020,01E,2,026,1,012,2,
01A,1,016,2,050,08C,14,8,(-4,-3),0
*00161,29,c231
2,14,8,(-2,-6),1,030,012,016,028,016,012,030,2,027,1,012,2,01A,1,
016,2,050,06C,14,8,(-4,-3),0
*00164,28,c155
2,14,8,(-2,-8),064,1,040,2,028,1,06C,2,074,1,012,2,01A,1,016,2,
050,08C,14,8,(-4,-3),0
*00165,26,c156
2,14,8,(-2,-6),044,1,040,2,026,1,05C,01E,012,2,054,1,01A,2,030,
05C,14,8,(-4,-3),0
*0016E,27,c222
2,14,8,(-2,-9),064,1,05C,01E,020,012,054,2,027,1,012,016,01A,01E,
2,040,07C,14,8,(-4,-3),0
*0016F,31,c133
2,14,8,(-2,-7),044,1,03C,01E,010,022,2,024,1,04C,2,028,054,1,012,
016,01A,01E,2,040,05C,14,8,(-4,-3),0
*00170,28,c235
2,14,8,(-2,-6),064,1,05C,01E,020,012,054,2,016,1,03C,2,028,1,034,
2,07C,050,14,8,(-4,-3),0
*00171,30,uue
2,14,04B,044,1,03C,01E,010,022,2,024,1,04C,2,074,018,1,02C,2,028,
1,024,2,07C,050,14,8,(-4,-3),0
*00179,25,c141
```

```
2,14,8,(-2,-6),064,1,040,8,(-4,-6),040,2,038,074,1,021,2,08C,030,
14,8,(-4,-3),0
*0017A,28,c171
2,14,04B,044,1,040,04A,040,2,3,2,8,(-5,10),1,021,2,029,8,(9,-10),
4,2,14,8,(-4,-3),0
*0017B,32,c189
2,14,8,(-2,-6),064,1,040,8,(-4,-6),040,2,084,028,1,3,4,01A,01E,
012,016,4,4,2,040,08C,14,8,(-4,-3),0
*0017C,34,c190
2,14,04B,044,1,040,04A,040,2,3,2,8,(-4,11),3,4,1,01A,01E,012,016,
4,4,2,8,(8,-11),4,2,14,8,(-4,-3),0
*0017D,29,c166
2,14,8,(-2,-8),064,1,040,8,(-4,-6),040,2,028,074,1,012,2,01A,1,
016,2,050,08C,14,8,(-4,-3),0
*0017E,27,c167
2,14,8,(-2,-6),044,1,040,04A,040,2,028,054,1,012,2,01A,1,016,2,
050,06C,14,8,(-4,-3),0
*00410,21,ucra
2,14,8,(-2,-6),1,024,043,04D,02C,2,047,1,040,2,02E,14,8,(-4,-3),0
*00411,24,ucrb
2,14,8,(-2,-6),1,064,030,01C,014,038,03C,030,01E,01C,01A,038,2,
060,14,8,(-4,-3),0
*00412,29,ucrv
2,14,8,(-2,-6),1,030,012,014,016,028,2,020,1,012,014,016,038,2,
010,1,06C,2,050,14,8,(-4,-3),0
*00413,17,ucrg
2,14,8,(-2,-6),1,064,040,01C,2,05C,020,14,8,(-4,-3),0
*00414,23,ucrd
2,14,8,(-2,-6),01C,1,014,050,064,028,04B,02C,050,01C,2,014,020,
14,8,(-6,-3),0
*00415,25,ucre
2,14,8,(-2,-6),1,064,040,2,048,03C,1,020,2,028,03C,1,040,2,020,
14,8,(-4,-3),0
*00416,22,ucr!
2,14,8,(-2,-6),062,2,038,1,06C,2,038,064,1,06E,2,020,
14,8,(-6,-3),0
*00417,27,ucr!
2,14,8,(-2,-6),014,1,01E,020,012,014,016,018,010,012,014,016,028,
01A,2,060,05C,14,8,(-4,-3),0
*00418,20,ucri
2,14,8,(-2,-6),1,042,04C,064,2,048,1,06C,2,060,14,8,(-4,-3),0
*00419,24,ucrikr
2,14,8,(-2,-6),1,042,04C,064,2,018,1,028,2,018,1,06C,2,060,
14,8,(-4,-3),0
*0041A,23,ucrk
2,14,8,(-2,-6),1,064,2,040,1,03A,018,2,010,1,03E,2,020,
14,8,(-4,-3),0
*0041B,17,ukrl
2,14,8,(-2,-6),1,010,063,010,06C,2,020,14,8,(-5,-3),0
*0041C,17,ucrm
2,14,8,(-2,-6),1,064,04D,043,06C,2,020,14,8,(-4,-3),0
*0041D,22,ucrn
2,14,8,(-2,-6),1,064,2,03C,1,040,2,034,1,06C,2,020,14,8,(-4,-3),0
*0041E,23,ucro
2,14,8,(-2,-6),014,1,044,012,020,01E,04C,01A,028,016,2,060,01C,
14,8,(-4,-3),0
```

```
*0041F,16,ucrp
2,14,8,(-2,-6),1,064,040,06C,2,020,14,8,(-4,-3),0
*00420,19,ucrr
2,14,8,(-2,-6),1,064,030,01E,01C,01A,038,2,06F,14,8,(-4,-3),0
*00421,23,ucrs
2,14,8,(-2,-6),040,014,1,01A,028,016,044,012,020,01E,2,02E,03C,
14,8,(-4,-3),0
*00422,19,ucrt
2,14,8,(-2,-6),064,1,040,2,028,1,06C,2,040,14,8,(-4,-3),0
*00423,23,ucru
2,14,8,(-2,-6),014,1,01E,020,012,054,04C,038,016,034,2,060,06C,
14,8,(-4,-3),0
*00424,25,ucrf
2,14,8,(-2,-6),020,1,064,018,01A,02C,01E,020,012,024,016,018,2,
040,06C,14,8,(-4,-3),0
*00425,22,ucrx
2,14,8,(-2,-6),1,8,(4,6),2,048,1,8,(4,-6),2,020,14,8,(-4,-3),0
*00426,21,ucr!
2,14,8,(-2,-6),1,064,06C,040,064,06C,010,01C,2,014,020,
14,8,(-5,-3),0
*00427,19,ucrch
2,14,8,(-2,-6),064,1,03C,01E,030,044,06C,2,020,14,8,(-4,-3),0
*00428,21,ucrsh
2,14,8,(-2,-6),1,064,06C,030,044,04C,030,064,06C,2,020,
14,8,(-6,-3),0
*00429,24,ucr!
2,14,8,(-2,-6),1,064,06C,030,044,04C,030,064,06C,010,01C,014,2,
020,14,8,(-7,-3),0
*0042A,23,ucr'
2,14,8,(-2,-6),054,1,014,010,06C,030,012,014,016,038,2,060,03C,
14,8,(-5,-3),0
*0042B,24,ucrs
2,14,8,(-2,-6),1,030,012,014,016,038,03C,064,2,050,1,06C,2,020,
14,8,(-5,-3),0
*0042C,21,ucr]
2,14,8,(-2,-6),1,030,012,014,016,038,03C,064,2,060,06C,
14,8,(-4,-3),0
*0042D,25,ucr'
2,14,8,(-2,-6),014,1,01E,020,012,024,028,020,024,016,028,01A,2,
060,05C,14,8,(-4,-3),00,
*0042E,26,ucr!
2,14,8,(-2,-6),1,064,03C,010,024,012,010,01E,04C,01A,018,016,024,
2,050,03C,14,8,(-4,-3),0
*0042F,22,ucrya
2,14,8,(-2,-6),1,022,020,044,038,01A,02C,01E,030,02C,2,020,
14,8,(-4,-3),0
*00430,25,lcra
2,14,8,(-2,-6),014,1,024,012,020,01E,014,04C,014,01A,028,016,2,
060,01C,14,8,(-4,-3),0
*00431,22,lcrb
2,14,8,(-2,-6),044,030,1,038,04C,030,012,016,038,2,02C,060,
14,8,(-4,-3),0
*00432,24,lcrv
2,14,8,(-2,-6),1,044,020,10,(1,-36),028,030,10,(1,-36),038,2,060,
14,8,(-4,-3),0
*00433,16,lcrg
```

```
2,14,8,(-2,-6),1,044,030,2,04C,020,14,8,(-3,-3),0
*00434,24,lcrd
2,14,8,(-2,-6),01C,1,014,010,034,012,010,04C,028,030,01C,2,014,
020,14,8,(-4,-3),00,
*00435,20,lcre
2,14,04B,024,1,030,012,016,028,01A,02C,01E,020,2,030,
14,8,(-4,-3),0
*00436,23,lcrg
2,14,8,(-2,-6),1,042,2,048,1,04E,2,028,1,044,2,040,04C,
14,8,(-4,-3),0
*00437,25,lcrz
2,14,8,(-2,-6),034,1,012,020,01E,01A,018,010,01E,01A,028,016,2,
060,01C,14,8,(-4,-3),0
*00438,17,lcri
2,14,8,(-2,-6),044,1,04C,042,04C,2,020,14,8,(-4,-3),0
*00439,23,lcrii
2,14,8,(-2,-6),044,1,04C,042,04C,2,044,018,1,028,2,050,04C,
14,8,(-4,-3),0
*0043A,19,lcrk
2,14,8,(-2,-6),1,044,02C,020,022,02A,02E,2,020,14,8,(-4,-3),0
*0043B,16,lcrl
2,14,8,(-2,-6),1,043,020,04C,2,020,14,8,(-4,-3),0
*0043C,17,lcrm
2,14,8,(-2,-6),1,044,02E,022,04C,2,020,14,8,(-4,-3),0
*0043D,18,lcrn
2,14,8,(-2,-6),1,044,02C,040,024,04C,2,020,14,8,(-4,-3),0
*0043E,25,lcro
2,14,04B,14,8,(0,-2),014,1,024,012,020,01E,02C,01A,028,016,2,060,
01C,14,8,(-4,-3),0
*0043F,16,lcrp
2,14,8,(-2,-6),1,044,040,04C,2,020,14,8,(-4,-3),0
*00440,20,lcrr
2,14,8,(-2,-6),1,044,030,01E,01C,01A,038,2,060,01C,14,8,(-4,-3),0
*00441,23,lcrs
2,14,8,(-2,-6),040,014,1,01A,028,016,024,012,020,01E,2,020,03C,
14,8,(-4,-3),0
*00442,18,lcrt
2,14,8,(-2,-6),020,1,044,028,040,2,020,04C,14,8,(-4,-3),0
*00443,22,lcru
2,14,8,(-2,-6),014,1,01E,020,012,034,02C,028,026,2,060,04C,
14,8,(-4,-3),0
*00444,25,lcrf
2,14,8,(-2,-6),020,1,044,018,01A,01C,01E,020,012,014,016,018,2,
040,04C,14,8,(-4,-3),0
*00445,20,lcrh
2,14,04B,14,8,(0,-2),1,042,2,048,1,04E,2,020,14,8,(-4,-3),0
*00446,21,lcrc
2,14,8,(-2,-6),044,1,04C,030,044,04C,010,01C,2,014,020,
14,8,(-4,-3),0
*00447,18,lcrch
2,14,8,(-2,-6),044,1,03C,030,034,04C,2,020,14,8,(-3,-3),0
*00448,21,lcrsh
2,14,8,(-2,-6),1,044,04C,020,024,02C,020,044,2,04C,020,
14,8,(-4,-3),0
*00449,24,lcrshch
2,14,8,(-2,-6),1,044,04C,020,024,02C,020,044,04C,010,01C,2,014,
```

```
020,14,8,(-5,-3),0
*0044A,21,lcrtvznak
2,14,8,(-2,-6),044,1,010,04C,020,012,016,028,2,050,02C,
14,8,(-4,-3),0
*0044B,24,lcryyy
2,14,8,(-2,-6),1,044,02C,020,01E,01A,028,2,040,1,044,2,04C,020,
14,8,(-4,-3),0
*0044C,19,lcrmznak
2,14,8,(-2,-6),1,044,02C,020,01E,01A,028,2,050,14,8,(-3,-3),0
*0044D,25,lcreee
2,14,8,(-2,-6),014,1,01E,020,012,014,028,020,014,016,028,01A,2,
060,03C,14,8,(-4,-3),0
*0044E,26,lcryu
2,14,8,(-2,-6),1,044,02C,010,014,012,010,01E,02C,01A,018,016,014,
2,050,02C,14,8,(-4,-3),0
*0044F,22,lcrya
2,14,8,(-2,-6),1,022,018,016,012,030,02C,028,020,02C,2,020,
14,8,(-4,-3),0
*020A0,4,keuroRef2
7,0020AC,0
*020A7,32,kpes
2,14,06B,14,010,1,064,020,01E,01C,01A,028,2,8,(4,3),1,05C,01E,
012,2,025,1,028,2,03E,020,14,8,(-6,-3),0
*020AC,45,keuro
3,2,2,14,8,(-4,-12),080,024,1,01C,01A,048,026,044,5,044,022,040,01E,01C
6,2,8,(-1,-1),5,1,050,6,2,024,1,060,2,8,(6,-7),14,8,(-8,-6),4,2,0
*02126,24,komega
2,14,8,(-2,-6),1,010,014,025,024,012,020,01E,02C,02B,01C,010,2,
020,14,8,(-4,-3),0
*02205,28,kdiam
2,14,8,(-2,-6),012,1,016,024,012,020,01E,02C,01A,028,2,01B,1,063,
2,010,03D,03C,14,8,(-4,-3),0
*0221E,18,kinfin
2,14,04B,034,1,01E,022,01E,01A,026,01A,2,06F,14,8,(-4,-1),0
*02264,20,kleq
2,14,8,(-2,-6),014,1,040,2,054,1,049,04F,2,02E,14,8,(-4,-2),0
*02302,16,ktri
2,14,04B,1,024,022,02E,02C,048,2,060,14,8,(-4,-3),0
```

Big Font Descriptions

Some languages, such as Japanese, use text fonts with thousands of non-ASCII characters. In order for drawings to contain such text, AutoCAD supports a special form of shape definition file called a *Big Font* file.

Some languages, such as Japanese, use text fonts with thousands of non-ASCII characters. In order for drawings to contain such text, AutoCAD supports a special form of shape definition file called a *Big Font* file.

Define a Big Font

Special codes in the first line of a Big Font file specify how to read two-byte hexidecimal codes.

A font with hundreds or thousands of characters must be handled differently from a font containing the ASCII set of up to 256 characters. In addition to using more complicated techniques for searching the file, AutoCAD needs a way to represent characters with two-byte codes as well as one-byte codes. Both situations are addressed by the use of special codes at the beginning of a Big Font file.

The first line of a Big Font shape definition file must be as follows:

```
*BIGFONT nchars,nranges,b1,e1,b2,e2,...
```

where *nchars* is the approximate number of character definitions in this set; if it is off by more than about 10 percent, either speed or file size suffers. You can use the rest of the line to name special character codes (escape codes) that signify the start of a two-byte code. For example, on Japanese computers, Kanji characters start with hexadecimal codes in the range 90-AF or E0-FF. When the operating system sees one of these codes, it reads the next byte and combines the two bytes into a code for one Kanji character. In the `*BIGFONT` line, *nranges* tells how many contiguous ranges of numbers are used as escape codes; *b1*, *e1*, *b2*, *e2*, and so on, define the beginning and ending codes in each range. Therefore, the header for a Japanese Big Font file might look like this:

```
*BIGFONT 4000,2,090,0AF,0E0,0FF
```

After the `*BIGFONT` line, the font definition is just like a regular AutoCAD text font, except that character codes (shape numbers) can have values up to 65535.

Define an Extended Big Font File

To reduce the size of composite Kanji characters, you can define an extended Big Font file. Extended big fonts use the subshape code, followed immediately by a 0.

The first line of an extended Big Font file is the same as the regular Big Font file. This is the format for the remaining lines of the file:

```
*0,5,font-name
character-height, 0, modes, character-width,0
.
.
.
*shape-number,defbytes,shape-name
.
code,0,primitive #,basepoint-x,basepoint-y,width,height,
.
.
```

```
code,0,primitive#,basepoint-x,basepoint-y,width,height,
.
terminator
```

The following list describes the fields of a Big Font definition file:

character height

Used along with character width to indicate the number of units that define the font characters.

character width

Used along with character height to indicate the number of units that define the font characters. The `character-height` and `character-width` values are used to scale the primitives of the font. In this context, primitives are the points, lines, polygons, or character strings of the font geometrically oriented in two-dimensional space. A Kanji character consists of several primitives used repeatedly in different scales and combinations.

modes

The *`modes`* byte should be 0 for a horizontally oriented font and 2 for a dual-orientation (horizontal or vertical) font. The special 00E (14) command code is honored only when *`modes`* is set to 2.

shape-number

Character code.

defbytes

Byte size. It is always 2 bytes, consisting of a hexadecimal or a combination of decimal and hexadecimal codes.

shape-name

Character name.

code

Shape description special code. It is always 7 so that it can use the subshape feature.

primitive#

Reference to the subshape number. It is always 2.

basepoint-x

X origin of the primitive.

basepoint-y

Y origin of the primitive.

width

Scale of the width of the primitive.

height

Scale of the height of the primitive.

terminator

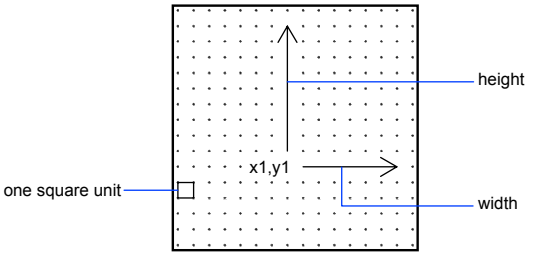
End-of-file indicator for the shape definition. It is always 0.

To arrive at the scale factor, AutoCAD scales down the primitive to a square unit and then multiplies it by the height and width to get the shape of the character. Character codes (shape numbers) in the Big Font shape definition file can have values up to 65535. The following table describes the fields of the extended Big Font file.

Fields of the extended Big Font file

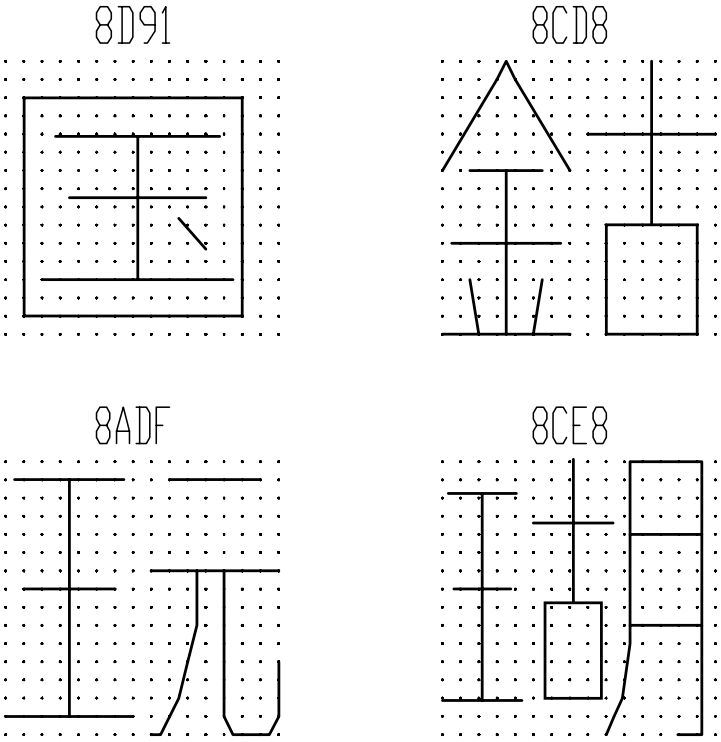
Variable	Value	Byte size	Description
shape-number	xxxx	2 bytes	Character code
code	7,0	2 bytes	Extended font definition
primitive#	xxxx	2 bytes	Refer to subshape number
basepoint-x		1 byte	Primitive *X* origin
basepoint-y		1 byte	Primitive *Y* origin
width		1 byte	Scale of primitive width
height		1 byte	Scale of primitive height
terminator	0	1 byte	End of shape definition

The following figure is an example of a 16 x 16 dot matrix that you could use to design an extended Big Font, such as a Kanji character. In the example, the distance between each dot is one unit. The callout points to a square unit.



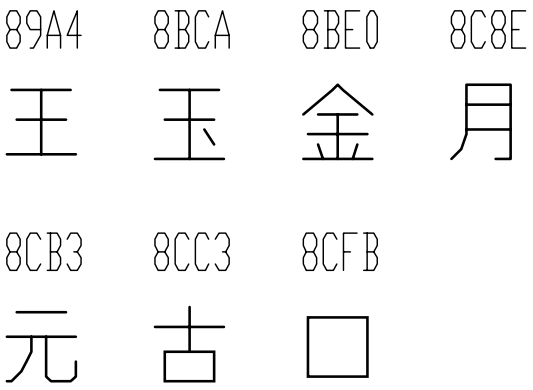
A square matrix for a Kanji character

The following figure shows examples of Kanji characters. Each character occupies an M x N matrix (matrices don't have to be square), similar to the one shown in the previous figure. The numbers above each figure are the associated shape numbers.



Examples of Kanji characters

The following figure shows Kanji primitives.



Examples of Kanji primitives

NOTE Not all fonts are defined in a square matrix; some are defined in rectangular matrices.

Example: Shape Definition File for an Extended Big Font

```
*BIGFONT 50,1,080,09e
*0,5,Extended Font
15,0,2,15,0
*08D91,31,unspecified
2,0e,8,-7,-15,
7,0,08cfb,0,0,16,16,7,0,08bca,2,3,12,9,
2,8,18,0,2,0e,8,-11,-3,0
*08CD8,31,unspecified
2,0e,8,-7,-15,
7,0,08be0,0,0,8,16,7,0,08cc3,8,0,8,16,
2,8,18,0,2,0e,8,-11,-3,0
*08ADF,31,unspecified
2,0e,8,-7,-15,
7,0,089a4,0,0,8,16,7,0,08cb3,8,0,8,16,
2,8,18,0,2,0e,8,-11,-3,0
*08CE8,39,unspecified
2,0e,8,-7,-15,
7,0,089a4,0,1,5,14,7,0,08cc3,5,2,5,14,7,0,08c8e,9,0,7,
16,2,8,18,0,2,0e,8,-11,-3,0
*089A4,39,primitive
2,0e,8,-7,-15,2,8,1,14,1,0c0,
2,8,-11,-6,1,0a0,2,8,-12,-7,1,
0e0,2,8,-7,13,1,0dc,2,8,11,-1,
2,0e,8,-11,-3,0
*08BCA,41,primitive
2,0e,8,-7,-15,2,8,1,14,1,0c0,
2,8,-11,-6,1,0a0,2,8,-12,-8,1,
```

```
0e0,2,0e5,1,0ec,2,063,1,8,
2,-3,2,06f,2,0e,8,-11,-3,0
*08BE0,81,primitive
2,0e,8,-7,-15,2,8,3,9,1,080,
2,8,-10,-4,1,0c0,2,8,-13,-5,1,
0e0,2,8,-7,9,1,09c,2,8,-1,14,
1,8,-6,-5,2,8,8,5,1,8,6,-5,
2,8,-11,-6,1,8,1,-3,2,8,7,3,
1,8,-1,-3,2,8,-3,15,1,01a,2,
012,1,01e,2,8,10,-14,2,0e,8,
-11,-3,0
*08C8E,44,primitive
2,0e,8,-7,-15,2,8,3,15,1,090,0fc,038,
2,8,-6,11,1,090,2,8,-9,-5,1,
090,2,096,1,0ac,8,-1,-3,01a,01a,2,8,
18,0,2,0e,8,-11,-3,0
*08CB3,61,primitive
2,0e,8,-7,-15,2,042,1,02b,02a,018,2,
0d0,1,012,034,2,069,1,01e,040,2,8,
-8,6,1,02b,2,8,4,5,1,08c,2,8,
-3,8,1,03c,2,8,-5,3,1,0e0,2,8,
-12,5,1,0a0,2,8,6,-14,2,0e,8,
-11,-3,0
*08CC3,34,primitive
2,0e,8,-7,-15,2,0c1,1,06c,0a8,064,0a0,2,8,
-5,9,1,09c,2,8,-7,5,1,0e0,2,8,
4,-11,2,0e,8,-11,-3,0
*08CFB,22,primitive
2,0e,8,-7,-15,2,0d2,1,0cc,0c8,0c4,0c0,2,8,
5,-13,2,0e,8,-11,-3,0
```

Use Big Font Text in a Drawing

To use a Big Font for drawing text, you set up a text style and then specify the name of the Big Font file.

To use a Big Font for drawing text, you must set up a text style by using the STYLE command and then specify the name of the Big Font file. The same text style can use a normal ASCII font as well; enter only the two file names, separated by a comma. The following example uses the command line version of the STYLE command. To enable Big Fonts from the Text Style dialog box, choose the Use Big Font option.

Command: –**style**
Enter name of text style or [?] <*current*>: *style_name*
Specify full font name or font file name (TTF or SHX): **txt,greek**

AutoCAD assumes that the first name is the normal font and that the second is the big font.

If you enter only one name, AutoCAD assumes it is the normal font and removes any associated Big Font.

By using leading or trailing commas when specifying the font file names, you can change one font without affecting the other, as shown in the following table.

Input for changing fonts	
Input	**Result**
normal, big	Both normal and Big Font specified
normal,	Normal font only (Big Font unchanged)
,big	Big Font only (normal font unchanged)
normal	Normal font only (if necessary, Big Font removed)
ENTER (null response)	No change

When you use the STYLE command to list styles or to revise an existing style, AutoCAD displays the normal font file, a comma, and the Big Font file. If the style has only a Big Font file, it is displayed with a leading comma: ,greek.

For each character in a text string, AutoCAD searches the Big Font file first. If the character is not found there, the normal font file is searched.

Use a Big Font to Extend a Font

To include special symbols in text strings, you can use a Big Font instead of extending a standard text font.

In some drafting disciplines, many special symbols can appear in text strings. The AutoCAD standard text fonts can be extended to include special symbols. However, extending standard text fonts has several limitations:

- The number of shapes is 255 per font file.
- Standard character set uses almost half the available shape numbers. Only codes 1 through 9, 11 through 31, and 130 through 255 are available.
- Multiple text fonts require duplication of the symbol definitions in each font.
- Special symbols require that you enter %% ***nnn***, where *nnn* is the symbol's shape number.

The Big Font mechanism avoids these problems. You can select one or more seldom-used characters, such as the tilde (~) or the vertical bar (|), as an escape code, and use the next character to select the appropriate special symbol. For instance, you can use the following Big Font file to draw Greek letters by entering a vertical bar (|, ASCII code 124) followed by the equivalent Roman

letter. Because the first byte of each character is 124, the character codes are biased by 124 x 256, or 31744.

```
*BIGFONT 60,1,124,124
*0,4,Greek
above, below, modes, 0
*31809,n,uca
. . . uppercase Alpha definition, invoked by "|A"
*31810,n,ucb
. . . uppercase Beta definition, invoked by "|B"
*31841,n,lca
. . . lowercase Alpha definition, invoked by "|a"
*31842,n,lcb
. . . lowercase Beta definition, invoked by "|b"
*31868,n,vbar
. . . vertical bar definition, invoked by "||"
. . .
```

Unicode Font Descriptions

Unicode shape definition files are virtually identical in format and syntax to regular AutoCAD shape definition files. The main difference is in the syntax of the font header.

The standard AutoCAD fonts correspond to the character mapping used by the host operating system. This is because characters are stored directly in the database in the format in which they are obtained from the keyboard. The same character codes are used to generate fonts. This becomes a problem when using accented (8-bit) characters for which many character encoding standards exist.

Due to character mapping limitations, AutoCAD provides a set of Unicode fonts for the various code pages that AutoCAD uses. These fonts, while essentially the same, have some characters located in different places, depending on the code page they are defined for. If the font encoding used does not match that of the text in the drawing, the wrong characters may be drawn.

With Unicode fonts, text strings are converted to Unicode *before* being drawn, so it is no longer necessary to provide additional fonts for other languages or platforms. A single Unicode font, due to its large character set, is capable of supporting all languages and platforms. This feature is transparent to the user because the drawings are, if needed (due to differing code pages), converted to the AutoCAD system code page at load time. The drawings are always saved in the AutoCAD system code page.

NOTE Unicode does not provide adequate support for all Asian languages, so Big Fonts are still used by some or all Asian versions.

Unicode shape definition files are virtually identical in format and syntax to regular AutoCAD shape definition files. The main difference is in the syntax of the font header as shown in the following code:

```
*UNIFONT,6,font-name
above,below,modes,encoding,type,0
```

The *font-name*, *above*, *below*, and *modes* parameters are the same as in regular fonts. The remaining two parameters are defined as follows:

encoding

Font encoding. Uses one of the following integer values.

0 Unicode

1 Packed multibyte 1

2 Shape file

type

Font embedding information. Specifies whether the font is licensed. Licensed fonts must not be modified or exchanged. Bitcoded values can be added.

0 Font can be embedded

1 Font cannot be embedded

2 Embedding is read-only

Another important, and sometimes confusing, difference is the handling of the code 7 subshape reference. If a shape description includes a code 7 subshape reference, the data following the code 7 is interpreted as a two-byte value. This affects the total number of data bytes (*defbytes*) in the shape description header. For example, the following shape description is found in the *romans.shp* file:

```
*00080,4,keuroRef
7,020AC,0
```

The second field in the header represents the total number of bytes in the shape description. If you are not used to working with Unicode font descriptions, you may be inclined to use three bytes rather than four, but this would cause an error during the compiling of the SHP file. This is true even if the shape number you are referencing is not in the two-byte range (below 255); the compiler always uses two bytes for this value, so you must account for that in the header.

The only other difference between Unifont shape definitions and regular shape definitions is the shape numbers. The Unifont shape definitions that AutoCAD provides use hexadecimal shape numbers as opposed to decimal values. Although hexadecimal numbers are not required, their use makes it easier to cross-reference the shape numbers with the `\U+` control character values.

Superscripts and Subscripts in SHX Files

You can modify shape definition files to improve their ability to display superscripts and subscripts.

The AutoCAD SHX fonts have limited superscript and subscript capabilities. However, it is relatively easy to modify shape definition files to improve superscript and subscript capability.

Creating superscripts and subscripts requires two steps. First, the "imaginary pen" that is creating the text, vector by vector, on your screen needs to be shifted up or down. Then, the font "scale" needs to be reduced. In addition, the reverse process has to take place to return to the normal font. The font needs to recognize four new keys: two for superscripts and two for subscripts. To avoid altering the existing font definitions, you can access these with the numeric keypad on your keyboard.

To add superscript and subscript definitions to a font

This example procedure is based on the AutoCAD Romans font file, although a similar method applies to any AutoCAD font. This procedure adds four new shape definitions to a font: super_on, super_off, sub_on, and sub_off, which control the position and size of the characters that follow. For simplicity, this example replaces the left- and right-bracket characters ([and]) and the left and right curly brace characters ({and}) with the new characters. You may choose to replace other characters or use a shape number in the extended range (ASCII codes 128 through 256). If you use an extended shape number, you need to use the %%*nnn* method (where *nnn* is the ASCII value of the character) for placing the new characters.

1. Edit your SHP file with an ASCII text editor.
2. Search for the shape definitions of the characters you are replacing. To comment out those definitions so the new definitions can take their place, insert a semicolon in front of each line of the shape definition. The shape definition may continue for a number of lines.

   The left- and right-bracket characters have ASCII values of 91 and 93 (05B and 05D hex values, if the font is Unicode). The left and right curly brace characters have ASCII values of 123 and 125 (07B and 07D hex).
3. Add the first and second values on the second line of the definition, and divide the total by 2 as shown in the following example:

   ```
   *UNIFONT,6,Extended Simplex Roman for UNICODE
   21,7,2,0 21 + 7 = 28, then 28 / 2 = 14. This number is used
   later.
   ```
4. Add the following lines to the end of the SHP file:

```
*91,8,super_on
2,8,(0,14),003,2,1,0
*93,8,super_off
2,004,2,8,(0,-14),1,0
*123,8,sub_on
2,8,(0,-14),003,2,1,0
*125,8,sub_off
2,004,2,8,(0,14),1,0
```

Notice the 14 and –14 values in the preceding lines. They are *Y* axis offsets for the imaginary pen. The value 14 is half the maximum height of a character in this font, which is the correct approximation for superscripts and subscripts. This value needs to be calculated for each font file, but you can modify it any way you want.

5 Save the file.

6 Use the COMPILE command to compile the SHP file.

Once the shape is compiled and an appropriate style is defined, you can access the new pen-up and pen-down commands by entering the [,], {, and } characters. The [character initiates superscript and the] character returns from superscript to normal. The { character initiates subscript and the } character returns from subscript to normal.

Index

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X