UTILITYDRAFTER.COM · PDF HELP

AutoCAD 2024–2026 Command Macro Guide

Valid for AutoCAD 2024, 2025 & 2026, AutoCAD LT and AutoCAD for Mac (see the compatibility table).

From your first button to DIESEL toggles and AutoLISP hybrids, with a complete symbol lexicon.

```
^C^C_.-LAYER;_M;E-OH-PRI;_C;1;;;_.PLINE;
```


1. Why Use Macros
2. Basic Macro Programming
3. Advanced Macro Creation
4. Lexicon & Symbols
5. Quick Reference Card

Interactive Macro Builder, Decoder & exports: **utilitydrafter.com/macro-guide.html#builder**

Anthony Squires · Senior Drafter · November 2026

For reference and education only. Verify macros in your own AutoCAD environment before production use. Not affiliated with Autodesk.

1 — Why You Should Use Macros

Every drafter repeats the same handful of sequences hundreds of times a week: set a layer, start a command, pick an option, type the same value. Each step is a chance to grab the wrong layer, forget an option, or burn seconds you don't get back. A macro records that sequence once and replays it perfectly every time.

⏱ Speed

A typical "set layer + start command" takes 8–15 keystrokes and 2 dialogs. A macro does it in one click. 40 uses a day × 10 seconds saved = almost 7 minutes a day, or about **28 hours a year** per drafter, from one button.

📏 Standards enforcement

The macro always puts conductors on `E-OH-PRI`, color 1, with the right linetype. Nobody has to remember the layer list. That makes it easier to pass a CHECKSTANDARDS review and keeps drawings consistent across the whole team.

⚠ Fewer errors

Wrong layer, wrong scale, wrong block, forgotten `ATTREQ` setting: macros remove the human step where mistakes happen. If the macro is right once, it's right forever.

🎓 Faster onboarding

New hires don't need to memorize your company's 14-step pole-symbol workflow. They click the *Insert Pole* button. The macro holds the knowledge.

🧰 No programming required

If you can type it at the Command prompt, you can macro it. Macros are just the Command-prompt conversation written as one line. That makes them the easiest automation to learn in AutoCAD.

📦 Portable & shareable

Macros live in CUIx files, tool palettes, and the Command Macros palette (2024–2026). Put them in a shared partial or enterprise CUIx and the whole office gets the same buttons.

💻 Works in AutoCAD LT

Macros run in full AutoCAD *and* AutoCAD LT for Windows. LT 2024+ also runs AutoLISP, so even hybrid LISP macros work across your licenses.

🤖 AutoCAD helps you find them

The **Macro Advisor** in AutoCAD 2024, 2025 and 2026 (`COMMANDMACROS`) watches the command sequences you repeat and suggests macros. Save them, edit them, and add them to the ribbon's *Automation* tab.

Version compatibility: AutoCAD 2024, 2025 & 2026

Command-macro syntax hasn't changed across these three releases, so every macro, symbol and export on this page works the same in **AutoCAD 2024, 2025 and 2026**. The differences are in *which product* you run (full AutoCAD, LT, or Mac), not which year.

Feature	2024	2025	2026	AutoCAD LT (Win)	AutoCAD for Mac
Core macro symbols (`^C ; \ _ . - * ' ^Z ^P ^Q ^R`, toggles)	✔	✔	✔	✔	✔ (Customize dialog)
`^]` select-if-needed (added in 2023)	✔	✔	✔	✔	—
DIESEL `$M=` expressions	✔	✔	✔	✔	✔
AutoLISP inside macros / Builder `.lsp` export	✔	✔	✔	✔ LT 2024+	✔ AutoCAD only (no LT for Mac)
Scripts / Builder `.scr` export	✔	✔	✔	✔	✔

Feature	2024	2025	2026	AutoCAD LT (Win)	AutoCAD for Mac
`CUI` editor, partial & enterprise CUIx	✔	✔	✔	✔	Customize dialog instead
Tool palette command strings	✔	✔	✔	✔	—
Command Macros palette / Macro Advisor (`COMMANDMACROS`)*	✔	✔	✔	✖	✖
Action Recorder (`.actm`)	✔	✔	✔	✖	✔
`UNDEFINE` (why the `.` prefix matters)	✔	✔	✔	✔ LT 2024+	✔

* The Macro Advisor (introduced in AutoCAD 2023) is a **My Insights** feature of full AutoCAD for Windows. It isn't in the AutoCAD-based toolsets, and the *Insights* tab needs a commercial named-user subscription with "My Insights – Macro Advisor" assigned to your Autodesk account. Saved macros, the Command Macro Editor and *Add to Ribbon* (Automation tab) work the same in 2024, 2025 and 2026.

💡 **Mixed-version office?** Write macros with the `_.` prefixes and dash commands taught here, and a single partial CUIx or `.lsp` file works on 2024, 2025 and 2026 machines side by side. If a prompt sequence ever differs between releases, the `^R` (command versioning) symbol is the escape hatch.

Macros vs. the other automation tools

Macros are one of five ways to automate AutoCAD. Pick the simplest one that does the job:

Tool	What it is	Best for	Limits
Command alias (.PGP)	A short name for *one* command (`L` = LINE)	Starting a single command faster	No options, no values, no sequences
Command macro	A one-line string of commands, options, values and pauses	Buttons, palettes, menus, double-click actions, shortcut keys	Mostly linear, with no loops; logic is limited to DIESEL/LISP snippets
Script (.SCR)	A text file of Command-prompt input, run with `SCRIPT`	Batch jobs across many drawings, drawing setup	No pauses for user input
Action Recorder (.ACTM)	Records what you do and plays it back	Quick personal recordings	Full AutoCAD only (not LT); harder to edit and share than a text macro
AutoLISP (.LSP)	A full programming language	Loops, logic, math, reading/writing data, new commands	Has to be loaded (APPLOAD/acaddoc.lsp); there's more to learn

💡 **Rule of thumb:** if the task is "do these same steps, maybe let me pick a point," it's a **macro**. If it needs "for each…" or "if this, then that," graduate to AutoLISP. The online Macro Builder exports both.

Where macros live (AutoCAD 2024–2026)

Location	How to get there	Notes
Ribbon button	`CUI` → Customize tab → Ribbon → Panels	Most visible; good for team tools
Quick Access Toolbar	`CUI` → Quick Access Toolbars	Always on screen, whatever ribbon tab is showing
Tool palette command tool	`TOOLPALETTES` (Ctrl+3) → tool Properties → *Command string*	Easy to drag and share; see Tool Palettes
Right-click (shortcut) menu	`CUI` → Shortcut Menus	Context-sensitive (Default, Edit, Command modes)
Keyboard shortcut	`CUI` → Keyboard Shortcuts → Shortcut Keys	Macros assigned to keys **cannot use** `\` **pauses**

Location	How to get there	Notes
Double-click action	`CUI` → Double Click Actions	Runs a macro when you double-click a given object type
Command Macros palette (2024–2026)	`COMMANDMACROS` → Saved tab	Stored in `commandmacros.cuix`; "Add to Ribbon" puts it on the Automation tab

2 — Basic Macro Programming

BEGINNER

The one idea you need: a macro is the Command-prompt conversation, typed out

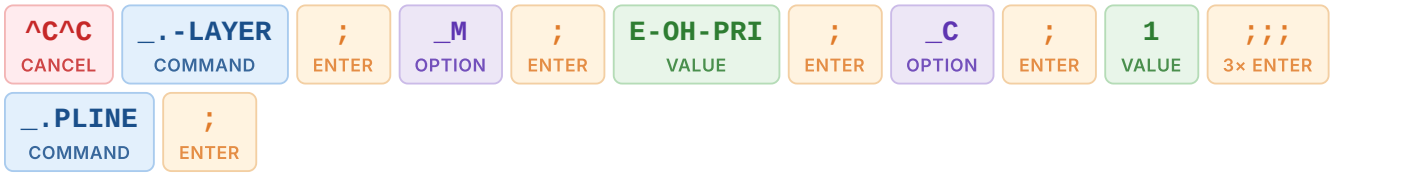
When you run a command, AutoCAD asks questions (prompts) and you answer them. A macro writes your answers down in order, with a symbol for every Enter and every "let the user pick." That's all it is.

Example: draw a circle with radius 5 wherever the user clicks.

AutoCAD asks…	You would…	Macro piece
Command:	Press Esc twice to clear anything running	`^C^C`
Command:	Type CIRCLE, press Enter	`_.CIRCLE;`
Specify center point for circle…	Click a point	`\` (pause)
Specify radius of circle…	Type 5, press Enter	`5;`

```
^C^C_.CIRCLE;\5;
```

Anatomy of a macro



- **Cancel** (`^C^C`): starts from a clean Command prompt. It equals pressing Esc twice. Two are needed to back out of dimension commands and nested prompts.
- **Command**: the command name. The `-` in `-LAYER` calls the *command-line* version, so no dialog box opens. Dialogs stop macros.
- **Option**: the capital letters of an option, e.g. `M` for Make, `C` for Color.
- **Value**: a typed answer: a layer name, a number, a coordinate.
- **Enter** (`;`): every answer has to be "entered." Extra semicolons accept defaults and exit commands.

Translate the Command prompt step by step: the "write the conversation" method

This is how pros build every macro. Run the command manually at the Command prompt, press **F2** to open the text window, and write down each prompt and your answer.

#	Prompt shown	Your answer	Macro
1	Command:	-LAYER	`_.-LAYER;`
2	Enter an option [?/Make/Set/New/.../Color/...]:	M	`_M;`
3	Enter name for new layer (becomes the current layer) <0>:	E-OH-PRI	`E-OH-PRI;`
4	Enter an option [?/Make/Set/...]:	C	`_C;`
5	New color [Truecolor/COlorbook]:	1	`1;`
6	Enter name list of layer(s) for color 1 <E-OH-PRI>:	(Enter = accept)	`;`
7	Enter an option [?/Make/Set/...]:	(Enter = exit LAYER)	`;`
8	Command:	PLINE	`_.PLINE;`

String the "Macro" column together and put `^C^C` in front:

```
^C^C_.-LAYER;_M;E-OH-PRI;_C;1;;;_.PLINE;
```

The 10 beginner rules

1. **Start with `^C^C`** so the macro never types into a command that's already running.
2. **Use the dash version of commands** (`-LAYER`, `-INSERT`, `-PURGE`, `-HATCH`, `-PLOT`) so no dialog box opens. A macro can't click inside a dialog.
3. **Use `;` for Enter**, not spaces. A space is Enter *most* of the time, but inside text it's a real space, and you can't see it at the end of a line. Semicolons are visible and predictable.
4. **Use `\` to pause** for the user to pick a point, type a value or select objects.
5. **Prefix commands with `_.`**: `_` makes it work in any language version, and `.` uses AutoCAD's built-in command even if someone redefined it.
6. **Prefix options with `_`** too (`_M`, `_E`, `_W`). Don't put it on your own values like layer names.
7. **Know the end-of-macro space.** AutoCAD adds a space (Enter) to the end of every macro, *unless* the macro ends in `;`, `\` or a control character such as `^Z`. So `^C^C_.LINE` and `^C^C_.LINE;` both start LINE.
8. **Use forward slashes in file paths** (`C:/CAD/Blocks/pole.dwg`). A backslash is a pause.
9. **Avoid spaces in names and paths** (they act as Enter). If you must use them, use the AutoLISP form (see Advanced).
10. **Test at the Command prompt first.** Type the sequence by hand. If it works by hand, it'll work as a macro.

Create your first macro button in the CUI (AutoCAD 2024–2026)

1. Type `CUI` and press Enter. The **Customize User Interface** editor opens.
2. On the **Customize** tab, in the **Command List** pane (lower left), click the **Create a new command** button (the star icon).
3. In the **Properties** pane on the right, set **Name** (e.g. *Overhead Primary PLINE*) and **Description** (the tooltip).
4. Click in **Macro** and replace the default `^C^C` with your full macro. For long macros, click **[...]** to open the Long String Editor.
5. Optional: assign a **Small Image** / **Large Image** (16×16 and 32×32 BMP/PNG) and an **Element ID** like `UD_OHPRI`.
6. Drag the new command from the Command List onto a **Ribbon → Panel**, a **Quick Access Toolbar**, a **Shortcut Menu**, or under **Keyboard Shortcuts → Shortcut Keys**.
7. Click **Apply**, then **OK**. Your button is live. Click it and watch the Command prompt to check each step.

📎 **Faster option: Tool Palettes.** Open `TOOLPALETTES` (Ctrl+3), drag any existing command (for example LINE from a ribbon button) onto a palette, right-click it → **Properties**, and paste your macro into **Command string**. See the Tool Palettes guide.

🆕 **AutoCAD 2024–2026: Command Macros palette.** Type `COMMANDMACROS`. The *Insights* tab shows macros AutoCAD suggests from your own command history (named-user commercial license required). *Try* or *Save* them, then right-click a saved macro → **Edit Macro** to paste in your own string, or → **Add to Ribbon** to place it on the *Automation* tab. These macros are stored in `commandmacros.cuix`.

Starter macros to try today

Macro	What it does
`^C^C_.ZOOM;_E;`	Zoom Extents
`^C^C_.UCS;_W;_.PLAN;;`	Reset to World UCS and plan view
`*^C^C_.ERASE;_SI;`	Erase one picked object at a time, repeating until you press Esc
`^C^C_.-LAYER;_M;E-DIM;;_.DIMLINEAR;`	Make/set layer E-DIM, then start a linear dimension
`^C^C_.CIRCLE;\5;`	Circle with radius 5 at a picked center
`^C^C_.-PURGE;_A;*;_N;`	Purge all unused items with no confirmation prompts
`^C^C_.LAYFRZ;`	Freeze the layer of whatever you pick

Debugging a macro that "doesn't work"

- **Press F2** and read the Command history line by line. The macro stops at the exact prompt where its answer didn't fit.
- **A dialog opened?** Use the dash command, or set `FILEDIA` to 0 for file prompts. Remember to set it back to 1.
- **Too many or too few `;`?** Count the prompts again. Command options change between releases and settings (e.g., the Y-scale prompt in `-INSERT` disappears for uniformly scaled blocks).
- **Attribute prompts appeared?** That's `ATTREQ` / `ATTDIA`. Either answer them in the macro or set `ATTREQ` to 0 first.
- **Works for you but not a co-worker?** Add `_` and `.` prefixes, and check they have the same layers, blocks and support paths.

3 — Advanced Macro Creation

ADVANCED

Repeating macros: `*^C^C`

A leading asterisk makes the macro repeat until you press Esc or pick another macro that starts with `^C`. Pair it with **Single** selection mode so each pick runs the command once:

```
*^C^C_.ERASE;_SI;          ← erase one object per pick, forever
*^C^C_.MOVE;_SI;           ← pick one object, then base/second point; repeats
*^C^C_.ROTATE;_SI;         ← pick one object, then base point/angle; repeats
```

> ⚠ The `*` must be the very first character, and you can't use it to repeat *options*. It also doesn't work in image tile menus.

Transparent commands: `'`

An apostrophe runs a command *inside* another command without cancelling it, for example zooming or changing a setting mid-LINE. Never put `^C` in front of a transparent macro, because that would cancel the command you're trying to help.

```
'_.ZOOM;_E;                ← zoom extents without leaving the current command
'_ORTHOMODE;1;             ← set a system variable transparently
```

Selection-set tricks

Input	Meaning	Example
`_SI`	Single: select once, then continue automatically	`^C^C_.MOVE;_SI;\\\`
`_P`	Previous selection set	`^C^C_.CHPROP;_P;;_LA;E-DEMO;;`
`_L`	Last object created	`^C^C_.CHPROP;_L;;_C;1;;`
`_ALL`	Everything not on frozen/locked layers	`^C^C_.-PURGE;_A;*;_N;`
`\` at Select objects	Each `\` accepts one pick or one window; add `;` to finish selecting	`^C^C_.ERASE;\;`
`^]`	Prompt to select only if nothing is preselected (2023+)	`^C^C^]_.CHPROP;_P;;_C;3;;`

Coordinates in macros

- `@10,0`: relative to the last point. `@10<45`: relative polar (distance<angle).
- `#0,0`: absolute coordinate even when Dynamic Input would make it relative.
- `0,0` after an option like `_.MOVE;_L;;0,0;\` moves the last object from the origin to a picked point.
- Object snap overrides work as inputs: `_END;\` forces Endpoint for that pick, `_MID;\` Midpoint, `_NON;` no snap.

```
^C^C_.LINE;_END;\@0,-10;;   ← line from a picked endpoint straight down 10 units
```

Echo control: `^P`, `^Q`, `MENUECHO`, `CMDECHO`

Macros "type" into the Command prompt, so every step echoes. `^P` toggles `MENUECHO` to hide the noise, and `^Q` keeps the macro's prompts and input out of the Command history. In LISP-based macros, set `CMDECHO` to 0 and restore it afterwards.

```
^C^C^P_.-LAYER;_M;E-OH-SEC;_C;5;;;^P_.PLINE;
```

Suppressing the automatic final space: `^Z`

AutoCAD appends a space (Enter) to any macro that doesn't end in `;`, `\` or a control character. Usually that's what starts the last command. When it isn't wanted, especially after a DIESEL test that might return nothing (an extra Enter would *repeat the last command*), end with `^Z`.

DIESEL: logic inside a macro with `$M=`

DIESEL (Direct Interpretively Evaluated String Expression Language) lets a macro make decisions. `$M=` tells AutoCAD "evaluate the rest of this as DIESEL." DIESEL functions are written `$(function,arg1,arg2,…)`, with **commas and no spaces**.

Toggle any 0/1 system variable: subtract the current value from 1.

```
^C^C_GRIDMODE;$M=$(-,1,$(getvar,gridmode))
^C^C_MIRRTEXT;$M=$(-,1,$(getvar,mirrtext))
^C^C_FILLMODE;$M=$(-,1,$(getvar,fillmode))
```

If/then, e.g. switch between Model and Paper space inside a layout (the trailing space inside each result is the Enter, because `^Z` suppresses the automatic one):

```
^C^C$M=$(if,$(=,$(getvar,cvport),1),_.MSPACE ,_.PSPACE )^Z
```

Cycle text height 0.08 → 0.10 → 0.12:

```
^C^C_TEXTSIZE;$M=$(if,$(=,$(getvar,textsize),0.08),0.10,$(if,$(=,$(getvar,textsize),0.10),0.12,0.08))
```

DIESEL function	Does	Example → result
`$(getvar,name)`	Reads a system variable	`$(getvar,clayer)` → `E-OH-PRI`
`$(if,test,then,else)`	Branch: test is true if non-zero	`$(if,$(getvar,gridmode),ON,OFF)`
`$(=,a,b)` `$(!=,a,b)`	Numeric equal / not equal (1 or 0)	`$(=,$(getvar,tilemode),0)`
`$(<,a,b)` `$(>,a,b)` `$(<=,a,b)` `$(>=,a,b)`	Numeric comparisons	`$(>,$(getvar,ltscale),1)`
`$(eq,a,b)`	String equal	`$(eq,$(getvar,clayer),0)`
`$(+,..)` `$(-,..)` `$(*,..)` `$(/,..)`	Math (up to 9 arguments)	`$(*,$(getvar,dimscale),0.125)`
`$(and,..)` `$(or,..)` `$(xor,..)`	Bitwise logic, useful for bit-coded sysvars	`$(and,$(getvar,osmode),16384)`
`$(fix,n)`	Truncates to an integer	`$(fix,48.7)` → `48`
`$(rtos,n,mode,prec)`	Number to text (mode 2 = decimal, 4 = architectural)	`$(rtos,$(getvar,ltscale),2,0)`
`$(angtos,a,mode,prec)`	Angle to text	`$(angtos,$(getvar,snapang),0,2)`
`$(edtime,time,format)`	Formats a date	`$(edtime,$(getvar,date),MO/DD/YYYY)`
`$(strlen,s)` `$(substr,s,start,len)` `$(upper,s)`	String length / substring / uppercase	`$(substr,$(getvar,dwgname),1,6)`

DIESEL function	Does	Example → result
`$(index,n,list)` `$(nth,n,a,b,..)`	Picks an item from a comma list / argument list	`$(nth,1,A,B,C)` → `B`
`$(getenv,name)`	Reads a registry environment value	`$(getenv,MaxArray)`
`$(linelen)`	Width of the status display, in characters	Rarely needed
`$(eval,str)`	Evaluates a string as DIESEL	Advanced nesting

`edtime` picture codes: `D` `DD` `DDD` (Mon) `DDDD` (Monday), `M` `MO` `MON` `MONTH`, `YY` `YYYY`, `H` `HH` `MM` `SS` `MSEC`, `AM/PM`.

> 🏷 **DIESEL in menu labels:** in classic pull-down/shortcut menu *names*, a DIESEL expression that returns `!.` shows a check mark and `~` greys the item out, e.g. `$(if,$(getvar,gridmode),!.)Grid`. Ribbon buttons don't evaluate label DIESEL.

AutoLISP inside macros (hybrid macros)

Full AutoCAD and LT 2024+ evaluate AutoLISP expressions typed at the Command prompt, so a macro can include them. That gets around most macro limits: spaces in paths, prompts, math, and restoring settings.

```
;; Ask for a layer name once, then make it current
^C^C(setq lyr (getstring T "Layer name: "));_.-LAYER;_M;!lyr;;

;; Insert a block whose path contains spaces (macros can't do this alone)
^C^C(command "_.-INSERT" "C:/CAD Library/Utility/POLE-WOOD.dwg" pause 1 1 0)

;; Load a LISP file, then run its command
^C^C(load "C:/CAD/Lisp/poletag.lsp");POLETAG;

;; Date-stamp text using DIESEL from inside LISP (assumes text style height = 0)
^C^C(command "_.TEXT" pause "" "0" (menucmd "M=$(edtime,$(getvar,date),MO/DD/YYYY)"))

;; Set LTSCALE & DIMSCALE from one scale factor the user types
^C^C(setq sf (getreal "Scale factor (e.g. 48): "));_LTSCALE;!sf;_DIMSCALE;!sf;
```

- `!variable` feeds the value of a LISP variable into any prompt.
- `pause` inside `(command ...)` works like `\`, and `""` works like `;`.
- Avoid backslashes inside LISP code in a macro (such as `"\n"` in prompts). The macro reader can treat them as pauses. Use forward slashes in paths and keep prompts simple.

Undo-safe macros

Wrap multi-step macros in an undo group so one `U` reverses the whole thing:

```
^C^C_.UNDO;_BE;_.-LAYER;_M;E-POLE;;_.-INSERT;POLE-WOOD;\1;;0;_.UNDO;_E;
```

Running scripts from a macro

```
^C^C_FILEDIA;0;_.SCRIPT;C:/CAD/Scripts/sheet-setup.scr;
```

Set `FILEDIA` back to 1 as the *last line of the script*. Anything after `SCRIPT` in the macro would be fed into the script's prompts.

Deploying macros to a team

- **Partial CUIx**: in `CUI` → **Transfer** tab, create a new file (e.g. `UtilityDrafter.cuix`), drag your commands, ribbon panels and toolbars into it, and save it to a network folder. Each user loads it with `CUILOAD`, or you attach it as a partial customization file.
- **Enterprise CUIx**: a read-only, company-wide CUIx set in **OPTIONS → Files → Customization Files → Enterprise Customization File**. Users get your buttons and can't break them.
- **Tool palettes**: export a palette (`.XTP`) or point everyone's *Tool Palettes File Locations* at a shared folder.
- **LISP wrappers**: a macro wrapped as `(defun c:OHPRI ...)` in an auto-loaded `.lsp` becomes a typed command. The online Macro Builder (utilitydrafter.com/macro-guide.html#builder) generates this for you.

Common advanced pitfalls

> • **Keyboard shortcuts can't pause.** `\` isn't supported in macros assigned to shortcut keys. Use a LISP wrapper instead.
> • **Prompts change with settings.** `-INSERT` asks for Y scale only if the block allows non-uniform scale; attribute prompts depend on `ATTREQ`; `TEXT` asks for height only if the style's height is 0.
> • **Layer prompt nesting.** Autodesk notes that some `-LAYER` option states need `^C^C^C` to fully cancel. Use three cancels if a layer macro misbehaves after another layer macro.
> • **Don't `^C` a transparent macro.** Transparent macros start with `'`, not `^C^C`.
> • **Command versioning (`^R`).** If a macro written for an old release breaks in a newer release (e.g. a 2024 macro in 2026) because a command's prompts changed, `^R` turns command versioning on so the older prompt sequence is used where it's supported.

4 — Lexicon: Every Macro Symbol Explained

Every character in a macro means something, even a space. Use this table as your decoder ring.

Symbol	Name	What it does	Example	Level
`^C`	Cancel	Same as pressing Esc. Cancels the active command or option.	`^C_.LINE`	BASIC
`^C^C`	Double cancel	Standard macro opener. Two cancels are needed to back out of dimensioning and nested prompts. Some `-LAYER` states need `^C^C^C` .	`^C^C_.CIRCLE;\5;`	BASIC
`;`	Enter	Presses Enter: submits a value, accepts a default, or ends a command. No space is added after a trailing `;` .	`_.PLAN;;`	BASIC
`(space)`	Spacebar	Same as pressing the Spacebar. It acts as Enter at most prompts but types a real space inside text strings. Use `;` for clarity.	`^C^C_.circle \5`	BASIC
`\`	Pause	Waits for user input: a point, distance, value, or one pick/window at *Select objects* (follow it with `;` to end selection). Not allowed in keyboard-shortcut macros.	`_.LINE;\\;`	BASIC
`_`	Global name	Uses the English (global) name of a command or option, so the macro works in every language pack.	`_.ZOOM;_E;`	BASIC
`.`	Built-in command	Uses AutoCAD's standard command even if it was changed with `UNDEFINE` or redefined by LISP. Combine as `_.` or `._` .	`_.ERASE`	BASIC
`-`	Command-line version	Runs the command without its dialog box, which macros need. Not every command has one.	`_.-LAYER` `_.-INSERT`	BASIC
`*`	Repeat	As the first character (`*^C^C`), repeats the macro until Esc or another `^C` macro runs. Can't repeat options.	`*^C^C_.ERASE;_SI;`	ADVANCED
`'`	Transparent	Runs a command inside another command without cancelling it. Don't put `^C` in front.	`'_.ZOOM;_E;`	ADVANCED
`^Z`	Null / no final space	Stops AutoCAD from adding its automatic final space (Enter) to the end of the macro. Vital after DIESEL tests.	`…$M=$(if,…)^Z`	ADVANCED
`^P`	MENUECHO toggle	Turns macro echo off/on so the Command prompt stays clean.	`^C^C^P_.-LAYER;…;^P`	ADVANCED
`^Q`	Quiet history	Keeps the macro's prompts and input out of the Command Line history.	`^C^C^Q_.REGENALL;`	ADVANCED

Symbol	Name	What it does	Example	Level
`^]`	Select if needed	Prompts the user to select objects only if nothing is preselected, and stores the selection as *Previous* (AutoCAD 2023+).	`^C^C^]_.CHPROP;_P;;_C;1;;`	ADVANCED
`^R`	Command versioning	Turns command versioning on/off, so a macro written for an older release keeps its expected prompt sequence where supported.	`^C^C^R_.-HATCH;…`	ADVANCED
`^M`	Enter (control)	Same as `;`, written as a control character.	`_.ZOOM^M_E^M`	ADVANCED
`^I`	Tab	Presses Tab. Cycles Dynamic Input fields / object snap candidates.	`^I`	ADVANCED
`^H`	Backspace	Presses Backspace.	`^H`	ADVANCED
`^B`	Snap toggle	Snap on/off (Ctrl+B).	`^B`	ADVANCED
`^G`	Grid toggle	Grid on/off (Ctrl+G).	`^G`	ADVANCED
`^O`	Ortho toggle	Ortho on/off.	`^C^C^O_.LINE;`	ADVANCED
`^D`	Dynamic UCS toggle	Dynamic UCS on/off (Ctrl+D).	`^D`	ADVANCED
`^E`	Isoplane	Cycles to the next isometric plane (Ctrl+E, Windows).	`^E`	ADVANCED
`^V`	Next viewport	Makes the next viewport current (in macros; on the keyboard Ctrl+V is Paste).	`^V`	ADVANCED
`^T`	Tablet toggle	Digitizer tablet on/off (Windows only).	`^T`	LEGACY
`$M=`	DIESEL macro	Evaluates everything after it as a DIESEL expression and feeds the result to AutoCAD.	`_GRIDMODE;$M=$(-,1,$(getvar,gridmode))`	ADVANCED
`$( , )`	DIESEL function	A DIESEL function call: `$(name,arg,arg)`, commas only, no spaces.	`$(getvar,clayer)`	ADVANCED
`( )`	AutoLISP expression	Anything in parentheses is evaluated as AutoLISP (full AutoCAD and LT 2024+).	`(setq sf 48)`	ADVANCED
`!`	LISP variable	Returns a LISP variable's value at any prompt.	`_LTSCALE;!sf;`	ADVANCED
`@`	Relative coordinate	Measures from the last point: `@x,y` or `@dist<angle`.	`_.LINE;\@10<90;;`	BASIC
`#`	Absolute coordinate	Forces absolute coordinates when Dynamic Input defaults to relative.	`_.MOVE;_L;;#0,0;\`	ADVANCED
`<`	Polar angle	Separates distance and angle in polar input.	`@25<45`	BASIC
`,`	Coordinate separator	Separates X,Y(,Z). In DIESEL it separates arguments.	`0,0`	BASIC

Symbol	Name	What it does	Example	Level
`/`	Path separator	Use forward slashes in file paths, because `\` is a pause.	`C:/CAD/pole.dwg`	BASIC
`"`	String quotes	Delimit text inside AutoLISP expressions. Plain macros don't use quotes.	`(command "_.ZOOM" "_E")`	ADVANCED
`!.`	Check mark (label)	In a classic menu label's DIESEL, returning `!.` shows a check mark.	`$(if,$(getvar,gridmode),!.)Grid`	ADVANCED
`~`	Grey out (label)	In a classic menu label, greys out (disables) the item.	`$(if,$(=,$(getvar,tilemode),1),~)MSPACE`	ADVANCED
`=*`	Show current menu	Displays the current top-level pull-down, shortcut, or image tile menu (Windows).	`$P0=*`	LEGACY
`$Px=`	Menu swap	Legacy MNU menu-section loading/swapping.	`$P1=POP1`	LEGACY
`+`	Line continuation	Legacy MNU files: a trailing `+` continues a macro on the next line. Not needed in CUIx.	`^C^C_.LINE;+`	LEGACY

Common option keywords used in macros

Keyword	Means	Used with
`_E`	Extents	ZOOM
`_W`	World / Window	UCS / ZOOM, selection
`_M` `_S` `_N`	Make / Set / New	-LAYER
`_C` `_L` `_LW`	Color / Ltype / Lweight	-LAYER
`_F` `_T` `_LO` `_U`	Freeze / Thaw / Lock / Unlock	-LAYER
`_ON` `_OFF`	On / Off	-LAYER
`_SI`	Single selection	Any Select objects prompt
`_P` `_L` `_ALL`	Previous / Last / All	Select objects
`_A` `_R`	All / Regapps	-PURGE
`_Y` `_N`	Yes / No	Confirmation prompts
`_BE` `_E`	Begin / End	UNDO groups
`_END` `_MID` `_CEN` `_INT` `_PER` `_NEA` `_NON`	Object snap overrides	Any point prompt

Glossary

Term	Definition
CUI / CUIx	Customize User Interface editor / the XML-based file it saves (acad.cuix, partial and enterprise CUIx).
Partial CUIx	An extra CUIx loaded alongside the main one. The best way to share your macros without editing acad.cuix.
Enterprise CUIx	A company-wide, usually read-only CUIx set in OPTIONS → Files.
Element ID	The unique ID of a CUI command (e.g. `UD_OHPRI`), used to reference it across the interface.
Command string	The tool palette name for a macro.
Accelerator	A keyboard shortcut (e.g. Ctrl+Shift+P) assigned to a CUI command.

Term	Definition
DIESEL	Direct Interpretively Evaluated String Expression Language: a tiny string language for logic in macros, labels and `MODEMACRO`.
MENUECHO	System variable that controls whether macro input and prompts echo.
CMDECHO	System variable that controls echo of commands run from AutoLISP.
FILEDIA	0 = file prompts appear on the Command line instead of in dialogs (macro-friendly). Set it back to 1.
ATTREQ / ATTDIA	Whether INSERT asks for attribute values / whether it uses a dialog to do so.
Macro Advisor	AutoCAD feature (2023 and later, so all of 2024–2026) that suggests macros from your command history (`COMMANDMACROS` , `MACRONOTIFY` , `MACROINSIGHTSSUPPORT`).

5 — Quick Reference Card

Every macro symbol on one page. Print it and keep it by your keyboard.

Symbol	Meaning
`^C^C`	Cancel ×2 (clean start)
`;`	Enter
`(space)`	Spacebar / Enter
`\`	Pause for input
`_`	Global (English) name
`.`	Built-in command
`-`	Command-line version (no dialog)
`*`	Repeat (first char: *^C^C)
`'`	Transparent command
`^Z`	No automatic final Enter
`^P`	Toggle MENUECHO
`^Q`	Keep out of history
`^]`	Select if nothing preselected
`^R`	Command versioning
`^M`	Enter
`^I`	Tab
`^H`	Backspace
`^B / ^G / ^O`	Snap / Grid / Ortho
`^D / ^E / ^V`	Dyn UCS / Isoplane / Next viewport
`$M=`	DIESEL expression
`$(fn,a,b)`	DIESEL function
`( )`	AutoLISP expression
`!var`	LISP variable value
`@ # <`	Relative / Absolute / Polar
`/`	Path separator (never \)

The 5-second checklist: starts with `^C^C` · dash commands (no dialogs) · `;` for every Enter · `\` only where the user picks · forward slashes in paths · `_.` on commands · tested at the Command prompt (F2).

Build, decode and export macros (.lsp / .scr / .txt / .json) at **utilitydrafter.com/macro-guide.html#builder**