BadBit CAD

User guide — 2D drafting

BADBIT · Development version · Generated on November 23, 2026
This guide follows development: every new tool is added here with screenshots of the program itself.

Contents

Introduction
- What this manual covers
- Requirements
- Conventions

Installation and updates
- How it is installed
- Terms of use on first run
- The new version notice
- The Updates window
- When the new version takes effect
- Gradual rollout
- For company IT: turning off automatic updates

The interface
- Title bar
- Ribbon
- Options and profiles (OPTIONS, alias OP)
- File tabs
- Start screen (Start tab)
- Drawing area
- Command line
- Status bar
- Light and dark themes
- Help and About

Navigation
- REGEN (aliases RE, REA, REGENALL)
- Model viewports: VPORTS and -VPORTS
- Named views: VIEW, -VIEW, and VIEWGO

Entering points and answering commands
- Confirming, canceling, and repeating
- Coordinates
- Dynamic input
- Command options
- User coordinate system (UCS)
- Scripts (SCRIPT, alias SCR)
- Command aliases (ALIASEDIT)
- LISP (APPLOAD, alias AP)
- BadBit Scripts: the LISP and Python editor (SCRIPTEDITOR, alias SED)
- The command line in the terminal (console)

Drawing commands
- LINE (alias L)

PLINE (alias PL)
CIRCLE (alias C)
ARC (alias A)
RECTANG (alias REC)
POLYGON (alias POL)
ELLIPSE (alias EL)
SPLINE (alias SPL)
XLINE (alias XL)
RAY
POINT (alias PO)
DONUT (alias DO)
REVCLOUD
BOUNDARY (aliases BO, -BO)
DIVIDE (alias DIV)
MEASURE (alias ME)
MLINE (alias ML)
MLSTYLE
WIPEOUT

Text and dimensions

The stroke font
MTEXT (alias T, MT)
Formatting pieces of text: the Text Editor
TEXT (DTEXT, alias DT)
STYLE (alias ST)
TEXTEDIT (alias ED, DDEDIT) and double-click
Find and replace (FIND)
Dimensions
Leaders (MLEADER and QLEADER)
Geometric tolerance (TOLERANCE, alias TOL)
Center marks and centerlines (CENTERMARK, CENTERLINE)
Annotative objects and the annotation scale (CANNOSCALE)
Selecting text with the mouse in the editor
Fields (FIELD)
The color of a piece of text: the Select Color dialog
MTEXT: column break, ruler indents, reversed flow and mask in draw order
AutoCAD SHX fonts
Texts into MTEXT: TXT2MTXT
Outline-only TrueType letters: TEXTFILL
Scaling texts in place: SCALETEXT and TSCALE

Hatches and gradients

HATCH (aliases H, BH)
-HATCH (aliases -H, -BH)
GRADIENT (alias GD)
Patterns
Associative hatch

HATCHEDIT (alias HE, double-click)
Selection, grips, and editing
Draw order
Fill on the screen

Tables

TABLE (alias TB)
Typing in the cells (TABLEDIT and double-click)
Formulas
Number format
Choosing cells
Editing the whole table
TABLEEXPORT
TABLESTYLE (alias TS)
User cell styles
Rotated text in the cell
Border linetype
MATCHCELL and MATCHPROP in tables
CTABLESTYLE
Table breaks

Selection and editing

Selecting
Isolating and hiding objects (ISOLATEOBJECTS, HIDEOBJECTS, UNISOLATEOBJECTS)
Grips
MOVE (alias M)
COPY (aliases CO, CP)
ROTATE (alias RO)
SCALE (alias SC)
MIRROR (alias MI)
OFFSET (alias O)
TRIM (alias TR) and EXTEND (alias EX)
FILLET (alias F) and CHAMFER (alias CHA)
BREAK (alias BR)
Break at Point (BREAKATPOINT)
JOIN (alias J)
PEDIT (alias PE)
Join Ends (JOINENDS)
EXPLODE (alias X)
STRETCH (alias S)
ARRAY (alias AR)
LENGTHEN (alias LEN)
ALIGN (alias AL)
MATCHPROP (aliases MA, PAINTER)
OVERKILL
SPLINEDIT (alias SPE)
ERASE (alias E)

- Groups (GROUP, alias G; UNGROUP; GROUPEDIT)

Inquiry

- ID
- DIST (alias DI)
- AREA (alias AA)
- MEASUREGEOM (alias MEA)
- LIST (aliases LI, LS)

Blocks, associative arrays and regions

- BLOCK (alias B)
- -BLOCK (alias -B)
- INSERT (aliases I, -I, -INSERT)
- Exploding a reference
- BURST (Express Tools)
- Associative array
- ARRAYEDIT
- REGION (alias REG)
- UNION, SUBTRACT and INTERSECT
- MASSPROP
- Attributes
- Block Editor (BEDIT, alias BE)
- WBLOCK (alias W) and -WBLOCK (alias -W)
- BASE
- INSERT from a file
- External references
- Clipping references (XCLIP, alias XC)
- Editing a reference in place (REFEDIT)
- Purging unused blocks
- Dynamic blocks
- Creating dynamic blocks in the editor
- Exporting and importing attributes: ATTOUT and ATTIN

Calculator, DesignCenter and tool palettes

- QuickCalc (QUICKCALC, alias QC, Ctrl+8)
- CAL
- DesignCenter (ADCENTER, aliases ADC, DC, DCENTER; Ctrl+2)
- Tool palettes (TOOLPALETTES, alias TP; Ctrl+3)
- Blocks palette (BLOCKSPALETTE)

Parametric drawing

- Geometric constraints: GEOMCONSTRAINT (alias GCON)
- AUTOCONSTRAIN and CONSTRAINTINFER
- Dimensional constraints: DIMCONSTRAINT (alias DCON)
- Parameters Manager: PARAMETERS (aliases PAR, PARAM)
- Constraint bars: CONSTRAINTBAR (alias CBAR)
- Constraint Settings: CONSTRAINTSETTINGS (alias CSETTINGS)
- Editing with constraints

Constraints in DXF and DWG
Parametric drawing variables

Undo and redo
OOPS

Precision: Ortho, Polar, object snaps and tracking
Ortho (F8)
Polar (F10)
Object snaps (F3)
Acquisition, extension and tracking (F11)

Grid and units
Grid (F7)
Grid snap (F9)
Drawing units (UNITS, alias UN)

Layers
Layer Properties Manager (LAYER, LA)
Select Color
Ribbon layer list

Object properties
Properties palette (PROPERTIES, Ctrl+1)
Lineweight on screen (LWDISPLAY, LWEIGHT)
Linetypes (LINETYPE, alias LT)
Quick Properties (QP, Ctrl+Shift+P)
Multiple objects

Layer tools
Faded locked layers (LAYLOCKFADECTL)
Walking through layers (LAYWALK)
Layer Translator (LAYTRANS)
Layer states (LAYERSTATE, LAS)
Layer of the current object (LAYMCUR)
Previous layer (LAYERP, LAYERPMODE)

Color, ByLayer and names
Color of new objects (COLOR, COL)
Changing to ByLayer (SETBYLAYER)
Renaming named objects (RENAME, REN)

Files
Commands
Opening from Finder or File Explorer (double-click)
Copying and pasting between drawings
Purging the drawing (PURGE, aliases PU, -PU)
Save changes?
Saving is fast, even in large drawings
Recovery after a crash

Automatic save (SAVETIME)
Drawing recovery
Files from newer versions
Comparing drawings (DWGCOMPARE)
Importing PDF (PDFIMPORT)

Layouts and plotting
The layout tabs
LAYOUT (alias LO)
Viewports: MVIEW (alias MV)
Inside and outside the viewport: MSPACE and PSPACE (MS, PS)
Layers per viewport: VPLAYER
Plot: PLOT (Ctrl+P)
Printing to system printers
Page setup: PAGESETUP
Preview: PREVIEW (PRE)
Export: EXPORT, EXPORTPDF, PUBLISH
Still missing
Changing space: CHSPACE

DXF and DWG files
Commands
Opening a DWG
Opening a DXF
Saving as DWG
Saving as DXF
What goes natively to AutoCAD
Round trip through .bbd
External references and file blocks
Civil 3D objects
3D solids (3DSOLID, REGION, BODY) and surfaces
Known limits

LandXML (Civil 3D)
Commands
Importing (LANDXMLIN)
Exporting (LANDXMLOUT)
Performance
Not done yet

Images, PDF underlays, transmittal and password
Commands
Attaching an image (IMAGEATTACH)
Clipping (IMAGECLIP and PDFCLIP)
Adjustments (IMAGEADJUST, TRANSPARENCY, PDFADJUST)
Attaching a PDF (PDFATTACH)
The External References manager (IMAGE)
Plotting

DXF and DWG
Transmittal package (ETRANSMIT)
Password (SECURITYOPTIONS)

System variables
Dimension variables
Parametric drawing
Object groups
Properties and text scale
Center marks and centerlines
User coordinate system (UCS)
Multilines
LISP
Wipeout
View and layouts (read only)

Performance and large drawings
Drawing on the processor (no GPU)
Memory
Test neighborhood

Command and shortcut reference
Commands
Keyboard shortcuts

Introduction

BadBit CAD is the 2D drafting program of the BadBit suite. Anyone who already uses AutoCAD will find the same commands, the same aliases, the same shortcuts, and the same screen layout: the idea is that there is nothing to relearn. The look (colors, icons, typography) is BadBit's own.

What this manual covers

This manual follows the program's development: every new tool is added here together with the tool itself, with screenshots of BadBit CAD itself. It describes what exists today. The chapters follow the order of use: the interface, navigation, how to enter points, the drawing and editing commands, the precision tools (snaps, polar, tracking), the grid and units, and a reference of all commands and shortcuts.

Requirements

BadBit CAD works well in three situations, all of them optimized:

Machine	How it draws
With a dedicated graphics card	On the GPU (Metal, Vulkan, or DirectX 12).
With the processor's integrated GPU (Intel, AMD)	On the integrated GPU, through the same path.
Without a GPU (or only with the system's software adapter)	On the processor, with its own renderer that keeps the image and redraws only what changes.

The choice is automatic. The chapter *Performance and large drawings* explains how to force drawing on the processor.

Conventions

- Commands appear in capital letters: **LINE**. In a localized interface the command also has a translated name (in Portuguese, **LINHA**); the English name works in any language, with or without an underscore (`_LINE`), as in AutoCAD.
- Keys appear like this: `Enter`, `F3`, `Ctrl`+`Z` (on the Mac, `Cmd`).
- What you type on the command line appears like this: `@3,0`.
- Internally, the drawing stores everything in meters. The project unit (meter, centimeter, millimeter...) changes what you type and what you see.

Installation and updates

BadBit CAD updates itself from inside the program: nobody needs to go back to the website to download an installer. Updating works **without signing in to the BadBit Account**: the program only reads the version information published by BadBit, which is digitally signed, and checks the signature before using anything in it.

How it is installed

System	Where	First installation
macOS	`BadBit CAD.app`, wherever you put it (usually in Applications)	From the `.dmg`: drag BadBit CAD to Applications.
Windows	In your user folder (`%LOCALAPPDATA%\Programs\BadBit\CAD`), without asking for an administrator password: one folder per version and the `BadBit CAD.exe` launcher, which the shortcuts point to	With the BadBit installer.
Linux	In your user folder (`~/.local/share/badbit/programas/cad`), without asking for an administrator password, the same way as on Windows, with the `badbit-cad` launcher	With the BadBit installer, which comes in a `.tar.gz`.

> On macOS, open BadBit CAD from the Applications folder. When it is opened straight from the `.dmg` or from the Downloads folder, macOS runs the program in a protected copy and does not allow it to be replaced: the Updates window asks you to move the program to Applications first.

Installing on Linux

1. Download `BadBit-CAD-<version>-linux-x64.tar.gz` from your BadBit Account.
2. In the file manager, right-click it and choose **Extract Here**. A single file comes out, the installer `BadBit-CAD-<version>-linux-x64-setup`.
3. Double-click the installer (if the file manager asks, choose **Run**). From the terminal, the same thing: `tar xzf BadBit-CAD-<version>-linux-x64.tar.gz` and `./BadBit-CAD-<version>-linux-x64-setup`.
4. In the **Install BadBit CAD** window, check or clear **Create a desktop shortcut** and click **Install**. At the end, **Open BadBit CAD now** opens the program.

The installer adds BadBit CAD to the applications menu (with "Open with" for DWG and DXF) and makes it the program for `.bbd` files: double-clicking a `.bbd` opens the drawing. The default program for DWG and DXF stays whatever you were already using. Running the installer again with BadBit CAD installed offers **Update** (newer version) or **Reinstall** (the same one).

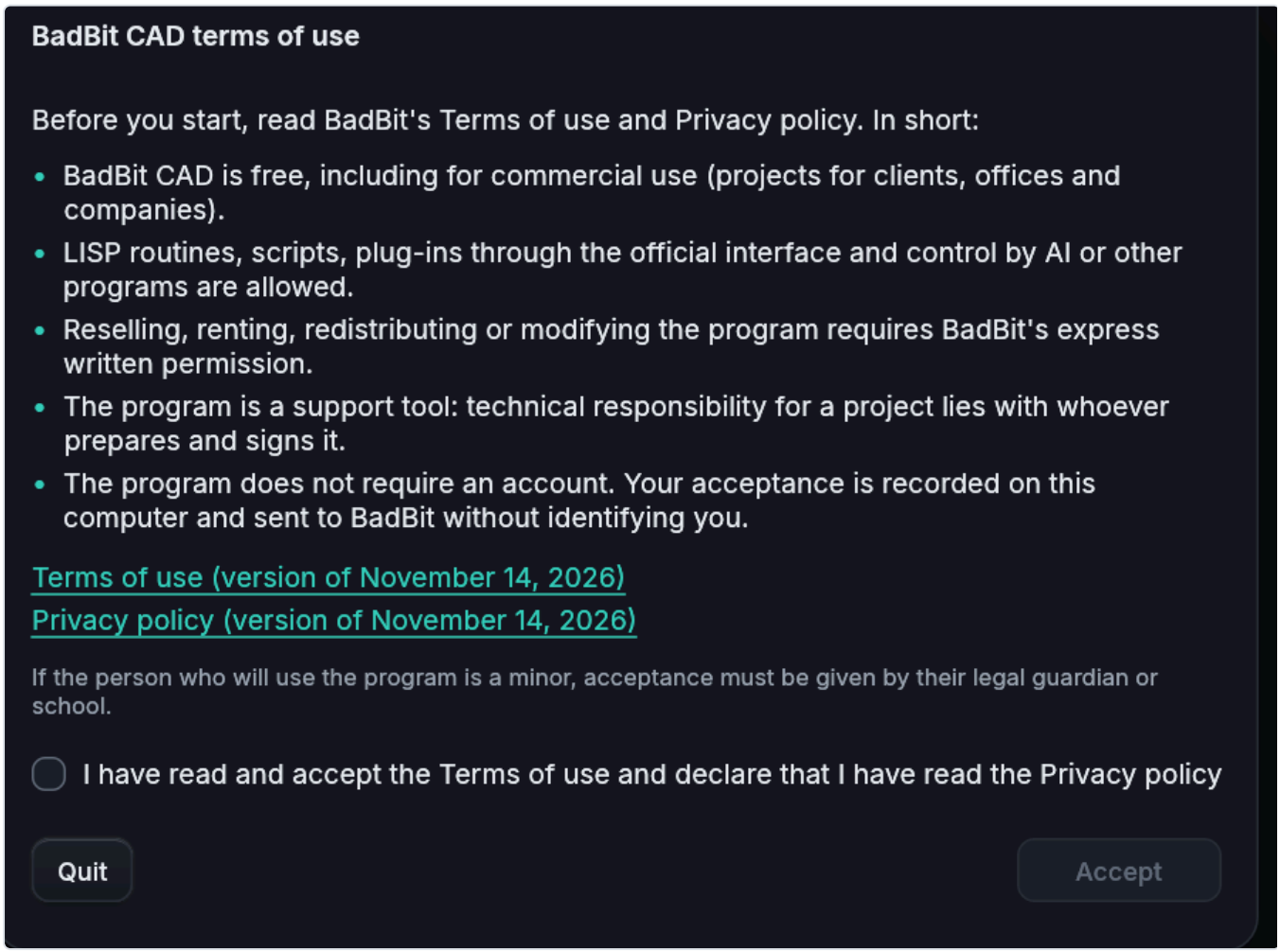
Uninstalling on Linux. Right-click BadBit CAD in the applications menu and choose **Uninstall BadBit CAD** (or run `~/.local/share/badbit/programas/cad/desinstalar-badbit-cad --desinstalar`). The window asks you to close BadBit CAD if it is open and asks whether to keep the preferences, the

profiles, and the drawing recovery (checked). Your drawings are not deleted.

Installation without a window (IT). `BadBit-CAD-<version>-linux-x64-setup --silencioso` installs without a desktop shortcut; `desinstalar-badbit-cad --desinstalar --silencioso` uninstalls and keeps the preferences. The exit code tells whether it succeeded.

BadBit CAD for Linux needs a 64-bit (x86-64) distribution with Wayland or X11 and glibc 2.31 or newer: Ubuntu 20.04, Linux Mint 20, Debian 11, Fedora 32, RHEL 9 and their later versions, as well as openSUSE, Arch, and others. The same installer serves all of them, whether they use `.deb`, `.rpm`, or other packages, because it does not go through the package manager: it installs in your home folder, without an administrator password.

Terms of use on first run



The first-run window: the summary, the links, and the acceptance box.

The first time BadBit CAD opens on a computer (for each user of the system), before any use, the **BadBit CAD terms of use** window appears, as in the installers of other CAD programs. It also applies to anyone who received a copy of the program from someone else and never went through the BadBit Account sign-up. The window shows:

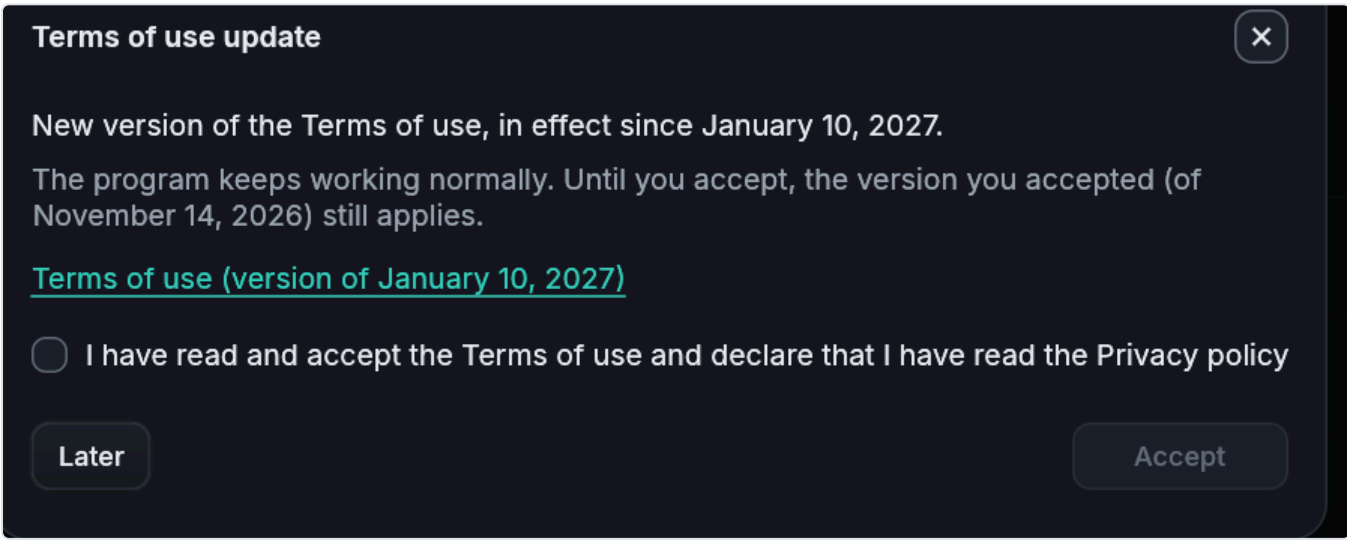
- a **summary** of the main points (the program is free, including for commercial use; automation through LISP, scripts, plug-ins, and AI is allowed; reselling, redistributing, or modifying it only with BadBit's permission; technical responsibility for the project lies with whoever signs it);
- the **Terms of use** and **Privacy policy** links, with the date of the version in effect: clicking opens the full text in the browser, in the program's language (Portuguese, English, or Spanish);
- the notice that, if the person who will use the program is a **minor**, acceptance must be given by their **legal guardian or school**;
- the **I have read and accept the Terms of use and declare that I have read the Privacy policy** box and the **Accept** (only with the box checked) and **Quit** buttons.

While the window is open, nothing of the program appears behind it. **Quit** closes BadBit CAD without recording anything, and the window comes back the next time it opens. After acceptance, it no longer appears on that computer.

When the terms change

Each new version of the Terms of use or of the Privacy policy is marked by BadBit with the size of the change, and BadBit CAD acts accordingly:

- **Minor**: nothing appears. You do not need to accept again, and the acceptance you already gave still applies.
- **Relevant**: the **Terms of use update** window, which **does not block** the program and can be left for later (below).
- **Mandatory**: from the effective date, the program only continues once you accept the new version (see *Mandatory change*, next). Before that date, it appears like a relevant change, with a notice of the date.

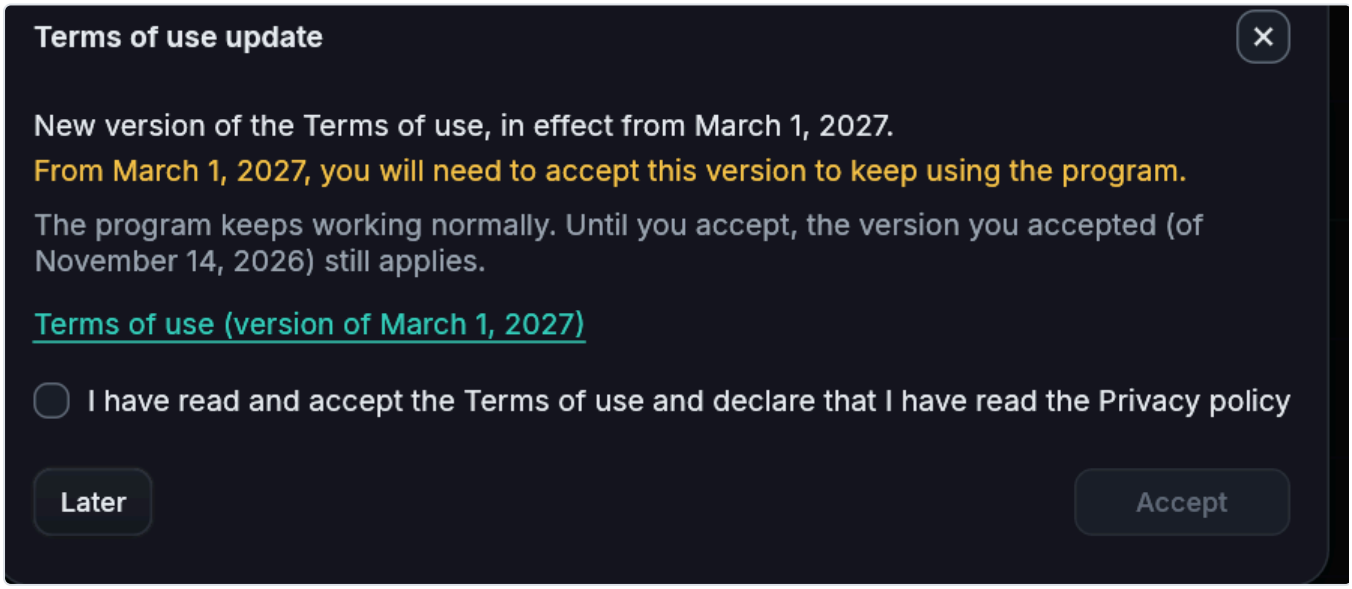


The new-version notice: the effective date, the link, and the acceptance, which can be left for later.

For a relevant change, BadBit CAD shows the **Terms of use update** window, with the date the new version comes (or came) into effect and the link to read it. It **does not block** the program: **Later** closes the notice, work continues normally, and until you accept, the version you accepted still applies. The notice opens by itself once each time the program starts and stays in the title bar (**Terms of use updated**, to the left of the **?**) until you accept; clicking it reopens the window.

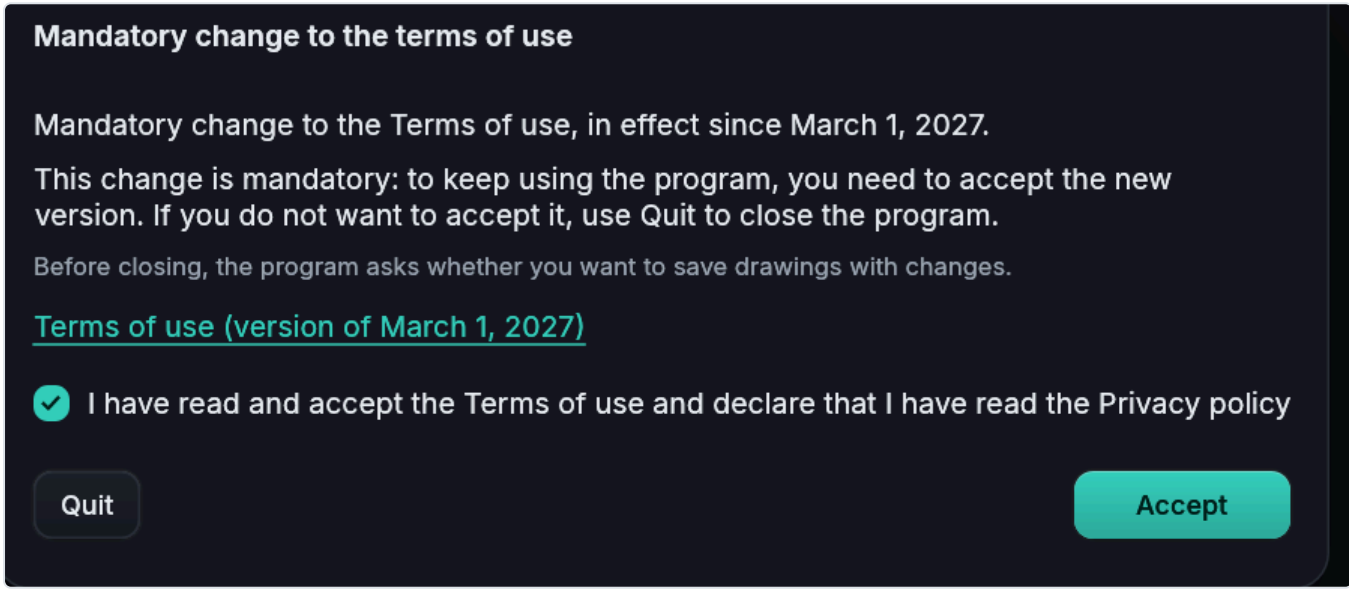
BadBit CAD learns about the new version from the same signed information it already reads for updates (no account and nothing that is not signed by BadBit); without internet, the version that came with the program applies.

Mandatory change



Before the effective date: the notice, with Later and the date from which you will need to accept.

Before the effective date, a mandatory change appears like a relevant one: the **Terms of use update** window, with **Later**, plus a highlighted line: "From March 1, 2027, you will need to accept this version to keep using the program." You can accept right away, or leave it for later and keep working until that date.



From the effective date: the window blocks the program, with only Accept and Quit.

From the effective date, the **Mandatory change to the terms of use** window blocks the program like the first-run one: it says which document changed and since when, explains that you need to accept the new version to continue, and has the link to read it, the **I have read and accept...** box and only two

buttons, **Accept** (with the box checked) and **Quit**. There is no **Later** and no × to close it, and `Esc` does not close the window. If the date arrives while the program is open, the window appears right away, on top of your work.

Quit closes BadBit CAD without recording the acceptance. Before that, as when you close the program window, it asks whether you want to save each drawing with changes (**Save changes to Drawing1?**, with **Yes**, **No** and **Cancel**); **Cancel** goes back to the terms window, which keeps blocking, and the drawings stay as they were. Closing the program window with the terms window open does the same as **Quit**. The next time the program opens, the window comes back until you accept. After **Accept**, the program continues normally and the new version is recorded.

Where the acceptance is recorded

The acceptance is stored on the computer, in the user folder (`aceites.json`): the document, the version, the jurisdiction (Brazil, or international for other countries, based on the system's country), the language, the date and time, and the program version. It is also sent to BadBit **without identifying you**: the only thing that goes with it is a random number created for this purpose (it is not the one used for the gradual rollout and has no relation to your account). Sending happens in the background and, without internet, is left for later, without getting in the way of your work. With updates turned off by IT (`"updates": "off"`), nothing is sent: the acceptance stays only on the computer.

The **About** window (**ABOUT**) shows the accepted version and the links to the Terms of use and the Privacy policy.

The new version notice



The discreet notice, to the left of the ?

in the title bar.

BadBit CAD checks for a new version when it opens and then every 4 hours, in the background: nothing freezes meanwhile. When there is one, it starts downloading right away (only the files that changed, and for each large file only the difference) and the notice **Update *version* available** appears in the title bar. Clicking the notice opens the Updates window.

The download resumes where it stopped if the internet drops or if you close the program midway: the part already downloaded is reused the next time. Each downloaded file is verified (size and SHA-256 digest of what BadBit published), including what was already stored from a previous attempt; whatever does not match is discarded and downloaded again. Right before switching versions, BadBit CAD verifies all the files of the assembled version again and, on macOS, BadBit's signature on the new program (Apple's, with the BadBit team); if anything does not match, the version is discarded and the window says so.

BadBit CAD only talks to the update server over a secure connection (`https://`) and never offers a version older than the installed one (unless BadBit has withdrawn the installed version; in that case, going back is only through the full installer). The version information published by BadBit has an

expiration date: once expired, it is not valid.

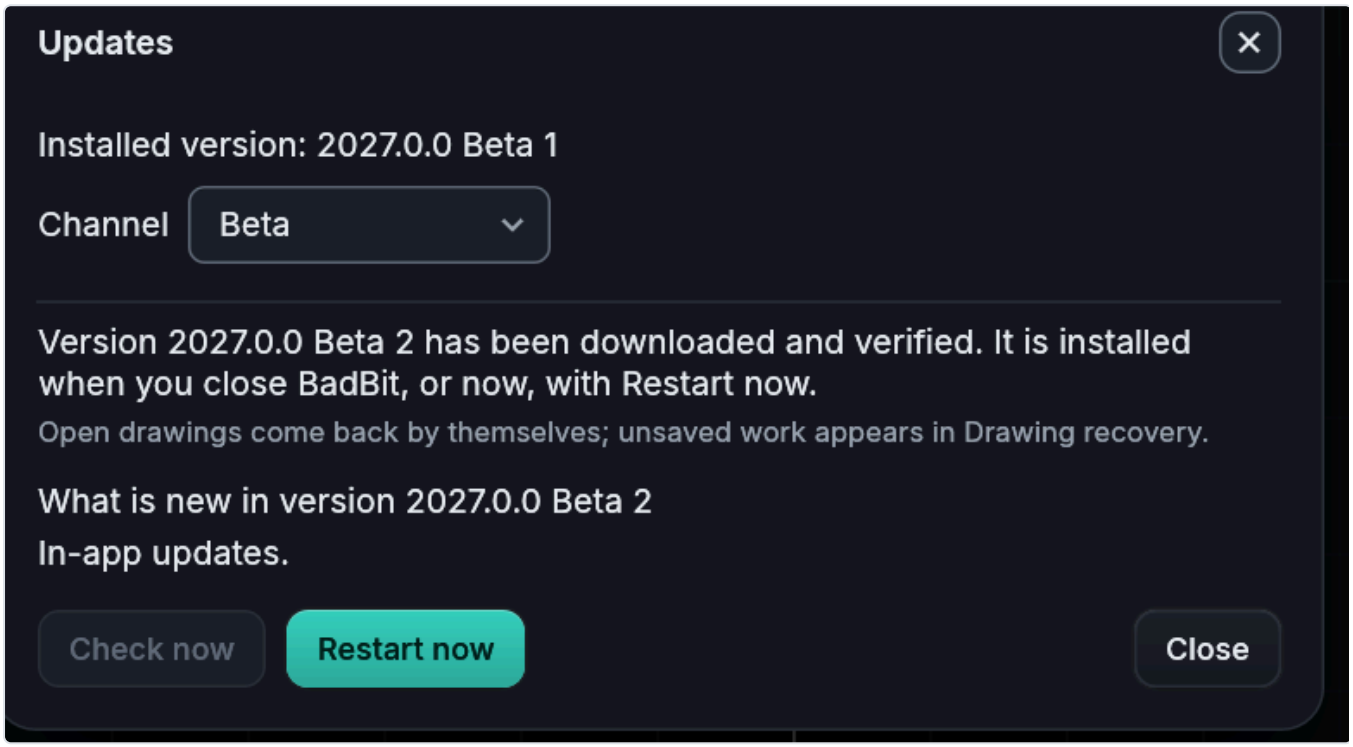
Not checked for more than 14 days



The notice when BadBit CAD has been unable to check for updates for more than 14 days.

If BadBit CAD goes more than 14 days without being able to verify BadBit's version information (no internet, a company proxy blocking it, or expired information), the notice **Updates not checked for *N* days** appears in the title bar, and the Updates window explains: check the internet connection or the company proxy, because without checking, security fixes do not arrive. The notice disappears on the first successful check. With updates turned off by IT, there is no notice; with manual updates only, the notice appears only in the window.

The Updates window



The Updates window: the new version downloaded and verified, with what is new.

It opens from the notice, from **?** > **Check for updates** (which also checks right away), and from the **About** window (**ABOUT**) > **Update**. It shows:

- the **installed version**;
- the **channel**: **Beta** (the test versions, which arrive earlier) or **Stable**. Switching channels checks again right away. The beta version of BadBit CAD starts on the Beta channel;
- the **progress**: *Checking for updates…*, *BadBit CAD is up to date.*, *Version … is available.*, *Downloading version … NN%* (with the bar), or *Version … has been downloaded and verified*;
- **what is new in the version**, in the interface language;

- the **Check now**, **Download** (when automatic updates are turned off), **Restart now** (with the version ready), and **Close** buttons.

When the new version takes effect

- **When you close BadBit CAD**: the downloaded version is installed by itself when you close the program; the next time it opens, it is already the new version.
- **Restart now**: BadBit CAD closes immediately, without asking whether to save, switches versions, and opens again. The drawings that were open **come back by themselves**: saved ones open as they were, and what was not saved comes back through **Drawing recovery** (the same as when the program crashes, chapter *Files*), with the question of whether to recover the changes.

If the new version does not open

The previous version is kept until the new one opens properly. If the new one closes with an error before it is up and running, or does not get up and running in three consecutive starts, BadBit CAD **goes back to the previous version by itself**, opens the previous one, and no longer offers that version on this computer. BadBit can also withdraw a version with a problem: anyone on it is taken to the good version on the next check.

No permission in Applications (macOS)

If your macOS account cannot write to the folder where BadBit CAD is (Applications on a managed machine, for example), the window says so and offers **Install in ~/Applications**: the new version goes to your own applications folder and opens from there; the old one stays where it was.

Gradual rollout

Each new version is released gradually: first to a portion of the computers, then to all of them. That is why a colleague may get the version a day before you. The draw uses a random number created the first time BadBit CAD checks for updates, stored only on your computer (it does not identify you and is not sent to BadBit). Required updates (security fixes) reach everyone at once.

For company IT: turning off automatic updates

IT can turn off automatic updates (and distribute the versions on its own) in two ways. The **more restrictive** of the two applies: a variable does not turn back on what the file turned off.

The machine's **policy file**, `politica.json`:

System	Path
macOS	`/Library/Application Support/BadBit/CAD/politica.json`
Windows	`%ProgramData%\BadBit\CAD\politica.json`
Linux	`/etc/badbit/cad/politica.json`

```
{
  "updates": "manual",
  "channel": "stable",
  "baseUrl": "https://espelho.empresa.local/badbit/"
}
```

- `updates`: `auto` (the default: checks and downloads by itself), `manual` (only checks when the person asks, in Check for updates, and only downloads with **Download**), or `off` (turned off: the window says IT turned it off and nothing is queried). An unreadable file counts as `manual`.

The file only applies in full if **only the administrator** can change it: on macOS and Linux, the file and its folder must belong to `root`, with no write permission for the group or for others. A file that fails this check only restricts (its `updates` applies), and its `channel` and `baseUrl` are ignored. On Windows, create the `%ProgramData%\BadBit\CAD` folder with write permission only for Administrators and SYSTEM (the BadBit installer does this); BadBit CAD does not yet check this permission on Windows. - `channel` (optional): pins the channel (`beta` or `stable`); the window shows *set by your company's IT* and does not allow switching. - `baseUrl` (optional, `https://` only): an internal mirror with the same structure as `https://downloads.badbit.com.br/` (the files published by BadBit, copied unchanged; the signature is still verified).

Environment variable `BADBIT_UPDATES` with `auto`, `manual`, or `off` (`0` and `false` count as `off`), for deployment scripts.

PDF manual. The installer does not include the manual: the program downloads it the first time someone asks for it (**Manual** button or `F1`), from `cad/releases/<version>/manual/` on the download base (BadBit's or the mirror's `baseUrl`, which must also copy that folder). On a network without internet access, copy `BadBit-CAD-Manual-pt-BR.pdf` (or the PDF for the interface language, such as `BadBit-CAD-Manual-en.pdf`) to the `manual` folder next to the executable (on macOS, `BadBit CAD.app/Contents/Resources/manual/`): the copy next to the program takes precedence over everything and nothing is downloaded.

For company IT: acceptance of the terms by the company

In a corporate installation (or in a school), the company can accept the Terms of use and declare that it has read the Privacy policy **on behalf of the people who use its machines**. Then the acceptance window, the new-version notice and the mandatory change window do not appear for anyone. This goes in the same `politica.json`:

```
{
  "termsAcceptance": {
    "acceptedBy": "Construtora Exemplo Ltda",
    "termsVersion": "2026-11-14",
    "privacyVersion": "2026-11-14"
  }
}
```

- `acceptedBy`: who accepted (the legal name of the company or school, up to 200 characters). Without it, the company's acceptance does not apply and the window appears normally.
- `termsVersion` and `privacyVersion` (optional, in the `YYYY-MM-DD` format): the versions the company accepted. Without them, the version in effect applies, including the next ones BadBit publishes.

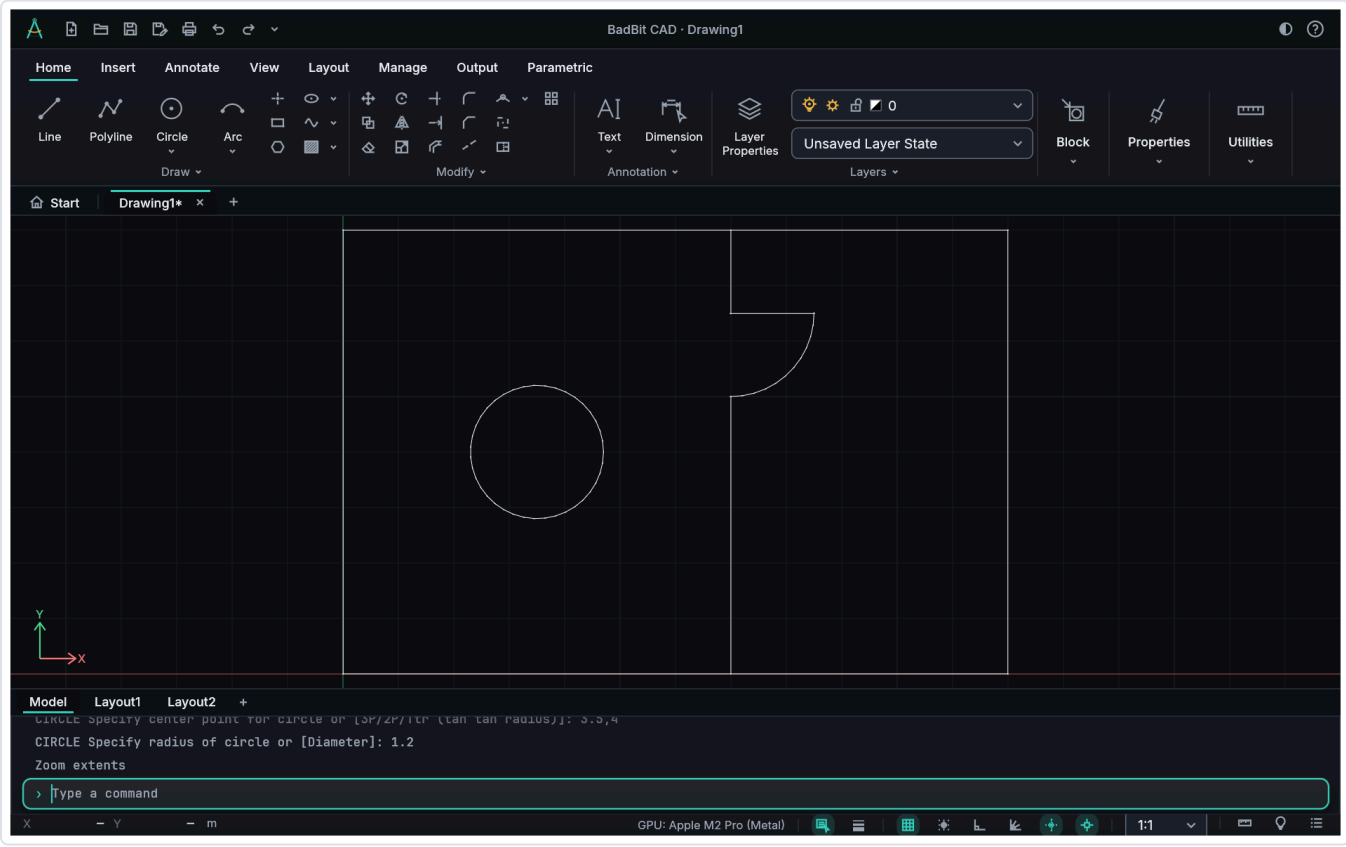
The company's acceptance only applies with the file protected (only the administrator can change it, as explained above): in a file that any user can write, or through an environment variable, nobody is exempted from accepting. BadBit CAD records the acceptance on behalf of the company (with its name) and sends it to BadBit like any other acceptance. The **About** window shows *accepted by* and the company's name.

Updater folders and variables

What	Where
Updater state (downloads in progress, versions that did not open, the gradual-rollout number)	macOS: `~/Library/Application Support/BadBit/CAD/atualizacao`; Windows: `%LOCALAPPDATA%\BadBit\CAD\atualizacao`; Linux: `~/.local/share/badbit/cad/atualizacao`
`BADBIT_UPDATE_URL`	Changes the address from which BadBit CAD fetches the versions. Only accepts `https://`; the local server (`http://127.0.0.1`) only works in a development build. A rejected address is ignored and noted in the log.
Updater log	`registro.log` in the state folder: the queries that failed, the discarded versions, and what the switch helper did.
`BADBIT_UPDATE_STATE`	Changes the updater's state folder (development and testing).
Record of the terms acceptances (`aceites.json`)	macOS: `~/Library/Application Support/BadBit`; Windows: `%APPDATA%\BadBit`; Linux: `~/.config/badbit`
`BADBIT_CONSENT_DIR`	Changes the folder of the acceptance record (development and testing).
`BADBIT_API_URL`	Changes the address to which the acceptance is sent. Only accepts `https://`; the local server (`http://127.0.0.1`) only works in a development build.

> In the browser there is no updater: the new version arrives when the page is reloaded. A development build (the program running outside an installation) only reports that there is a new version; the switch is through the installer.

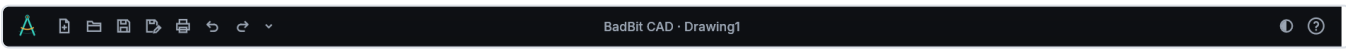
The interface



The BadBit CAD window with a simple floor plan.

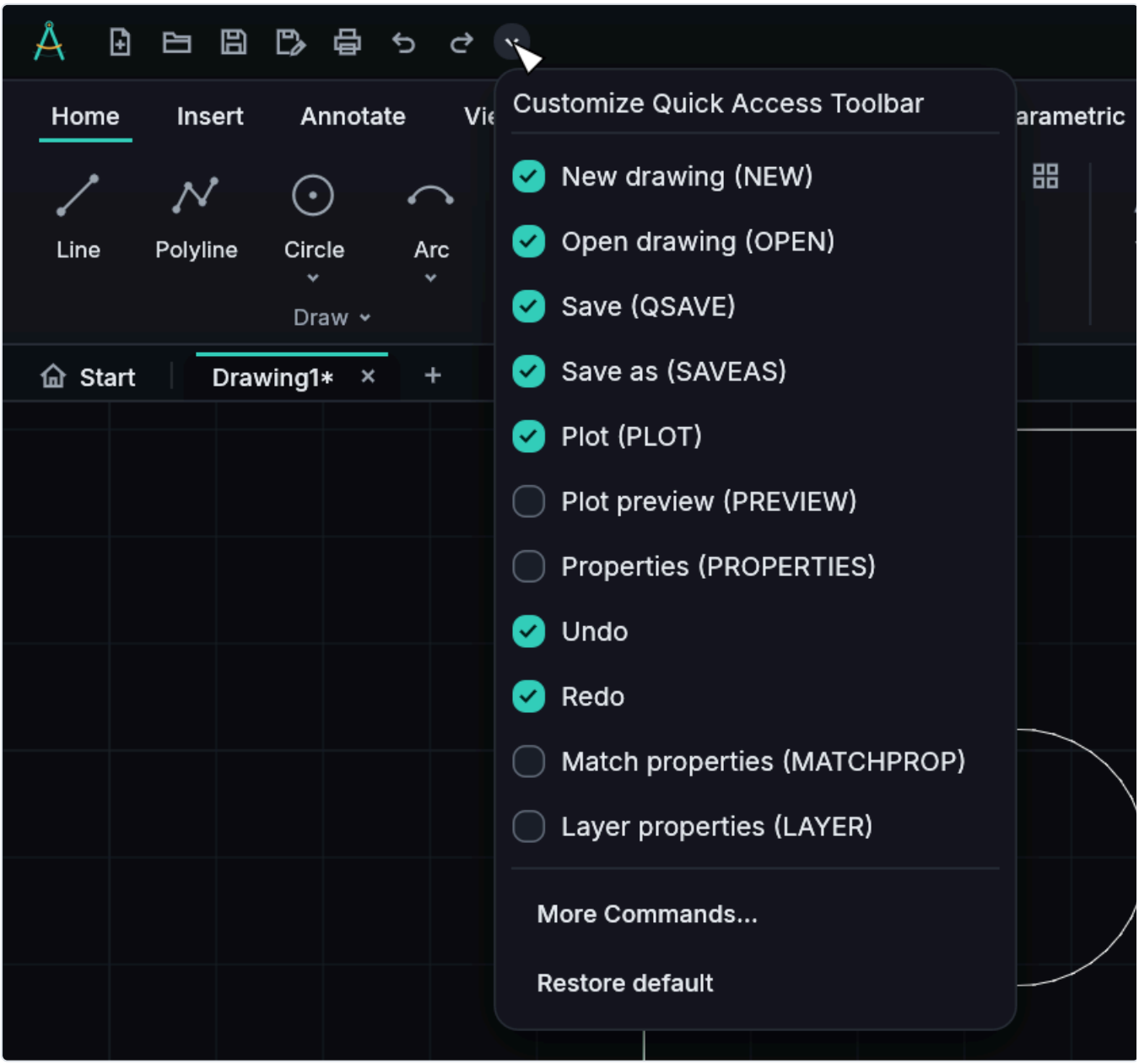
The window follows the AutoCAD layout, from top to bottom: the title bar, the ribbon with the command tabs, the file tabs, the drawing area, the Model and layout tabs (see the chapter Layouts and plotting), the command line, and the status bar.

Title bar



On the left are the BadBit CAD symbol (the compass; BadBit is the name of the whole suite) and the **Quick Access Toolbar**: new (`Ctrl` + `N`), open (`Ctrl` + `O`), save (`Ctrl` + `S`), save as (`Ctrl` + `Shift` + `S`), undo, and redo (see *Files*). The drawing name appears in the middle; on the right, the light or dark theme button and, in the corner, the help **?** (see *Help and About*).

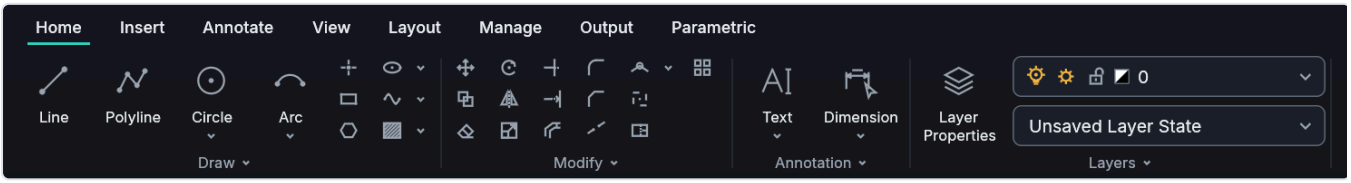
Customizing the Quick Access Toolbar



As in AutoCAD, the small arrow at the end of the toolbar opens the **Customize Quick Access Toolbar** list: check or clear New, Open, Save, Save as, Plot, Plot preview, Properties, Undo, Redo, Match Properties, and Layer Properties. **More Commands...** opens the list of all commands, with search, to put any of them on the toolbar (a command without an icon appears with its typed name). **Restore default** goes back to the factory toolbar: New, Open, Save, Save as, Plot, Undo, and Redo.

The toolbar is stored in your profile (Appearance section; see *Options and profiles*) and goes along when you export or import the profile.

Ribbon

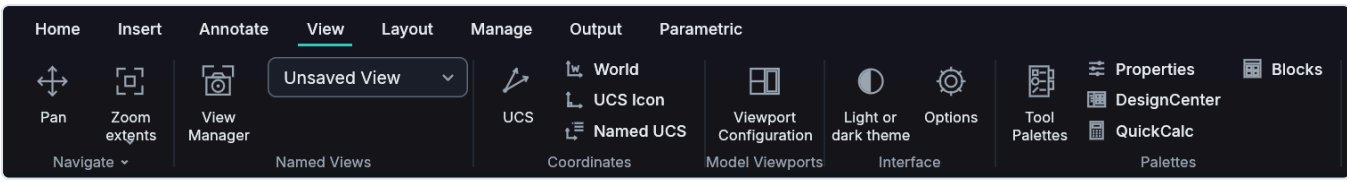


The ribbon groups the commands into tabs and panels, with the same organization as AutoCAD:

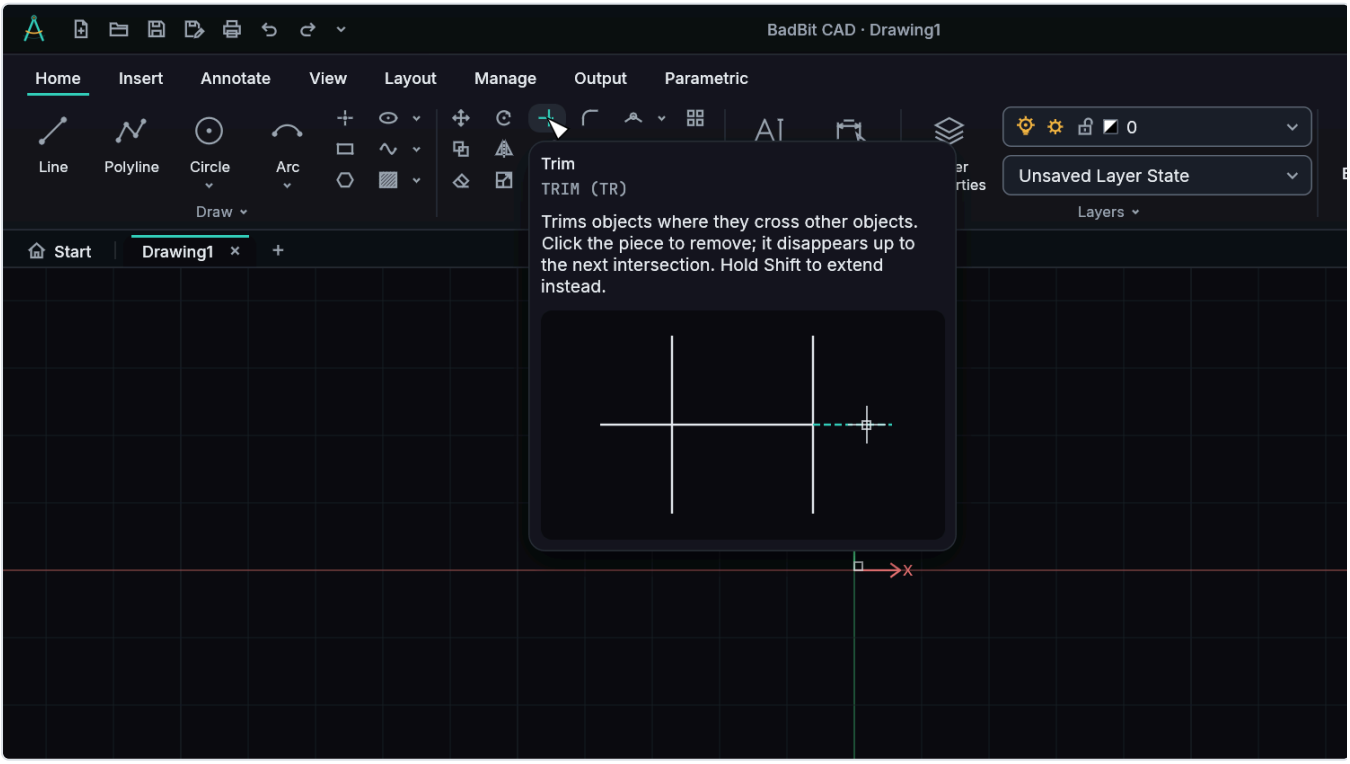
Tab	Panel	Commands
Home	Draw	Line, Polyline, Circle, Arc, Point, Rectangle, Polygon, Ellipse, Spline; in the expanded panel, Construction Line, Ray, Boundary, Revision Cloud (rectangular, polygonal, and freehand), Donut, Divide, and Measure
Home	Modify	Move, Copy, Erase, Rotate, Mirror, Scale, Trim, Extend, Offset, Fillet, Chamfer, Break, Join (with Join Ends in the list), Explode, Stretch, Array; in the expanded panel, Break at Point, Edit Polyline, Edit Spline, Lengthen, Align, and Delete Duplicate Objects
Home	Annotation	Text (MTEXT, with TEXT in the list), Dimension (DIM, with Linear, Aligned, Angular, Arc Length, Radius, Diameter, and Ordinate in the list) and, with a wide window, the lists of the current text style (AA icon) and dimension style (dimension icon); in the expanded panel, Text Style, Dimension Style, Continue, Baseline, and Find Text
Home	Layers	Layer Properties and the layer list (see *Layers*)
Home	Block	Insert and Create (after Layers, as in AutoCAD: with a narrow window, it becomes a single button before the layer list)
Home	Properties	Match Properties (the brush) and the properties palette (`Ctrl` + `1`); in the expanded panel, List and Linetype
Home	Utilities	Measure (with Distance, Radius, Angle, Area, Volume, and Quick in the list), ID Point, Quick Select, Units, Object snap settings
Home	Clipboard	Paste (with Paste as Block and Paste to Original Coordinates in the list), Cut, and Copy Clip (with Copy with Base Point in the list); only with a wide window (a ribbon of 1,400 points or more)
View	Navigate	Pan and Zoom (Extents; in the list, Window, Previous, All, Center, Object, and Scale); in the expanded panel, Regen
View	Interface	Light or dark theme, Options (user profiles)
Layout	Layout	New layout, Page setup, and the LAYOUT options
Layout	Layout viewports	Rectangular and Polygonal (MVIEW), Enter viewport (MSPACE), Back to paper (PSPACE); in the expanded panel, VPLAYER
Manage	Cleanup	Purge (PURGE) and Delete Duplicate Objects (OVERKILL)
Output	Plot	Plot (PLOT), Page setup (PAGESETUP), Preview (PREVIEW)
Output	Export	Export PDF (EXPORTPDF), Export (EXPORT), Publish (PUBLISH)

A panel title with the small arrow (**Draw ▾**) opens the **expanded** panel, with the commands that do not fit in the strip, as in AutoCAD. Buttons with the small arrow next to them (Arc, Ellipse, Spline) have a list of forms; the button repeats the last one chosen. In the Measure and Zoom lists, each item starts the command with the chosen option already selected, as in AutoCAD.

In a narrow window, the small buttons show only the icon (the name appears in the tooltip). If space is still short, the panels on the right turn into a single button, with the title and the small arrow: clicking it opens the whole panel right below, as in AutoCAD.



The button of the command in progress (and those of the open palettes, such as Layers and Properties) is highlighted in the accent color while it is active. Hovering the mouse over a button shows the name and the alias (for example, "Line (L)"). If the mouse stays still (a little over a second), the tooltip grows, like AutoCAD's extended tooltip: the name, what you type on the command line, a short explanation of what the tool does and how to use it and, for the drawing and editing commands, an **animation** of the tool in action, in a loop (the cursor crosshairs, the clicks, and the result). The animation is drawn by BadBit CAD itself, with the theme colors. A click closes the tooltip.



The extended tooltip for TRIM: the command, the explanation, and the animation, with the piece that will be removed already highlighted.

The button runs the same command you type on the command line. The tabs grow along with the new commands.

Options and profiles (OPTIONS, alias OP)

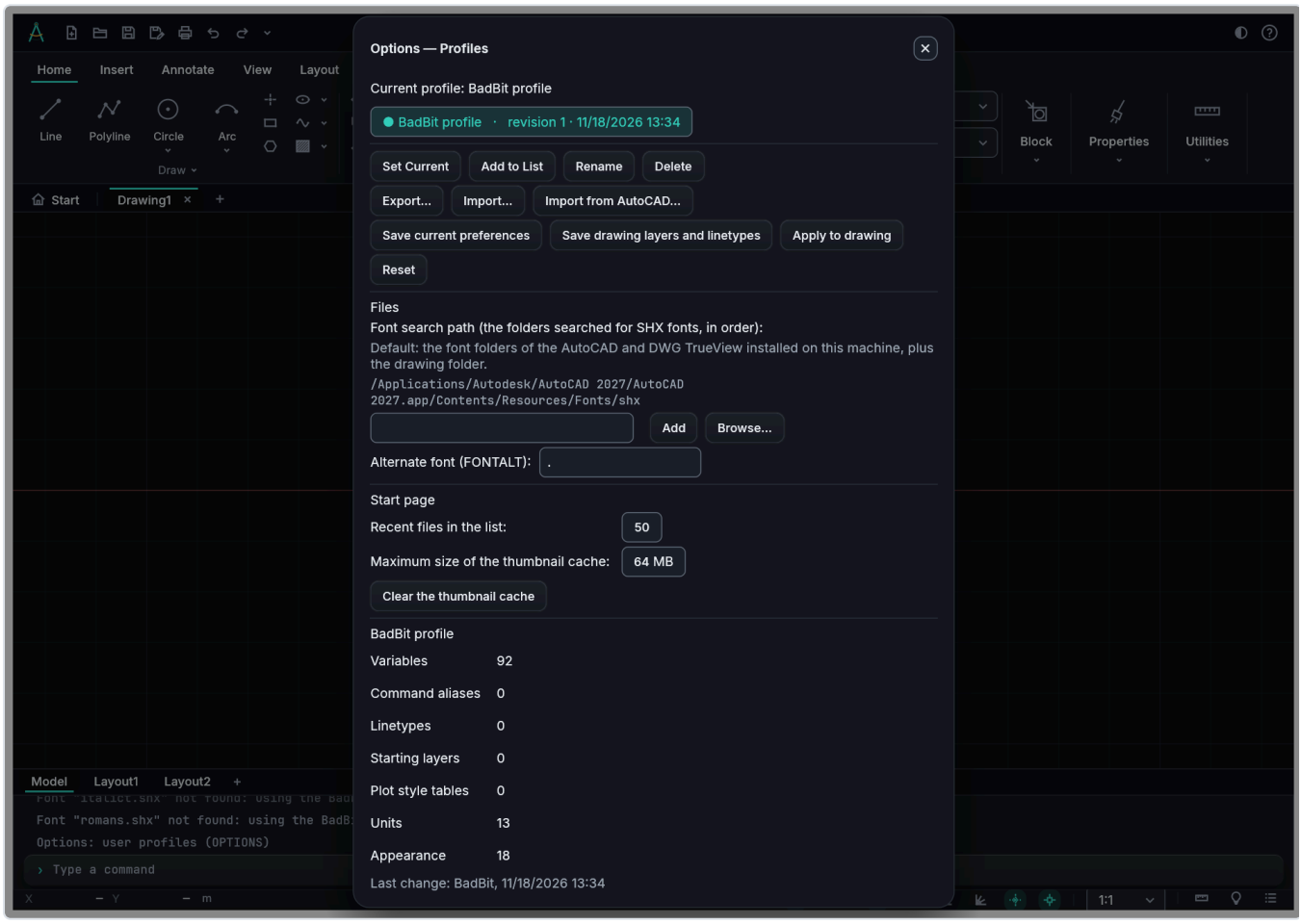
The **Options** button on the View tab (or `OP` on the command line) opens the **Profiles** tab, like AutoCAD's. A profile stores the preferences of the person using BadBit CAD: the variables (those of SETVAR, except the ones AutoCAD stores in the drawing, such as CLAYER and LTSCALE), the command shortcuts, the linetypes, the starting layers, the pen tables, the units, and the theme.

This window, like every window in BadBit CAD, never goes past the visible area: on a short screen (or with the program window made smaller), the title and the buttons at the bottom always stay in view and the middle scrolls inside, with the scroll bar on the right. Dragged by the title, the window does

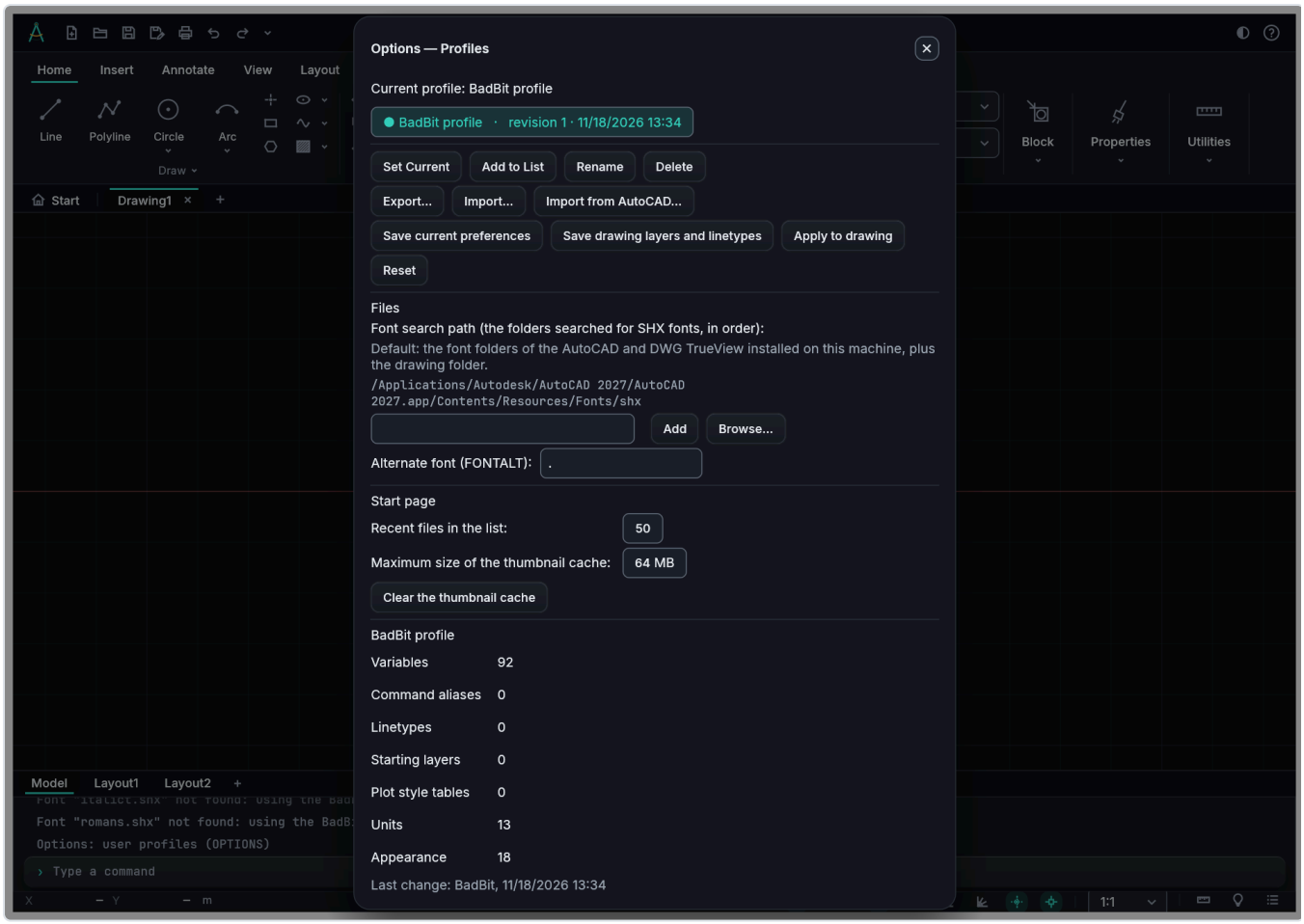
not leave the screen.

Fields too: **FIELDDISPLAY** (the gray background of fields) and **FIELDEVAL** (when fields recalculate by themselves) are in the profile's **Variables** section, come back when BadBit CAD opens, and go along when exporting; the FIELDDISPLAY of an AutoCAD `.arg` profile is imported too.

The **Files** part of the window has the **font search path** (AutoCAD's *Support File Search Path*, as far as fonts are concerned): the folders in which **SHX** fonts are searched for, in order. With no folder in the list, the font folders of the AutoCAD and DWG TrueView installed on the computer and their user support folders apply (the window shows which ones it found; on the Mac, the `Fonts/shx` inside `AutoCAD 20xx.app`), plus the folder of the open drawing. **Add** adds the typed folder; **Browse...** picks the folder in a dialog; **Remove** removes one; **Restore default** goes back to the folders of the installations. Right below is the **Alternate font (FONTALT)**: the SHX used in place of a missing one (`.` for none: BadBit CAD's equivalent is used, of the same size; see *AutoCAD SHX fonts* in the text chapter). The search path goes in the profile (with the variables) and FONTALT in the **Variables** section; when importing an AutoCAD `.arg` profile, its support path (`ACAD`) becomes the font search path.



The Files part of the Options window: the font search path (here, the default, with no AutoCAD installed) and FONTALT.



The Profiles tab: the list, the buttons, and what the selected profile contains.

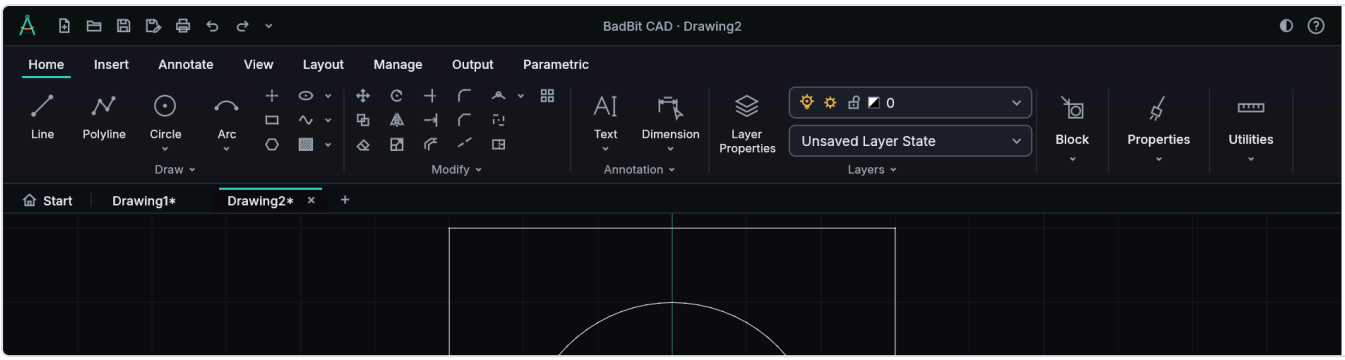
Button	What it does
Set Current	The selected profile takes effect immediately (variables, shortcuts, units, and theme).
Add to List	A new profile, a copy of the selected one.
Rename and **Delete**	Changes the name of or deletes the selected profile (the current one cannot be deleted).
Export... and **Import...**	Writes and reads the profile in a `.bbd-profile` file (readable JSON text), to take it to another machine.
Import from AutoCAD...	Reads what you already have in AutoCAD: shortcuts (`acad.pgp`), linetypes (`.lin`), pen tables (`.ctb` by color and `.stb` by named style), and the exported profile (`.arg`).
Save current preferences	Copies the current variables, units, and theme into the selected profile.
Save drawing layers and linetypes	The layers and linetypes of the open drawing become the profile's starting ones.
Apply to drawing	Adds to the open drawing the profile's layers and linetypes that are not in it yet (a single undo step).
Reset	Returns the selected profile to BadBit CAD's default values, as in a freshly installed program: variables, shortcuts, units, theme, layers, linetypes, and pen tables. It asks for confirmation first, because it cannot be undone. If it is the current profile, the program goes back to the defaults right away (the language stays the same). To keep what it has beforehand, use **Export...**.

Importing shows what came in, section by section, with the count of each: clear what you do not want and choose **Create new profile** or **Merge into selected profile** (what comes in takes precedence, and lists are merged by name). Whatever BadBit CAD did not use appears under **Not used**, never silently disappearing (for example, the external commands of `acad.pgp`).

The current profile saves the variables, units, and theme by itself when BadBit CAD closes, and each new drawing starts with its starting layers and linetypes. Profiles are kept in the system's configuration folder (in Application Support on macOS, in `%APPDATA%` on Windows, and in `~/.config` on Linux). The pen tables are kept for plotting.

File tabs

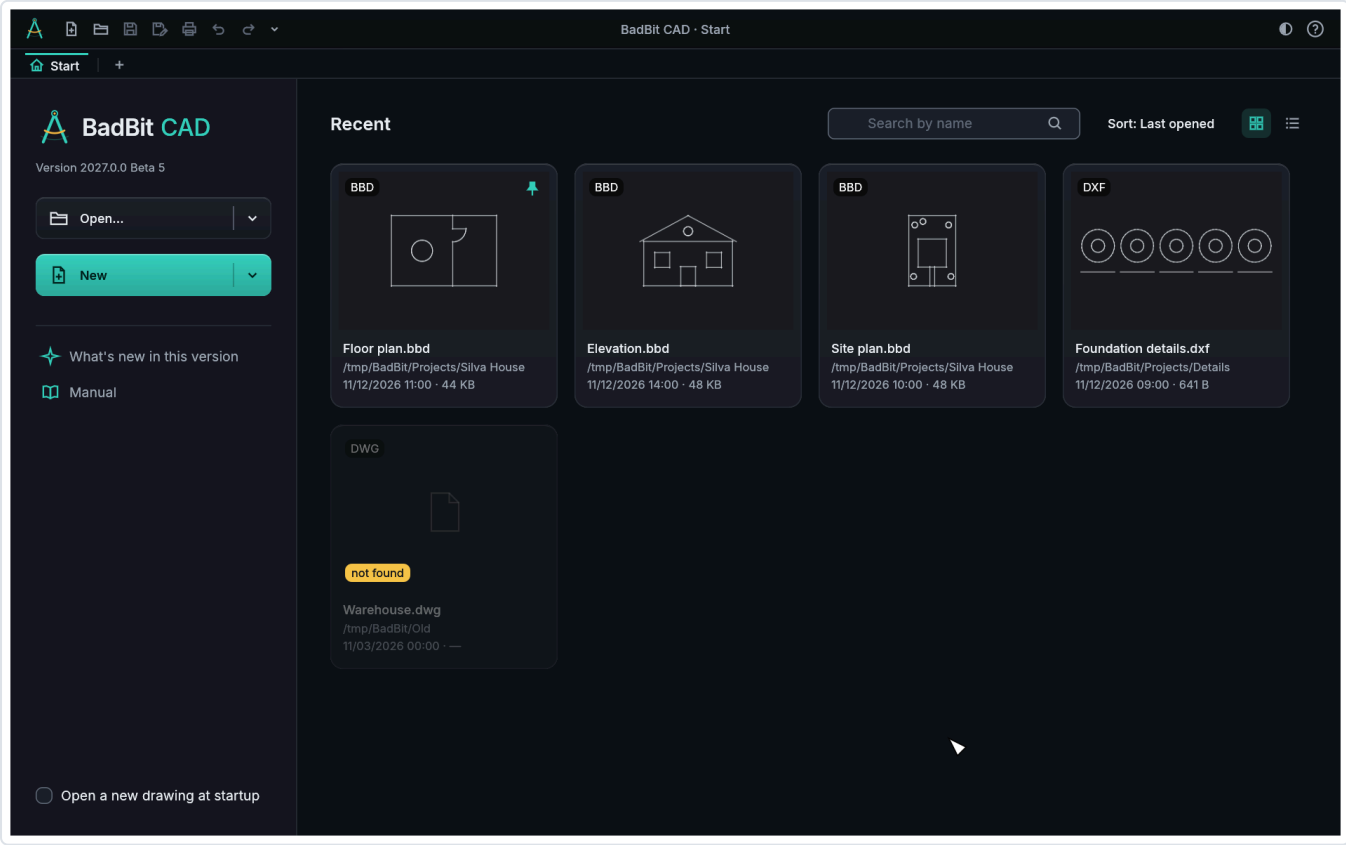
Right below the ribbon are the tabs of the open drawings, as in AutoCAD: one tab per drawing, with the file name (or Drawing1, Drawing2... while it has no name). A `*` after the name indicates unsaved changes.



Two open drawings: the active tab highlighted, the `×` to close, and the `+` for a new drawing.

- **NEW** (`Ctrl`+`N`, the New button, or the `+` at the end of the bar) opens a new tab with an empty drawing, already with the profile's starting layers and linetypes. The previous drawing stays open in its tab.
- **OPEN** (`Ctrl`+`O`) also goes to a new tab. If the current tab is the empty drawing from the start, with nothing done, the file opens in it. A file that is already open does not open again: BadBit CAD just goes to its tab.
- **Switching tabs**: click the tab, or `Ctrl`+`Tab` (the next one) and `Ctrl`+`Shift`+`Tab` (the previous one). Each tab keeps its own undo, selection, zoom, and automatic save. A half-finished command is canceled when you switch tabs.
- **Closing a tab**: the `×` (it appears on the active tab and on the one under the mouse) or the **CLOSE** command. With unsaved changes, it asks whether to save first. When you close the last tab, BadBit CAD goes back to the **Start** tab (see below); with STARTMODE 0, an empty drawing remains.
- **Closing BadBit CAD** asks about each tab with changes, one at a time: BadBit CAD goes to the tab and asks. **No** discards only that tab and moves on to the next; **Cancel** stops everything.

Start screen (Start tab)



The Start tab with four recent drawings: the floor plan pinned at the top and Galpão.dwg, which disappeared from the disk, dimmed.

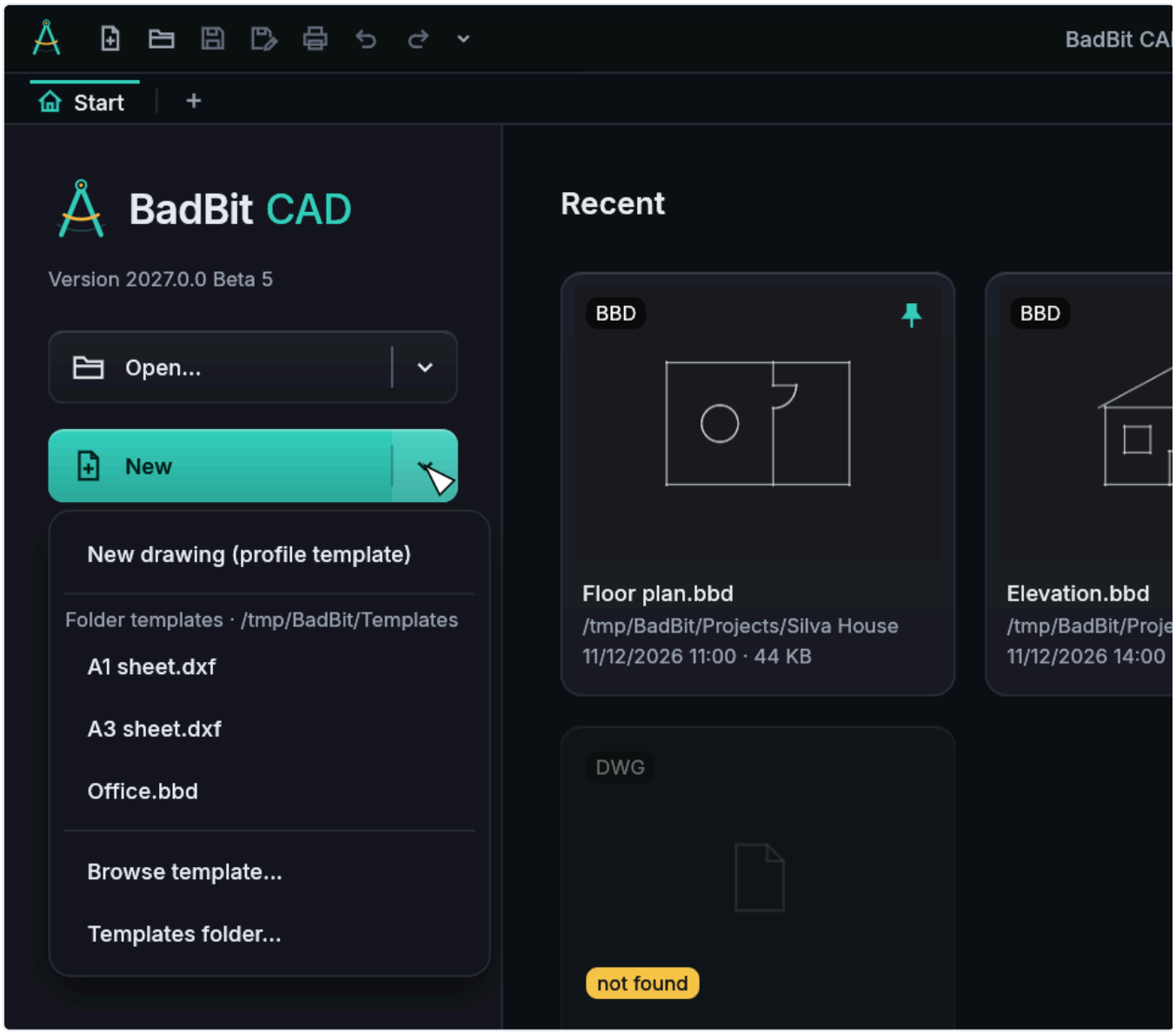
At the top of the left column are the **BadBit CAD** signature (the symbol and the name) and the version. Like AutoCAD's Start tab, the **Start** tab is the first one in the tab bar, fixed (without the `×`), with the little house. It is what appears when BadBit CAD opens, in place of the empty drawing. The tab has no drawing: the ribbon, the command line, the Model and layout tabs, and the status bar go away, and only the file commands work (`Ctrl` + `N`, `Ctrl` + `O`, and the New and Open buttons on the Quick Access Toolbar). As in AutoCAD, the Quick Access Toolbar buttons that need a drawing (Save, Save as, Plot, Undo, Redo...) are disabled while the Start tab is in front; the small arrow that customizes the toolbar still works. The open drawings stay in their tabs.

- **Going back to the Start tab**: click it, `Ctrl` + `Home`, or the **GOTOSTART** command. A half-finished command in the drawing is canceled, as when switching tabs. On the Start tab, `Ctrl` + `Tab` goes back to the previous drawing; among the drawings, `Ctrl` + `Tab` still cycles only among them.
- **STARTMODE**: 1 (the default) shows the tab; 0 hides it, and GOTOSTART only reports that it is turned off. The variable is stored in your profile.
- **Open a new drawing at startup**: the box at the foot of the left column makes BadBit CAD open straight into a Drawing1, as before; the Start tab stays in the bar. The choice is stored in the profile.

Open and New

The left column has the two buttons with the small arrow next to them:

- **Open...** opens the file dialog (DWG, DXF, and `.bbd`). The small arrow shows **Open file...** and the **recent folders** (the folders of the files in the list): choosing one opens the dialog already in it.
- **New** opens a new drawing with the profile's starting layers and linetypes, like `Ctrl` + `N`. The small arrow shows the **templates**: the profile's, the DWT, DWG, DXF, and `.bbd` files of the **templates folder** (the default is Documents/BadBit/Templates, named Modelos with the interface in Portuguese and Plantillas in Spanish; after a language change the folder that already exists is kept; **Templates folder...** picks another one, which is stored in the profile), and **Browse template...**, for a file from anywhere. The drawing that comes from a template is new and unnamed (Drawing2, Drawing3...): the template is not changed on save.



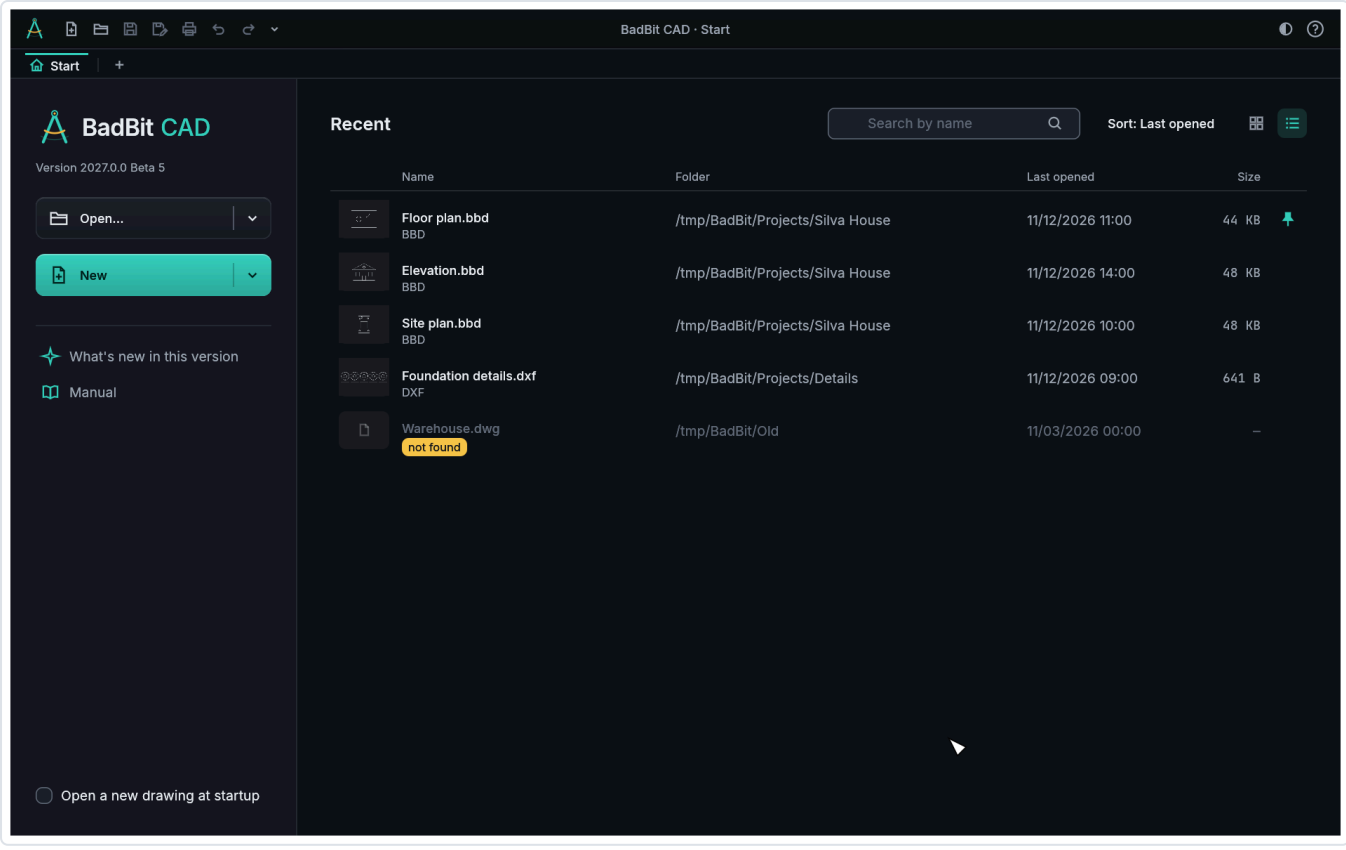
The New button's small arrow: the profile template, the folder templates, Browse template, and Templates folder.

At the bottom are the links **What's new in this version** (a window with what changed) and **Manual** (this PDF manual, in the system's reader). The manual does not come inside the installer (the PDF is larger than the program itself): the first time, **Manual** downloads the one for the installed version, with the notice **Downloading the manual...** at the bottom, and opens it when it finishes; from the second time on it opens the copy stored on the computer, also without internet (see **Help**, just below).

Recent

In the center are the files opened and saved most recently (DWG, DXF, and `.bbd`), from newest to oldest. Each one shows the name, the folder, the date it was last opened, and the size; the badge in the corner of the thumbnail tells the type.

- **Open**: double-click the file. A file that is already open does not open again: BadBit CAD goes to its tab.
- **Grid or list**: the two buttons next to the search switch between the cards with the thumbnail and the rows with the columns (name, folder, last opened, and size).
- **Sort**: by last opened, by name, or by size.
- **Search**: the field filters by file name, ignoring case and accents ("fundacao" finds "Fundação.dwg").
- **Pin**: the pin in the corner of the card (or at the end of the row) pins the file to the top of the list, in any sort order; another click unpins it.
- **Right-click**: **Open**, **Open folder** (shows the file in the Finder, in Explorer, or in the file manager), **Remove from list**, and **Pin** (or Unpin).
- **File that disappeared**: a deleted or moved file appears dimmed, with the **not found** badge; double-clicking only shows a notice. Remove it with Remove from list.

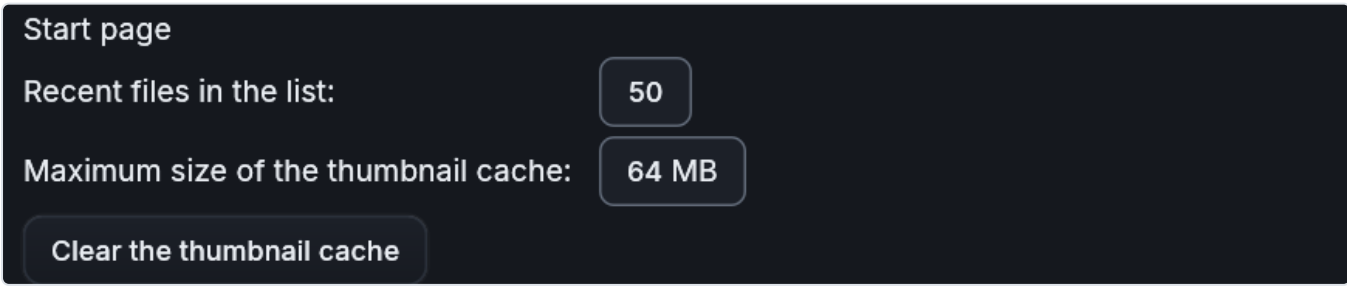


The same recent files as a list.

The list is stored in your profile (Appearance section), with up to 50 files; the maximum (from 1 to 500) is changed in the **Options** window (OPTIONS), **Start page** part, **Recent files in the list** field (like AutoCAD's "Number of recently-used files"). Lowering the number removes the oldest ones from the list immediately (pinned ones go last). The list is updated each time a file is opened and each time it is saved.

Thumbnails

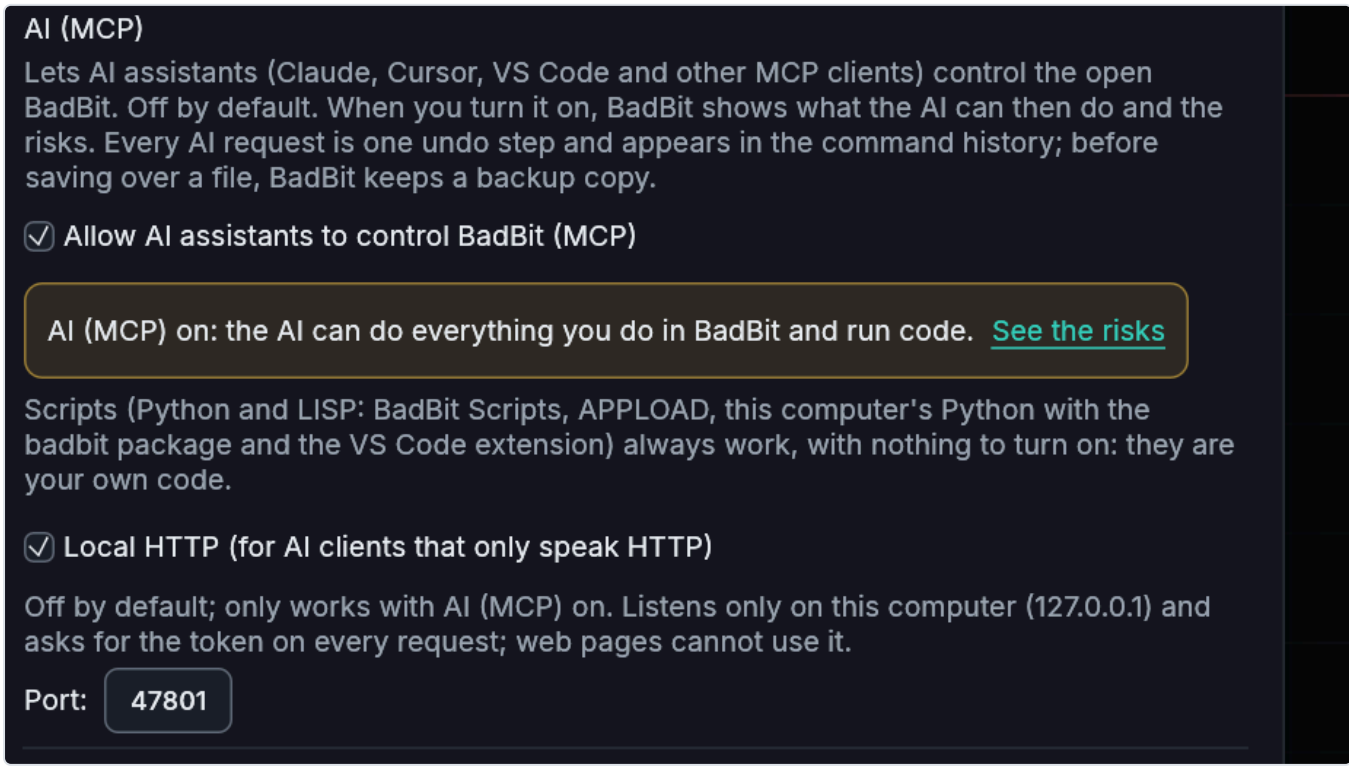
- A `.bbd` gets the thumbnail inside it each time it is saved, drawn by the CPU in a thread (the screen does not stop). As in AutoCAD, the thumbnail shows the **current view**: the model the way it is on the screen (with the current zoom and pan) or, with a layout on the screen, that layout's sheet with the viewports. For the thumbnail to show the whole drawing, do a zoom extents (Z, E) before saving.
- A DWG shows the preview that AutoCAD writes into the file (BMP up to 2010, PNG from 2013 on; in old DWG files that only carry the preview in **WMF**, the Windows metafile, BadBit CAD draws it with its own interpreter: polylines, polygons, rectangles, ellipses, arcs, and the images inside it); a DXF, the THUMBNAILIMAGE section, both in text DXF and in **binary DXF**. DWG and DXF files **saved by BadBit CAD** also carry the thumbnail (drawn by the CPU at save time, in the same format as AutoCAD): it appears here, in AutoCAD's Open dialog, and in Windows File Explorer.
- With no thumbnail in the file (a DWG written by another program, a DXF without the section, an old `.bbd`), BadBit CAD opens the file in a thread, draws the thumbnail, and keeps it in a cache on your computer (the system's cache folder, up to 64 MB; the oldest ones go first). Files over 200 MB get only the placeholder.
- In the **Options** window, **Start page** part: **Maximum size of the thumbnail cache** (from 1 to 4096 MB; lowering the limit trims the cache immediately, in a thread) and **Clear the thumbnail cache** (deletes the stored thumbnails; they are read or drawn again when the Start tab appears). Both choices are stored in the profile.



The Start page part of the Options window: the number of recent files, the cache limit, and the button that clears it.

- While the thumbnail has not arrived, the card shows the placeholder (the sheet); with many files, the thumbnails arrive one by one, without freezing the screen. With everything loaded and the mouse still, the Start tab requests no frames, like the rest of BadBit CAD.
- In the browser, the Start tab has only New and Open (with no disk, there are no recent files).

AI (MCP)



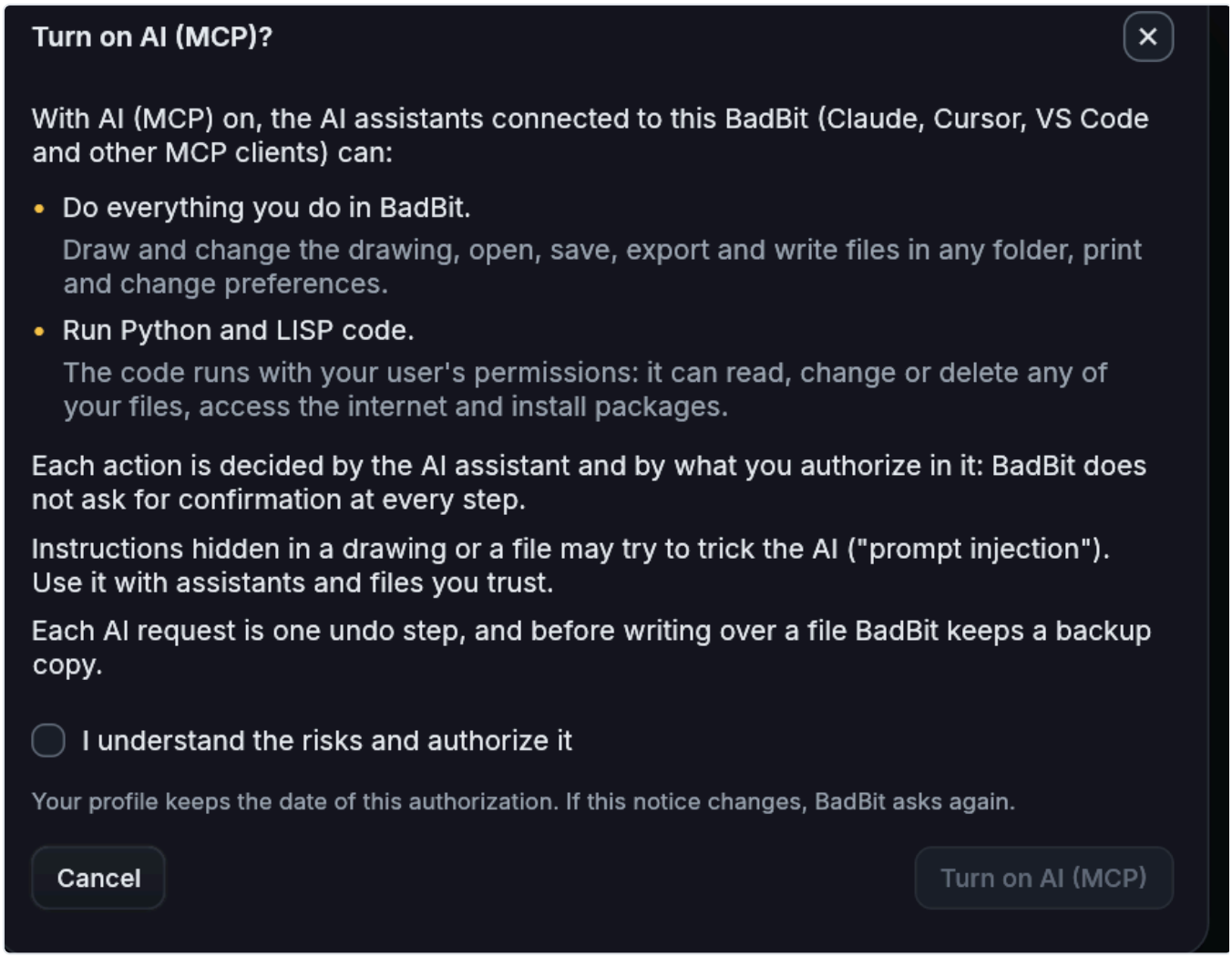
The AI (MCP) part of the Options window, turned on: the banner with the See the risks link and the local HTTP on at the default port.

AI assistants (such as Claude, Cursor and VS Code) can control BadBit CAD over **MCP**: drawing, querying and changing objects and layers, running any command with the answers to its questions, changing the view and taking an image of it, opening, saving, exporting and closing files (a drawing with changes only closes when the assistant says to save it first or to discard, and never while a question of yours is open on screen), printing and running code. AI comes **turned off**: it only works after you turn it on, in the **Options** window, part **AI (MCP)**, or with the **AI** button of the status bar.

- **Allow AI assistants to control BadBit (MCP)**: turns it on and off. Turning it on shows the risk notice first (below); turning it off disconnects whoever is connected right away.
- **Local HTTP (for AI clients that only speak HTTP)**: off by default, and only works with AI (MCP) on. When on, BadBit CAD serves the assistants that only connect over HTTP at `http://127.0.0.1:<port>/mcp`, on this computer only; the default **Port** is 47801. While it is open, the **Address** appears with the **Copy token** button (the client sends the token in the `Authorization: Bearer` header) and **Generate another token** (the old one stops working at once). Web pages cannot use the local HTTP, and every request without the token is refused. The MCP guide, in the AI and scripts documentation, shows how to configure the client.

Scripts do not depend on this switch: BadBit Scripts, APPLOAD, LISP, the Python on this computer with the `badbit` package, the VS Code extension and the console always work, with nothing to turn on, because they are your own code (see the chapter *Entering points and answering commands*). Connected scripts do not show on the status bar.

The risk notice



The window that opens when you turn on AI (MCP): what the AI can then do, the caveats, the authorization box and the Turn on AI (MCP) button, greyed out until the box is ticked.

Turning on AI (MCP) opens the **Turn on AI (MCP)?** window, which lists what the connected AI assistants can then do:

- **Do everything you do in BadBit**: draw and change the drawing, open, save, export and write files in any folder, print and change preferences.
- **Run Python and LISP code**: the code runs with your user's permissions and can read, change or delete any of your files, access the internet and install packages.

And the caveats: each action is decided by the AI assistant and by what you authorize in it (BadBit does not ask for confirmation at every step); instructions hidden in a drawing or a file may try to trick the AI (*prompt injection*), so use it with assistants and files you trust; and each AI request is one undo step, with the backup copy before writing over a file.

The **Turn on AI (MCP)** button only works after you tick **I understand the risks and authorize it**; **Cancel** (or `Esc`) leaves AI off. Your profile keeps the date and the notice version: if the notice changes, BadBit asks again. With AI on, the AI (MCP) part always shows the banner "AI (MCP) on: the AI can do everything you do in BadBit and run code.", with the **See the risks** link (which reopens the list, with the **Turn off AI (MCP)** button), and the **AI** button of the status bar turns to the warning color. A profile without this consent (from an earlier version or edited by hand) comes in with AI (MCP) off.

When it is off, an assistant that tries to use the open BadBit gets, on every request, the message asking to turn on AI (MCP) here; as soon as you turn it on, the next request works, without restarting the assistant.

Without a window (BadBit from the terminal or from the Claude Desktop package), there is no window for the notice, so the consent is explicit in the configuration: in the Claude Desktop package, the required box **I understand the risks and authorize it**; in the terminal, `--aceito-ia` with the notice version. Without it, BadBit does not start and says why; when it starts, it writes a short notice with these powers to the client's log. The MCP guide, in the AI and scripts documentation, explains each case.

BadBit does not interrupt the assistant with questions: each action is decided by your AI client (which usually asks for your permission). Before the first time the AI saves over a file in each connection, BadBit silently keeps a **backup copy** next to it (`name.ia-backup-YYYYMMDD-HHMMSS.bbd`). The AI cannot change its own switch: this part of Options (the switch, the local HTTP and the token) only changes here, by you. Network and Windows device paths are not read or written by the AI's requests, and commands that would open the file window do not run through them (the AI uses the file operations). With no assistant connected, none of this changes your work.

Each request from an assistant becomes **a single undo step**, with the name of its origin (for example "AI: cad.entity.create"): `U` takes back everything that request did. The command line shows who connected and who disconnected. The choices are kept in your profile (importing a profile from another computer does not turn AI on). Only programs of your user, on your machine, can connect: the channel is local and asks for a new key every time BadBit CAD opens.

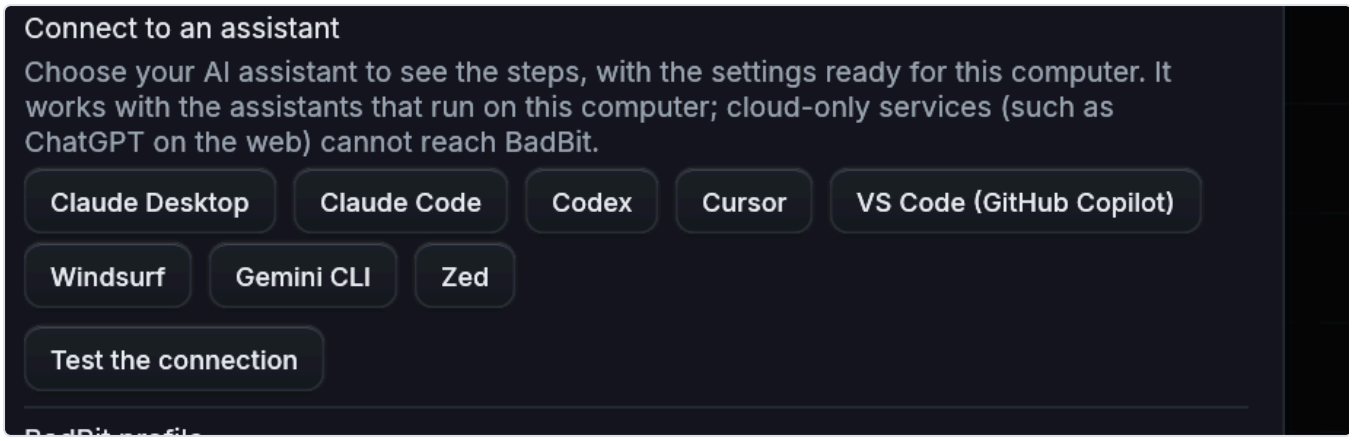
If you close BadBit CAD with an assistant connected and open it again (with AI (MCP) on), the assistant comes back by itself on its next request; the requests that were in progress get the notice that the program closed. The click that disconnects, on the other hand, holds until you restart the MCP server in the assistant, even with BadBit reopened. A batch from the assistant that fails leaves nothing behind, not even the system variables it changed (such as LTSCALE).

Watching the AI draw. With BadBit CAD open, the assistant draws in short stages you follow on screen (for example: layers and walls, then doors and windows, then dimensions and hatches), saying before each one what it is going to do. Each stage shows up little by little, as its steps are applied, and is one undo step. Between one stage and the next the assistant reads what you wrote, so you can ask or correct things along the way. If you start a command while a stage is being drawn, the stage waits for your command to finish; what you draw in the meantime goes into the same undo step as the stage. Without a window (from the terminal, or the package without BadBit open), the assistant still sends the whole project at once.



The status bar with an AI assistant connected: "AI connected (name given by the client): Claude Desktop" (clicking disconnects) and the AI button turned on, in the warning color.

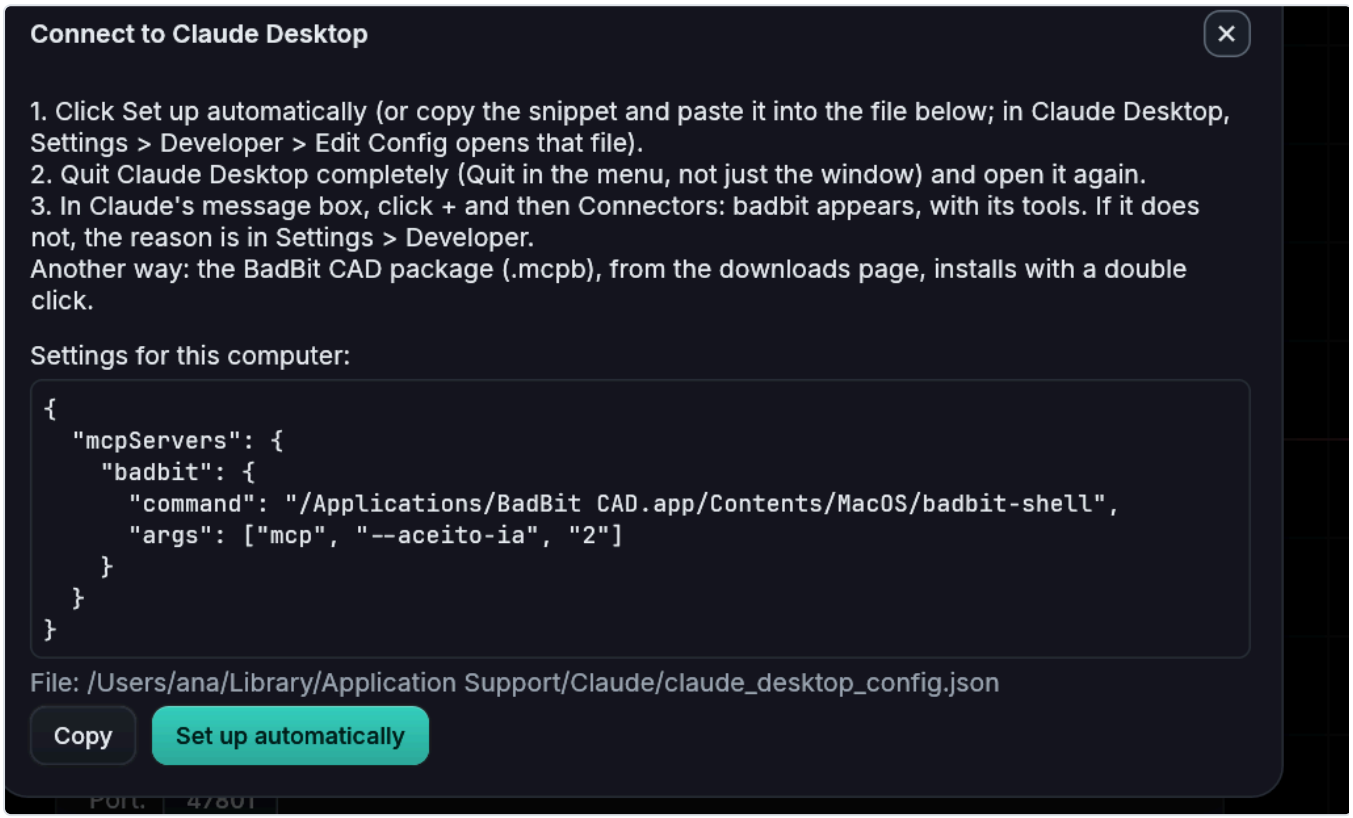
Connect to an assistant



Options > AI (MCP) > Connect to an assistant: a button for each AI assistant that runs on the computer and Test the connection.

Below the AI (MCP) switch there is **Connect to an assistant**, with a button for each AI assistant that runs on the computer: **Claude Desktop**, **Claude Code**, **Codex**, **Cursor**, **VS Code (GitHub Copilot)**, **Windsurf**, **Gemini CLI** and **Zed**. The button opens the **Connect to** (assistant name) window, with:

- the short steps (what to do and how to tell it connected);
- the settings ready for this computer, with the path of the installed BadBit CAD (the snippet that goes in the assistant's file or, for Claude Code, the terminal command) and the file where it goes;
- **Copy**: copies all the settings;
- **Set up automatically**: BadBit tells you which file it will change and asks first; when you confirm, it keeps a copy of the file next to it (`<file>.badbit-backup-YYYYMMDD-HHMMSS`) and adds only the `badbit` server (the other servers and settings in the file stay as they are; a file it cannot read is not touched). For Claude Code, BadBit runs the `claude mcp add` command for you.



An assistant's window: the steps, the settings ready (the snippet scrolls sideways when a line is long), the file and the Copy and Set up automatically buttons.

The settings carry your acceptance of the risks (with BadBit closed, the assistant opens it without a window), so they only appear with AI (MCP) on; when it is off, the window shows the **Turn on AI (MCP)...** button. Cloud-only services (such as ChatGPT on the web) cannot reach BadBit, which only answers on this computer.

Test the connection checks the BadBit side: it tells you whether the local channel is answering, whether AI (MCP) is on, which assistant connected last (with the name it gave) and the local HTTP address, if on. The guide "Connecting your AI assistant to BadBit", in the AI and scripts documentation, has the manual steps for each assistant, on each system, and the common problems.

Drawing area

This is where the drawing appears. The cursor changes with what the command asks for, as in AutoCAD:

Situation	Cursor
No command	The crosshairs with the pickbox in the center
The command asks for a point (LINE, CIRCLE, the hot grip...)	Only the crosshairs, closed at the point
The command asks for objects (Select objects, the object to trim...)	Only the pickbox

The size of the crosshairs follows the **CURSORSIZE** variable (default: 5% of the screen) and that of the box, **PICKBOX**. In the lower-left corner is the icon of the X (red) and Y (green) axes.

Command line



The command line occupies the bottom strip, across the full width. Above the input are the last three lines of the history (prompts and answers); `F2` expands the history. While a command runs, its prompt appears before what you type, with the command name: "LINE Specify first point:".

You do not need to click the command line: what you type over the drawing goes straight to it, as in AutoCAD. As you type, command suggestions appear; the arrows choose and `Enter` runs.

Status bar

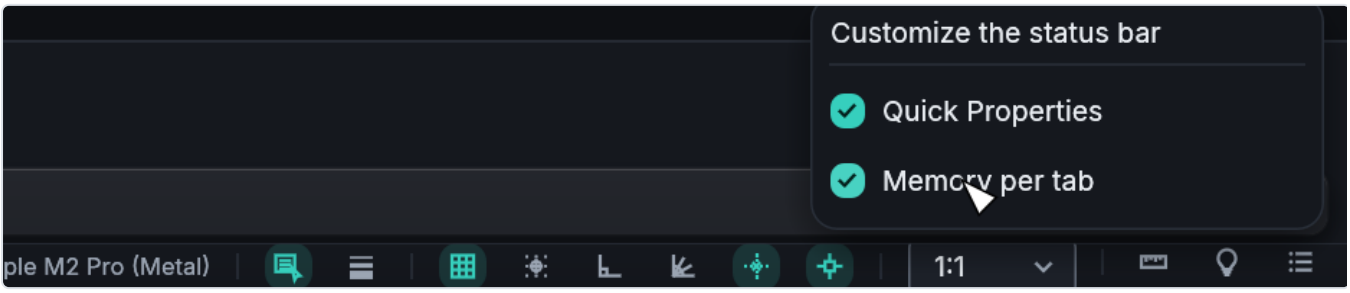


On the left, the cursor coordinates in the project unit and, with objects selected, how many there are. On the right, the drawing modes as icons, in AutoCAD's order:

Icon	Mode	Key
Grid	Reference grid	`F7`
Snap mode	The cursor moves from grid point to grid point (SNAP)	`F9`
Ortho	Horizontal and vertical only	`F8`
Polar	Polar tracking	`F10`
Tracking	Object snap tracking (OTRACK)	`F11`
Object snap	Snaps (OSNAP)	`F3`
Lineweight	Shows or hides lineweights (LWDISPLAY)	—
Quick Properties	The small properties window for the selection (QPMODE)	`Ctrl` + `Shift` + `P`
Annotation scale	The CANNOSCALE list (1:1, 1:50, 1:100…), the scale of annotative objects	—
Units	Opens the units window	—
Isolate objects	The light bulb: lit with objects hidden by ISOLATEOBJECTS or HIDEOBJECTS; clicking it opens the menu Isolate Objects, Hide Objects, and End Object Isolation (see *Isolating and hiding objects*, in the selection chapter)	—
AI	Turns AI (MCP), the control by AI assistants, on or off (see *AI (MCP)*, above); turning it on shows the risk notice first, and when on the button is in the warning color. With an assistant connected, "AI connected (name given by the client): Name" appears on its left (the name is the one the assistant itself gives; when it only sends its technical name, known ones show with their display name, such as "claude-ai", which becomes Claude Desktop), and the command line reports the connection with the same name; clicking it disconnects. Scripts do not show here	—

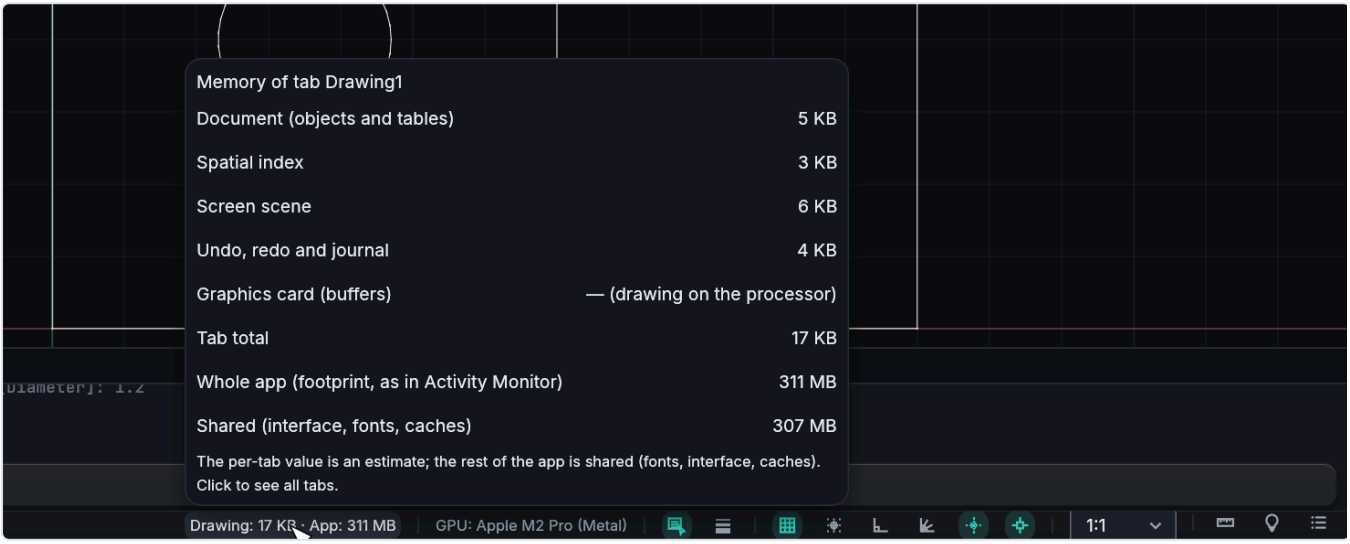
A mode that is on appears highlighted. Right-clicking the object snap icon opens the Object snap tab of the Drafting settings; on the polar icon, the Polar tracking tab; on the lineweight icon, the Lineweight settings (LWEIGHT). Hovering the mouse over the GPU name shows the zoom, how many lines are on the screen, and the time of each frame.

Customizing the status bar: memory per tab



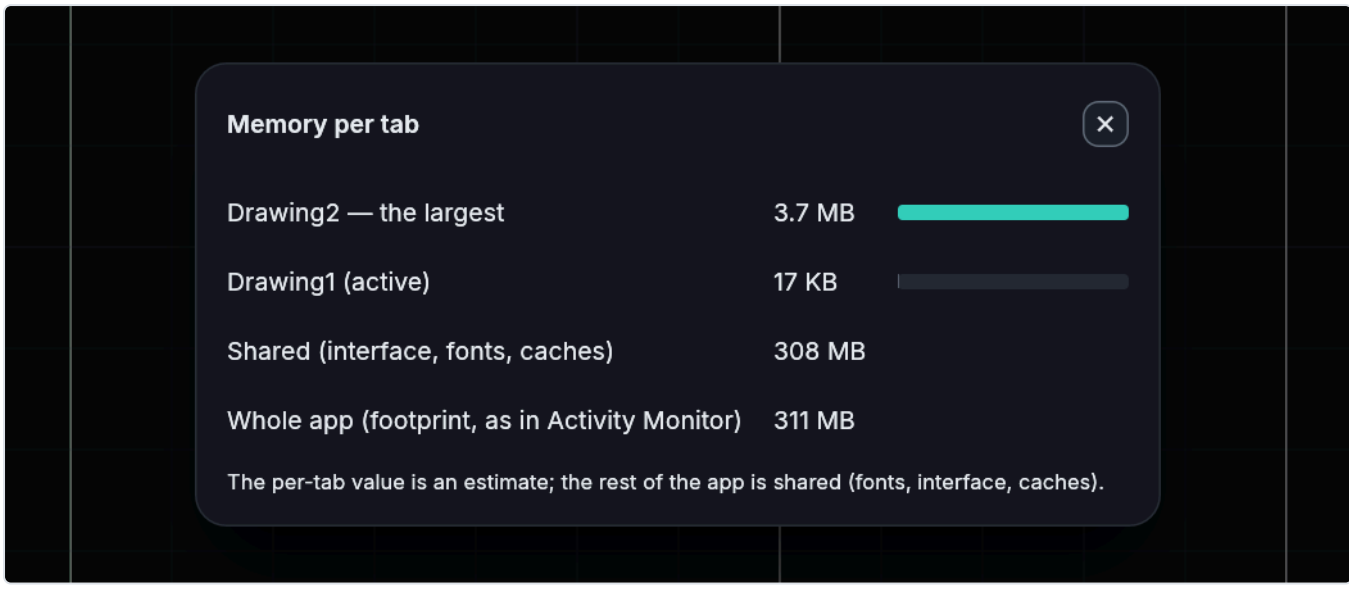
The customize button, in the right corner, with Memory per tab checked.

The button in the right corner of the status bar (the one with the lines, like AutoCAD's "Customization") opens the list of items that can be put on the bar. For now, **Memory per tab**, which comes turned off. When checked, the bar shows, to the left of the GPU name, how much the drawing of the active tab and the whole app take up, for example `Drawing: 96 MB · App: 182 MB`. The drawing value changes when you switch tabs. The choice is stored in the user profile (Appearance section) and goes along when exporting the profile.



The tooltip: the details of the active tab, the whole app, and the shared part.

Hovering the mouse over the item shows the details of the active tab: the document (objects, layers, blocks, and styles), the spatial index, the screen scene, the undo and redo with the recovery journal, the graphics card buffers (when the drawing goes through the GPU), and the tab total. It also shows the whole app, as the system counts it (on macOS, the same number as Activity Monitor; on Windows, Task Manager's working set; on Linux, the resident memory), and the shared part: what belongs to no tab (the interface, the fonts, the caches, and the program itself). The per-tab value is an estimate made from the drawing's content; the app's is the system's. In the browser, only the drawing appears.

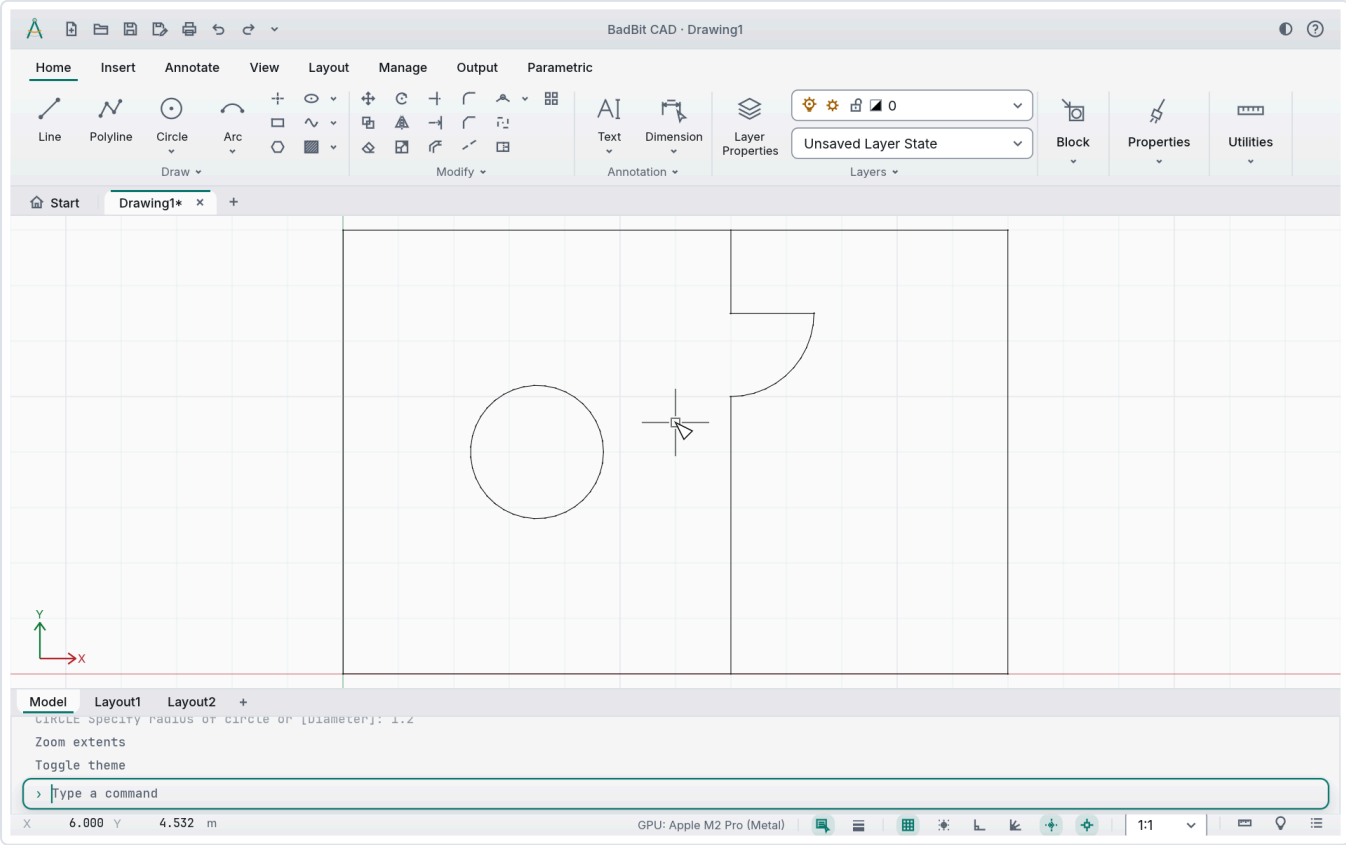


The window with all open tabs, from the one that takes up the most to the one that takes up the least.

Clicking the item opens the **Memory per tab** window: all open tabs, from the one that takes up the most to the one that takes up the least, with a proportional bar; the first one is marked as "the largest" and the active one as "(active)". At the bottom, the shared part and the whole app. An idle tab has no screen scene (it is rebuilt when you go back to the tab), which is why it takes up less than the same tab when active.

The meter only works with the item turned on and measures at most every 2 seconds, taking advantage of the frames the screen was going to draw anyway: with the app idle, the number stays as it was and the app keeps sleeping (moving the mouse updates it). The size of each drawing is measured once and then follows each command, undo, and redo, without going through the drawing again.

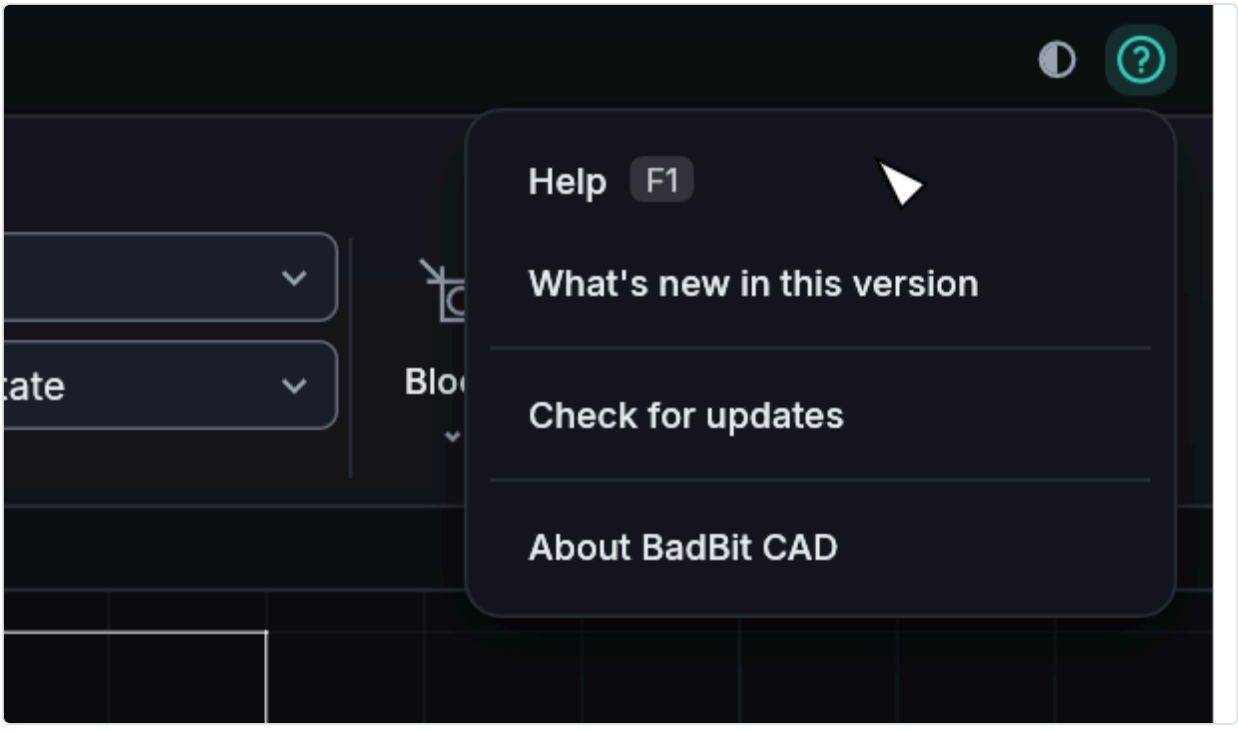
Light and dark themes



The light theme.

The button in the right corner of the title bar, or the **THEME** command, switches themes. Objects of color 7 (white/black) follow the background, as in AutoCAD.

Help and About



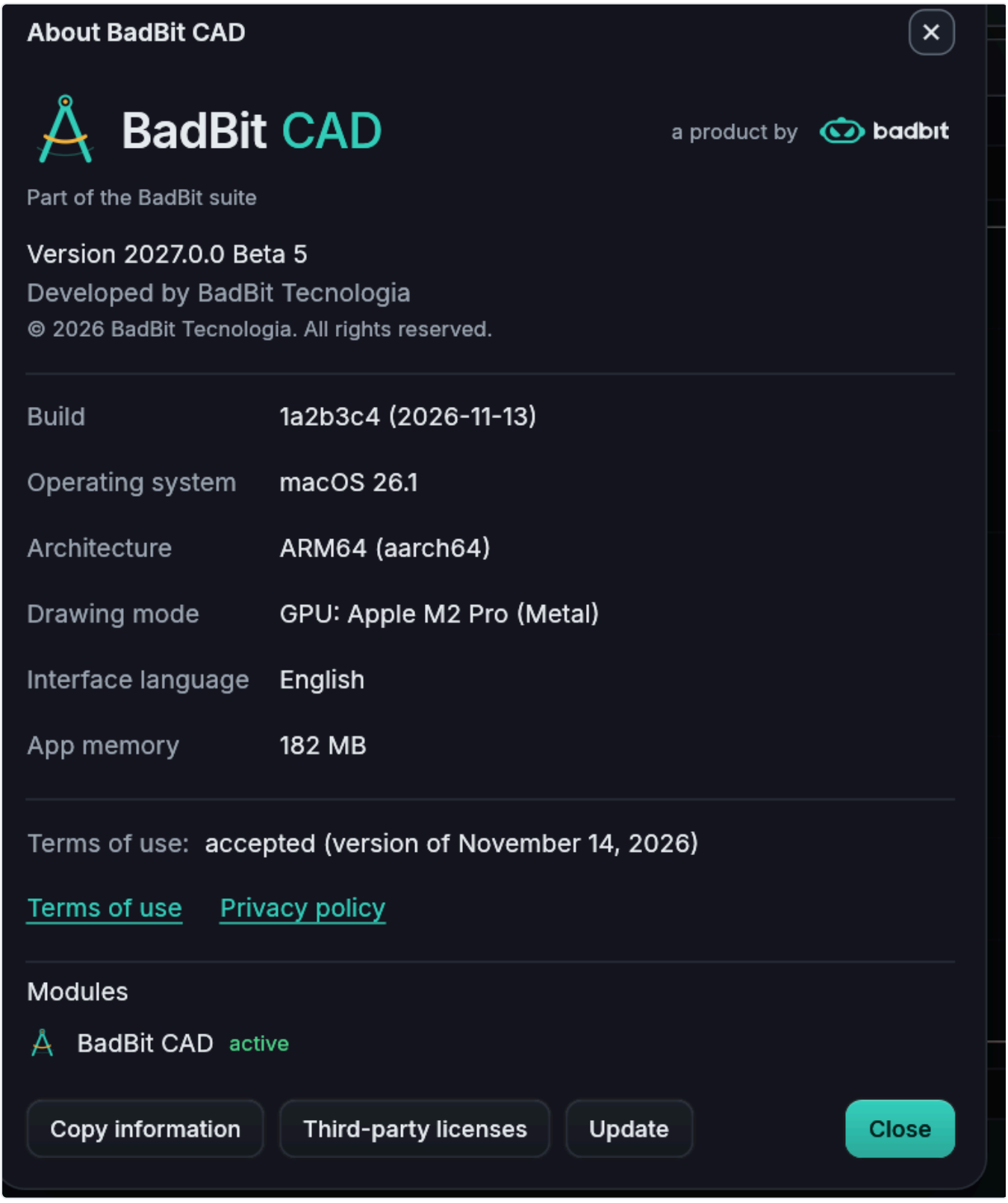
The ?

menu, in the right corner of the title bar.

The **?** in the right corner of the title bar takes the place of AutoCAD's InfoCenter and also works on the Start tab. Clicking it opens the menu:

- **Help** (`F1`, or the **HELP** command): opens this PDF manual in the system's reader, the same as the **Manual** link on the Start tab. The program looks for the file `BadBit-CAD-Manual-<language>.pdf` (or the Portuguese one) in this order: first **next to the program** (for IT staff who copy the PDF at installation: the `manual` folder next to the executable, the executable's own folder, `../Resources/manual` in the macOS `.app` bundle, and `../share/badbit/manual` on Linux); then the **already downloaded copy** for this version, in the program's data folder (`~/Library/Application Support/BadBit/CAD/manual/<version>` on macOS, `%LOCALAPPDATA%\BadBit\CAD\manual\<version>` on Windows, `~/.local/share/badbit/cad/manual/<version>` on Linux); with neither of the two, it **downloads** the manual for the installed version from `https://downloads.badbit.com.br/cad/releases/<version>/manual/` (or from the company mirror, the policy's `baseUrl`, chapter *Installation and updates*), in the interface language or, if that one does not exist yet, in Portuguese. The download runs in the background: you can keep drawing while **Downloading the manual...** appears (on the command line; on the Start tab, at its bottom), and the PDF opens by itself when it arrives. From the second time on, the manual opens from the stored copy, also without internet. In a new version of the program, its manual is downloaded the first time and the copy for the previous version is deleted. Without internet and without a copy, the notice says it could not be downloaded right now and shows the manual's address, to open in the browser when the connection comes back. In the browser the manual does not have its own address yet; the notice says so.
- **What's new in this version**: the window with what changed, the same as on the Start tab.
- **Check for updates**: opens the **Updates** window and checks for a new version right away (chapter *Installation and updates*). When there is a new version, the notice **Update *version* available** also appears to the left of the **?**. In the browser the item is disabled: the new version arrives when the page is reloaded.
- **About BadBit CAD**: the About window, also through the **ABOUT** command.

About window (ABOUT)



The About window with the version, the build, the system, and the modules.

The window shows:

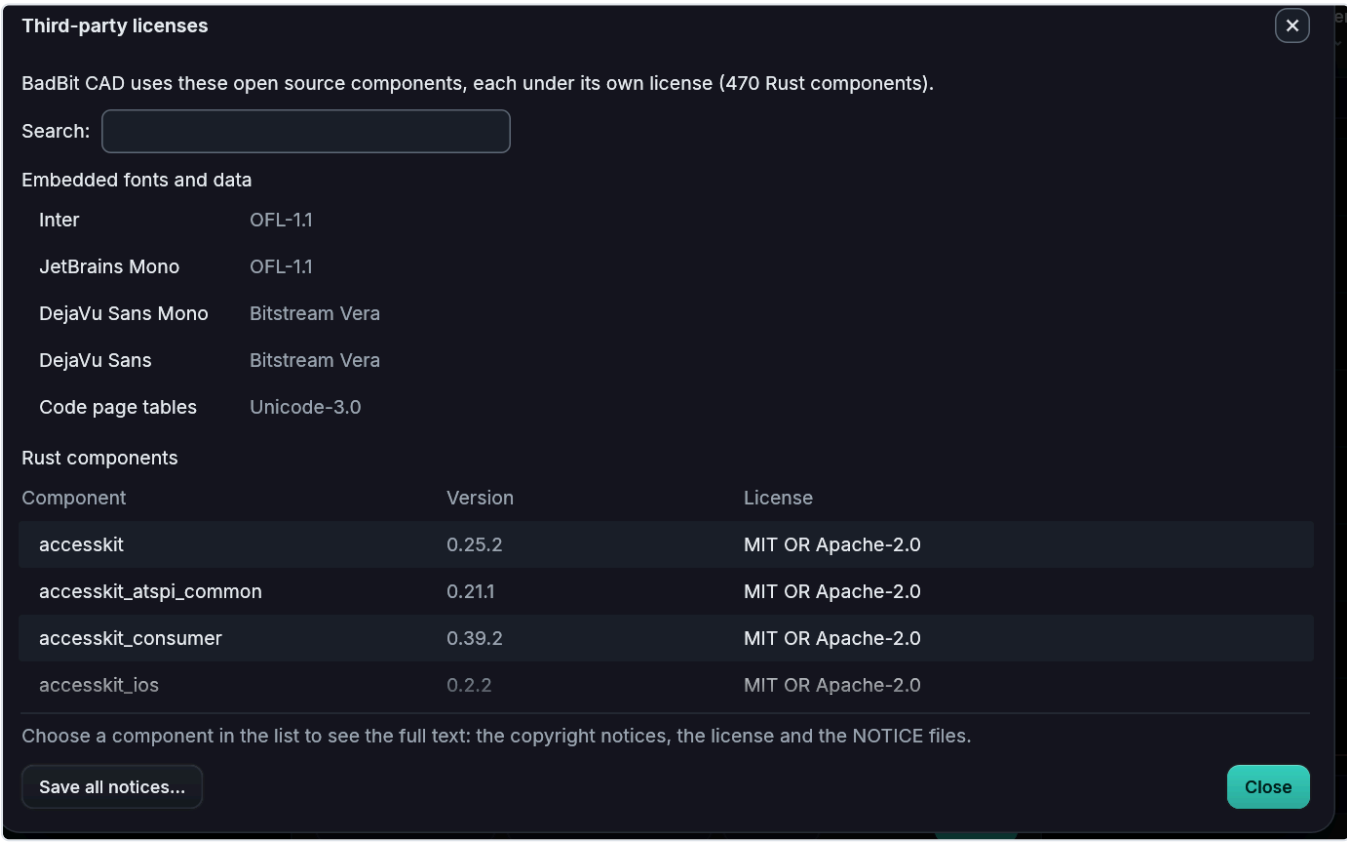
- at the top, the **BadBit CAD** signature (the compass symbol and the name) and, on the right, *a product by* with the **BadBit** logo; below, that it is part of the BadBit suite, and who develops it (**BadBit Tecnologia**), with the copyright. The screen reader reads the name with the version

year (**BadBit CAD 2027**);

- the **version** in the product format, `YYYY.N.P` (model year, release in the year's line, and fix; as at Autodesk, a year's line starts in the previous year), with the channel before the public release: **Alpha N**, **Beta N**, **Release Candidate N**, or **Nightly** with the date; the stable version has no channel. E.g.: *Version 2027.0.0 Beta 1*. It is the same version BadBit CAD writes into the files (`.bbd`, DWG, and LandXML);
- the **build**: the short commit code and its date (without the source code history, *unknown* and the build date);
- the **operating system** with the version, the **architecture** (ARM64, x64), the **drawing mode** (GPU, with the name of the card, or CPU, with the number of threads), the **interface language**, and the **app memory** of the whole app (the same measurement as the status bar meter; in the browser it does not appear);
- the **Terms of use** accepted on this computer (*accepted (version of 11/14/2026)*, or *accepted by* and the company's name, when IT accepted on its behalf) and the **Terms of use** and **Privacy policy** links, which open the version in effect in the browser (chapter *Installation and updates*; in the browser it does not appear);
- the platform **modules**, each with the product symbol, and whether they are active. For now, only BadBit CAD; the other modules of the suite appear here according to the license.

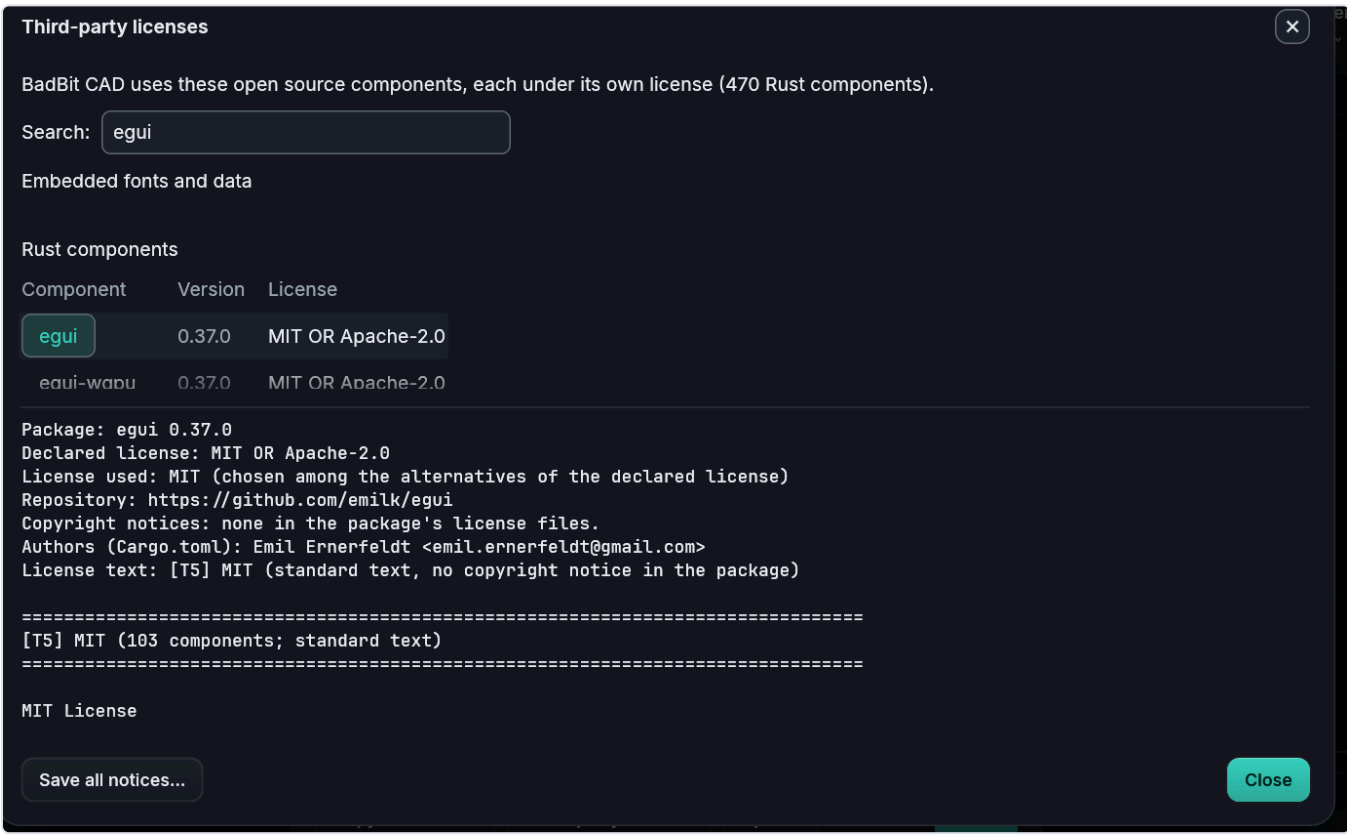
Copy information copies all of this as text, one item per line, to paste into a bug report. **Update** opens the Updates window and checks for a new version (chapter *Installation and updates*). **Close**, the **×**, or `Esc` close the window.

Third-party licenses



The embedded fonts and data and the Rust components, with the version and license of each one.

The **Third-party licenses** button in the About window lists the third-party fonts and data that ship with the program (Inter and JetBrains Mono, OFL 1.1 license; the subset of DejaVu Sans Mono and the DejaVu Sans used for the drawing's texts, Bitstream Vera license; the code page tables of old DXF, Unicode license) and all the open source components that go inside the program, with the version and the license (the SPDX expression of each one). The **Search** field filters by name or by license; hovering the mouse over the name shows the address of the source code.



egui chosen in the list: the license used, the copyright notices, and the license text.

Clicking a component shows its **full text** below the list, the same as in the third-party notices file, with the labels and explanations in the interface language and the copyright notices and licenses in the original: the declared license and the **license used** (in a component with a choice of licenses, such as *MIT OR Apache-2.0*, BadBit CAD uses MIT when the package includes its file), the **copyright notices**, the **license text** and, in full, the NOTICE files and the third-party code files the package embeds. When the package does not include the license file, the license's standard text applies, marked *standard text, no copyright notice in the package*. The text can be selected and copied.

Save all notices... writes the whole file, in the interface language (a text file with all the components), wherever you choose. In the browser, the button is **Copy all notices** and copies the text to the clipboard. Each system's installer ships the three languages with the program: `THIRD-PARTY-NOTICES` (Portuguese), `THIRD-PARTY-NOTICES.en` (English), and `THIRD-PARTY-NOTICES.es` (Spanish).

Navigation

Action	How
Zoom at the cursor	Mouse wheel; on the trackpad, pinch with two fingers.
Drag the view (pan)	Middle button held down; on the trackpad, drag with two fingers.
Zoom extents (fit everything)	Double-click the middle button, `ZE`, **ZOOM** Extents (`Z` and `E`), or the button on the ribbon (View tab).
Zoom with options	**ZOOM** (alias `Z`), as in AutoCAD; see below.
Pan by two points	**PAN** (alias `P`): the base point and the second point; `Enter` at the second uses the first as the displacement.

Zooming out stops when the view reaches 20 times the size of the drawing (at least 10 km, in an empty or small drawing; a drawing larger than that always fits whole). This way nobody gets lost millions of meters from the origin without noticing: the grid adjusts to the zoom and looks the same at any distance. Zooming in has no such limit. In an empty drawing, zoom extents goes back to the initial view (about 25 m wide around the origin); with a single point, the view shows 20 m around it.

ZOOM asks for a window corner, a factor, or an option:

Answer	What it does
Click (two corners) or **Window** (`W`)	Fits the window.
All (`A`) and **Extents** (`E`)	Fit the whole drawing.
Previous (`P`)	Goes back to the previous view (up to ten back).
Center (`C`)	The point goes to the middle of the view, with the typed height (or the factor `nX`).
Scale (`S`) or a factor `2x`, `0.5x`	Zooms the current view in or out by that factor. A number without `x` is relative to the whole drawing.
Object (`O`)	Fits the objects selected before the command.

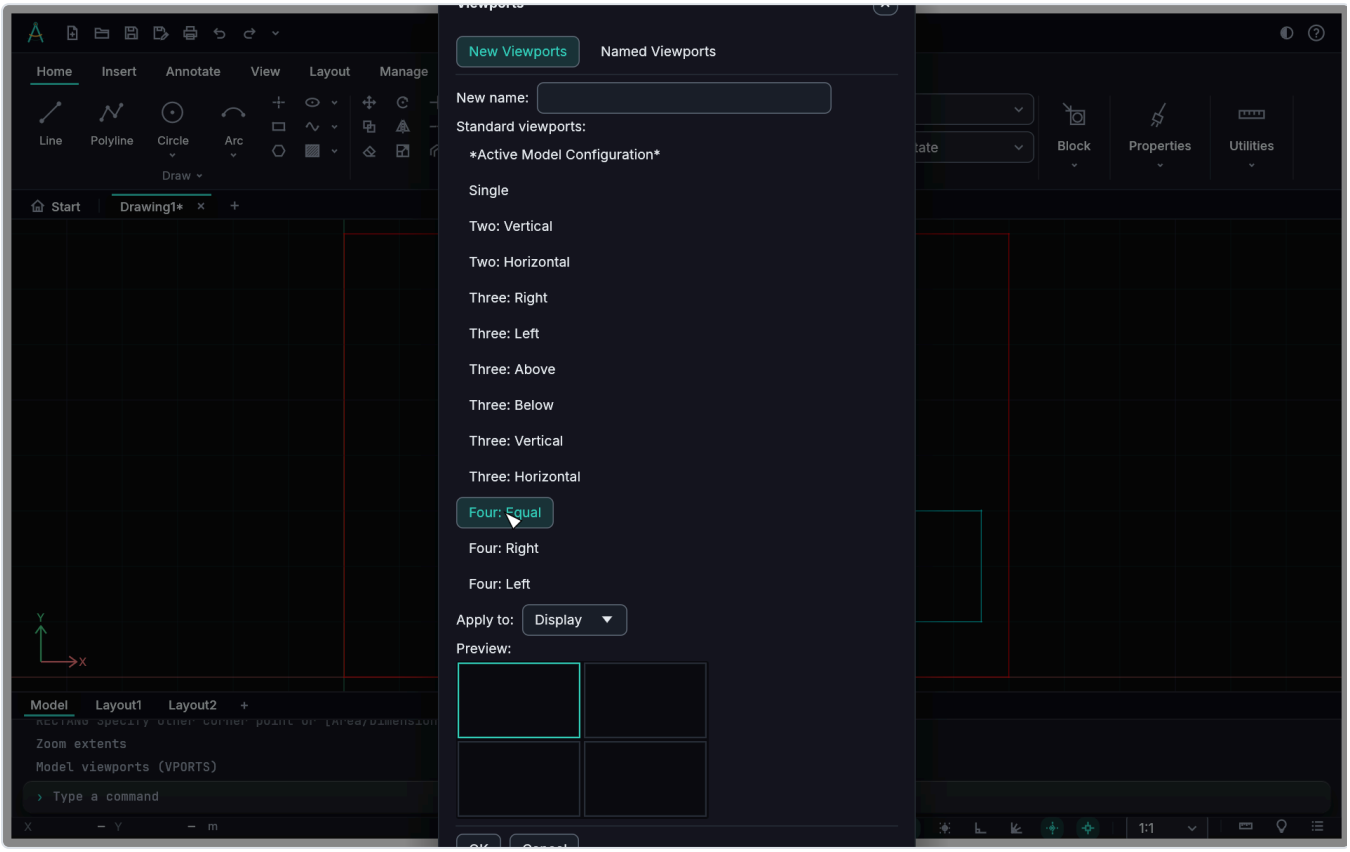
Zooming is always smooth, even with millions of lines. Circles and arcs are redrawn with more or less detail depending on the zoom, gradually, without freezing the screen.

REGEN (aliases RE, REA, REGENALL)

Rebuilds the image of the whole drawing (the curves, the linetypes, and the points). BadBit CAD already keeps the image up to date by itself; REGEN is there for those used to AutoCAD or who suspect something on the screen.

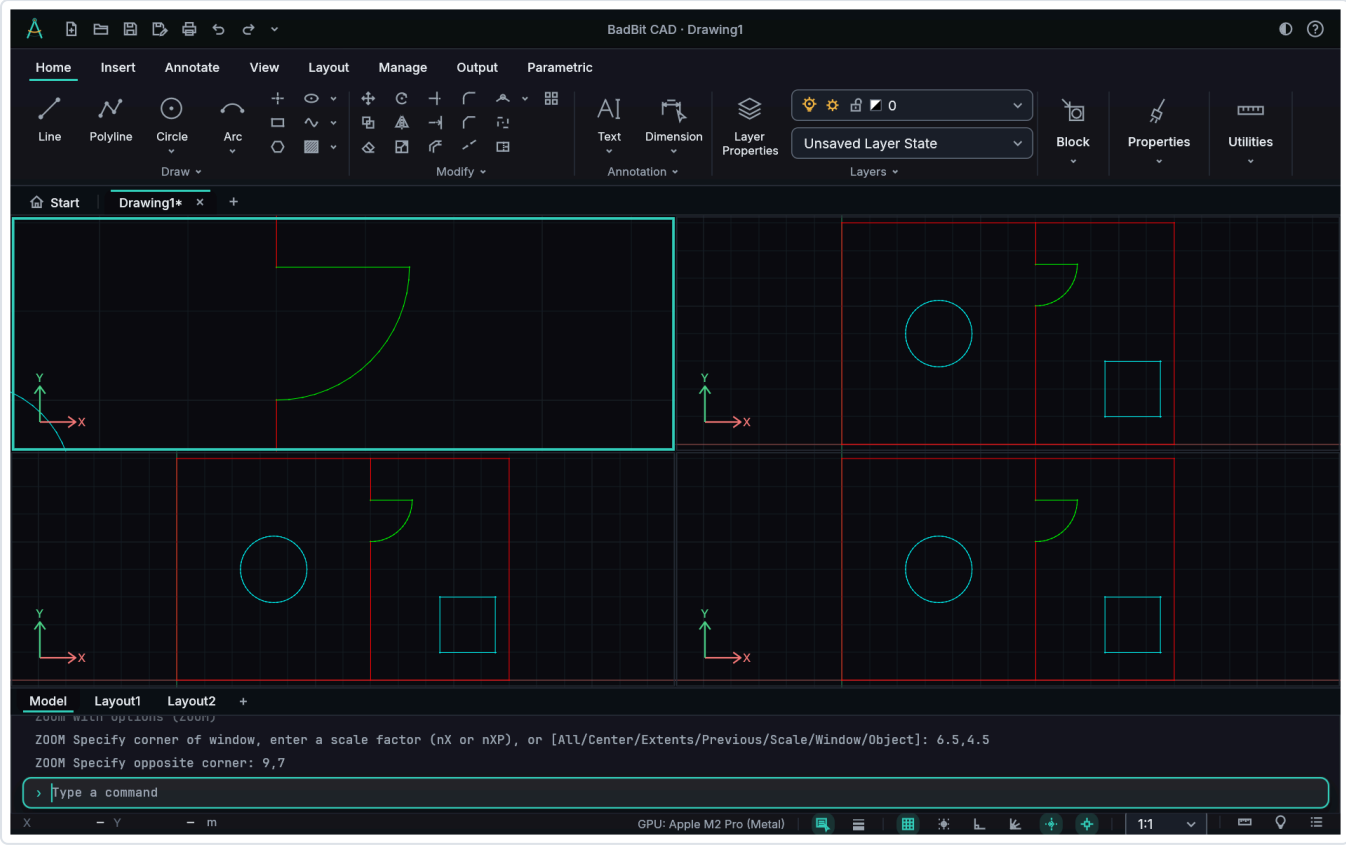
Model viewports: VPORTS and -VPORTS

VPORTS splits the drawing area into side-by-side viewports, as in AutoCAD: each one with its own zoom and pan, the active one with its border highlighted. The **Viewports** window (View tab, **Model Viewports** panel) has the standard configurations (Single; Two: Vertical and Horizontal; Three: Right, Left, Above, Below, Vertical, and Horizontal; Four: Equal, Right, and Left), with the preview; **Apply to**: **Display** (the whole screen) or **Current Viewport** (splits only the active one). A **New name** saves the configuration; the **Named Viewports** tab restores, renames, and deletes the saved ones.



The Viewports window with Four: Equal.

Click a viewport to make it the active one (the click does not become a point); `Ctrl`+`R` moves to the next one. A command continues from one viewport to the other: LINE starts at a point in one viewport and ends in another, after the click that activates it. The wheel and the middle button zoom and pan in the viewport under the cursor, even without activating it. ZOOM, PAN, and zoom extents apply to the active one.



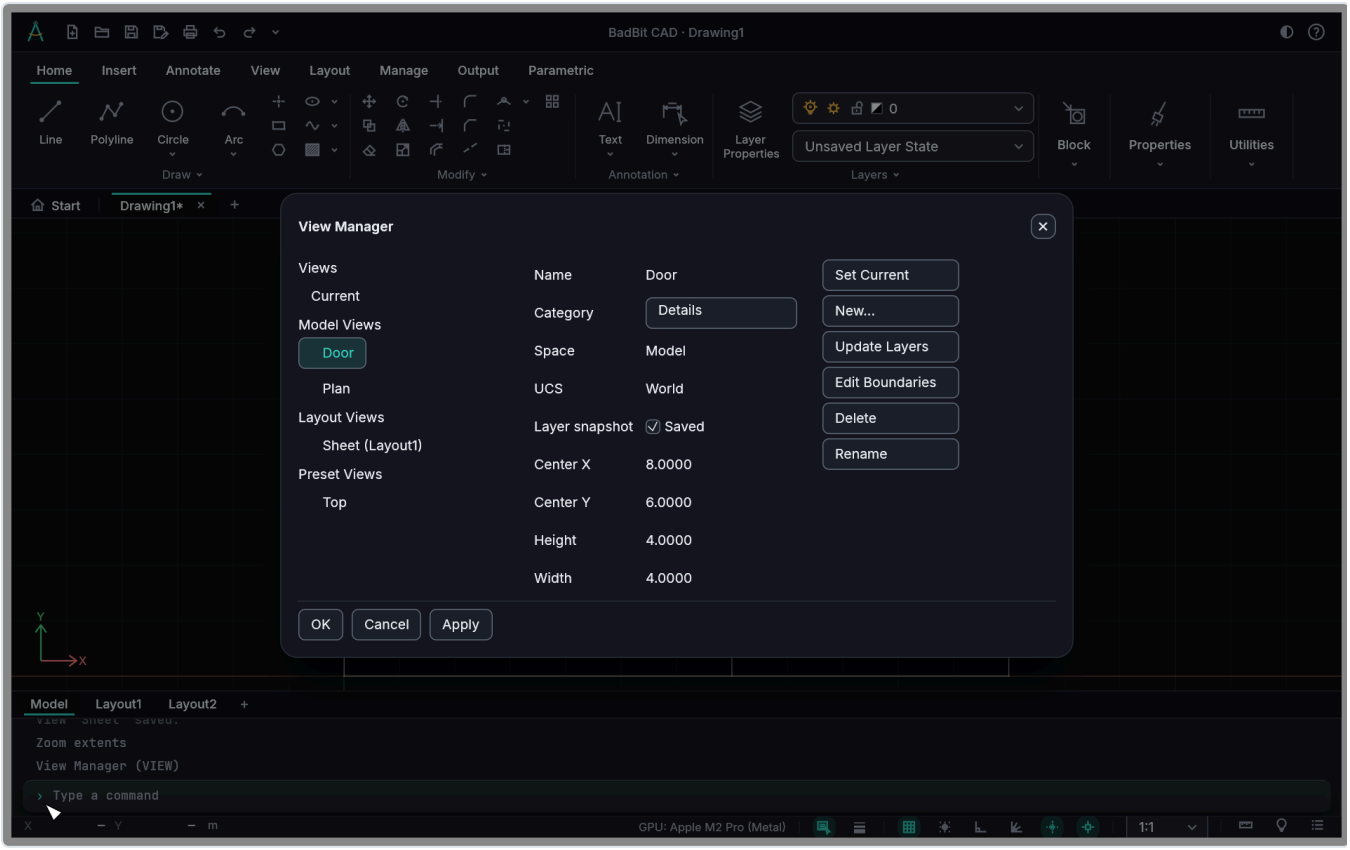
Four model viewports: the active one (with the border) zoomed in on the door.

-VPORTS does the same on the command line: **Save** (`S`), **Restore** (`R`), **Delete** (`D`), **Join** (`J`: the dominant viewport, with `Enter` the active one, and the neighbor to join; the two must form a rectangle), **SIngle** (`SI`), `?` (lists the saved ones), **2** (Horizontal or Vertical), **3** (Horizontal, Vertical, Above, Below, Left, or Right), and **4** split the active viewport, and **Toggle** (`T`) shows only the active one and goes back. In a layout, VPORTS and -VPORTS are MVIEW.

The viewports draw the same scene of the drawing (without copying anything), with and without the GPU. The configuration on the screen goes with the drawing (in DWG and DXF, in the `*Active` entries of the VPORT table, the way AutoCAD reads them) and so do the saved ones (in DWG and DXF, the named entries of the same table).

Named views: VIEW, -VIEW, and VIEWGO

A **named view** stores what is on the screen (the center and the size) under a name, to go back to it later, as in AutoCAD. **VIEW** (alias `V`; **View** tab, **Named Views** panel, **View Manager** button) opens the **View Manager**:



The View Manager with two model views and one layout view; in the middle, the properties of the chosen one (the Door, with the Details category).

- On the left, the list: **Current** (the view on the screen), **Model Views**, **Layout Views** (with the layout name in parentheses), and **Preset Views** (**Top**, the only one in a 2D drawing: it fits the whole drawing). Right-clicking a view offers Set Current, Rename, and Delete.
- In the middle, the properties of the chosen one: the name, the **Category** (type to change it; categories group the views, as in AutoCAD's Sheet Set Manager), the space (Model or the layout), the UCS (always World), the **Layer snapshot** (Saved or Not saved; checking it saves the current one), the center, the height, and the width.
- On the right, the buttons: **Set Current** (the view comes back to the screen on OK), **New...**, **Update Layers** (the view's layer snapshot becomes the current one), **Edit Boundaries** (the window disappears, the view appears highlighted with the rest darkened, and two clicks define the new boundary; `Enter` keeps what it was), **Delete**, and **Rename** (the field for the new name appears below; the second click on Rename confirms).

What you change in the Manager only goes into the drawing on **OK** or **Apply**, in one undo step; **Cancel** gives up on everything. **New...** opens the **New View** window: the name, the category, the boundary (**Current display** or **Define window**, with **Pick on screen** for the two corners), and **Save layer snapshot with view**. The new view goes into the drawing immediately (the Manager's Cancel does not remove it). A name that already exists asks for a second OK to replace it.

Next to the button, on the panel, is the list of the drawing's views: choosing one restores it (the first item, **Unsaved View**, only shows that the screen is not a saved view).

Restoring a view is like a ZOOM: the whole view fits on the screen (or in the viewport) and the previous one goes into **ZOOM Previous** (`Z` `P` goes back). If the view has the layer snapshot, the on/off and frozen state of each layer comes back too, in one undo step (the current layer is not frozen; layers created later stay as they are). Views apply in the space where they were saved:

- A **model view** restored on the Model tab fills the screen; in a layout, it goes to the active viewport; on paper, BadBit CAD asks **Select viewport for view:** and clicking a viewport enters it (MSPACE) and shows the view there.
- A **layout view** can only be restored in the layout: on the Model tab, the command line says so. Restored in another layout, BadBit CAD goes to its layout; inside a viewport, it goes back to paper.

-VIEW (alias `-V`) does the same on the command line, with AutoCAD's options:

Option	What it does
`?`	Lists the views that match the names (commas, `*`, and `?`), with the space: `M` (model) or `P` (paper).
Delete (`D`)	Deletes the views by name (with wildcards: `A*`).
Orthographic (`O`)	**Top** (`T`, the default) fits the whole drawing; the others are 3D and do not exist in a 2D drawing.
Restore (`R`)	Restores the view by name (`?` lists).
Save (`S`)	Saves the screen under the name. A name that already exists asks **Replace it? [Yes/No] <N>**.
sEttings (`E`)	For new views: **Categorize** (`C`, the category; `.` removes it), **Layer snapshot** (`L`, Yes or No; the default is Yes), and **Ucs** (`U`; a 2D drawing only has World).
Window (`W`)	Saves the window of two corners as a view (without saving the whole screen).

VIEWGO restores the view by name, directly (`VIEWGO` `Enter` and the name).

Views go with the drawing: in the `.bbd` and in the VIEW table of DWG and DXF (AutoCAD reads them in its View Manager). The category, the layout, and the layer snapshot go in the `BADBIT_VIEW` XDATA (AutoCAD keeps what it does not understand). The 3D views of an AutoCAD file (side, rotated, or perspective views) do not appear in BadBit CAD, but they come back unchanged when saving in the same format and version.

Entering points and answering commands

Confirming, canceling, and repeating

Key	Effect
`Enter`, spacebar, or right mouse button	Confirms; with no active command, repeats the last command.
`Esc`	Cancels the command (LINE and PLINE keep the segments already drawn).
`Delete`	Erases the selected objects.

Coordinates

A point can be clicked in the drawing or typed. The comma separates the coordinates and the decimal separator is the period, as in AutoCAD.

Format	Example	Meaning
Absolute	`10,5`	The point x = 10, y = 5.
Relative	`@3,0`	3 units in x from the last point.
Relative polar	`@5<45`	5 units at 45° from the last point.
Direct distance	`5`	5 units in the direction of the cursor (with Ortho or Polar, in the locked direction).
With a unit	`250 cm`, `@1.2 m,0`	Converts to the project unit.

With no unit written, the number is in the project unit (see *Grid and units*).

Dynamic input

Dynamic input is always on, in any command, whether started from the ribbon or typed. While the command asks for a point, a box next to the cursor shows the prompt. From the second point on, a **temporary dimension** appears along the segment, with the length in the project unit, and an arc with the angle from the X axis. For the circle and for the radius of the arc, the dimension shows the radius. None of this becomes an object: it disappears when the command ends. Real dimensions, which stay in the drawing, will come with the dimension commands.



LINE from (0, 0): the temporary dimension shows 5.385 m at 22°.

Typing in the fields, with Tab

You do not need to click anywhere to type: the number goes straight to the highlighted field next to the cursor, as in AutoCAD and Fusion 360.

- **Before the first point**, the fields are **X** and **Y**. Type X, press `Tab`, and type Y; `Enter` confirms the point.
- **After the first point**, the fields are the **length** and the **angle**. Type the length, press `Tab`, and type the angle (in degrees, from the X axis); `Enter` confirms.
- `Tab` **locks** the typed value (a padlock appears) and moves to the other field; another `Tab` goes back. With a locked value, the preview and the click respect it: with the length locked, just point the direction with the mouse and click.
- Only the length and `Enter`, without `Tab`, is the direct distance, in the direction of the cursor (with Ortho or Polar, in the locked direction).
- Fully typed coordinates (`3,4`, `@5<45`) still work and appear in the cursor box.



At the first point: X locked at 5 (padlock) and Y receiving what is typed.



After the first point: length locked at 3 and the angle receiving 30.



Typing 5 without Tab: the value goes to the dimension box, and Enter uses the direction of the cursor.

For those coming from AutoCAD who prefer to work without it, `F12` turns dynamic input off and on.

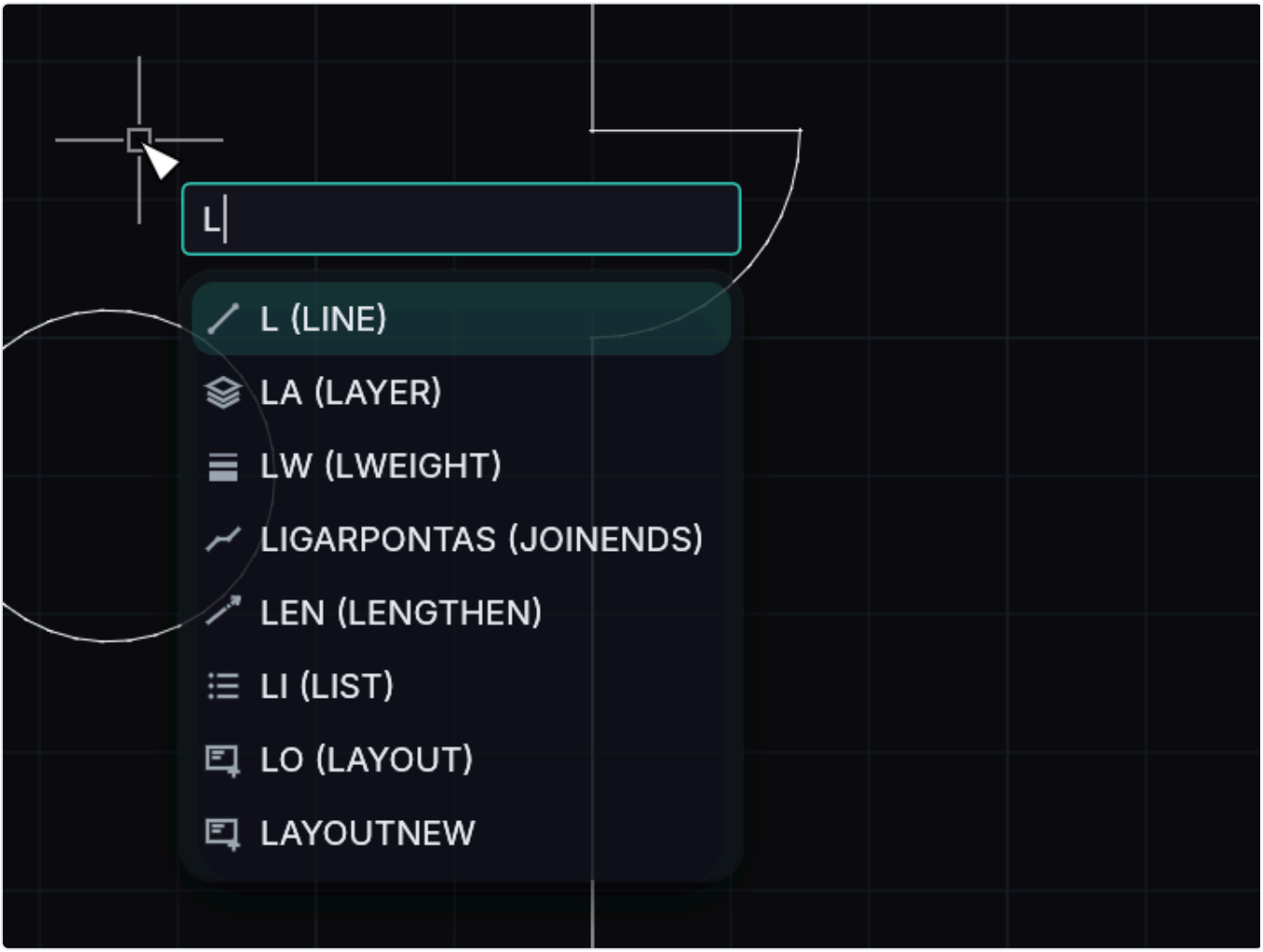
Command and prompts next to the cursor (DYNPROMPT)

With dynamic input on, what you type over the drawing appears **next to the cursor**, as in AutoCAD 2026, without having to look at the command line:

- **With no command**, when you type (for example `L`), a small box with the text appears just below and to the right of the cursor, with the **suggestion list** underneath: the command icon (the same as on the ribbon), the alias, and the name in capitals, such as `L (LINE)`, `LA (LAYER)`. The order is the same as the command line suggestions (see *Suggestion order*, below). Eight fit at once; the others appear when you scroll the list with the mouse wheel.

- The ↑ and ↓ arrows choose the suggestion; `Enter` or the spacebar run it; `Esc` clears the text and closes the small box. Clicking a suggestion also runs it.
- **During a command**, the current prompt ("Specify next point or [Undo]:") appears next to the cursor, and what you type appears in the same box. The length and angle fields stay the same.
- When the prompt has **options in square brackets**, a small down arrow appears next to it. The ↓ key opens the **small list of options** right there; the arrows choose, `Enter` or the spacebar confirm, and clicking the option also chooses it. `Esc` closes only the small list, without canceling the command.

The command line at the bottom stays the same and shows the same thing: there is a single text, typed in one place it appears in the other, and the answer chosen in the small list goes into the history as if it had been typed. While the suggestion list is next to the cursor, the one above the command line does not appear, so as not to be repeated, as in AutoCAD; when typing in the command line with the mouse outside the drawing, the list appears there, as before.



Typing L over the drawing: the small box next to the cursor and the suggestions, with L (LINE) first.



LINE after three points: the prompt next to the cursor and the small list of options opened with the down arrow.

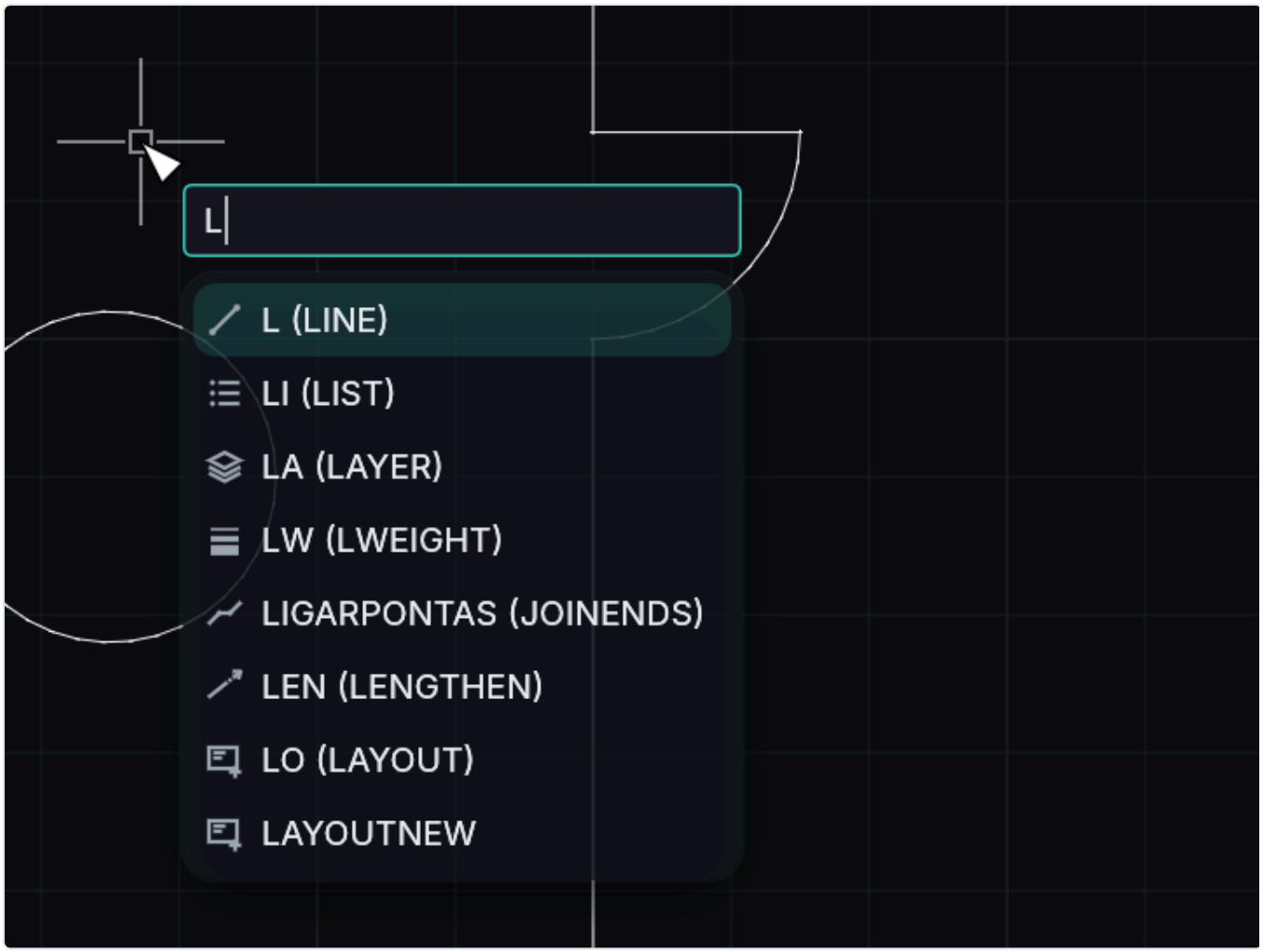
The **DYNPROMPT** variable turns the command and prompts next to the cursor on (1, the default, as in AutoCAD) and off (0); with 0, or with `F12` off, everything stays only on the command line.

Suggestion order and search options (INPUTSEARCHOPTIONS)

As in AutoCAD, the suggestion list (next to the cursor and on the command line) is sorted **by frequency of use**: each command run from the command line (typed, chosen in the list, or repeated with an empty Enter) earns a point, and the most used ones move up. Three rules stay fixed:

- the **exact name or alias** always comes first: `L` and `Enter` is still LINE, no matter how much another command with L is used;
- next come the ones that **start** with the typed text, from most used to least used;
- finally, the ones that only **contain** the text (in the friendly name or, with mid-string search, in the command name), also by use.

With no use at all, the order is the usual one. The count is stored in your **profile** (Appearance section), goes along when exporting and importing the profile, and goes back to zero with the profile's **Reset**.



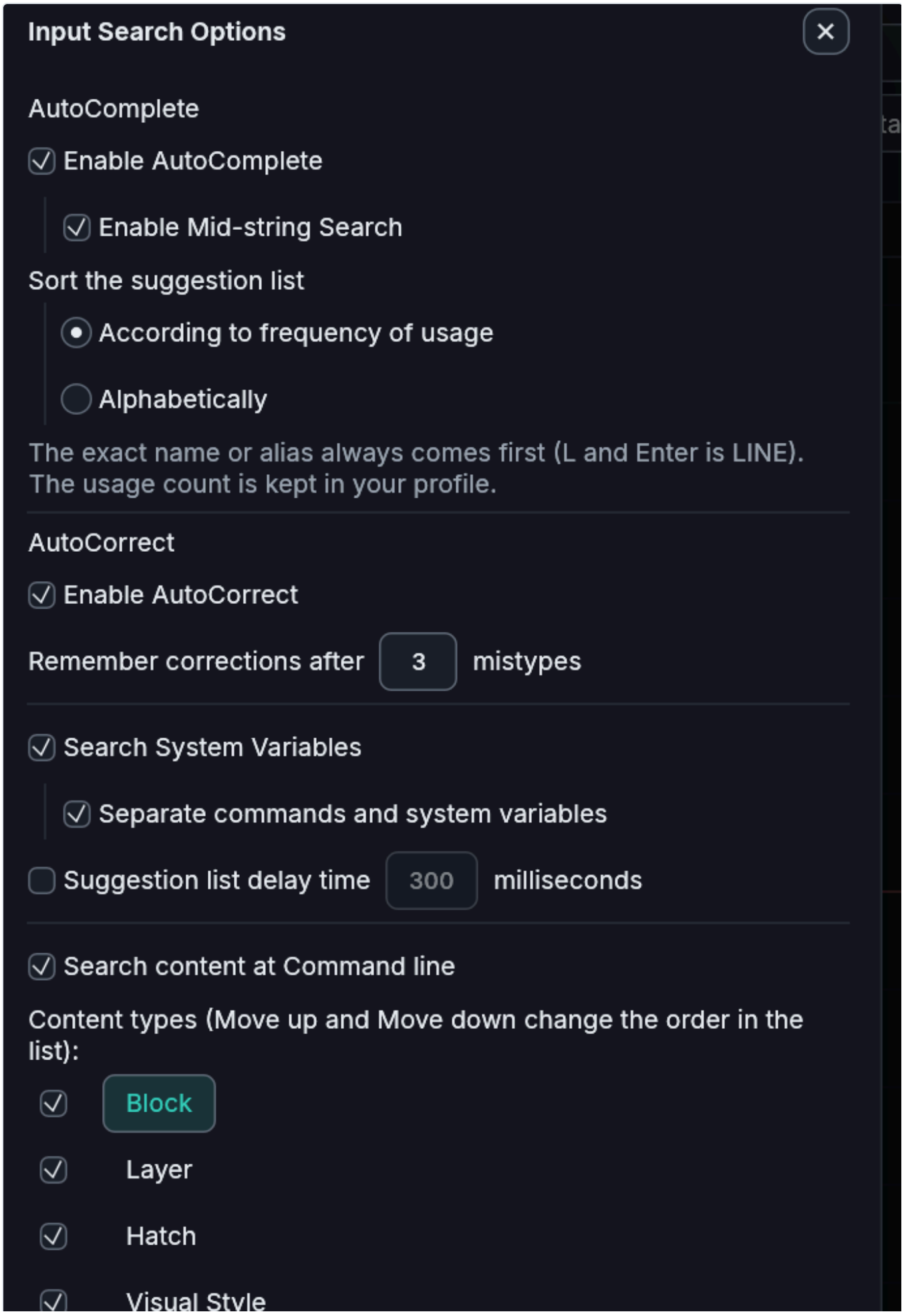
After using LIST a few times: typing L, LINE is still first and LIST moves up to right after it.

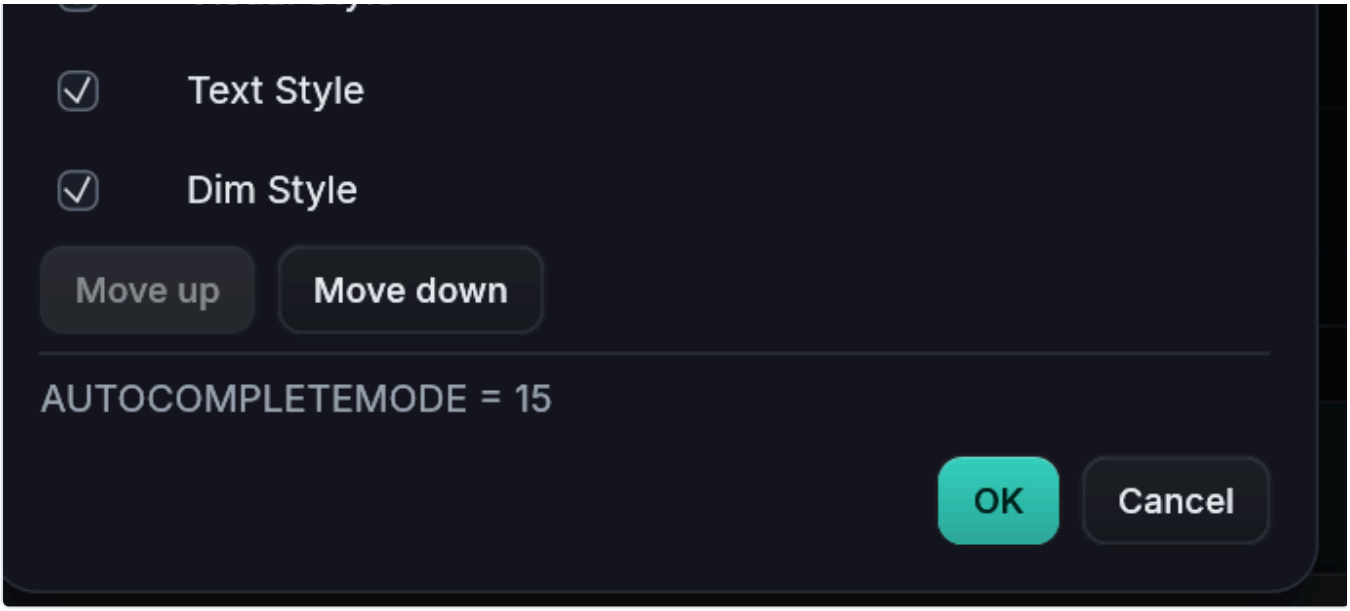
INPUTSEARCHOPTIONS opens the **Input Search Options** window, like AutoCAD's:

- **AutoComplete** — **Enable AutoComplete**: the commands that start with what you type go into the list and, from the third character on, the rest of the name of the first suggestion appears grayed out after the text (`PLI` shows `PLI` **`NE`**); when off, only the exact name or alias works. **Enable Mid-string Search** (only with AutoComplete): also the ones that only *contain* the text (`LINE` finds `PLINE` and `XLINE`). **Sort**: **According to frequency of usage** (the default) or **Alphabetically** (the exact match still comes first; the rest by command name).
- **AutoCorrect** — **Enable AutoCorrect**: BadBit CAD learns your typing mistakes. When what you type finds nothing (`LIEN`) and the command run right after resembles it (`LINE`, up to two mistakes), the mistake counts once; after **Remember corrections after N mistypes** (3 to start with), `LIEN` becomes LINE. The learned list is stored in the profile; AutoCAD's (`autoCorrectUserDB.pgp`, lines `WRONG, *RIGHT`) comes in through the profile's **Import**.
- **Search System Variables**: the variables (PICKADD, OSMODE…) go into the list; when off, they leave the list, but the exact name still works. **Separate commands and system variables** (only with the previous one; on to start with): the variables leave the middle of the commands and go into a separate list, below them, behind the **System Variables (N) ▼** line. The `Tab` key opens that list (the first variable is chosen) and closes it again; clicking the line, or `Enter` with it chosen, does too. It works on the command line and in the list next to the cursor; the exact name of a variable (`PICKADD`) still comes first, among the commands. With the box off, the variables mix with the commands again.

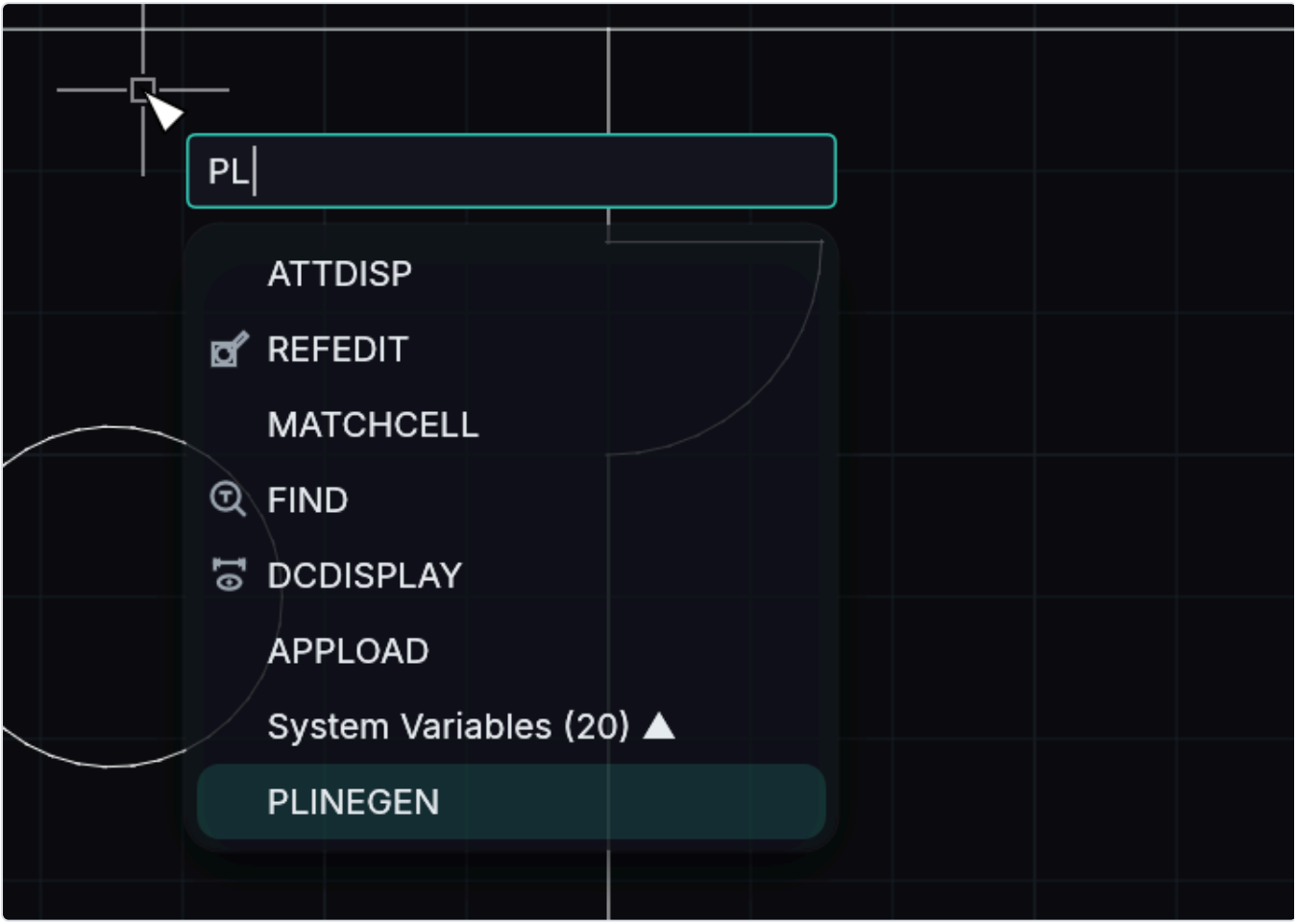
- **Suggestion list delay time**: when on, the list waits the given time (in milliseconds, from 100 to 10,000; this is the **INPUTSEARCHDELAY** variable) after the last keystroke.
- **Search content at Command line**: the name of a **block**, a **layer**, a **hatch** pattern, a **visual style**, a **text style**, or a **dimension style** in the drawing also appears in the list, after the commands ("Layer: PAREDES"). Choosing the layer or a style makes it current; choosing the block starts INSERT with it; choosing the pattern starts HATCH with it. The box for each type turns it on or off, and **Move up** and **Move down** change the order of the types in the list. BadBit CAD draws in 2D, always in wireframe: the only visual style is **2D Wireframe** (AutoCAD's name), which is already current; choosing it in the list ("Visual Style: 2D Wireframe") only confirms it. AutoCAD's shaded and 3D styles do not exist here.

At the bottom of the window appears the value of the **AUTOCOMPLETEMODE** variable, which moves together with the boxes (AutoCAD's bits: 1 turns everything on, 2 AutoComplete, 4 the list, 8 the icons of the list next to the cursor, 16 removes the system variables, 32 the delay; the default is 15). Changing the variable changes the boxes, and vice versa; with 0, only the exact name or alias works and no list appears. **-INPUTSEARCHOPTIONS** does the same from the command line: `Autocomplete (A)`, `autocorRect (R)`, `System variables (S)`, `sEparate (E)` (separate commands and system variables), `Content (C)`, `Midstring (M)` with Yes (Y) or No (N), and `Delay (D)` in milliseconds; `Enter` finishes. All of this is stored in the **profile** (Appearance section) and applies both on the command line and in the list next to the cursor.


Input Search Options
AutoComplete
Enable AutoComplete
Enable Mid-string Search
Sort the suggestion list
According to frequency of usage
Alphabetically
The exact name or alias always comes first (L and Enter is LINE).
The usage count is kept in your profile.
AutoCorrect
Enable AutoCorrect
Remember corrections after 3 mistypes
Search System Variables
Separate commands and system variables
Suggestion list delay time 300 milliseconds
Search content at Command line
Content types (Move up and Move down change the order in the list):
Block
Layer
Hatch
Visual Style



The Input Search Options window: AutoComplete with mid-string search, AutoCorrect, system variables (with the box that separates them from the commands), delay, and content search with the six types, with AUTOCOMPLETEMODE at the bottom.



Typing PL with the variables separated: the commands at the top and the System Variables line; Tab opened their list (PLINEGEN), with the first one chosen.



Typing the beginning of a layer name: it appears in the list next to the cursor; Enter makes it current.

Command options

Options appear in square brackets in the prompt, as in "[Close/Undo]". You type the letter (or the word): `C` for Close, `U` for Undo. The letters are AutoCAD's English ones, in any interface language; in a translated interface, the letters of the translated words also work.

User coordinate system (UCS)

The **UCS** (in the Portuguese version of AutoCAD, SCU) changes the origin and rotation of the coordinates you type and see, without touching the drawing: entities remain stored in the world coordinate system (WCS, World). Since BadBit CAD draws plans, the UCS is always an origin and a rotation in the XY plane (or that plane seen from below).

With a current UCS:

- `10,5`, `@3,0`, and `@5<45` apply on the UCS axes (angles count from its X, with ANGBASE and ANGDIR on top);
- `*` in front forces the world, as in AutoCAD: `*10,5`, `@*3,0`, `*5<45`; `#` forces absolute (`#10,5`), useful with dynamic input;
- Ortho, Polar, orthogonal tracking, the **grid**, and **grid snap** follow the UCS axes, and the cursor crosshairs rotate with it;
- the status bar coordinates, **ID**, **DIST** (deltas and angle), and **LIST** show the values in the UCS;
- **TEXT** and **MTEXT** with rotation 0, **insertion** with rotation 0, the Horizontal and Vertical linear dimensions (and the automatic choice), the **ordinate** dimension (UCS origin and axes), **RECTANG**, **XLINE** Hor/Ver/Ang, and the **hatch** (pattern angle and default origin) are created aligned with the UCS.

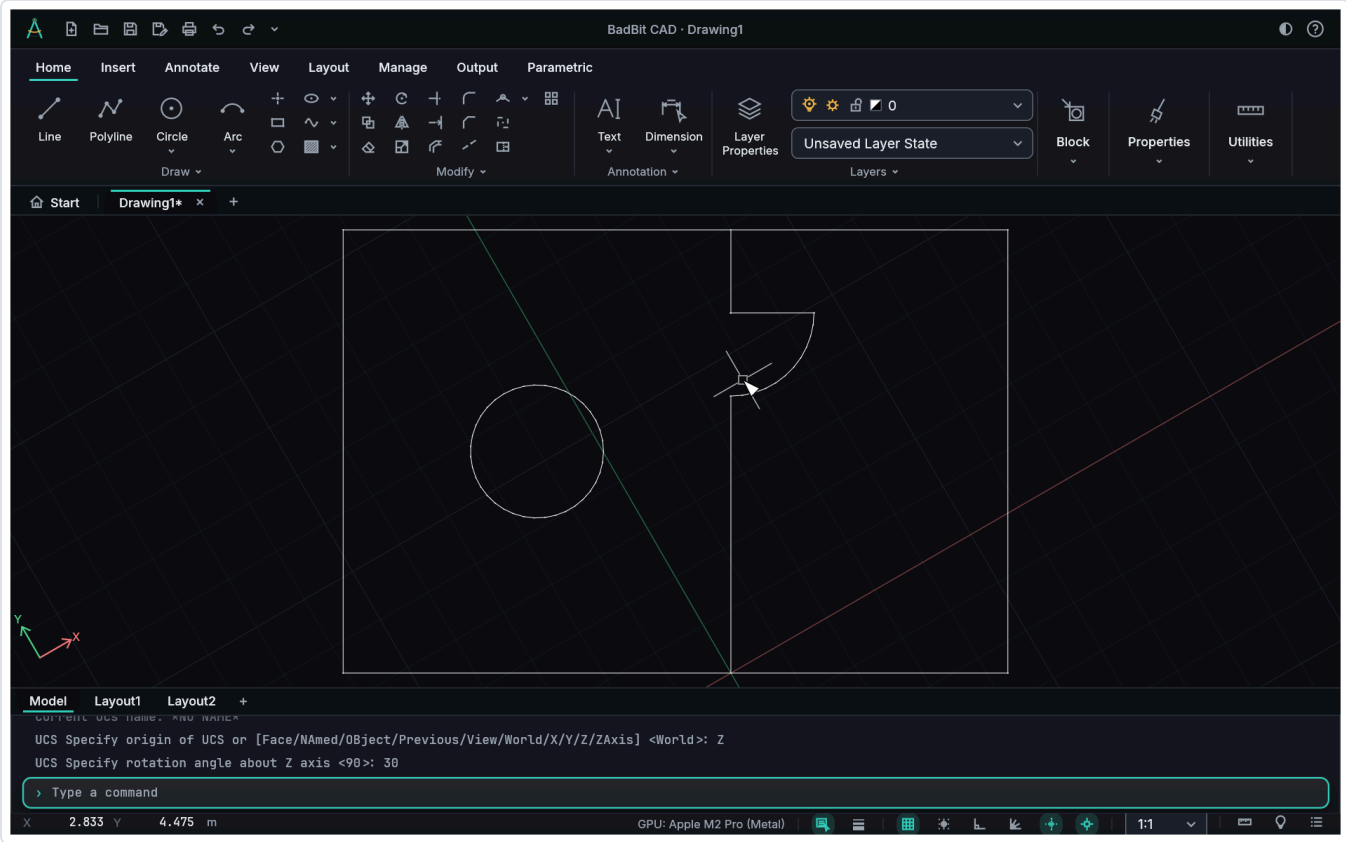
The **UCS** command asks "Specify origin of UCS or [Face/NAmed/OBject/Previous/View/World/X/Y/Z/ZAxis] <World>":

Answer	What it does
a point	Moves the origin. A second point gives the positive X and a third tells which side Y is on (3 points, in the plane); `Enter` accepts.
Enter or W	Goes back to World.
P	The previous UCS (BadBit CAD keeps the last 10 for each space, model and paper, like AutoCAD).
Z	Rotates around Z (the angle, <90> from UCSAXISANG, or two points).
X, Y	Only 180° (the plane seen from below): any other angle would take the plane out of the plan, and BadBit CAD reports that it is 2D and asks again (AutoCAD would rotate the plane and show the broken pencil).
OB	Aligns with the object: for the line and the polyline, the origin at the endpoint nearest the click and X along it; for the arc, at the center with X toward the nearest endpoint; for the circle, at the center with X toward the click; for text, insertion, and dimension, their origin and rotation.
V	The screen axes (the plan does not rotate), at the same origin.
ZA	Only the new origin (Z stays pointing up).
NA	Restore (R), Save (S), Delete (D, with `*` and `?`), and `?` (lists with the origin and axes relative to the current UCS); R, S, D, and `?` work directly at the first prompt.
A	Applies the current UCS to all model viewports (VPORTS).
F	There is no solid face in 2D: BadBit CAD says so.

Each UCS change is an **undo** step (U brings back the UCS) and goes into the file. With the screen split (VPORTS) and **UCSVP** 1, each viewport keeps its own UCS.

UCSICON turns the icon on (ON) and off (OFF), puts it at the origin (OR) or in the corner (N), says whether it can be selected (S), and opens the **Properties** (P: 2D or 3D style, size, and line width). **UCSMAN** (alias UC) opens the **UCS** window: the list of named ones (Set Current, Rename, Delete), the orthographic ones that work in plan (Top and Bottom), and the Settings (the icon, UCSVP, and UCSFOLLOW). **RENAME** also renames named UCSs (the UCS type, `U` in -RENAME), and a **named view** stores the current UCS and restores it along with it (UCSVIEW 1, the Ucs option of -VIEW sEttings).

The UCS goes into and comes back from `.bbd`, DXF, and DWG: the UCS table, the UCSNAME, UCSORG, UCSXDIR, UCSYDIR variables (and the paper's PUCS* ones), the `*Active` viewports of the VPORT table, and the UCS of each view in the VIEW table.



The UCS with the origin at 7,0 and rotated 30°: the grid, the colored axes, and the crosshairs follow the UCS, the icon shows its axes, and the status bar gives the coordinates in the UCS.

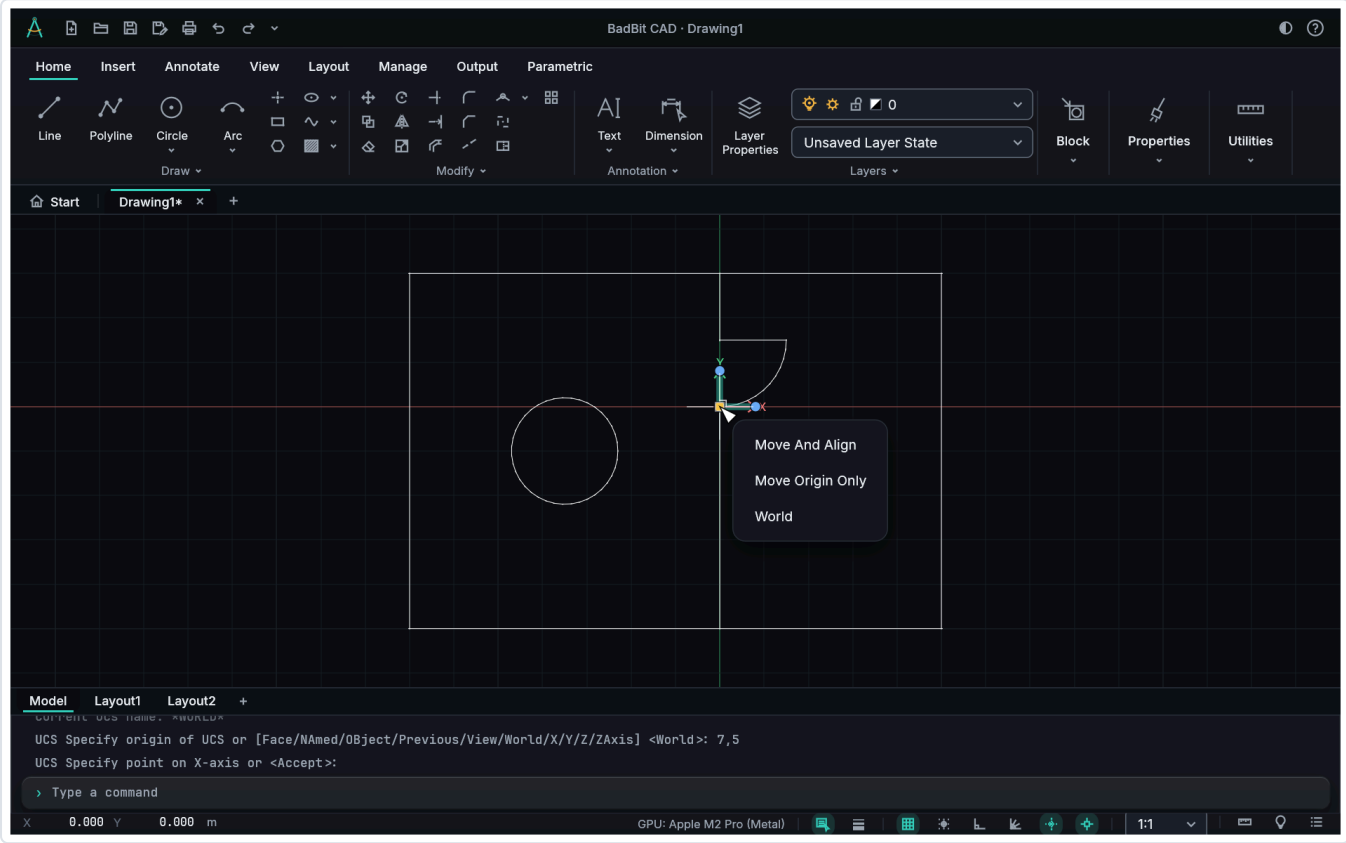
Moving and rotating the UCS with the icon

As in AutoCAD, the UCS icon can be picked with the mouse (UCSICON Selectable, the **UCSSELECTMODE** variable 1, the default; with 0, the icon only shows the axes). With no command, moving the cursor over the icon highlights it; clicking selects it and shows the **grips**: the square at the origin and the round ones at the ends of the X and Y axes.

- **Origin grip**: click it (or drag it) and give the new origin point. Object snaps, the grid, Ortho, polar, and typed coordinates work (`10,5` in the current UCS, `@2,0` from the grip, `*10,5` in the world). The axes stay as they were.
- **Axis end grips**: the axis now points from the icon to the given point (for example, the Endpoint of a slanted wall); the origin stays and the UCS rotates in the plane, with Z pointing up.
- **Grip menu**: hovering half a second over a grip, the menu appears. On the origin grip: **Move And Align**, **Move Origin Only**, and **World** (goes back to the world UCS). On the X grip: **Rotate Around Y Axis** and **Rotate Around Z Axis**; on the Y grip: around X and around Z. Around Z, type the angle (added to the current rotation) or give a point. Around X or Y the same rule as the X and Y options of the UCS applies: only 180° stays in plan (the plane seen from below); any other angle answers that BadBit CAD is 2D and the prompt continues.

What is left out in 2D: in AutoCAD, **Move And Align** aligns the XY plane with the face of a solid, a surface, or a mesh under the cursor; BadBit CAD draws plans and has no such faces, so it only moves the origin (the same as Move Origin Only). The Z axis grip of the 3D icon does not appear either: in a plan, Z points toward the viewer.

Each change through the icon is an **undo** step and goes into **UCS Previous** (UCS P), like the UCS command. The selected icon does not become part of the object selection (Delete erases nothing); **Esc**, a click outside, selecting objects, or starting a command deselect it. With the screen split (VPORTS), each viewport's icon responds when that viewport is the active one and changes its UCS; in the layout, it changes the paper's UCS. The icon and the grips are the same with and without the graphics card.



The icon selected at the UCS origin (7,5, the end of the wall), with the origin square, the round grips at the ends of the axes, and the origin grip menu.

Scripts (SCRIPT, alias SCR)

A script is an `.scr` text file with what you would type on the command line, like AutoCAD's: command names, answers, coordinates, and options. **SCRIPT** (`SCR`) runs it: with **FILEDIA** at 1 it picks the file in a dialog; with FILEDIA at 0 it asks for the name on the command line (without the extension, `.scr` is assumed; an empty `Enter` repeats the last one). Dragging an `.scr` from the file manager onto the BadBit CAD window also runs it (the command in progress is canceled first).

The file rules are AutoCAD's:

- **Each space and each end of line count as an `Enter`**. Two consecutive spaces are two Enters; an empty line is one Enter (with no command in progress, it repeats the last command, as on the keyboard). That is why the end of a LINE needs an extra space or an empty line.
- A line that starts with `;` is a comment.
- At prompts that accept spaces (the text of **TEXT**, a file name), the space is a character and only the end of the line confirms.

- The prefixes work as when typed: `_` (the English name), `.` (the original command), `'` (transparent), and `-` (the command-line version), combined if needed (`._-HATCH`). A command that has a command-line version runs that version in the script (`HATCH` becomes `-HATCH`, without the panel), and the file dialogs give way to the prompt (FILEDIA is 0 while the script runs).
- The file can be UTF-8 or ANSI (Windows-1252, the one used by the old Notepad): accented characters arrive correctly.

Example, which draws a rectangle of lines, a circle by its diameter, and a rectangle:

```
; quadro.scr
LINE 0,0 10,0 @0,6 C
CIRCLE 2P 3,3 7,3
RECTANG 12,0 @6,6
ZOOM E
```

The command line history shows each prompt with the script's answer, as if it had been typed, and each command in the script is an undo step.

Interrupting and resuming. `Esc` or `Backspace` interrupt the script (Esc also cancels the command in progress). An error also stops the script, at its line: a command that does not exist ("Unknown command") or an answer the command rejects (an invalid point); the command line tells the line of the error. **RESUME** continues from where it stopped (in the middle of a command, `'RESUME`).

Repeating and waiting. RSCRIPT goes back to the beginning of the script: on the last line of the file, the script repeats until Esc (for demonstrations). **DELAY** waits the given time, in milliseconds, before the next command (from 0 to 32767, a little under 33 seconds; outside that range, DELAY rejects it and the script stops).

Left out, for now: running a script when opening BadBit CAD from the system command line (AutoCAD's `/b`).

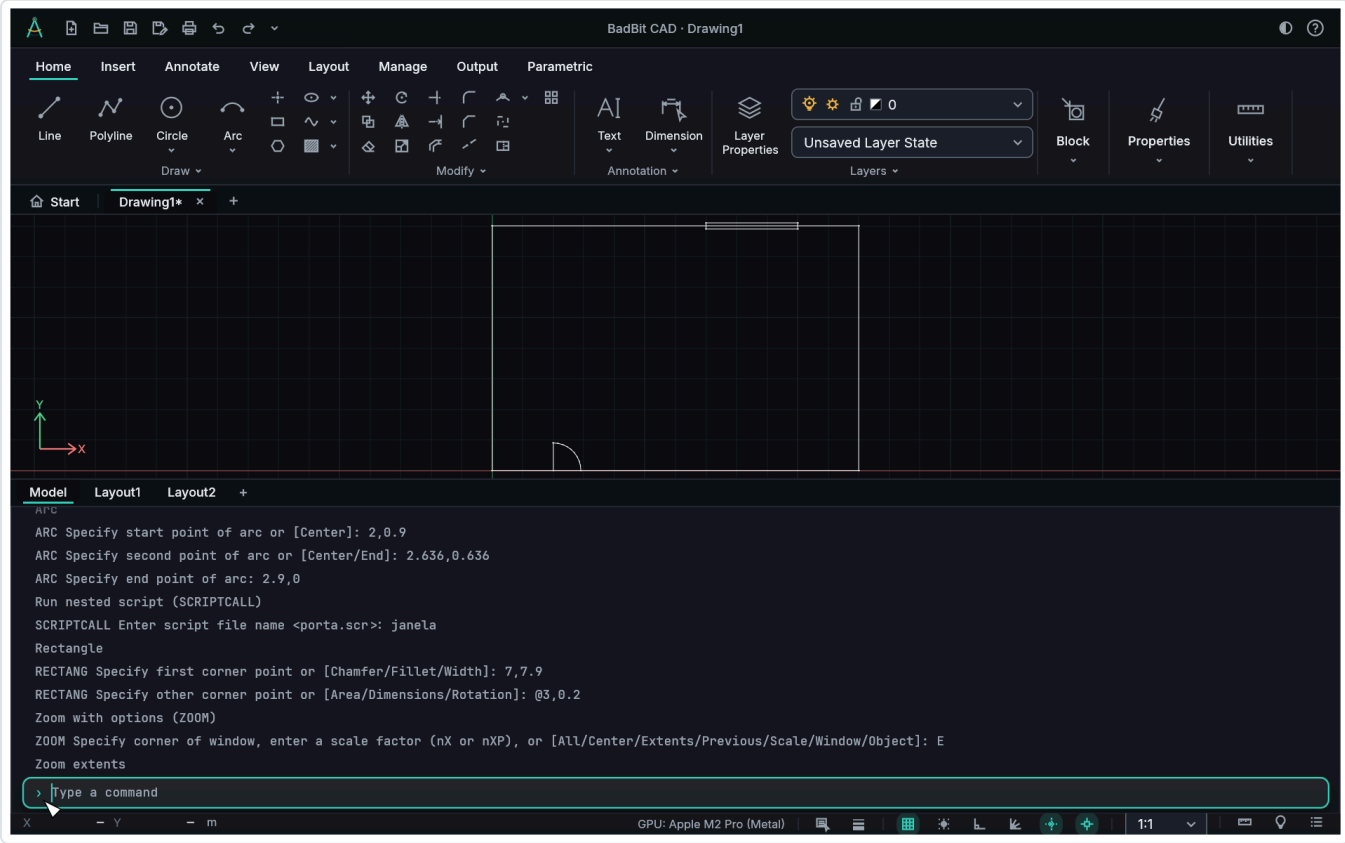
Nested scripts (SCRIPTCALL). SCRIPTCALL runs a script like SCRIPT does (the same dialog with FILEDIA at 1, the same prompt with FILEDIA at 0 or inside a script, `.scr` when the extension is missing, UTF-8 or ANSI), with one difference: inside a script, it **calls** the other one and, when the called script ends, the calling script continues on the next line. The called script can call others, on several levels. SCRIPT inside a script, on the other hand, replaces everything that is running with the new script, as in AutoCAD. A name without a folder is looked for first in the folder of the calling script, then in the working folder.

```
; planta.scr
RECTANG 0,0 @12,8
SCRIPTCALL porta
SCRIPTCALL janela
ZOOM E
```

Only the top script moves forward; the ones that called it wait for it. Therefore:

- an error at any level (unknown command, rejected answer, file that does not exist) stops everything, and the command line tells the script and the line of the error;
- `Esc` or `Backspace` interrupt the called script and, with it, the ones waiting;
- **RESUME** continues from the point where it stopped and, at the end of the called script, goes back to the ones that called it;
- **RSCRIPT** inside the called script repeats the called script (the caller keeps waiting, until Esc).

So that it does not grow forever, there is a limit of 16 nested scripts: a script that calls itself stops at the 16th level, with the message "Limit of 16 nested scripts reached" (RESUME continues from what comes after the SCRIPTCALL).

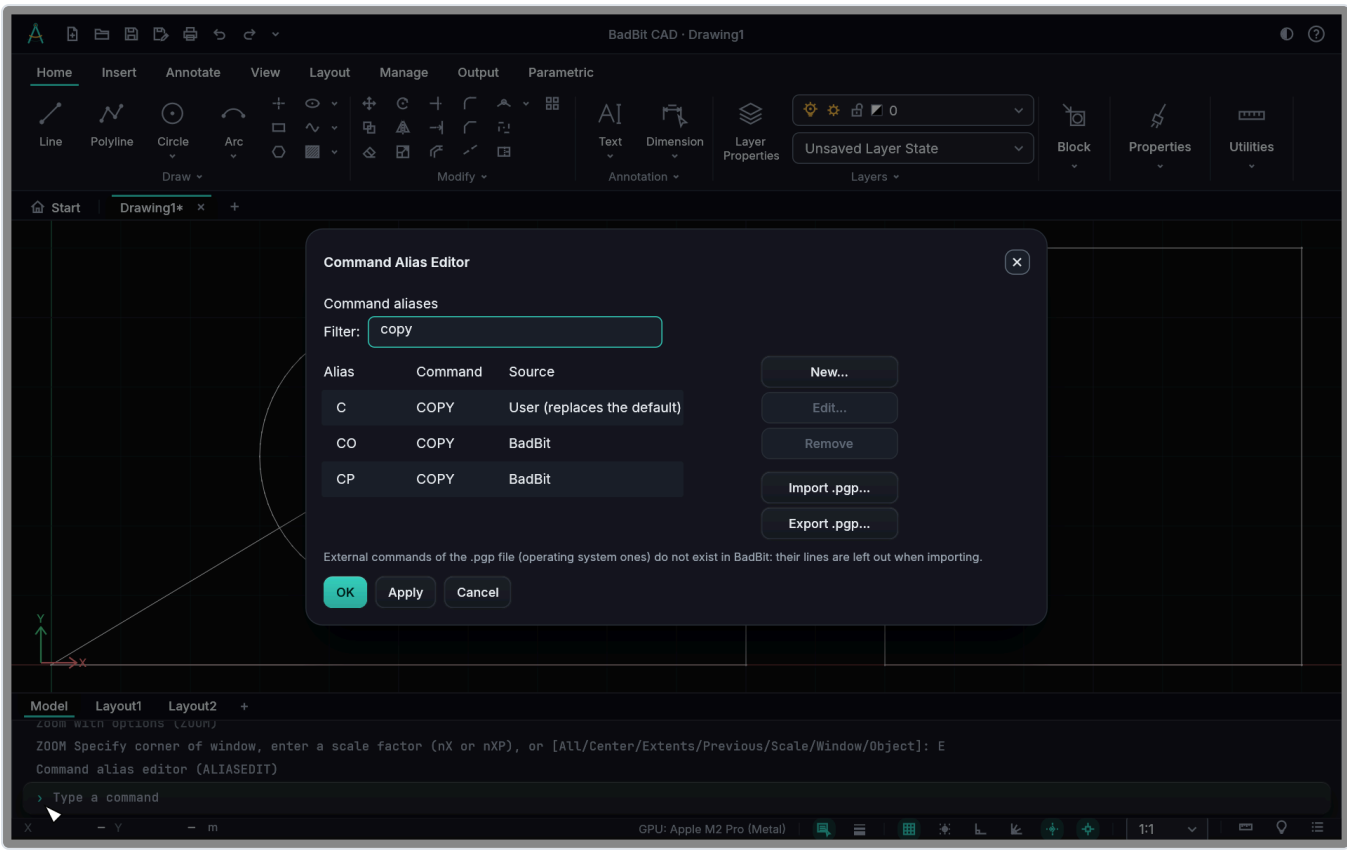


planta.scr drew the walls, called porta.scr and janela.scr, and continued with ZOOM; the history (opened with F2) shows the call to janela.scr, with the name typed next to the prompt.

For programmers: the script hands each piece to the same command line feed that LISP uses in `(command ...)`.

Command aliases (ALIASEDIT)

ALIASEDIT opens the command alias editor (the "Command Alias Editor" of AutoCAD's Express Tools): the list with the **alias**, the **command** (by its English name, as in `acad.pgp`), and the **source** (*BadBit*, the factory ones; *User*; *User (replaces the default)*). The **Filter** field shows only the ones that have the text in the alias or the command.



ALIASEDIT filtered by "copy": the user's C replacing CIRCLE's C, next to the factory CO and CP.

- **New...** creates an alias: the abbreviation (letters, numbers, and `- _ . +`) and the command (the name in any language or one of its aliases). A command name cannot be used as an alias.
- **Edit...** changes the command of the selected alias. For a factory alias, the user's takes its place (as in `acad.pgp`, the last definition wins: `C` can become COPY).
- **Remove** removes a user alias (factory ones cannot be removed; when edited, they come back when the user's is removed).
- **Import .pgp...** reads an `acad.pgp` (AutoCAD's or one exported from here): aliases that are already the same as the factory ones do not need to come in; lines that are not usable (external commands, from the operating system, which BadBit CAD does not have; command names; commands that do not exist) are left out, with the count.
- **Export .pgp...** writes a file in the `acad.pgp` text format: the factory ones and, at the end, the user's.
- **Apply** and **OK** make the aliases work immediately (AutoCAD's REINIT is not needed) and save them in the user profile (see *Options and profiles*); **Cancel** and the `×` discard what has not been applied.

LISP (APPLOAD, alias AP)

BadBit CAD has its own **LISP** (`.lsp`), compatible with AutoLISP, AutoCAD's automation language: the office routines that count blocks, number texts, swap layers, or draw with commands work without changes. The interpreter is BadBit CAD's own (ADR 0029) and runs in the background: the window keeps responding while the routine works, and `Esc` interrupts it (the routine receives the error `Function cancelled`, as in AutoCAD).

On the command line. What you type starting with `(` is an expression: the space is a character until the parentheses close, and `Enter` shows the value in the history (`(* 2 1.5)` gives `3.0`). An expression that does not close continues on the next line (the prompt becomes `(_>`). `!name` shows the value of a variable. At a command prompt, the expression is the answer: at CIRCLE's "radius", `(* 2 2.5)` gives a radius of 5.

New commands. `(defun c:NAME () ...)` becomes the command `NAME`, typed like any other (it appears in the command line suggestions, as "LISP command NAME"). Everything the command does is a single undo step. A BadBit CAD command with the same name still works.

Example, a routine that counts the blocks in the drawing:

```
; contablocos.lsp
(defun c:contablocos (/ ss i nome par res)
  (setq ss (ssget "X" '((0 . "INSERT"))) i 0)
  (repeat (sslength ss)
    (setq nome (cdr (assoc 2 (entget (ssname ss i)))) i (1+ i))
    (if (setq par (assoc nome res))
      (setq res (subst (cons nome (1+ (cdr par))) par res))
      (setq res (cons (cons nome 1) res))))
  (foreach par res
    (princ (strcat "\n" (car par) ": " (itoa (cdr par)))))
  (princ))
```

Loading. APPLOAD (`AP`) opens the **Load/Unload Applications** window: the file (typed or through **Browse...**), **Load**, the list of **Loaded applications** in this session, and the **Startup Suite**, the files loaded every time BadBit CAD opens (**Add...**, **Remove**, or **Add to Startup Suite** for one of the loaded ones). The Startup Suite is stored in the user profile (see *Options and profiles*). As in AutoCAD, an `.lsp` cannot be unloaded (the functions stay defined): the **Unload** button is grayed out. Inside a routine, `(load "file")` also loads (without the extension, `.lsp` is assumed), searching in the drawing's folder, in BadBit CAD's `lisp` folder, and in the trusted folders. The file can be UTF-8 or ANSI (Windows-1252), like old `.lsp` files.

ObjectARX (`.arx`, `.crx`), .NET (`.dll`), VBA (`.dvb`), and compiled LISP (`.fas`, `.vlx`), which AutoCAD's APPLOAD loads, are not supported by BadBit CAD: the command line says so.

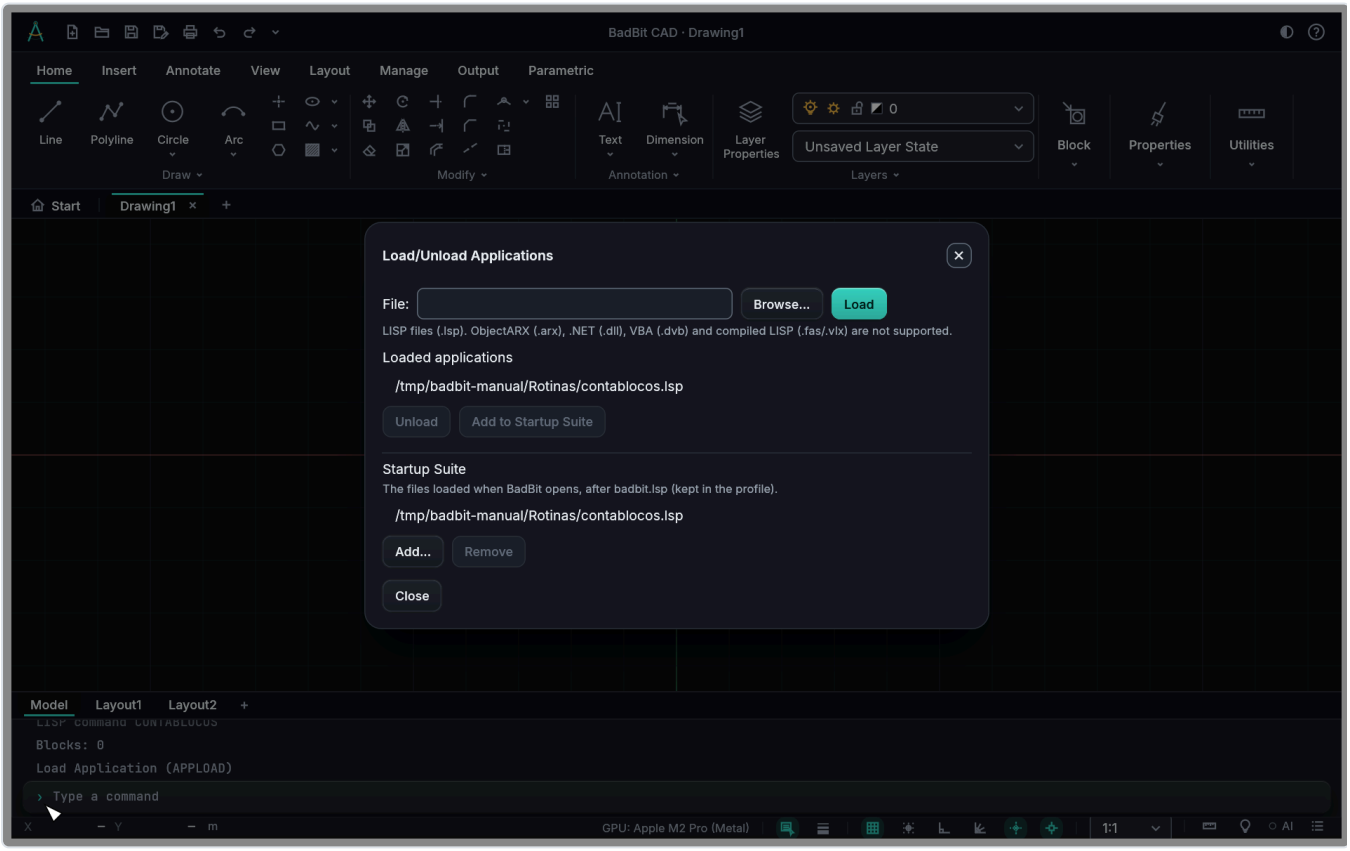
At startup. BadBit CAD loads, in this order: **`badbit.lsp`** (once, when the program opens; the equivalent of `acad.lsp`), **`badbitdoc.lsp`** (for each drawing opened or created; the equivalent of `acaddoc.lsp`), and the Startup Suite. Both files are searched for in BadBit CAD's `lisp` folder (on macOS, `~/Library/Application Support/BadBit/lisp`; on Windows, `%APPDATA%\BadBit\lisp`; on Linux, `~/.config/badbit/lisp`) and in the trusted folders. Routines belong to a single session: the ones defined in one drawing work in the others (AutoCAD has one namespace per drawing; that is why `badbitdoc.lsp` runs again for each drawing).

Security. A routine can change or erase the whole drawing. Like AutoCAD's **SECURELOAD**, BadBit CAD trusts the `lisp` folder and the **trusted folders** (**TRUSTEDPATHS**, separated by `;`). With SECURELOAD 1 (the default), loading an `.lsp` from another folder opens a confirmation: **Load once**

(in this session), **Always load** (the folder is added to the trusted ones), or **Do not load** (the `load` fails). With SECURELOAD 2, only from the trusted folders; with 0, without asking. Both variables are stored in the profile. Routines do not run system programs (`startapp` and external commands do not exist) and do not read environment variables.

What phase 1 includes. The language (lists, 32-bit integers and reals, strings, `defun` with local variables, `if` / `cond` / `while` / `repeat` / `foreach`, `lambda` / `mapcar` / `apply`, arithmetic, trigonometry, the `rtos` / `distof` / `angtos` / `angtof` conversions, the `vl-string-*` functions and the basic `vl-*` list functions, `wcmatch`), error handling (`*error*` called at the point of the error, with the routine's local variables still valid, and `vl-catch-all-apply`), `getvar` / `setvar` tied to BadBit CAD's variables, `(command ...)` and `(command-s ...)` (with `pause` for the user to answer), the `getpoint`, `getcorner`, `getdist`, `getangle`, `getorient`, `getreal`, `getint`, `getstring`, and `getkword` prompts with `initget` (they ask on the command line and accept clicks), `entsel`, `ssget` (the "Select objects" prompt, and the `X`, `W`, `C`, `WP`, `CP`, `I`, `P`, and `L` modes, with the filter list and wildcards), selection sets, `entget` / `entmod` / `entmake` / `entdel` / `entlast` / `entnext` / `handent` (LINE, CIRCLE, ARC, LWPOLYLINE, POINT, TEXT, MTEXT, and INSERT with the geometry; the others only with the layer, color, and linetype), and `tblsearch` / `tblnext` for the LAYER, LTYPE, STYLE, and BLOCK tables (`entmod` on a layer turns it on, off, freezes, and locks it).

Left for later: the routines' DCL dialogs, Visual LISP with ActiveX (`vla-*`, `vlax-*`) and reactors (`vlr-*`), text files (`open`, `read-line`, `write-line`), extended data, and dictionaries. LISP error messages stay in English (`bad argument type`, `divide by zero`), as in AutoCAD, because routines compare that text in `*error*`; the *handle* (group 5) belongs to the session.



The Load/Unload Applications window with one routine loaded and the Startup Suite; in the history, the typed expression and the result of a LISP command.

BadBit Scripts: the LISP and Python editor (SCRIPTEDITOR, alias SED)

BadBit Scripts is BadBit CAD's simple editor for **LISP** routines (`.lsp`) and **Python** scripts (`.py`): open, edit, save, load and run without installing anything. Type `SCRIPTEDITOR` (or `SED`) and `Enter`; the name is the same in the three languages. If you come from AutoCAD and type `VLIDE` or `VLISP`, a hint shows the right name (nothing opens under those names). It is not meant to be an IDE: the full editor will be the BadBit extension for VS Code.

Tabs. New LISP and **New Python** open an empty tab; **Open** opens a `.lsp` or `.py`; **Save** and **Save as** write it (a tab with unsaved changes has a dot after the name and, when closing it, the editor asks). The code has line numbers and simple colors for the language words, strings, numbers and comments.

Load and Run. In a `.lsp`, **Load** (or **Run**) sends the file to the LISP session, expression by expression: its functions and `c:` commands become available. In a `.py`, **Run** runs the script in a separate process (a stuck or failing script never freezes BadBit CAD) and **Stop** (or `Esc`) ends it; **Load** runs the file in the Python console, to try its functions. The first time you run Python, BadBit CAD downloads its Python (25 to 37 MB, depending on the system), with a progress bar and a **Cancel** button; without internet, the message says so.

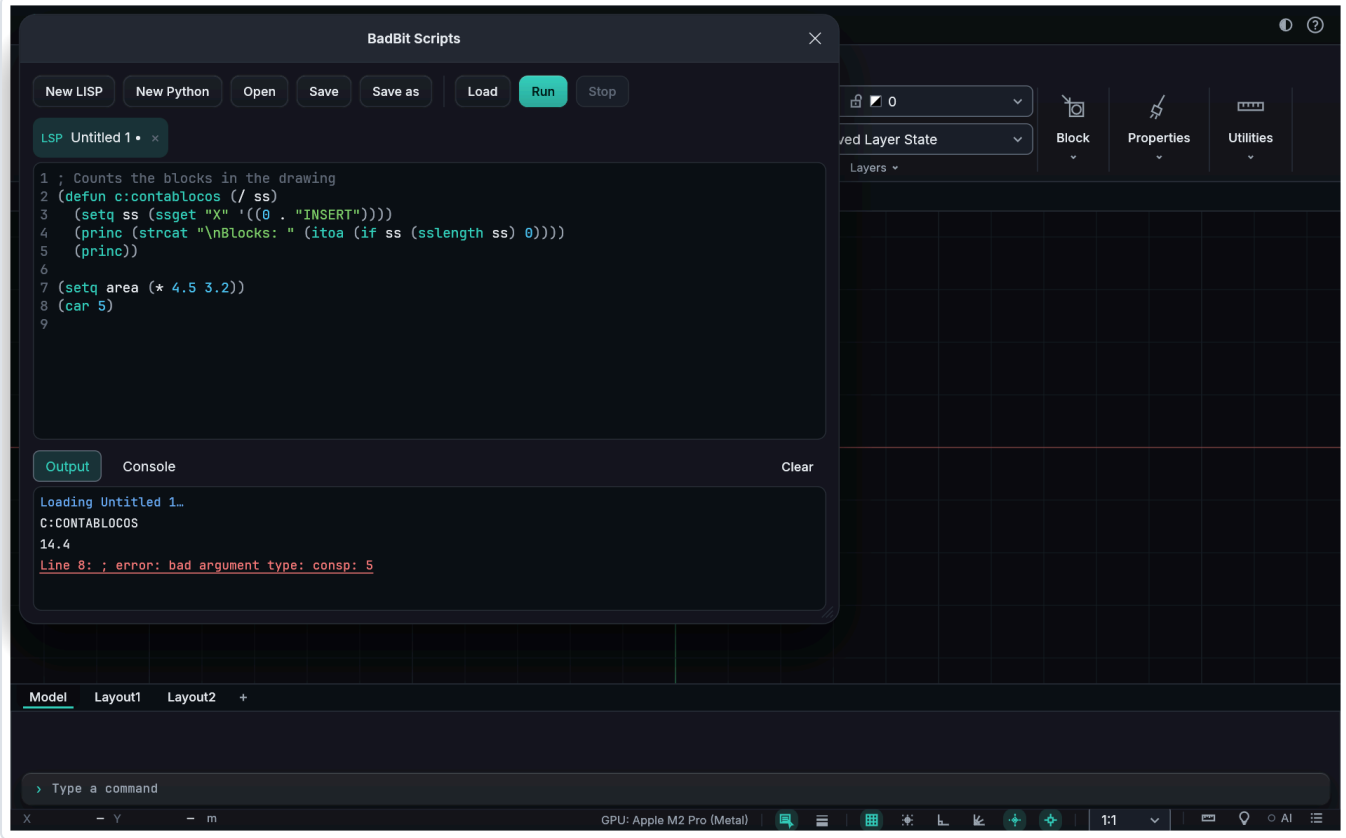
Output. The **Output** panel shows what the program prints and the errors, with the **line**: in a `.lsp`, the line where the expression starts ("Line 9: ; error: bad argument type"); in a `.py`, the lines of the Python error. Click the underlined line to go to it in the code.

Console. In the **Console** panel, choose **LISP** or **Python** and type one line at a time, with `Enter`; the value appears right below. In Python, the `badbit` package is already imported.

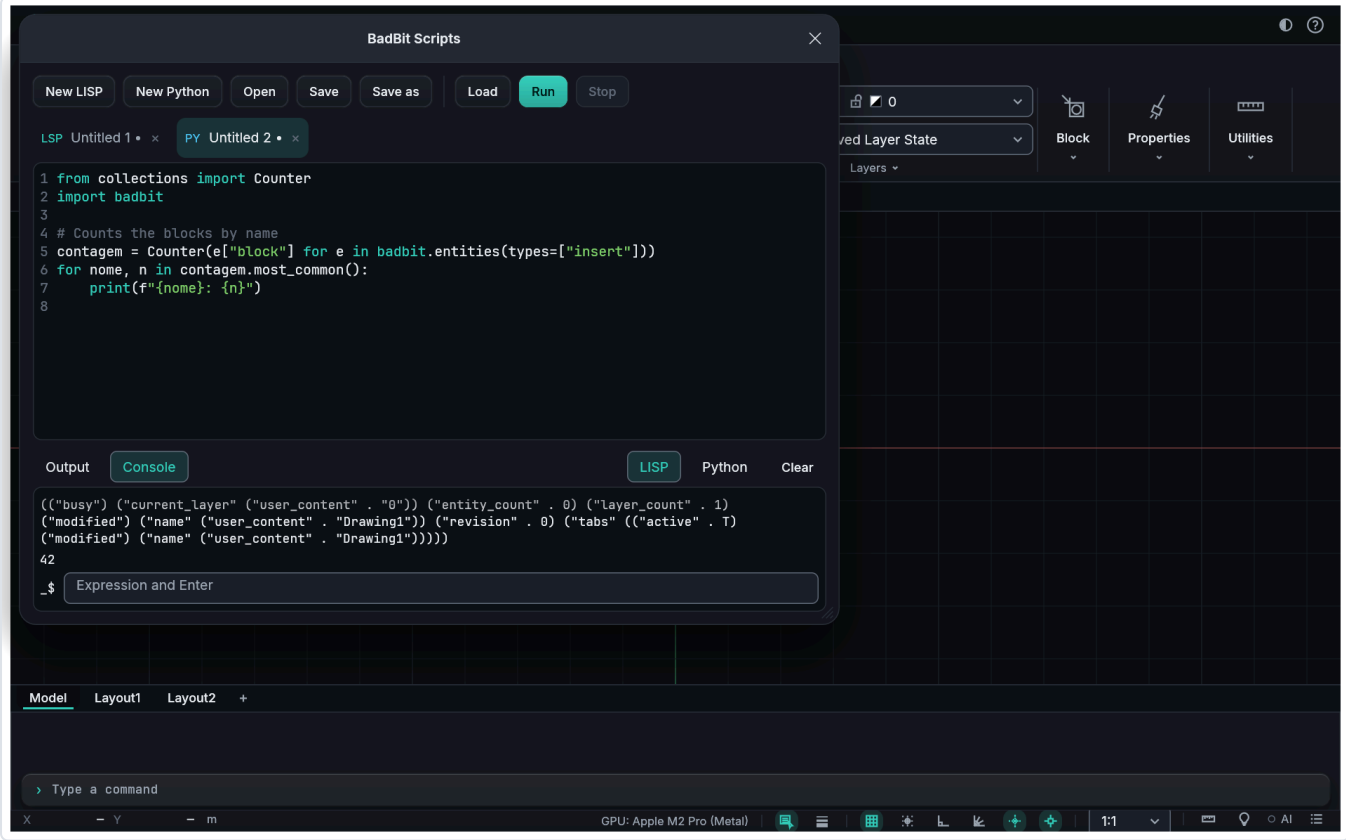
Packages (pip). In the Python console, a line that starts with `pip` (or `%pip`, as in IPython) runs **pip** in a package environment of your own, managed by BadBit CAD, and does not go to Python: `pip install numpy`, `pip uninstall numpy` (no prompt), `pip list`, `pip show`, `pip freeze` and `pip --version`; other subcommands are refused. The first time, the environment is created (a few seconds, no internet needed); after that, scripts and the console import what you installed. pip runs on its own, with its output in the console, and **Stop** (or `Esc`) ends it. Packages come from PyPI (or from the index you give, such as `--index-url` or `--no-index --find-links <folder>`) and are kept when you move to a new version of BadBit CAD. Installing a package runs third-party code with your account's permissions; no BadBit operation runs pip.

BadBit operations. From LISP, `(badbit-call "operation" '(("parameter" . value)))` calls any BadBit operation (the same ones AI and Python use), and each call is one undo step. In Python, each operation is a function of the `badbit` package (`badbit.cad.layer.create(name="Walls")`), with questions to the person (`badbit.getpoint`, `badbit.ssget`) on the command line. Scripts always work, with nothing to turn on: they are your own code, and the **AI (MCP)** switch in Options is only for AI assistants. The Python and LISP guides and the operations reference are in the BadBit AI and scripts documentation.

Security. The trusted folders and SECURELOAD apply to `.py` files too: a file outside the trusted folders asks for confirmation (Load once, Always load from this folder, Do not load) before loading or running.



BadBit Scripts with a loaded routine: in the Output, the value of an expression and the error with the line where it starts.



A Python tab and the LISP Console, with a badbit-call and a calculation.

The command line in the terminal (console)

BadBit CAD also has its command line in a **terminal**, without opening the window: call the program with `console` (on macOS, `"/Applications/BadBit CAD.app/Contents/MacOS/badbit-shell" console`; on Windows and Linux, the installed executable with `console`). You type commands and the answers to the prompts as here, LISP expressions (`(+ 1 2)`, `!variable`) and, on the lines that start with `:`, the BadBit operations (`:op cad.layer.list`), their list (`:ops`), the description of one (`:desc`), the session history (`:history`), pip (`:pip install <package>`), help (`:help`) and quitting (`:quit`). `Esc` and `Enter`, or `Ctrl`+`C`, cancel the command and interrupt LISP; `Ctrl`+`D` quits (anything not saved is lost).

The console opens and saves drawings in any folder, as here: a name without a folder goes to the terminal's current folder, and network and Windows device paths are refused. It needs nothing turned on, because you are the one typing. File dialogs become the file name prompt (`SAVEAS` asks for the file name). `--idioma pt-BR`, `en` or `es` chooses the language of the prompts. It also reads its input from a file (`console < script.txt`), to run a sequence of commands without the window. The console guide, in the AI and scripts documentation, has all the details.

Batch from Python, without opening the window. The `badbit` Python package (`pip install badbit`, in the Python on your machine) also processes drawings without opening BadBit CAD: `badbit.headless()` starts the program without a window, the script opens, changes and saves `.bbd`, DXF and DWG files, and at the end the program closes by itself; `badbit.each_drawing(folder)` goes through a whole folder, one drawing at a time. The same script runs from the terminal with `python -m badbit run script.py --headless`. It is your own script: it needs nothing turned on and nothing to accept, opens and saves files where you ask (with absolute paths) and keeps the backup copy before writing over a file. The Python guide, in the AI and scripts documentation, has the examples.

Drawing commands

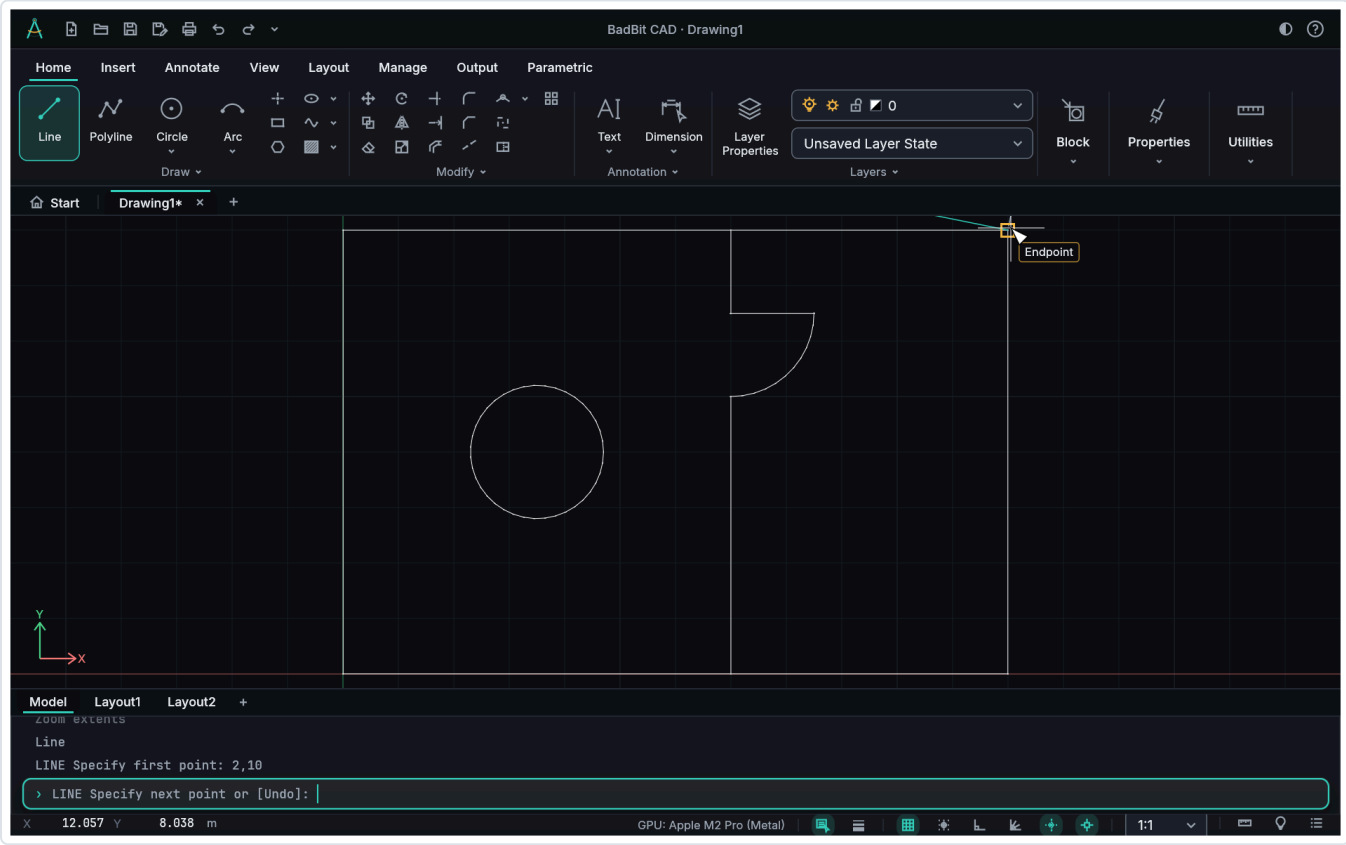
LINE (alias L)

Draws straight lines, one per segment. Each click creates the segment immediately, and the snaps already see the previous segments: you can close a rectangle by clicking the start point or follow the extension of a segment.

1. Type `L` and `Enter` (or click Line on the ribbon).
2. "Specify first point:" — click or type.
3. "Specify next point or [Undo]:" — keep clicking or typing.
4. `Enter` finishes.

Options: **Close** (`C`, from three points on) connects the last point to the first and finishes; **Undo** (`U`, or `Ctrl` + `Z` with the command active) removes the last segment and the command continues from the previous point. `Esc` finishes and keeps what was drawn. After the command, a single **U** (or `Ctrl` + `Z`) undoes the whole LINE.

Continue: `Enter` in place of the first point starts at the end of the last line or arc drawn, as in AutoCAD. After a line, the command already asks for the next point. After an arc, it asks "Specify length of line:" and the line comes out tangent to the arc; from then on, it goes on like a regular LINE. With no previous line or arc, the command line says so.



LINE in progress: the rubber-band line to the cursor and the endpoint snap.

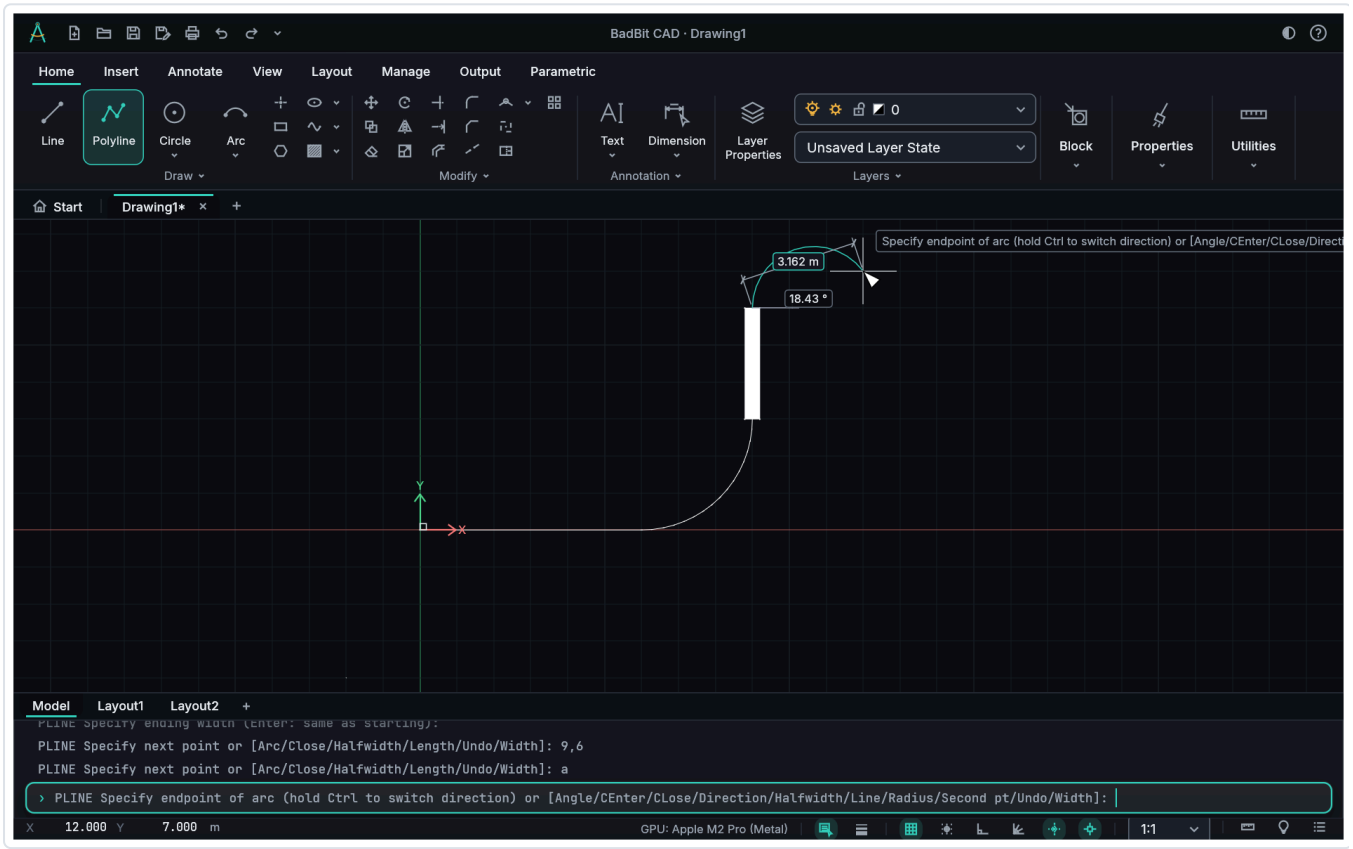
PLINE (alias PL)

A polyline: straight and arc segments that form a single object, with width, like AutoCAD's PLINE. It is created at the second point and grows with each click (the snaps already see the segments), and the whole command is a single undo step.

The option letters are AutoCAD's English ones, in any language (in the prompt, the capital letters of each option are the ones you type):

- **Arc** (`A`): switches to arc mode. The arc comes out tangent to the previous segment up to the clicked point (`Ctrl` switches the direction where the prompt says so). In arc mode: **Angle** (`A`, the included angle and then the end, the center, or the radius), **CEnter** (`CE`, and then the end, the angle, or the chord length), **CLose** (`CL`, closes with an arc), **Direction** (`D`, the tangent at the start), **Line** (`L`, goes back to straight segments), **Radius** (`R`, and then the end or the angle), and **Second pt** (`S`, the arc through three points).
- **Close** (`C`): connects the last point to the first and finishes.
- **Width** (`W`) and **Halfwidth** (`H`): the starting and ending width of the next segment. The ending width applies to the following segments, as in AutoCAD. A wide polyline appears as its outline (the two edges), like AutoCAD with fill turned off; the fill will come with solid hatching.
- **Length** (`L`): a straight segment of the typed length, in the direction of the previous segment (the tangent, if it is an arc).
- **Undo** (`U`): removes the last segment.

In the properties palette (`Ctrl`+`1`), the polyline shows the chosen vertex (X and Y), the **starting width** and **ending width** of the segment leaving it, the **global width** (the same on all segments; empty if they vary), whether it is closed, the length, and the area (with the arcs).



PLINE: a straight segment, a tangent arc, and a straight segment with width.

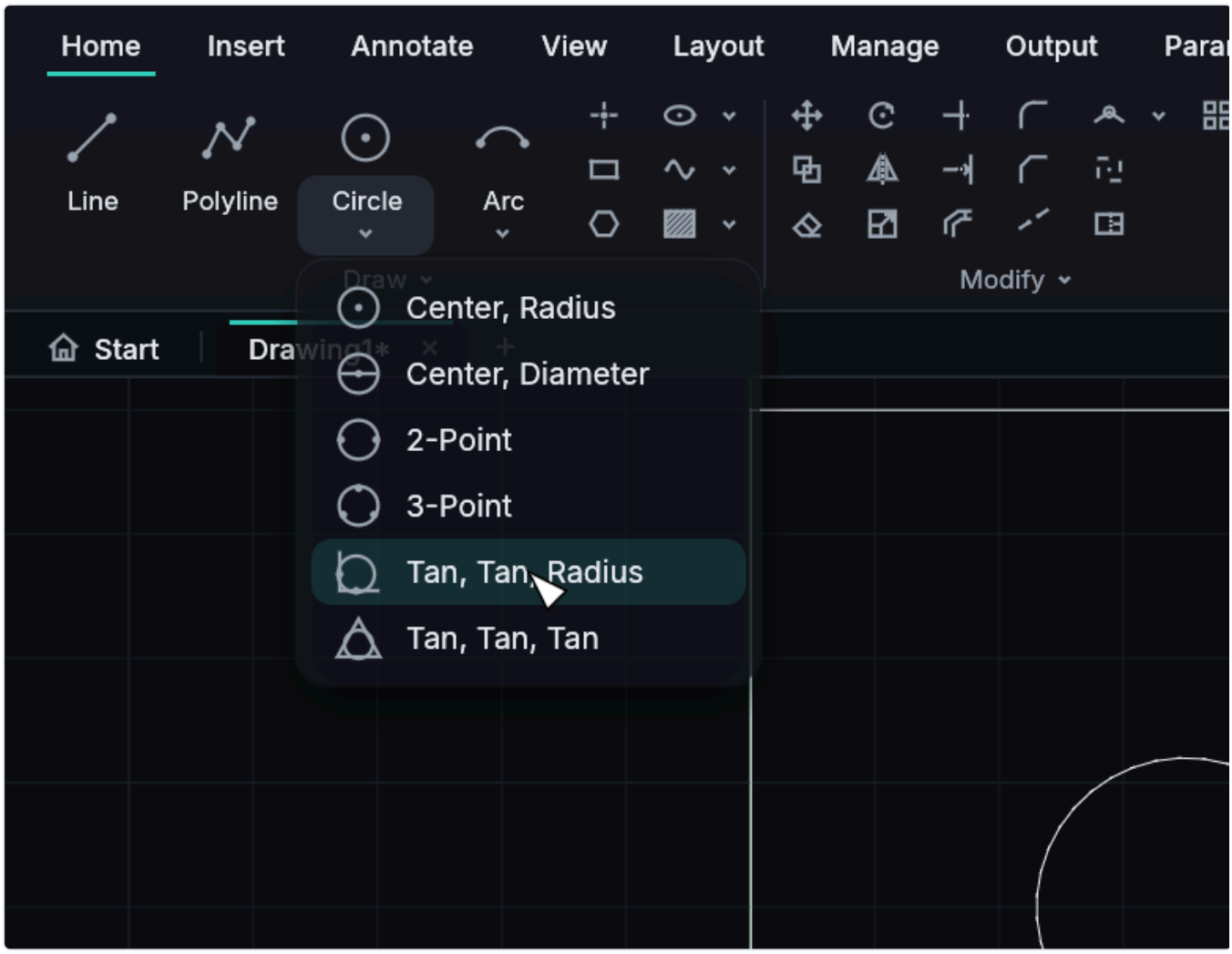
CIRCLE (alias C)

As in AutoCAD, the default is by **center and radius**: "Specify center point for circle or [3P/2P/Ttr (tan tan radius)]:", then "Specify radius of circle or [Diameter]:" (click or type the radius). The option letters are AutoCAD's English ones.

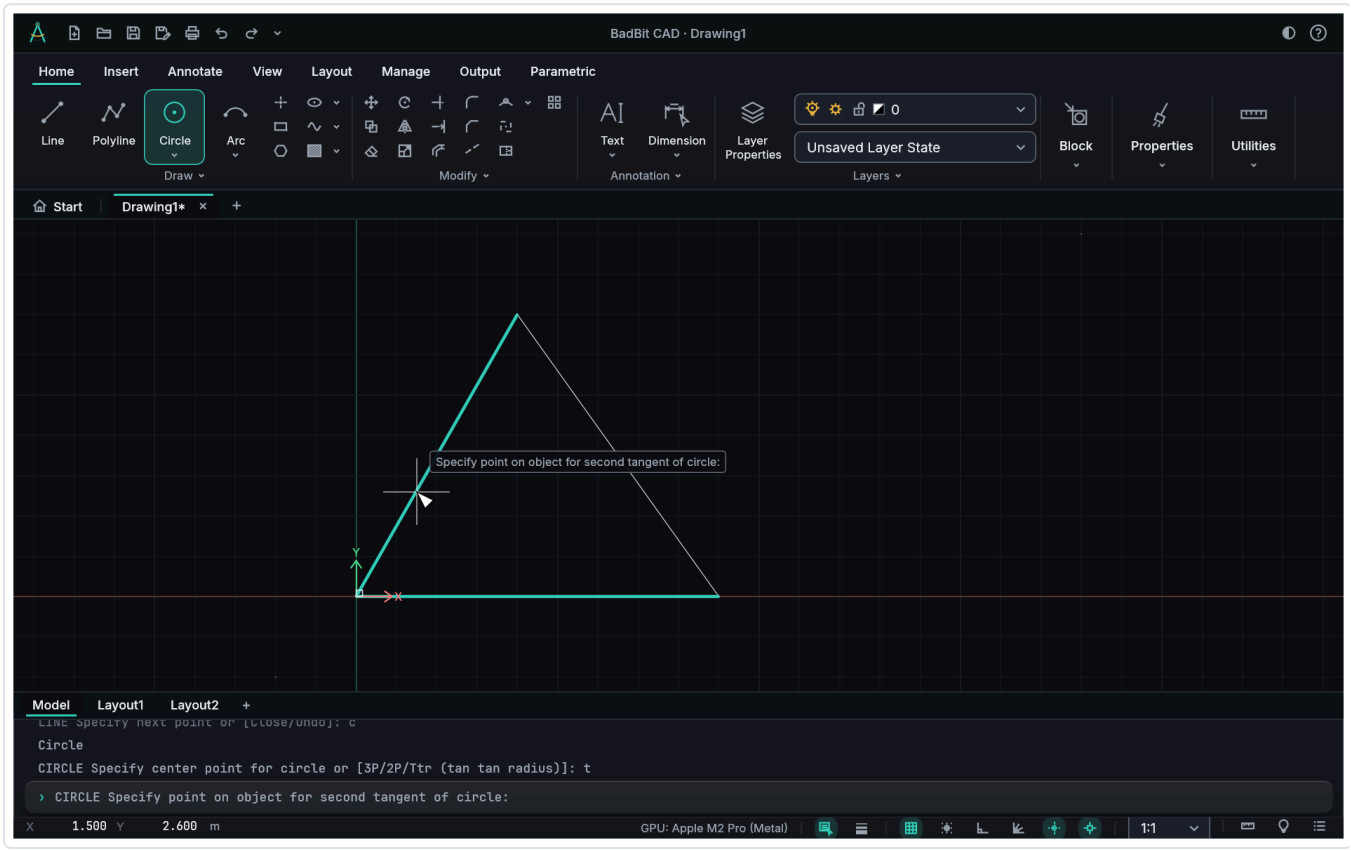
Form	Answers
Center, radius (default)	center, radius
Center, diameter	center, `D`, diameter
3 points	`3P`, three points on the circle
2 points	`2P`, the two ends of a diameter
Tan, tan, radius	`T` (or `TTR`), a point on each of the two objects, radius
Tan, tan, tan	only from the button's list: a point on each of the three objects

The **last radius** is remembered and appears between `< >` in the prompt: Enter accepts it (for the diameter, the default is twice that). For tangency, the objects can be lines, circles, arcs, construction lines, rays, or a polyline segment; among the possible circles, the one that touches closest to the clicked points is kept, as in AutoCAD. When there is no circle with that radius touching both objects, the command line says so and asks for the radius again. While it asks for each tangent, the object under the cursor is highlighted (thick, in the accent color), and the ones already chosen stay highlighted until the circle is created: you can see what was picked without reading the command line.

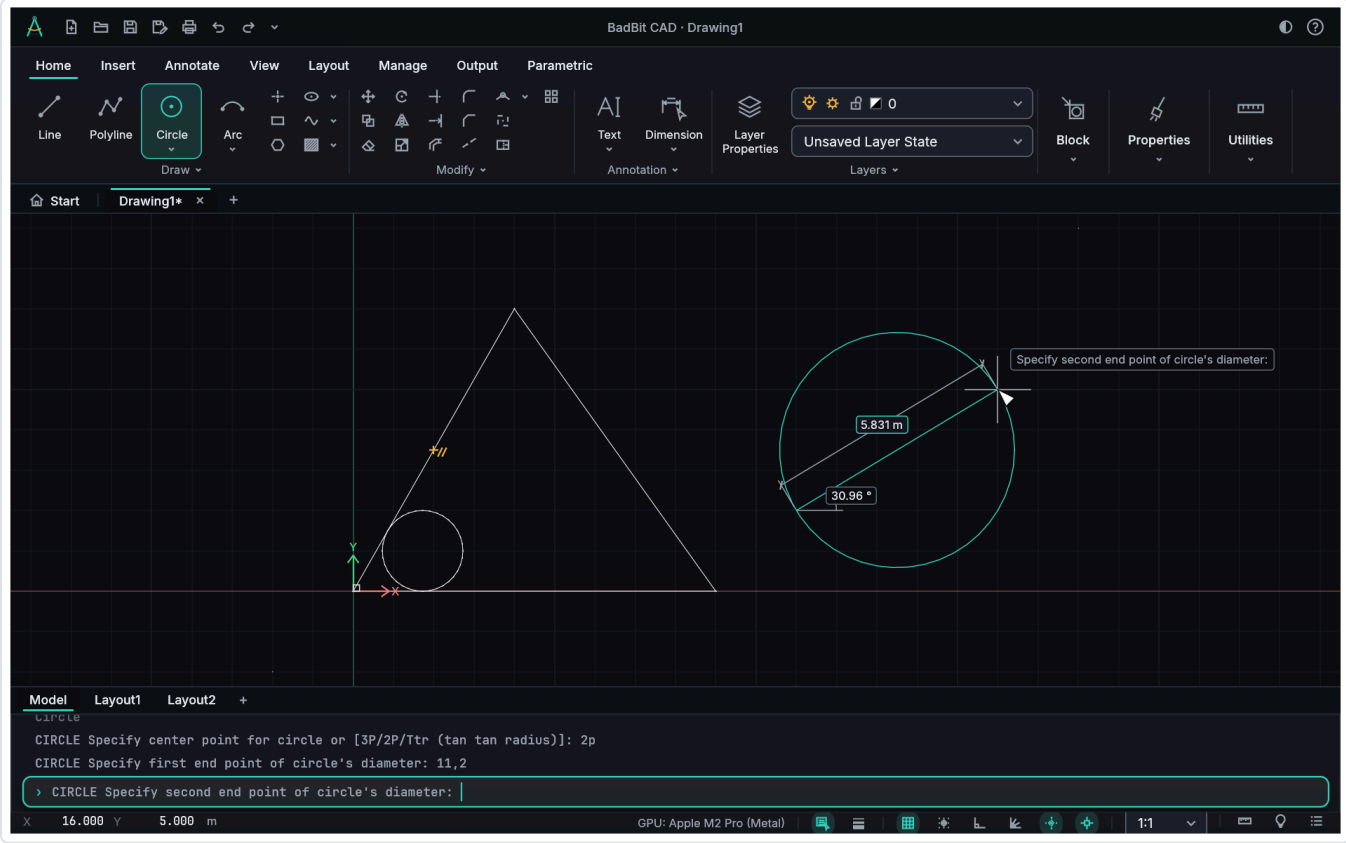
On the ribbon, the **Circle** button is split like the Arc button: the top part repeats the last form chosen, and the list has the six forms, each with its icon. When typed, CIRCLE always starts with the center, with the options in the prompt.



The circle forms on the ribbon.



Ttr: the bottom side, already chosen, stays highlighted, and the left side lights up under the cursor.



A circle of radius 1 tangent to two sides of the triangle (Ttr) and 2P in the preview, through the ends of the diameter.

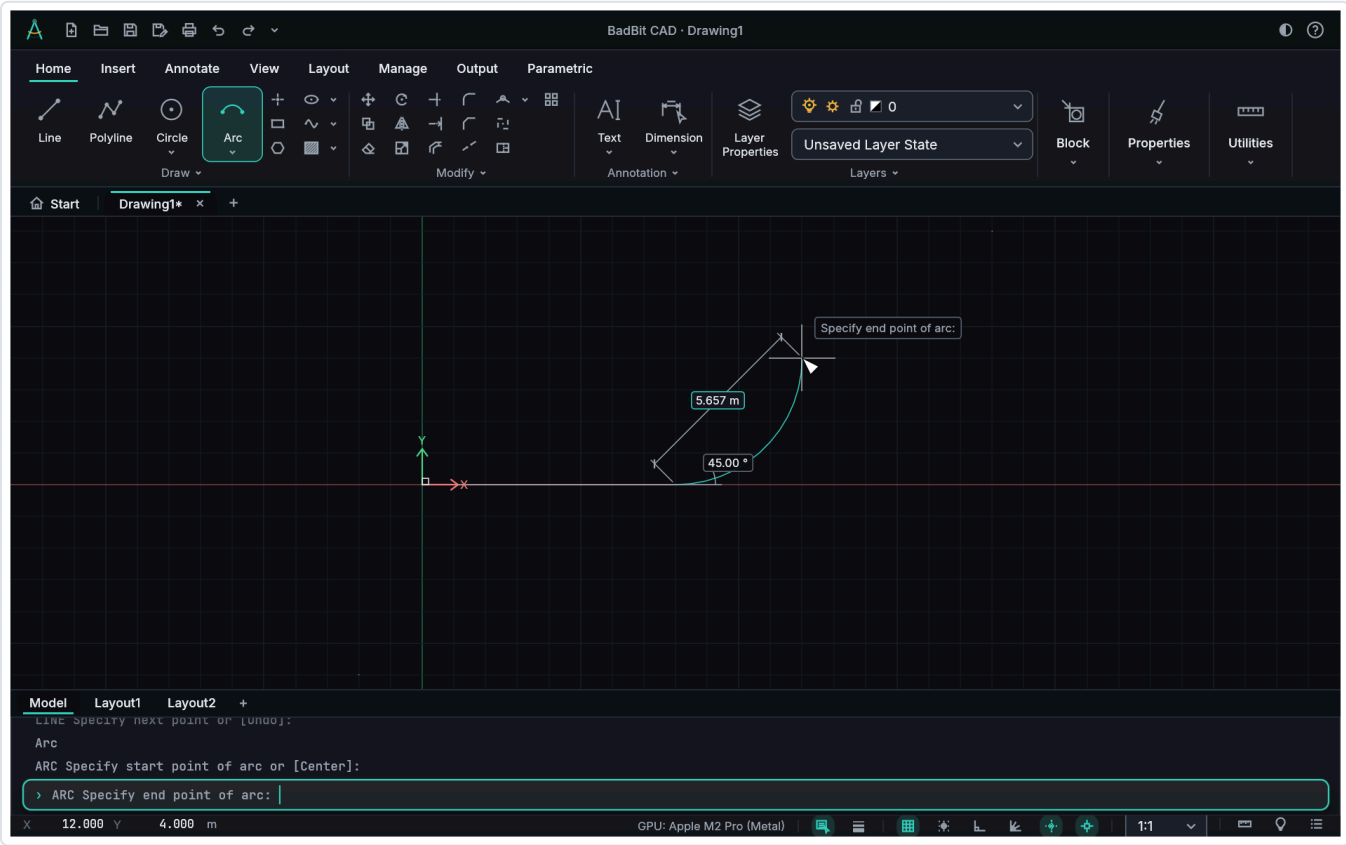
ARC (alias A)

As in AutoCAD, the default is by **three points**: the start, a point on the arc, and the end. Two points alone do not define an arc (infinitely many pass through them); the command options change the third piece of data:

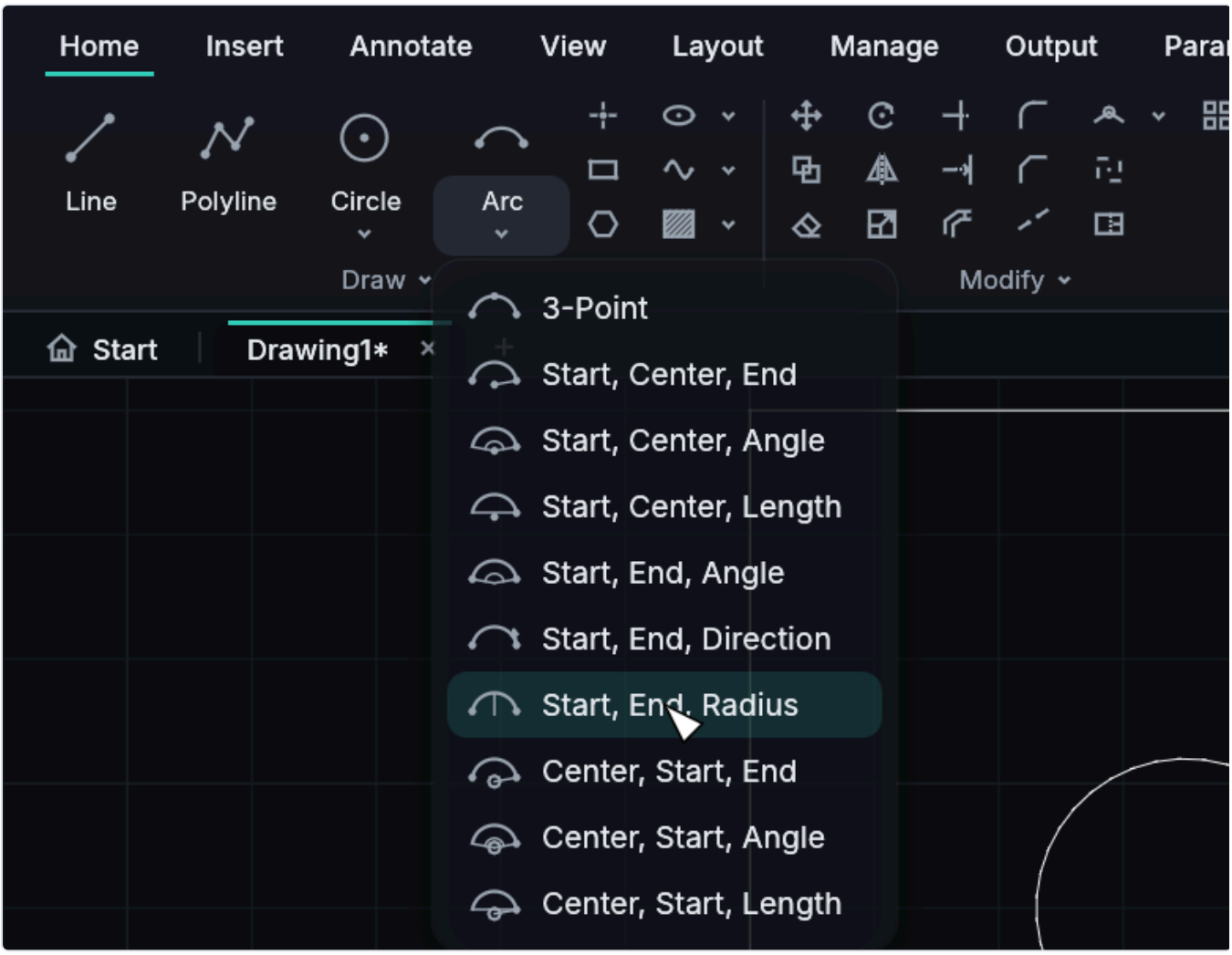
Form	Answers
Three points (default)	start, second point, end
Start, center, end	start, `C`, center, end
Start, center, angle	start, `C`, center, `A`, included angle
Start, center, chord	start, `C`, center, `L`, chord length
Start, end, center	start, `E`, end, center
Start, end, angle	start, `E`, end, `A`, included angle
Start, end, direction	start, `E`, end, `D`, tangent direction at the start
Start, end, radius	start, `E`, end, `R`, radius
Center, start, end	`C`, center, start, end (or `A` / `L` as above)

On the ribbon, the **Arc** button is split, as in AutoCAD: the top part (the icon) repeats the last form chosen in the list, and the button's icon becomes that form's (when the program opens, it is the three-point one); the bottom part (the name, with the arrow) opens the list of forms, each with an icon that marks what it asks for: solid points for what you click, the center as a ring when it comes first, and the angle, the chord, the tangent direction, or the radius drawn. Once a form is chosen, the command answers its options by itself and only asks for the data (for example, in *Start, End, Radius*: the start, the end, and the radius). When typed on the command line, ARC always starts with the three points, with the options in the prompt.

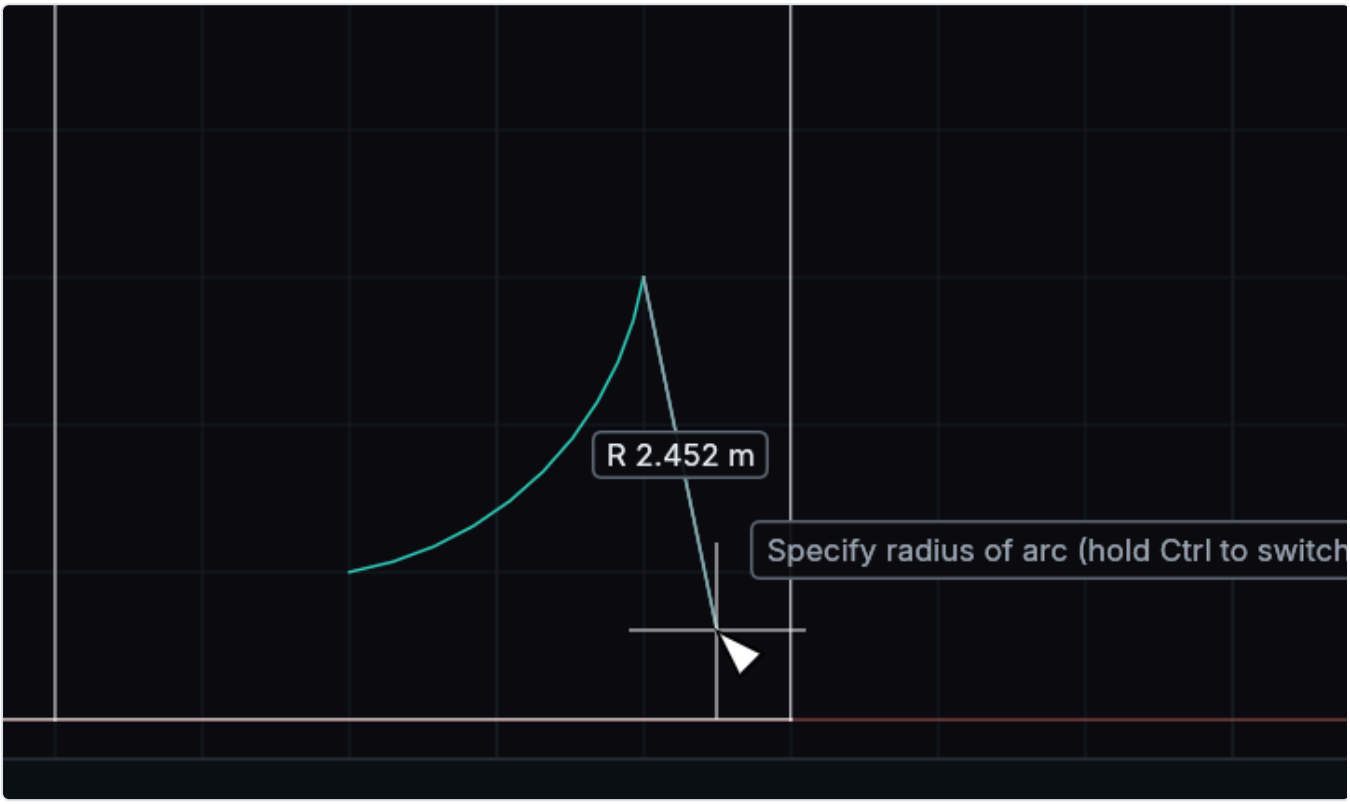
Continue: `Enter` in place of the start point makes the arc come out tangent from the end of the last line or arc; the command only asks for the end point.



Enter at the start: the arc comes out tangent from the end of the line, and the preview follows the cursor.



The list of arc forms on the ribbon.



Start, End, and Radius: the preview follows the cursor (the radius is the distance from the end to the cursor, shown by dynamic input) until the value is typed.

Arcs by center, angle, chord, and radius run **counterclockwise**, as in AutoCAD: with the start on the left and the end on the right, the arc goes below. For the other side, **hold Ctrl** (or Cmd, on the Mac) while dragging and clicking: the arc switches direction (with radius and angle, it goes to the other side of the chord; by center, it becomes the other piece of the circle). A negative angle (clockwise) and a negative chord or radius (the larger arc) also work. Angles are in degrees, and the value can be typed or pointed with the cursor. If the data do not form an arc (a radius smaller than half the chord, for example), the command line says so and asks again.

RECTANG (alias REC)

A rectangle as a closed polyline: the first corner and the opposite corner.

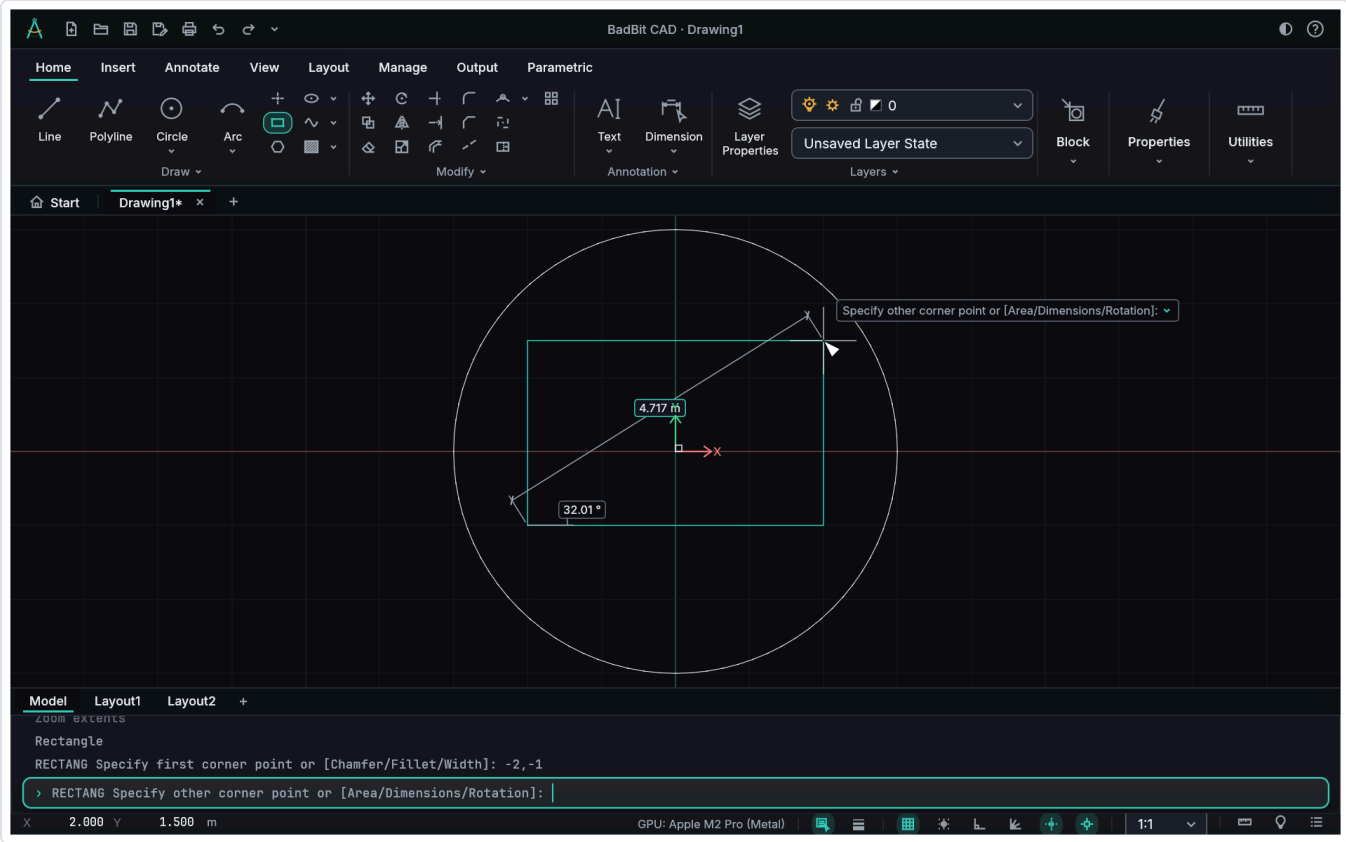
Before the first corner:

- **Chamfer** (`C`): the two chamfer distances at each corner;
- **Fillet** (`F`): the radius of the rounded corners (chamfer and fillet exclude each other: choosing one zeroes the other);
- **Width** (`W`): the line width of the polyline.

All three are remembered for the next rectangles (set back to 0 for sharp corners). The rectangle remains a single polyline, with the corners in it.

Before the opposite corner:

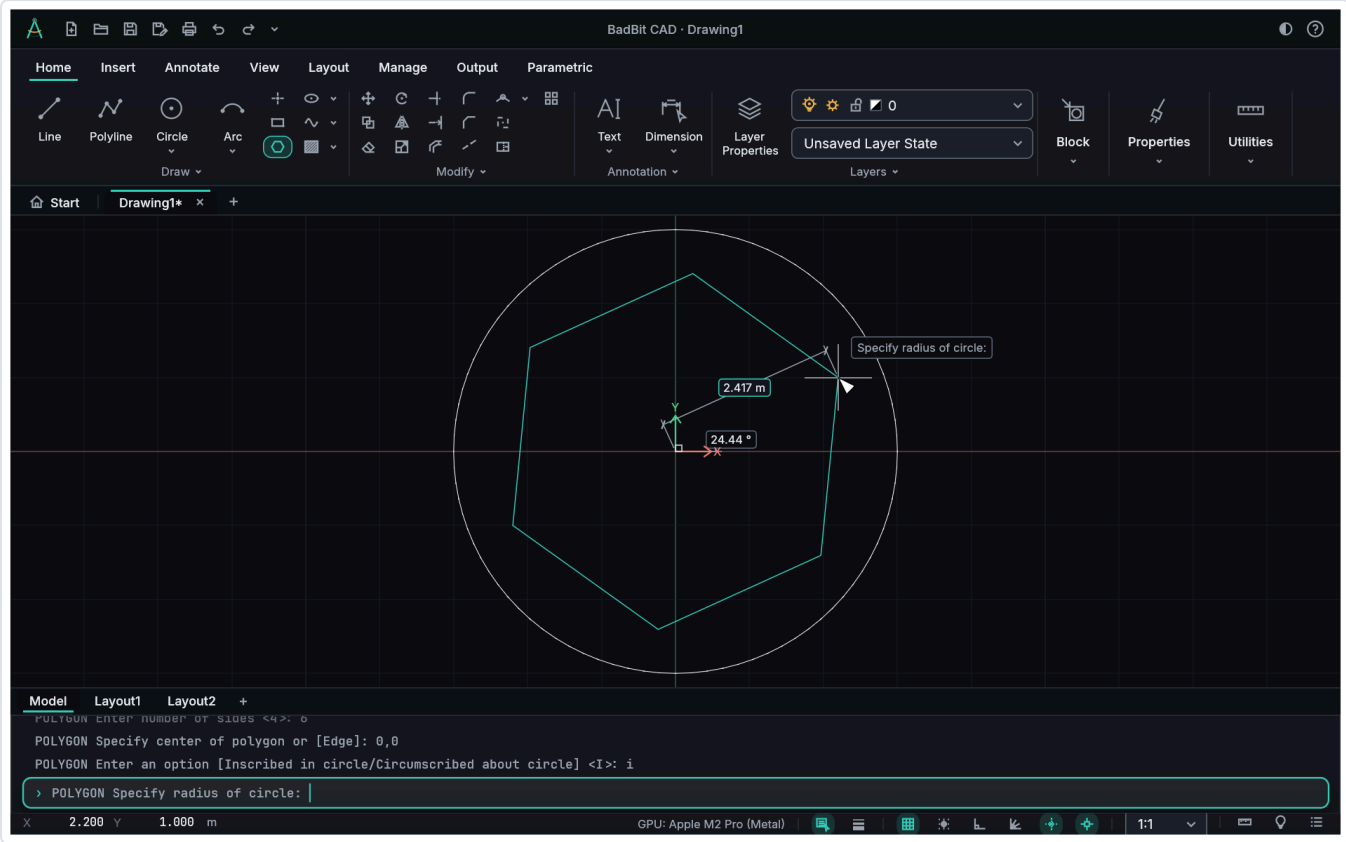
- **Area** (`A`): the area and then the length or the width; the other side comes from the calculation;
- **Dimensions** (`D`): the typed length and width; a click chooses which side the rectangle goes to.
- **Rotation** (`R`): the angle of the sides, typed or pointed. It is remembered for the next rectangles.



RECTANG: after the first corner, the preview follows the cursor.

POLYGON (alias POL)

A regular polygon as a closed polyline. Enter the number of sides (3 to 1024; it is remembered), the center, and whether it is **Inscribed** in the circle (the radius goes to the vertices) or **Circumscribed** (the radius goes to the middle of the sides); then the radius, typed or pointed (the point also gives the direction). With **Edge** (`E`) in place of the center, enter the two points of a side.



Inscribed 6-sided POLYGON: the radius and the direction of the first vertex come from the cursor.

Both are on the ribbon's Draw panel, below Point.

ELLIPSE (alias EL)

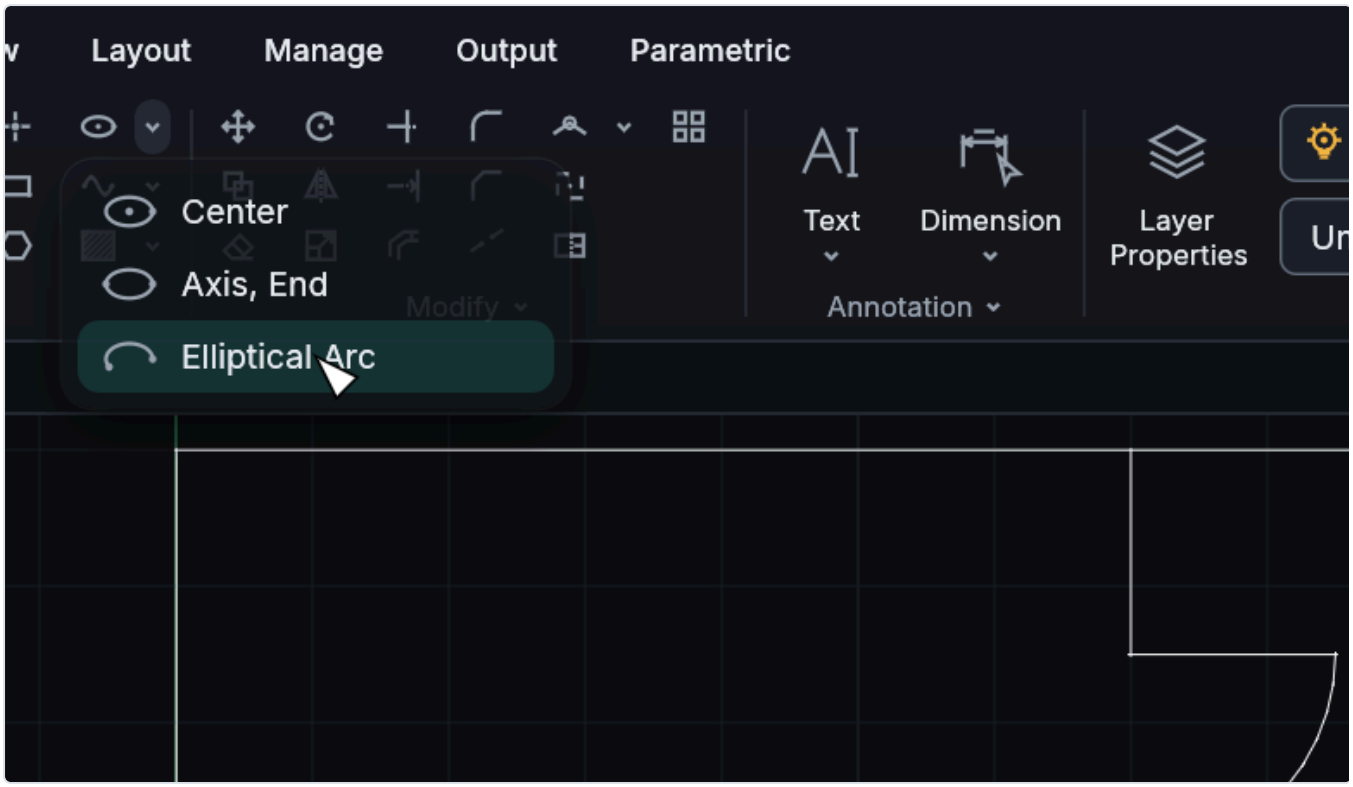
An ellipse, or an elliptical arc. When typed, the command starts with the ends of an axis, as in AutoCAD:

1. **Axis endpoint** and the **other endpoint**: the two ends of the first axis (the center is in the middle);
2. **Distance to other axis**: from the center to the end of the other axis, typed or pointed.

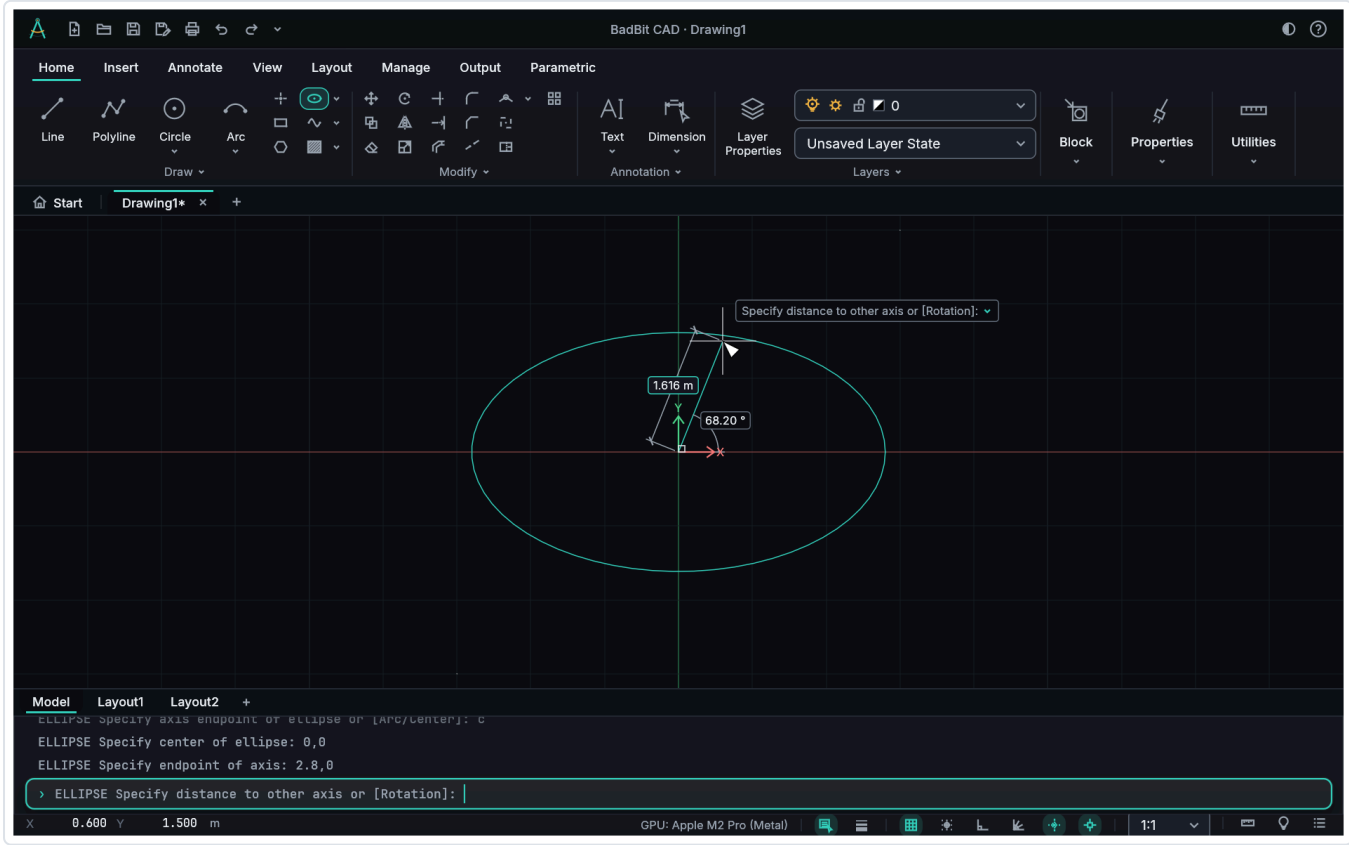
The first axis can turn out to be the major or the minor one: the ellipse figures out which is which by itself.

- **Center** (`C`): the center, the end of one axis, and the distance to the other.
- **Rotation** (`R`), in place of the distance: the ellipse is the circle of the first axis rotated around it, from 0 to 89.4° (the other axis is the first one times the cosine of the angle).
- **Arc** (`A`): the same ellipse, and then the **start angle** and the **end angle**, counted from the first axis given, counterclockwise. At the end angle, **Included angle** (`I`) gives the opening of the arc from the start angle (`Enter` accepts 180°). **Parameter** (`P`) swaps the angles for the parameters of the ellipse equation, and **Angle** (`A`) goes back.

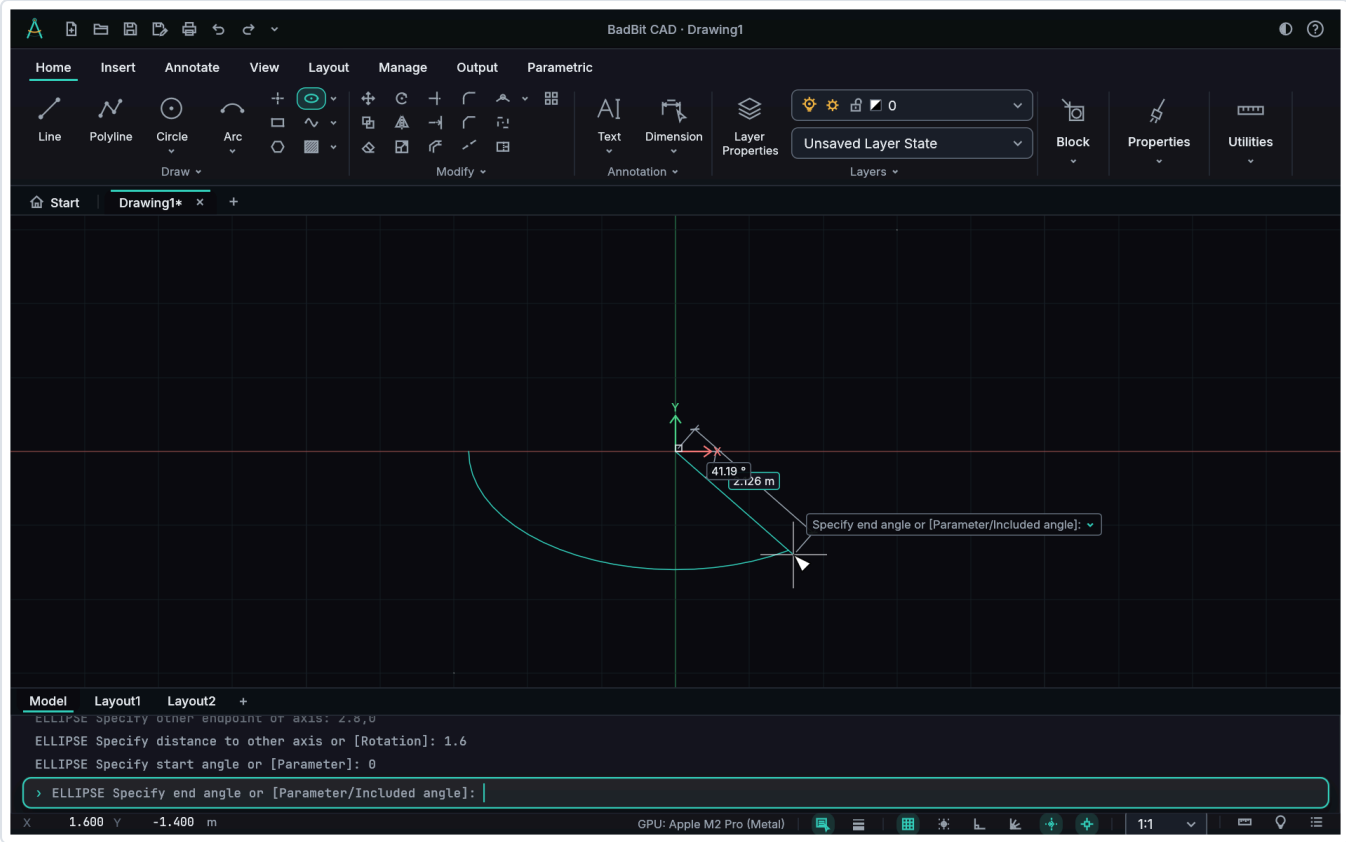
On the ribbon, the **Ellipse** button (Draw panel) has AutoCAD's list: **Center**, **Axis, End**, and **Elliptical Arc**. The button repeats the last form chosen in the list, and the icon changes with it.



The Ellipse button list: the small arrow on the right opens the forms.



ELLIPSE by center: after the end of the first axis, the preview follows the distance to the other.



Elliptical arc: with the axes defined, the end angle follows the cursor.

The ellipse has center, quadrant (the ends of the axes), nearest, and intersection snaps; the arc also has endpoint and midpoint. TRIM, BREAK, OFFSET, STRETCH, MOVE, COPY, ROTATE, SCALE, MIRROR, ARRAY, and grips work on it, and the properties palette shows the center, the major and minor radii, the ratio between them, the arc angles, and the area, calculated along the curve (for the arc, the area enclosed by the chord between the ends, like AutoCAD's Area property and AREA).

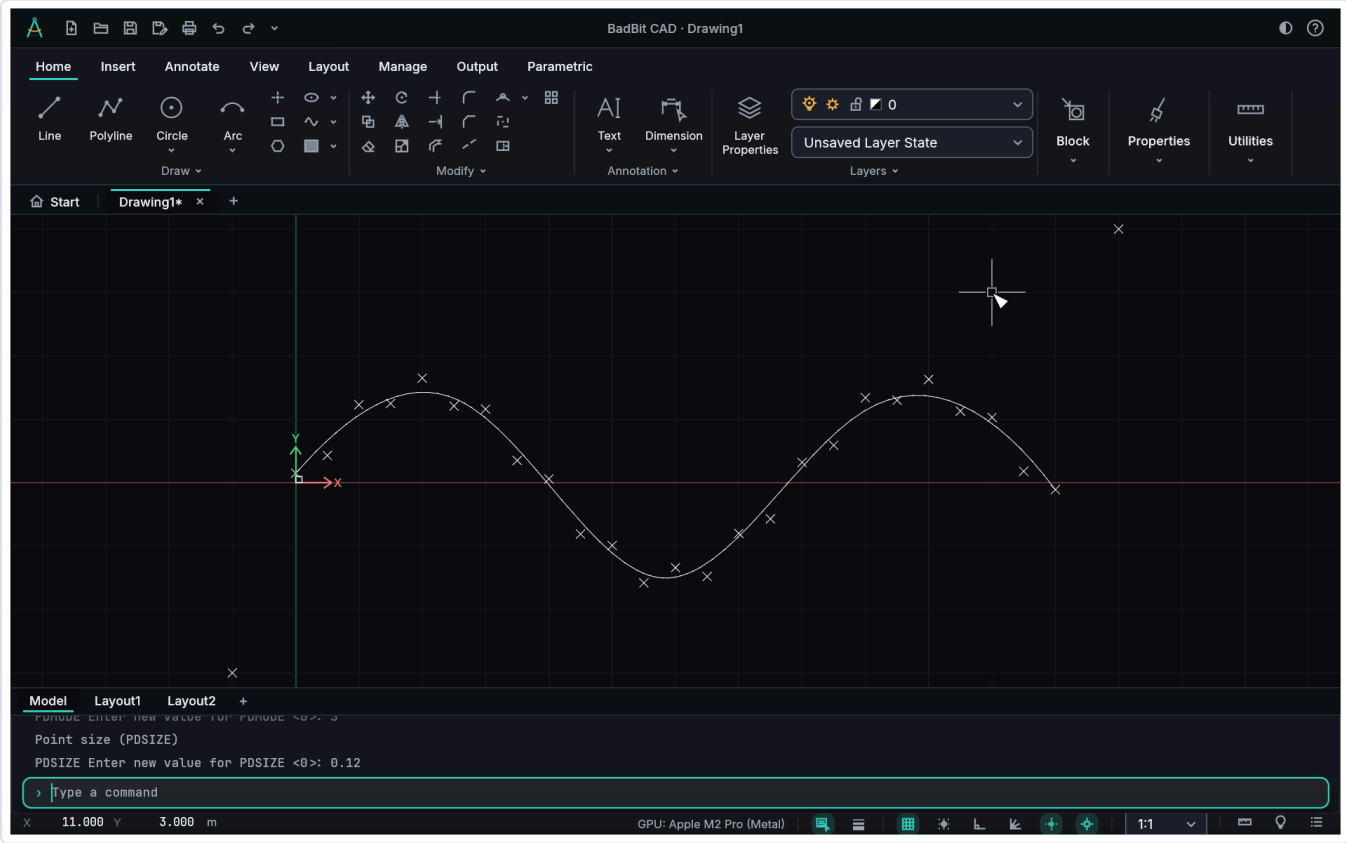
SPLINE (alias SPL)

A smooth curve. There are two methods, as in AutoCAD; the one chosen is remembered for the next spline.

Fit (the default): the curve passes through all the clicked or typed points (a cubic).

- Before the first point, **Knots** (`K`) chooses how the points are spaced along the curve: **Chord** (by the distance between them, the default), **Square root** (smoother on closed curves and with uneven points), or **Uniform**.
- After the first point, **start Tangency** (`T`) gives the direction of the start of the curve (a point from the first one). After the second, **end Tangency** gives the direction of the end and finishes the spline.
- Without tangencies, the ends follow the natural curvature of the first three and the last three points.
- After the first point, **toLerance** (`L`) gives the fit tolerance: with 0 (the default) the curve passes through the points; with a larger value, it passes at most that distance from them (the ends stay exact), with fewer vertices and smoother, good for survey points with noise. The tolerance is

remembered and appears between `< >` in the prompt; the spline stores it, and moving a grip rebuilds the curve with it.



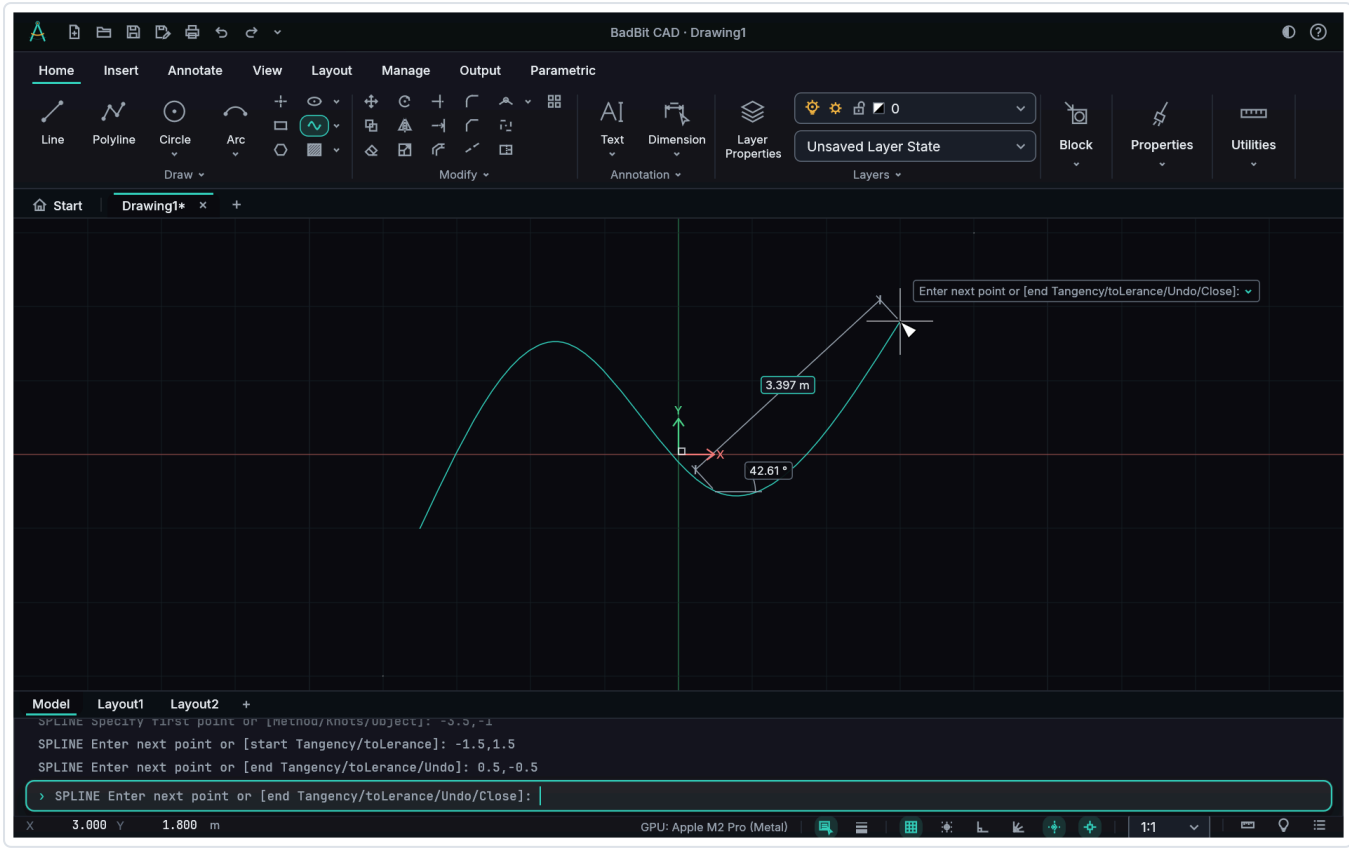
SPLINE with fit tolerance: the curve passes close to the survey points, without following the noise.

CV (control vertices): the points form the control polygon, and the curve is pulled by it, without passing through the middle vertices. Before the first point, **Degree** (`D`) chooses the degree, from 1 (polyline) to 10 (default 3).

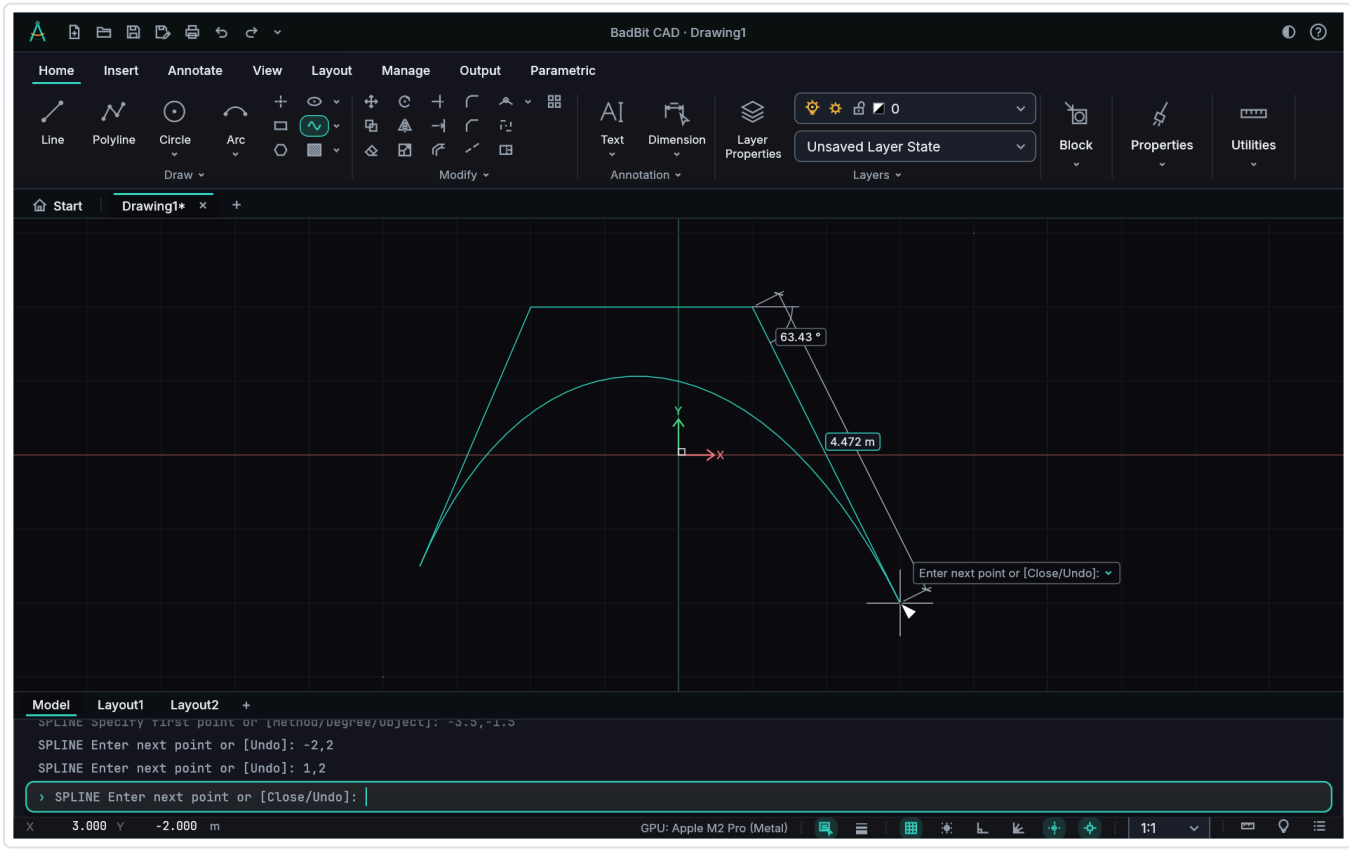
In both: **Undo** (`U`) removes the last point; **Close** (`C`, with three points or more) closes the curve without a corner and finishes; `Enter` finishes it open; `Esc` exits without creating it. **Method** (`M`), before the first point, switches between Fit and CV.

Object (`O`), before the first point: click a polyline and it becomes a spline in its place, with the polyline's vertices as the control polygon (like AutoCAD's spline-fit polyline). A single undo step gives the polyline back.

On the ribbon, the **Spline** button (Draw panel) has **Spline Fit** and **Spline CV** under its small arrow.



Fit SPLINE: the curve passes through the points and the preview follows the cursor.



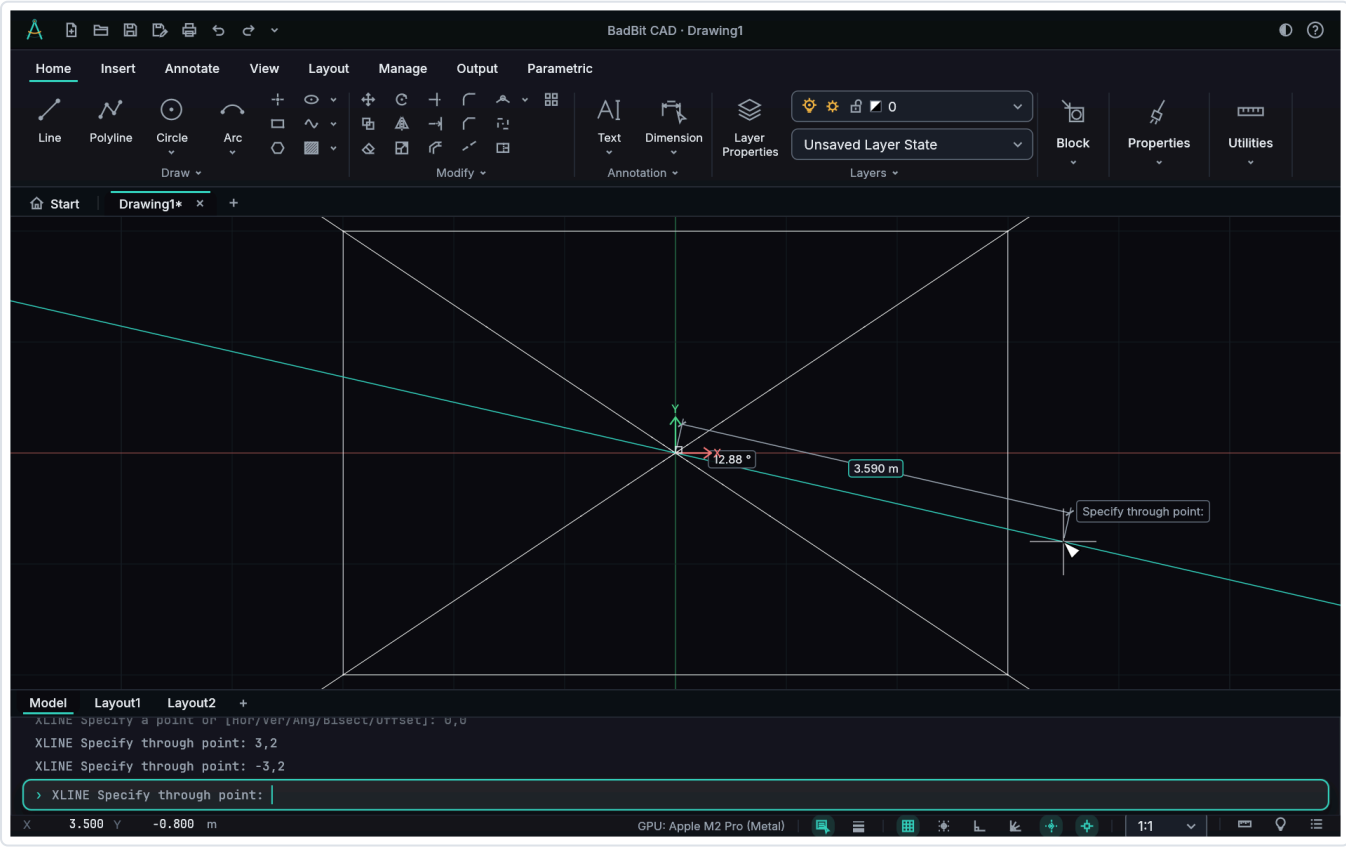
CV SPLINE: the control polygon and the curve pulled by it.

The spline has endpoint (when open), nearest, and intersection snaps. TRIM, BREAK, STRETCH, OFFSET, MOVE, COPY, ROTATE, SCALE, MIRROR, and ARRAY work on it. The grips are at the fit points (moving one rebuilds the curve through them) or at the control vertices. Trimming or breaking a fit spline turns it into a control-vertex spline, as in AutoCAD. The properties palette shows the method, the degree, whether it is closed, how many fit points and vertices it has, the length, and the area, both along the curve (for an open spline, the area enclosed by a straight line from the end to the start, like AREA).

XLINE (alias XL)

Construction lines: lines infinite in both directions, to guide the drawing (axes, alignments, angles). Enter a point (the root) and, repeatedly, through points: each one creates a line through the root. `Enter` finishes, and one U undoes all the lines of the command at once.

- **Hor** (`H`) and **Ver** (`V`): horizontal or vertical, one per clicked point.
- **Ang** (`A`): at the typed angle (`Enter` = 0°); with **Reference** (`R`), the angle counts from a chosen line.
- **Bisect** (`B`): the bisector of the angle given by the vertex, a start point and, repeatedly, end points.
- **Offset** (`O`): parallel to a line (line, construction line, ray, or polyline segment) at a distance, on the clicked side; with **Through** (`T`, or `Enter`), passing through the clicked point.



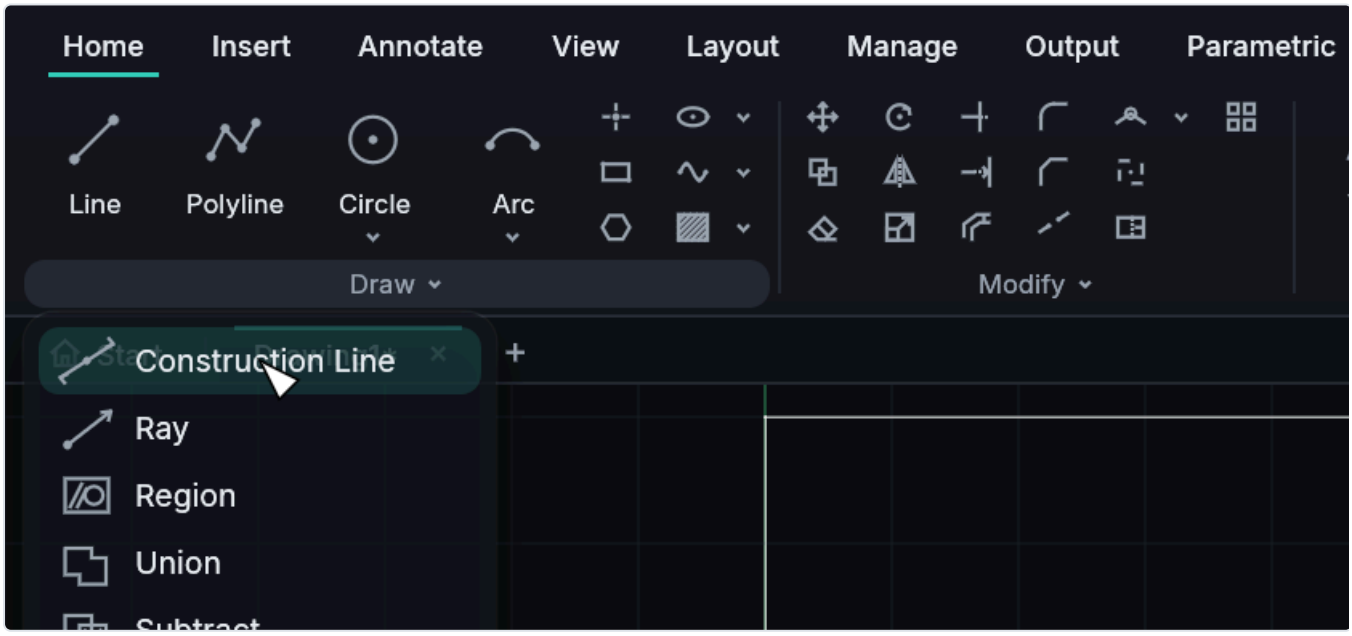
XLINE: the root stays in place and the line follows the cursor.

RAY

Like the construction line, but infinite in one direction only: enter the start and, repeatedly, through points.

Construction lines and rays appear whole at any zoom, but zoom extents (Z) counts only their base point, as in AutoCAD. Snaps: the root of the construction line is a midpoint, the start of the ray is an endpoint, and nearest, perpendicular, and intersection work (at any distance). TRIM turns the construction line into a ray (or a line) and the ray into a line; so does BREAK. OFFSET, MOVE, COPY, ROTATE, MIRROR, SCALE, and ARRAY work on them. Grips: the root moves, the other two points rotate the line around the root. The properties palette shows the base point, the second point, and the angle.

On the ribbon, both are on the **expanded** Draw panel, as in AutoCAD: click the **Draw** ▾ title to open the list.



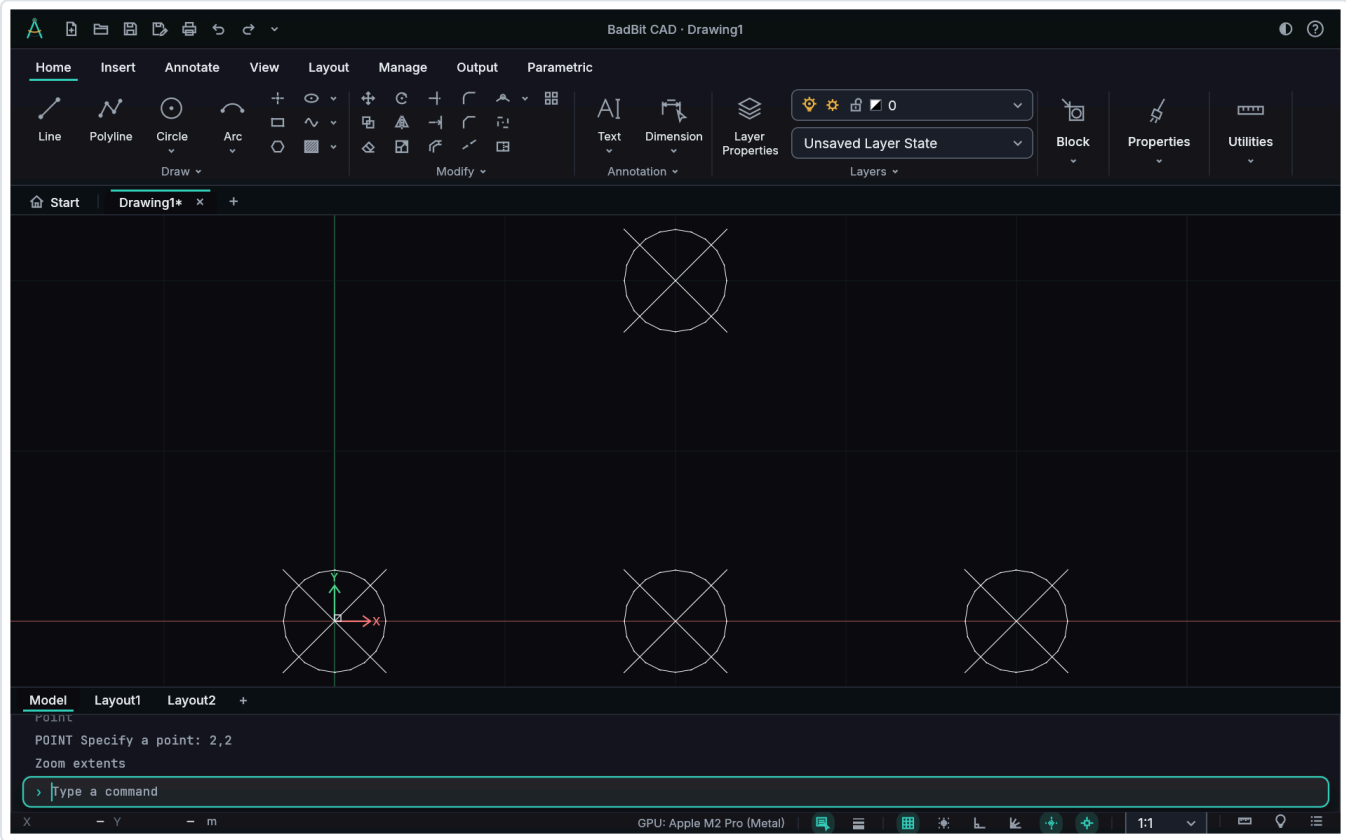
The expanded Draw panel, with Construction Line and Ray.

POINT (alias PO)

Marks a point in the drawing. The **Node** snap sticks to points. When typed, POINT marks one point and finishes (`Enter` repeats the command); the ribbon button is AutoCAD's **Multiple Points**, which keeps marking with each click until `Esc` or `Enter`. The symbol and size of the points follow the **PDMODE** and **PDSIZE** variables, which apply to all points in the drawing:

PDMODE	Symbol
0	Dot (default)
1	Nothing
2	Cross (+)
3	X
4	Upward tick
+ 32	With a circle around it
+ 64	With a square around it

For example, 35 = X with a circle. A positive **PDSIZE** is the size in the project unit; negative, a percentage of the screen height; zero, 5% of the screen.



Points with PDMODE 35 (X with a circle).

DONUT (alias DO)

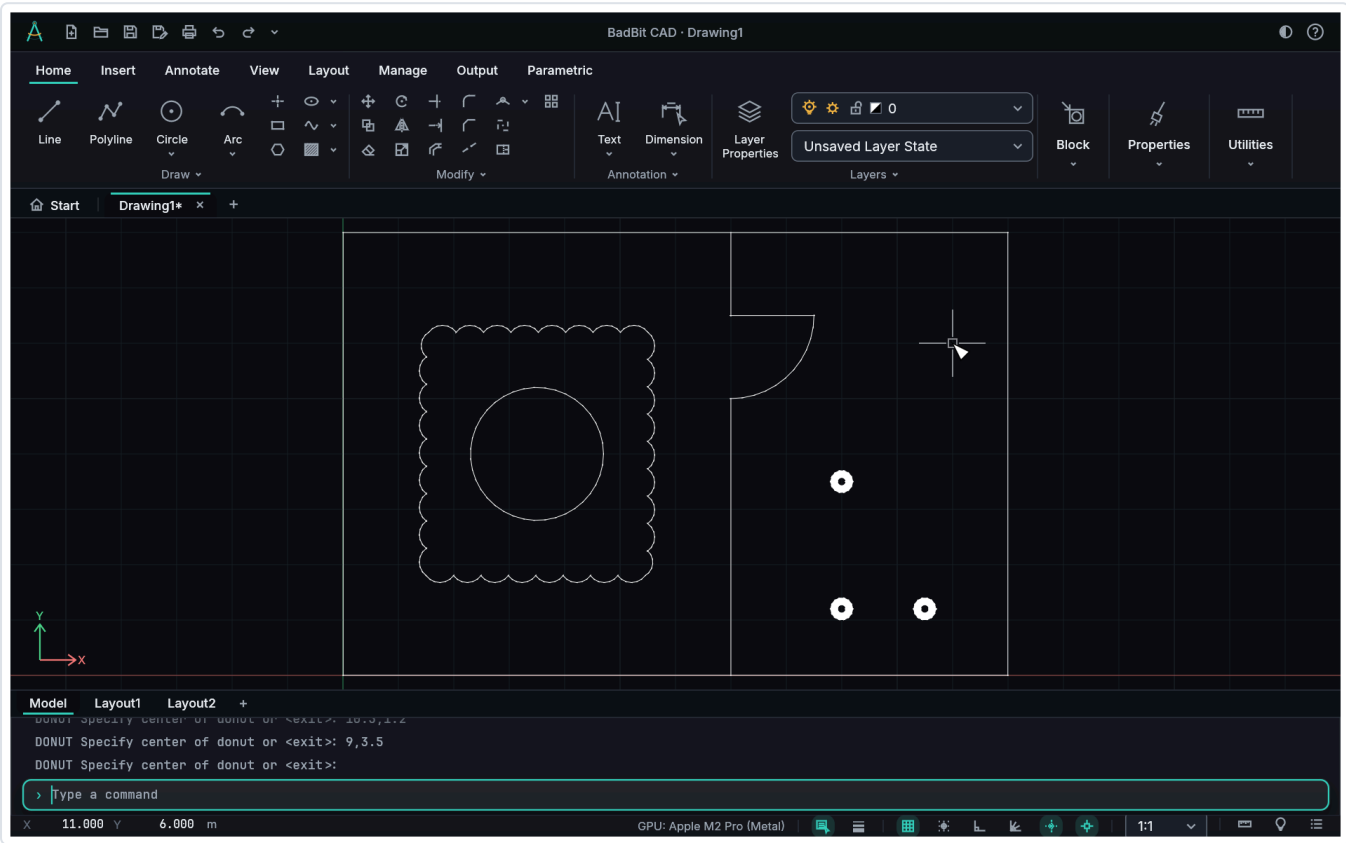
Donut: a ring made, as in AutoCAD, by a polyline of two arcs with width. The command asks for the **inside diameter**, the **outside diameter**, and the **centers**, one after the other, until `Enter`. With an inside diameter of 0, you get a disk. The donuts made in one go are a single undo step.

REVCLOUD

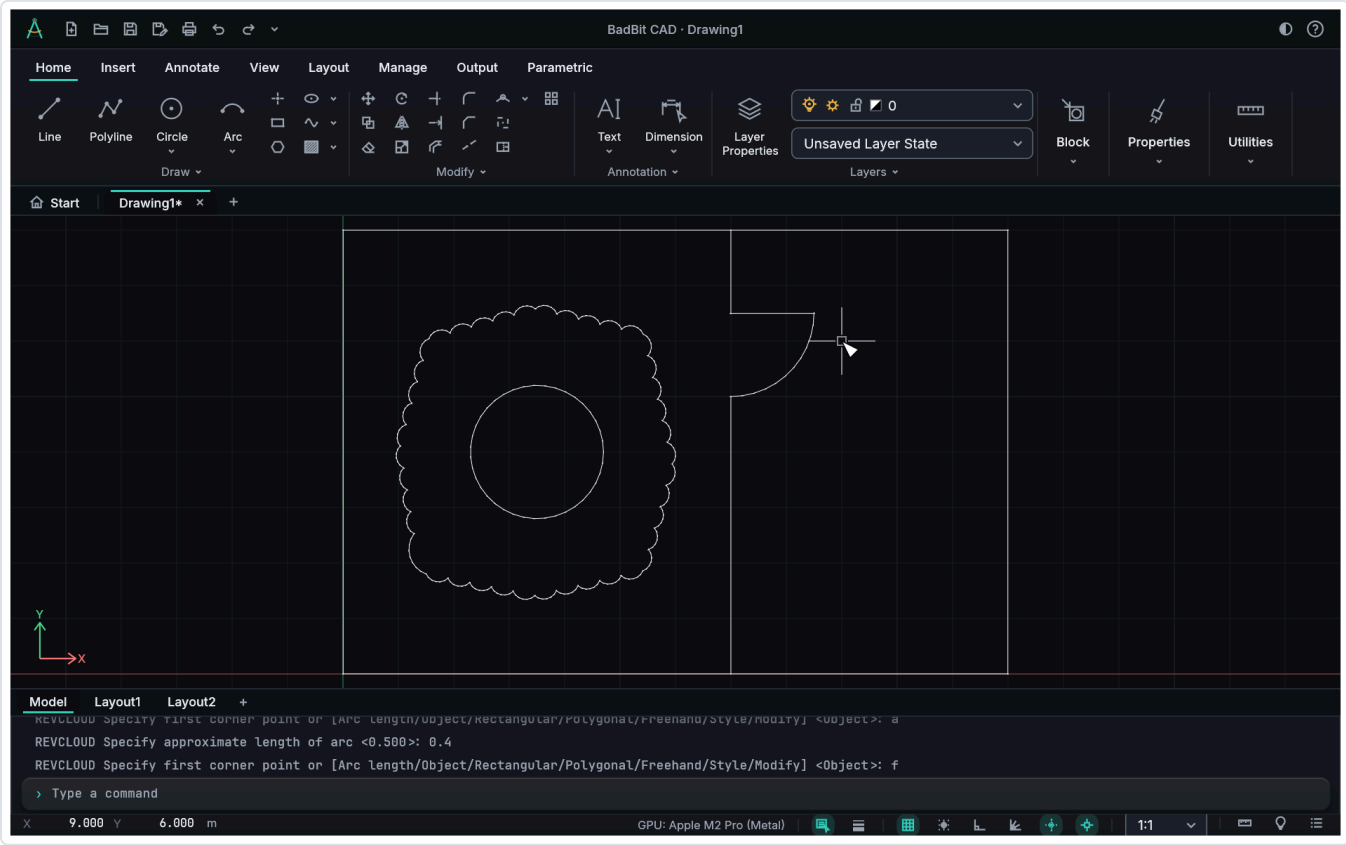
Revision cloud: a polyline of arcs around what changed in the drawing. Before the first point, the options:

Option	What it does
Arc length (`A`)	The approximate size of each arc (the arcs are spread evenly along the outline).
Object (`O`, or `Enter`)	A closed object (circle, ellipse, closed polyline or spline) becomes the cloud; then it asks whether to **reverse** the direction of the arcs (inward).
Rectangular (`R`)	The cloud by two corners (the default).
Polygonal (`P`)	The cloud by points, until `Enter` (`U` undoes the last one).
Freehand (`F`)	After the first point, the cloud follows the cursor: one arc for each arc length traveled. Coming back near the start, it closes by itself; `Enter` before that finishes the cloud open.
Style (`S`)	**Normal** or **Calligraphy** (each arc starts thin and thickens, like a pen).
Modify (`M`)	Redraws a piece of a cloud (or of any polyline): click the polyline (that is where the new path starts), the points of the new path, and the polyline again (or `Enter`, which takes the last point to it). Then **pick the side to erase** and whether to **reverse** the new arcs. The new path becomes cloud arcs.

The arcs bulge outward from the outline. Rectangular, Polygonal, and Freehand are remembered: the first click of the next cloud follows the last type chosen.



A rectangular cloud around the table and three donuts.



The freehand cloud around the table: the cursor went all the way around and it closed by itself.

BOUNDARY (aliases BO, -BO)

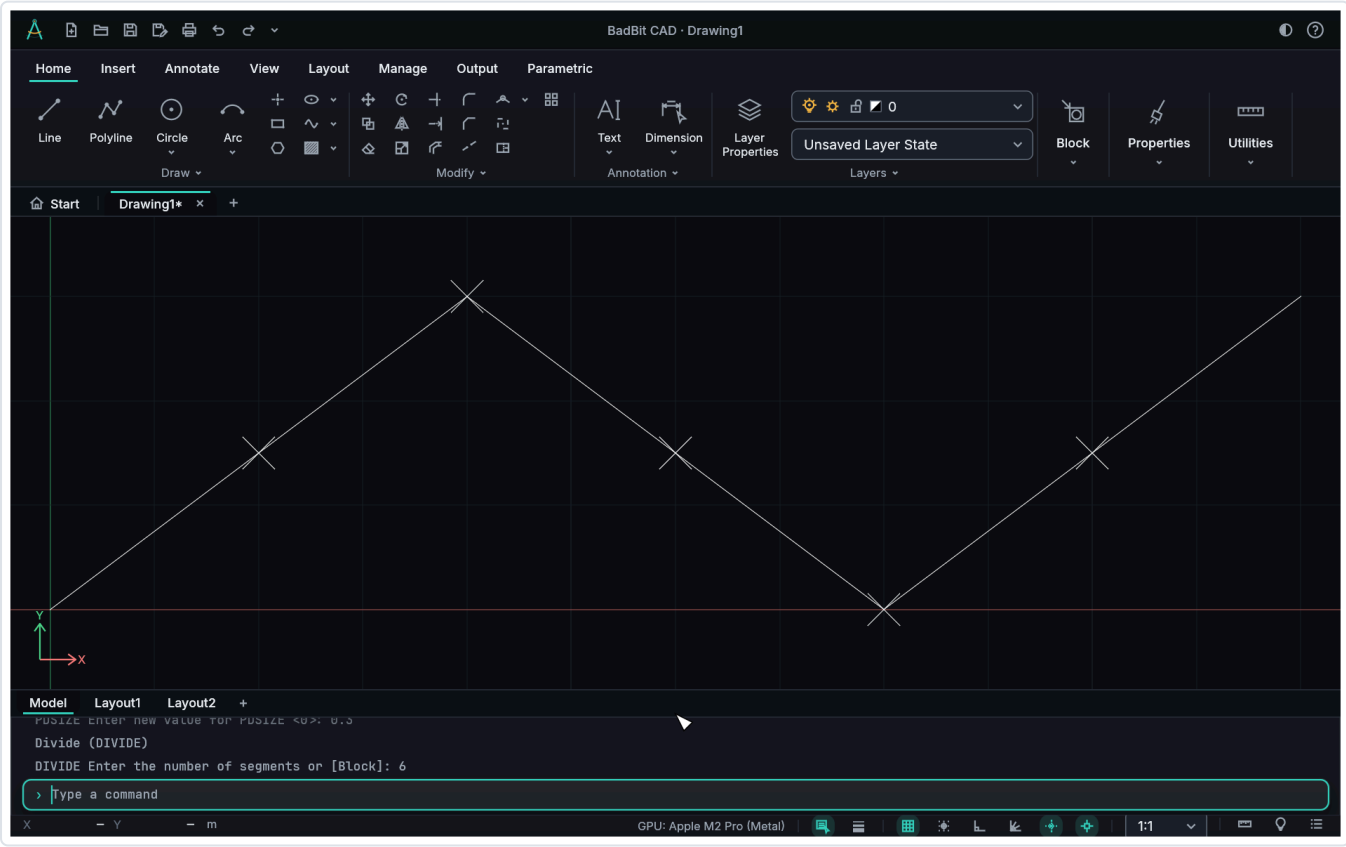
Creates a closed polyline around the region you click in, formed by the objects that intersect (lines, arcs, circles, polylines, ellipses, and splines; arcs stay arcs). It asks for points until `Enter`, one boundary per point, on the current layer. With **Island detection** (advanced options, `A` and `I`; on by default), the loose closed objects inside the region also get their boundary. A point outside a closed region gives a notice and keeps asking; a point between two nearly coincident curves (closer together than the drawing's tolerance along a stretch) also gives no boundary, with the notice that the curves are nearly coincident there. In the advanced options, **Object type** (`O`) swaps the polyline for a **Region** (`R`), with the islands as holes (see *Blocks, associative arrays and regions*).

A polyline has only straight segments and circular arcs: when the boundary passes along an **ellipse** or **spline** and the type is Polyline, BOUNDARY does not create a polyline of straight segments (it would measure a different area); like AutoCAD, it asks whether to create a **region** instead (`Y`, the default) or nothing (`N`). The region carries the area and perimeter of the curve itself. The type chosen remains Polyline for the next points.

DIVIDE (alias DIV)

Places points (POINT objects) that divide an object into parts of equal length: choose the object and the **number of segments**. On an open object, there is one point fewer than the parts; on a closed one (circle, closed polyline), one per part, starting at the start of the object. Use **PDMODE** and **PDSIZE** to

see the points and the **Node** snap to use them. With **Block** (`B`) in place of the number, the divisions receive insertions of a block from the drawing: the block name, whether it is **aligned** with the object (`Y`, rotates with the curve), and then the number of segments.



A polyline divided into 6 parts, with PDMODE 3.

MEASURE (alias ME)

Like DIVIDE, but by the **length of each segment**: the points are placed at each given distance, starting from the end of the object nearest to where it was clicked (on a closed object, from the start). The **Block** option (`B`) also works, as in DIVIDE.

MLINE (alias ML)

Draws a **multiline**: parallel lines (the **elements** of the current multiline style) that follow the given points, with the correct joints at the corners. When it starts, the command line shows the **current settings** (justification, scale, and style). Before the first point, these options work:

- **Justification** (`J`): **Top** (`T`, the element with the largest offset passes through the points; drawing from left to right, the multiline stays below), **Zero** (`Z`, the zero offset passes through the points), or **Bottom** (`B`, the one with the smallest offset passes through the points). This is CMLJUST.
- **Scale** (`S`): how many times the style's width (STANDARD has width 1: elements at +0.5 and −0.5). A negative scale reverses the order of the elements; zero merges everything into a single line. This is CMLSCALE (it does not change the linetype scale).
- **STyle** (`ST`): the name of a loaded style (CMLSTYLE); `?` lists the drawing's styles.

After the first point, each point adds a segment; with two points **Undo** appears (`U`, removes the last vertex) and, with two segments or more, **Close** (`C`, connects the last segment to the first, with the joint at the start). `Enter` finishes. The multiline exists in the drawing from the second point on (the snaps already see it) and the whole command is a single undo step.

The multiline has **grips** at the vertices (dragging one rebuilds the multiline through the new vertices), stretches by the vertices inside the **STRETCH** window, moves with MOVE, ROTATE, SCALE, and MIRROR, gives the snaps of its lines and arcs, and appears in **LIST** (style, scale, justification, length, and vertices). **EXPLODE** turns it into its lines and arcs, each element with its own color and linetype (the fill disappears, as in AutoCAD). **OFFSET** does not offset a multiline (AutoCAD does not either). In the **Properties** palette, **Style** is read-only and **Justification** and **Scale** can be changed (the multiline is rebuilt through the same vertices); whether it is **Closed**, the number of **Vertices**, and the length also appear.

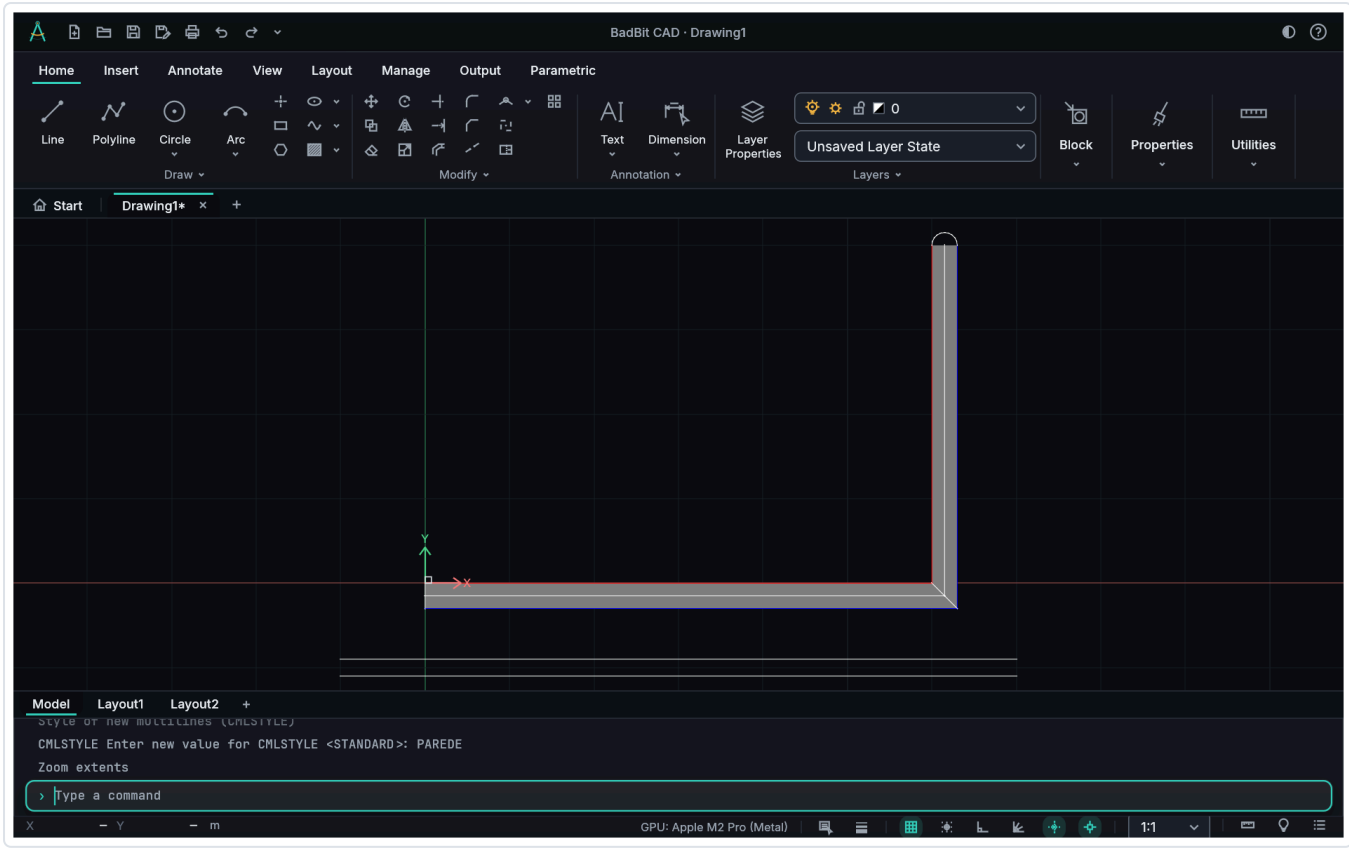
In **DXF** and **DWG** it goes as AutoCAD's MLINE, with the style in the multiline style dictionary; a multiline cut by MLEDIT in AutoCAD opens with the cuts. In R12, which has no multilines, it goes as its lines and arcs.

MLSTYLE

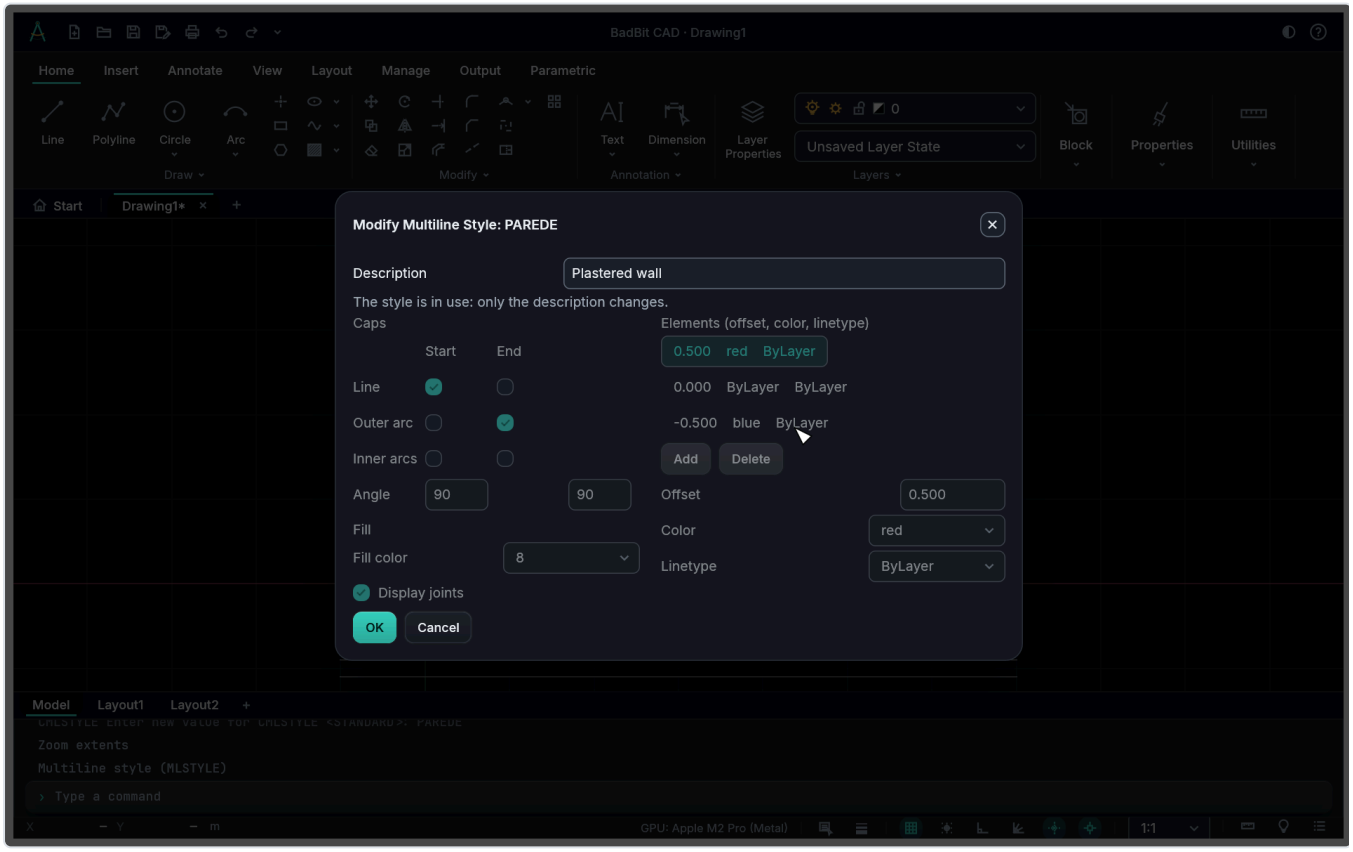
Opens the **Multiline Style** window: the current style, the list of the drawing's styles (the ● marks the current one), the **Description** and the **Preview** of the selected style, and the buttons:

- **Set Current**: the selected style becomes the one for new multilines (also with a double-click).
- **New...**: asks for the **New style name** and the style to **Start with**; **Continue** opens the new style's window.
- **Modify...**: opens the selected style's window. **STANDARD** and a style **in use** by some multiline only change their description (AutoCAD does not allow changing the elements and properties of a style in use: make the changes before drawing).
- **Rename**: changes the name (STANDARD cannot be renamed); the multilines of the style go along.
- **Delete**: removes the style from the drawing (not STANDARD, not the current one, not one in use).
- **Load...**: picks an **.mln** library file (like AutoCAD's `acad.mln`) and shows its styles to choose from; a style with the same name that is not in use is replaced by the one from the file.
- **Save...**: writes the selected style to an **.mln**; if the file already exists, the style goes in together with the ones already there (the one with the same name is replaced).

The style window (**New Multiline Style** or **Modify Multiline Style**) has the **Description**; the **Caps** for the **Start** and the **End**: **Line** (from one outer element to the other), **Outer arc** (between the two outermost elements), **Inner arcs** (between the inner pairs: the 2nd with the second-to-last and so on; with an odd number, the middle one is left loose), and the **Angle** (from 10° to 170°); the **Fill** (the **Fill color**, or None); **Display joints** (the lines at the vertices); and the **Elements**, in descending order of offset, each with the **Offset**, the **Color** (ByLayer and ByBlock take the multiline's; Select Color opens the color window), and the **Linetype** (the ones loaded in the drawing). **Add** adds an element at offset 0 (up to 16) and **Delete** removes the selected one.



A wall drawn with MLINE in a three-element style (red, ByLayer, and blue) with fill, joints, a line at the start, and an outer arc at the end; below, a STANDARD multiline with Zero justification.



The PAREDE style window (Modify): the caps, the fill, the joints, and the elements. Since the style is in use, only the description changes.

WIPEOUT

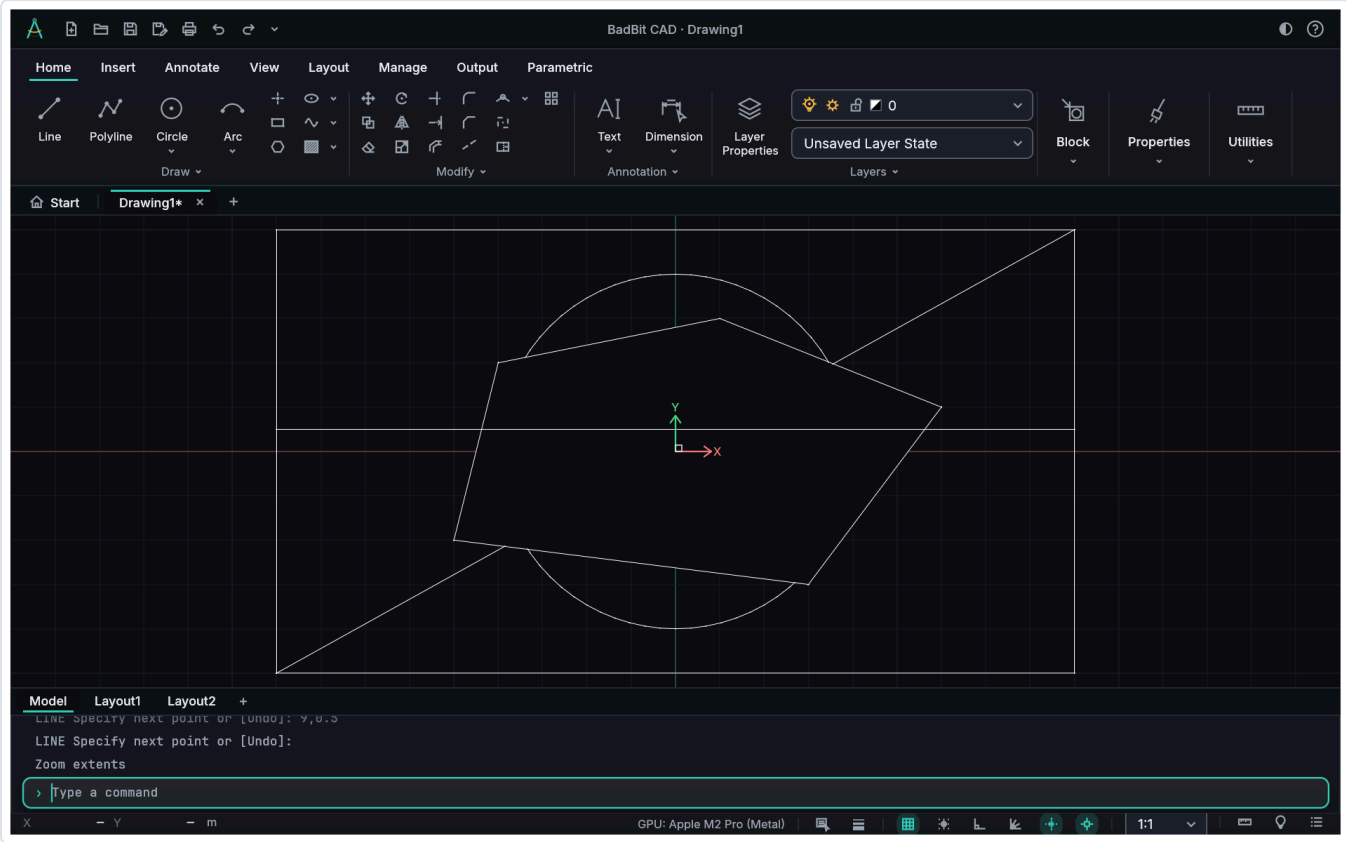
Creates a **wipeout** (AutoCAD's mask): a polygonal area painted with the background color of the drawing area that **hides what comes before it in the draw order**. What is drawn afterward stays on top; DRAWORDER, TEXTTOFRONT, and HATCHTOBACK change what hides what. In paper space, in plotting, and in the PDFs and PNGs generated by PLOT, the wipeout color is the paper's. A wipeout inside a block hides the block's items that come before it and what came before the insertion in the drawing. The English name, WIPEOUT, works in any language. The button is on the **Home** tab, on the expanded **Draw** panel (the panel's small arrow).

The first prompt is the **first point** or an option:

- **Points**: each point adds a vertex; with two points **Undo** (`U`) appears, with three or more, **Close** (`C`). `Enter` (or Close) creates the wipeout; with fewer than three points, Enter cancels.
- **Polyline** (`P`, the answer for `Enter`): choose a **closed** polyline of zero width; arc segments become straight, as in AutoCAD. Then comes **Erase polyline? [Yes (Y)/No (N)] <No>**. The wipeout and the erased polyline are a single undo step.
- **Frames** (`F`): **On** (`ON`, the frame appears and plots), **Off** (`OFF`, the frame disappears; it appears only in the selection highlight), or **Display but not plot** (`D`). It applies to all wipeouts in the drawing: it is the **WIPEOUTFRAME** variable, stored in the drawing. **TFRAMES** (from Express Tools) turns the frames of images (IMAGEFRAME) and of wipeouts on or off at once.

A wipeout is selected by its frame (even when hidden), has **grips** at the vertices, moves with MOVE, ROTATE, SCALE, and MIRROR, stretches by the vertices inside the **STRETCH** window, gives the endpoint, midpoint, and intersection snaps on the sides of the frame, and appears in **LIST** (area, perimeter, and vertices). In the **Properties** palette, the chosen **Vertex** is changed through **Vertex X** and **Vertex Y**, with the area and the perimeter. EXPLODE does not explode a wipeout (AutoCAD does not either).

In **DXF** and **DWG** it goes as AutoCAD's WIPEOUT (a 1×1 image clipped by the outline) and WIPEOUTFRAME in the object of the `ACAD_WIPEOUT_VARS` dictionary; an AutoCAD drawing with wipeouts opens with them hiding what is behind. In R12, which has no wipeouts, only the outline goes.



A rectangle, a circle, and a diagonal drawn before the wipeout disappear inside it; the horizontal line, drawn afterward, stays on top. The frame is visible (WIPEOUTFRAME 1).

Text and dimensions

The stroke font

BadBit CAD texts use its own stroke font, designed for CAD: each letter is made of lines and arcs, like the single-line fonts of drafting programs. That is why text stays sharp at any zoom, looks the same with or without a graphics card and plots with a single pen pass. It has the letters of Portuguese and Spanish (accents, ç, ñ, ¿, ¡) and the engineering symbols: °, ±, Ø (diameter), ², ³, µ, ×. Accents are typed as in any program: `´` and then `a` gives á; `~` and `a`, ã.

Besides the stroke font, text can use the **TrueType and OpenType fonts** installed on the computer (Arial, Calibri, Helvetica...), chosen in the text style (see STYLE). The letters of these fonts come out **filled**, with the font's own metrics (without kerning, as in AutoCAD: each letter takes up its own advance), the Portuguese and Spanish accents and whatever symbols the font has (°, ±, ², ³; the diameter `%%c` comes out as the font's ⌀ or, without it, as Ø, as in AutoCAD). They look smooth up close and light from afar: the curve of each letter is rebuilt with the precision of the zoom, like circles. The text of dimensions, leaders (MLEADER and QLEADER), the geometric tolerance frame (TOLERANCE) and block attributes also uses the text style's font and comes out filled, including inside blocks and external references; the tolerance symbols (⌖, ▱, ⌀, Ⓜ...) stay in the stroke font, as AutoCAD does with the GDT font.

In the stroke font (and in AutoCAD SHX fonts), the text height is the height of the capital letters. In TrueType fonts, as in AutoCAD, the height is that of the font's **A**: capitals come out at the text height (in Arial, the A has exactly that height and the O goes slightly beyond it), and accents and tall letters may go beyond it. The box of a TrueType text (the extents of ZOOM Extents) runs from the start to the end of the letter advances and from the font's descender to its ascender, also as in AutoCAD; the Center and Right justifications count from that box, and **Middle** centers on the height of the drawn letters. So a drawing made in AutoCAD with Arial opens in BadBit CAD with the texts the same size and in the same place. The height is the same for TEXT and MTEXT, like AutoCAD's TEXTSIZE: it starts at 0.25 m, changes with the **Height** option of both commands or with the **TEXTSIZE** variable, and is remembered for the next texts — unless the current text style has a fixed height (see STYLE). At the start of each command, the command line shows the style and height the text will have (`Current text style: "Standard" Text height: 0.250 m`); in a layout (on paper), the height is in sheet millimeters, like the prompt (`Text height: 2.50 mm`). In MTEXT, the rectangle gives the width of the block, not the height of the letters.

MTEXT (alias T, MT)

Multiline text, the most used kind, as in AutoCAD. It is what the **Text** button on the Annotation panel opens (its small arrow also gives Single Line Text):

1. **First corner** and **opposite corner** of a rectangle;
2. the text, typed **directly in the rectangle**, in the drawing: words wrap on their own at its width. The block keeps the height of the rectangle (the "defined height", seen in Properties in AutoCAD) and only grows downward when the text goes past it. `Enter` starts another

paragraph; `Ctrl` + `Enter` (`Cmd` + `Enter` on the Mac) or a click outside finishes; `Esc` cancels. The arrow keys move through the lines, and the other editing keys are the same as in TEXT.

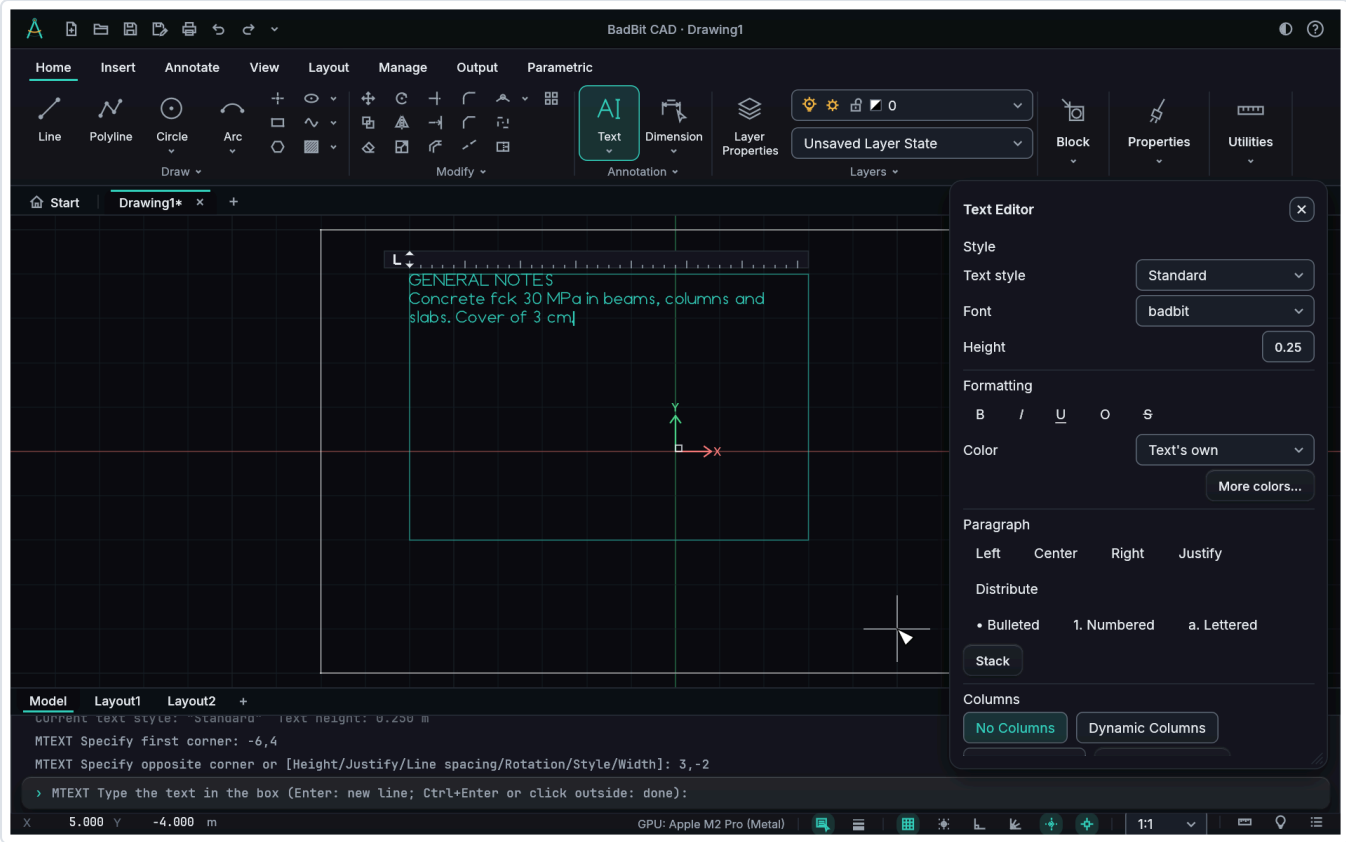
Before the opposite corner, the options:

- **Height** (`H`, or `A`): the height of the letters;
- **Justify** (`J`): the alignment point of the block, one of the nine (TL, TC, TR, ML, MC, MR, BL, BC, BR); the default is top left, at the first corner;
- **Line spacing** (`L`, or `E`): the type, **At least** (`A`, the default) or **Exactly** (`E`), and the factor (`1x` is normal; `1.5x`, one and a half times) or the distance between lines;
- **Rotation** (`R`): the angle of the block;
- **Style** (`S`): the name of the text style (`?` lists the styles in the drawing); it becomes the current one;
- **Width** (`W`): the typed width, instead of the opposite corner.

The values are remembered for the next one. A block with no text is not created.

With TrueType and SHX fonts, MTEXT is laid out by the **drawn letters**, as in AutoCAD: each line puts the letters flush against the alignment side (or centers them), the first line touches the top of the block and the bottom of the block is the bottom of the letters of the last line; a word fits on the line when its letters fit within the width (not the advance of the last letter); a justified paragraph stretches the spaces until the letters reach the right side; spaces at the end of the line do not count. Text with **mixed fonts** (TrueType with SHX, or two TrueType fonts) follows the same rule. In **columns**, each column does the same: the line moves to the next column when the bottom of its letters goes past the column height, and in MTEXT attached at the middle or bottom each column aligns by the height of its own letters. The alignment point counts from the height of the lines, not from the drawn rectangle (the defined height does not change where the text is).

MTEXT is **selected** by the box of its letters: a **window** catches the text when it surrounds all the letters (even if the MTEXT rectangle is wider) and a **crossing** catches it when it touches the outline of any letter, as in AutoCAD (this also applies to TEXT). **ZOOM Extents** counts the letters, not the rectangle.



MTEXT: the rectangle and the text wrapping at its width, typed in the drawing.

To change the text later, double-click it (see TEXTEDIT). The properties palette shows the contents (the running text, without the formatting codes), the style, the font, the alignment, the height, the rotation, the width and the spacing, and changes the alignment, height, rotation and width. When the pieces of text have different fonts or heights, the font and the height appear as **mixed**; typing a height changes the height of the text, and the pieces follow in the same proportion.

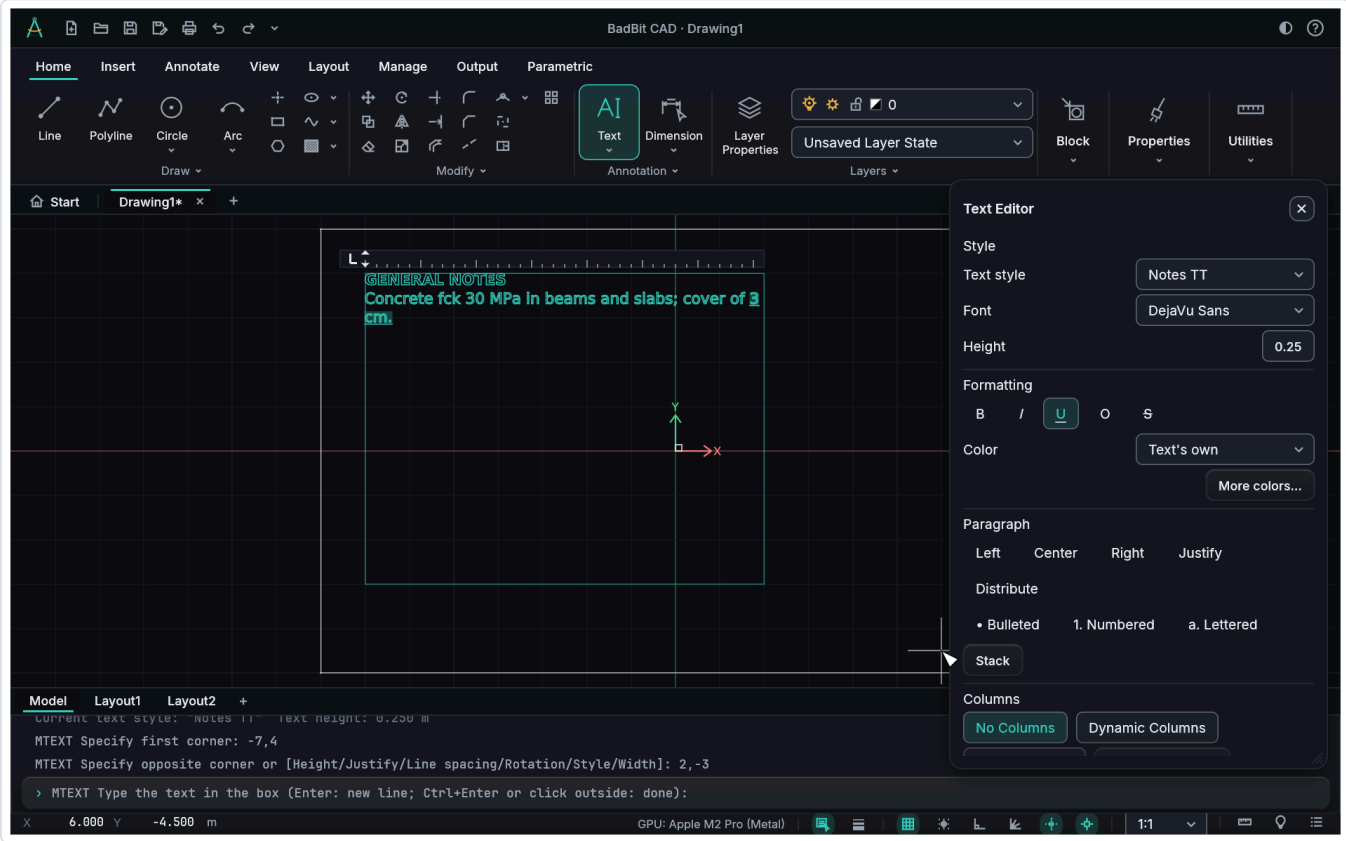
Formatting pieces of text: the Text Editor

While a multiline text is being edited in the drawing (MTEXT, TEXTEDIT of an MTEXT, a table cell, the text of a leader or of a multiline attribute), the **Text Editor** palette appears next to the drawing, like the contextual tab of the AutoCAD editor. Each change applies to the selected piece of text; with no selection, it applies to what is typed next (until the cursor moves).

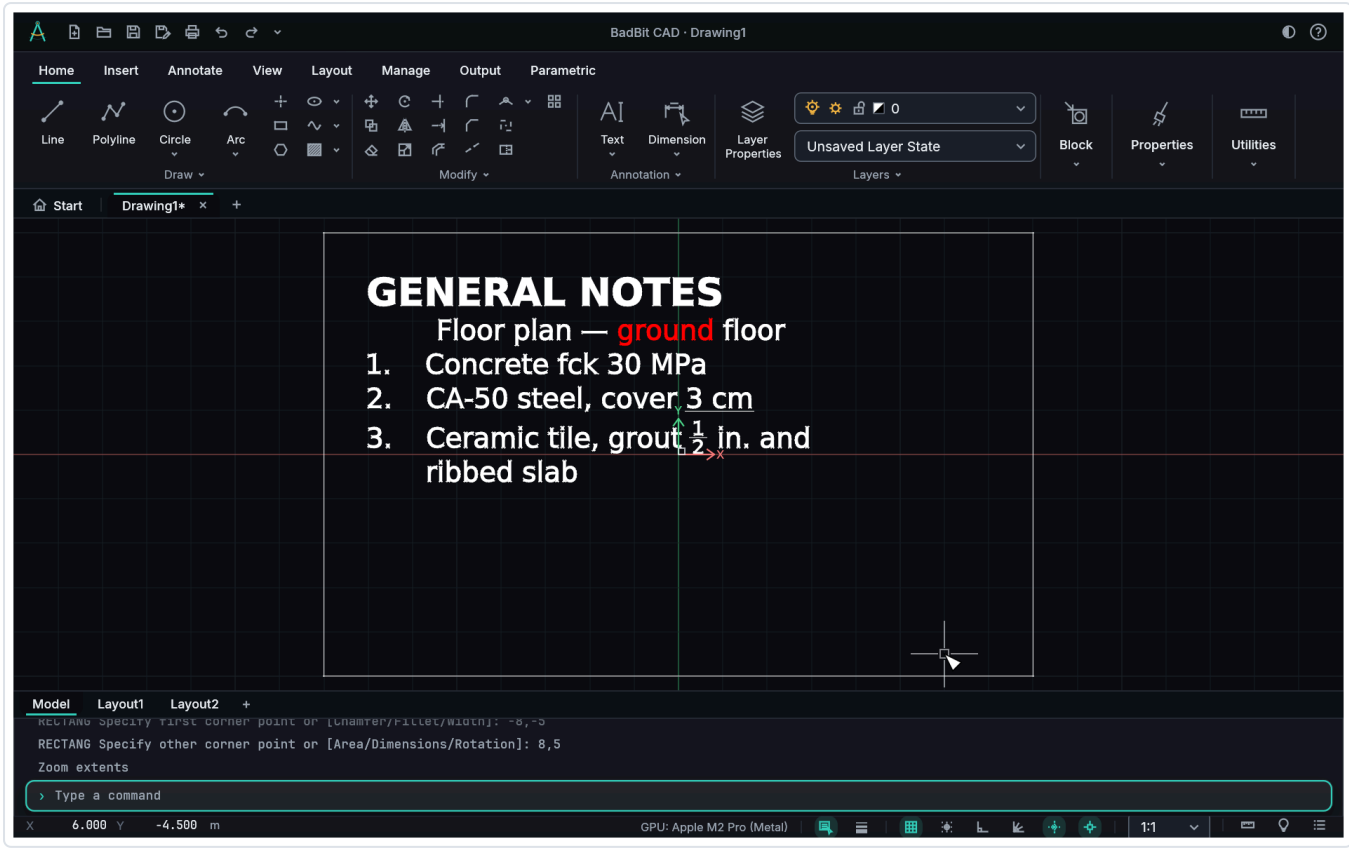
- **Style**: the text style of the whole MTEXT.
- **Font**: the font of the piece of text (the stroke font or an installed TrueType font).
- **Height**: the height of the letters of the piece; the line with larger letters becomes taller and farther from the one above (**At least** spacing), and letters of different heights stay on the same baseline.
- **B**, **I**, **U**, **O**, **S**: bold (`Ctrl` + `B`), italic (`Ctrl` + `I`), underline (`Ctrl` + `U`), overline and strikethrough; on the Mac, `Cmd`. The stroke font has no bold or italic, like AutoCAD SHX fonts.
- **Color**: the color of the piece (the text's own, ByLayer, ByBlock or AutoCAD colors 1 to 9).
- **Paragraph**: the alignment of the selected paragraphs (Left, Center, Right, Justify, Distribute; with none, the alignment point of the block applies) and **bulleted**, **numbered** and **lettered** lists (the bullet hangs in the indent and the numbering continues on its own).

- **Stack**: the selection `1/2` becomes a stacked fraction with a bar; `1#2`, a diagonal fraction; `1^2`, a tolerance (one above the other, without a bar). With the cursor next to a fraction, the same button unstacks it. The fraction moves like a single letter.
- **Symbol**: degrees, plus/minus, diameter, almost equal, angle, delta, not equal, identity, ohm, micro, squared, cubed, subscript 2 and the non-breaking space.
- **Find and Replace**: inside the text being edited, with **Find next**, **Replace** and **Replace all**; the replacement keeps the formatting of the replaced word.
- **Undo** and **Redo** (`Ctrl` + `Z` and `Ctrl` + `Y`) undo inside the editor, without leaving the text; the whole text is still a single undo step in the drawing.

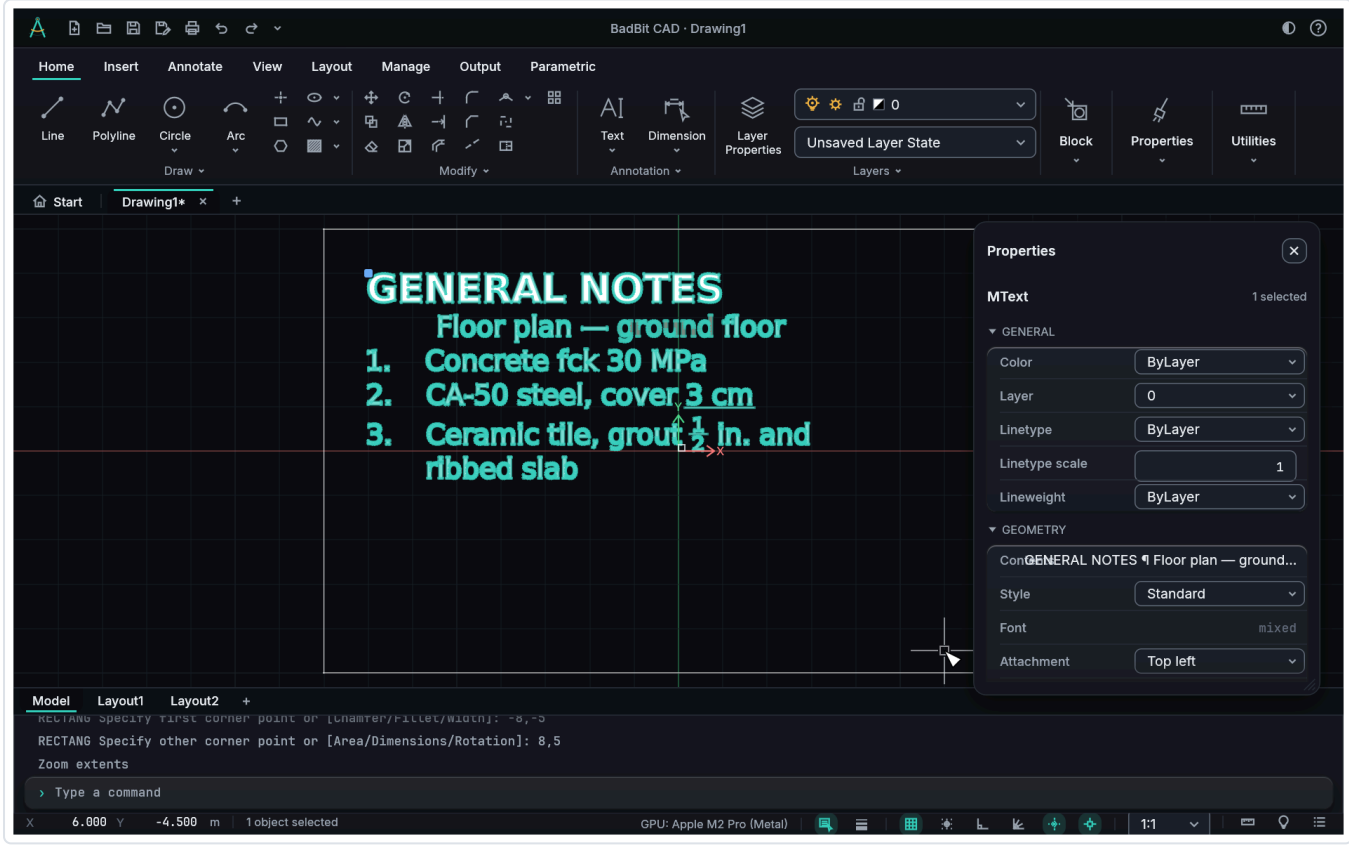
Copying and pasting within BadBit CAD carries the formatting (a bold piece stays bold); other programs receive plain text. The text is stored the way AutoCAD stores it, with the formatting codes (`{\fArial|b1;Título}`, `\H2x;`, `\C1;`, `\L`, `\S1/2;`, `\pxqc;`...): DXF and DWG open and save the same formatted text, and an AutoCAD file with formatted text opens with its formatting.



MTEXT being edited: the title in bold with Ctrl+B, the underlined piece selected and the Text Editor palette.



An MTEXT with a larger bold title, a centered paragraph with one word in red, a numbered list, underline, a stacked fraction and italic.



The same MTEXT selected: in Properties, the contents without the codes and the font as mixed.

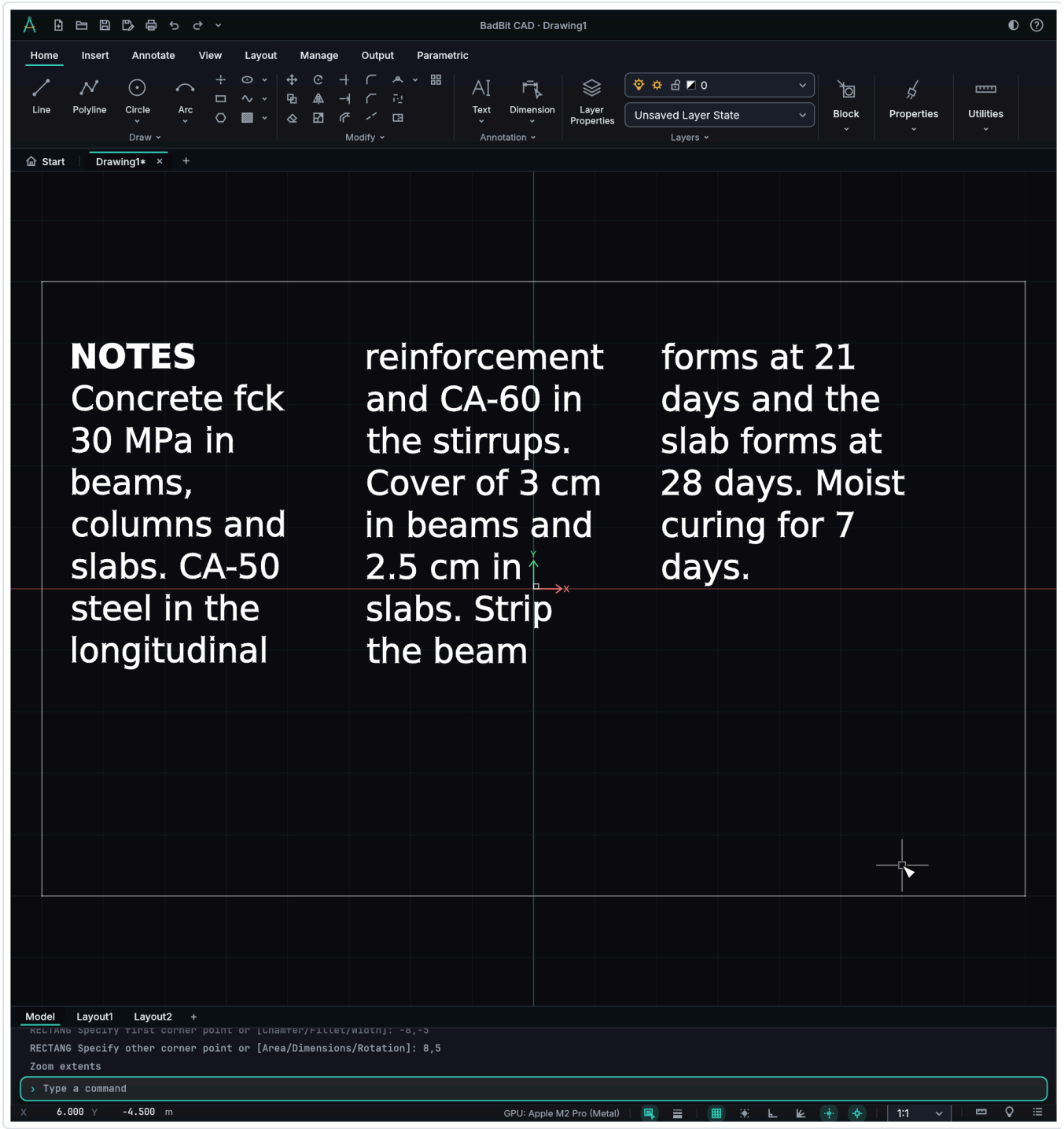
Columns

The text of an MTEXT can flow in **columns**, as in AutoCAD. In the Text Editor palette, the **Columns** group has:

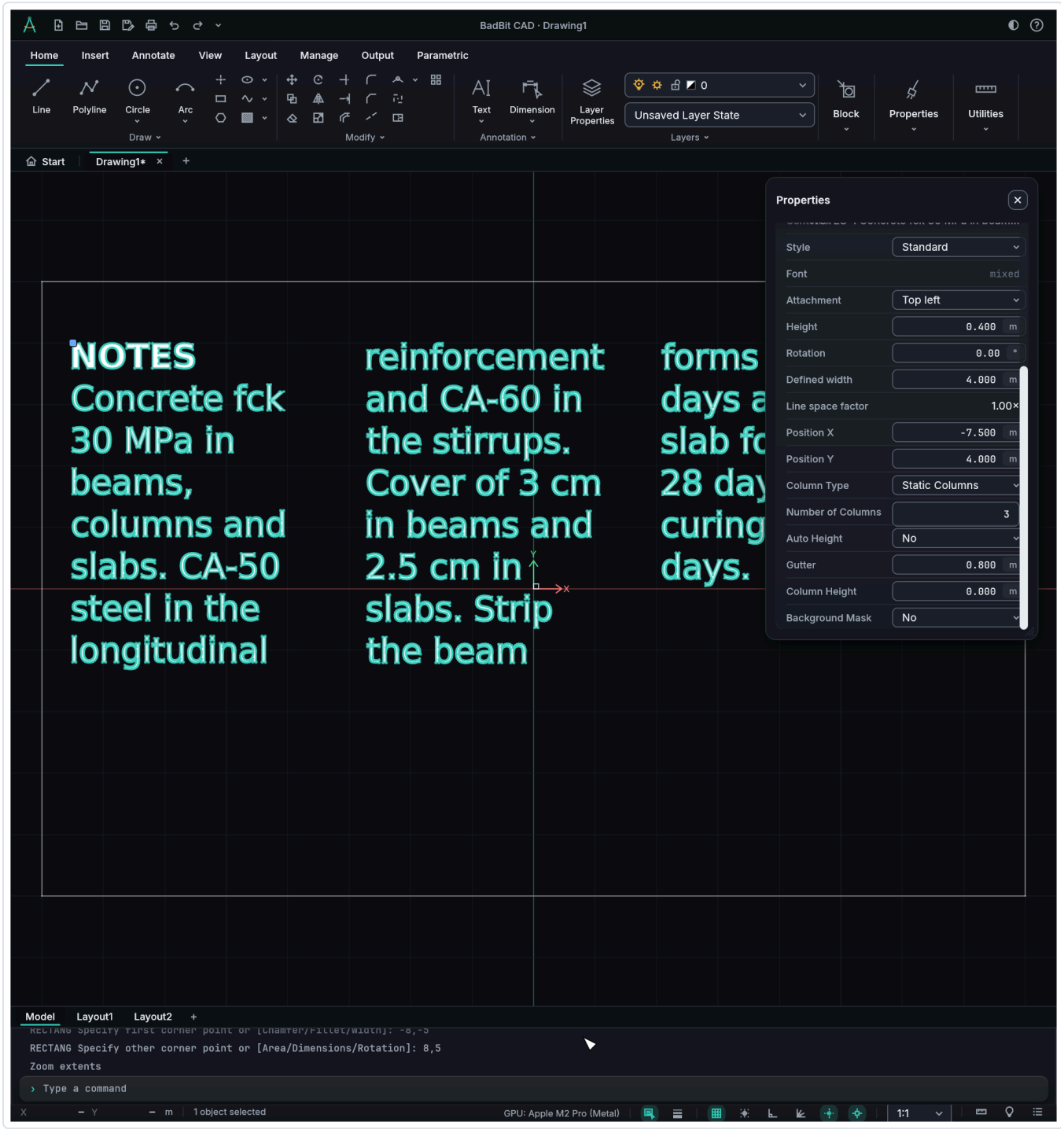
- **No Columns**: the text in a single block (the default).
- **Dynamic Columns** › **Auto Height** or **Manual Height**: columns appear as the text grows. With auto height, all of them have the height of the MTEXT rectangle (the defined height); when the text goes past it, it continues at the top of the next column. With manual height, each column keeps its own height.
- **Static Columns** › **2** to **6**: the number of columns is fixed and all have the same height; with no defined height, BadBit CAD divides the text equally among them. Text that does not fit stays in the last one.
- **Column Settings...**: the dialog with the **Column Type**, **Auto Height** (dynamic), **Number of Columns** (static), **Column Height**, **Column Width**, **Gutter** and **Total Width**.

Each column has the width of the MTEXT and the columns sit side by side with the gutter between them: two 10 m columns with a 2 m gutter start at 0 and at 12 m, and the block is 22 m wide. When columns are turned on, the gutter starts at five times the text height. The MTEXT **Properties** show the **Column Type**, **Number of Columns**, **Auto Height**, **Gutter** and **Column Height** (the column width is the **Width**), all editable.

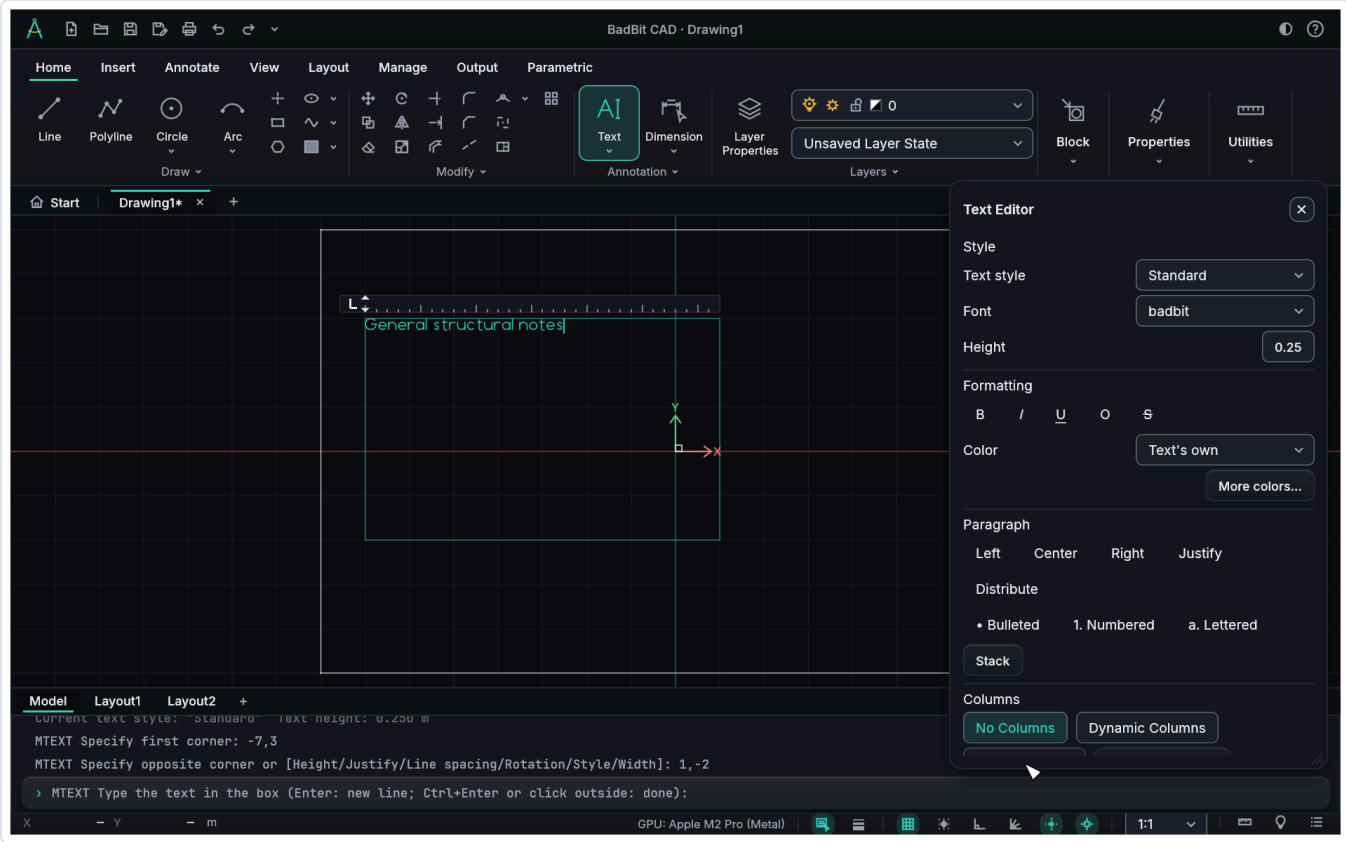
Columns go into the `.bbd`, DXF and DWG the way AutoCAD stores them: from AutoCAD 2018 onward, in the MTEXT itself; in versions 2000 to 2013, the main MTEXT carries the text of the first column and each following column is an MTEXT linked to it, which BadBit CAD joins back into a single MTEXT on opening.



An MTEXT in three static 4 m columns with 0.8 m between them: the text flows from one to the next.



The same MTEXT selected: Column Type, Number of Columns, Auto Height, Gutter, Column Height and Background Mask in Properties.



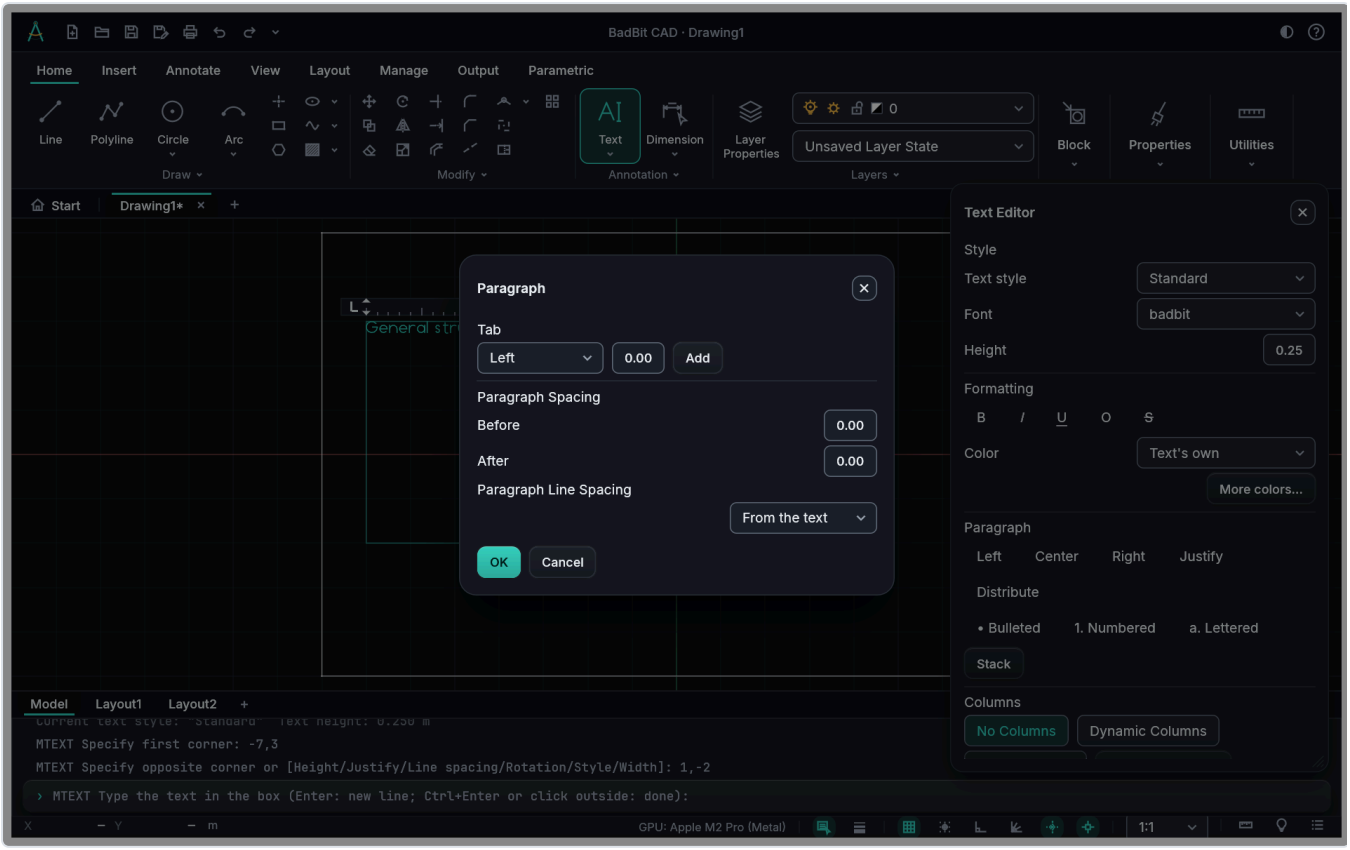
MTEXT being edited with the tab ruler on top and the Static Columns menu open in the Text Editor palette.

Paragraph: space before and after and line spacing

The **Paragraph...** button on the palette opens the **Paragraph** dialog, like the AutoCAD one, for the selected paragraphs (or the one at the cursor):

- **Tab**: the type (**Left**, **Center**, **Right**, **Decimal**), the **Position** and **Add**; the list of the paragraph's stops, each one with **Remove**.
- **Paragraph Spacing**: the space **Before** and **After** the paragraph. Between two paragraphs, the space is the after of the upper one plus the before of the lower one, added to the line spacing.
- **Paragraph Line Spacing**: **From the text** (that of the MTEXT), **Multiple** (times single spacing, which grows with the largest letter on the line), **Exactly** (the distance between baselines, always the same) or **At least** (the given distance, or more if the letters need it).

The distances in the dialog are in drawing units; in the text, they are stored in text heights, with the AutoCAD codes (`\pxb0.5,a1,se2;`): the paragraph follows the height when the text is scaled.



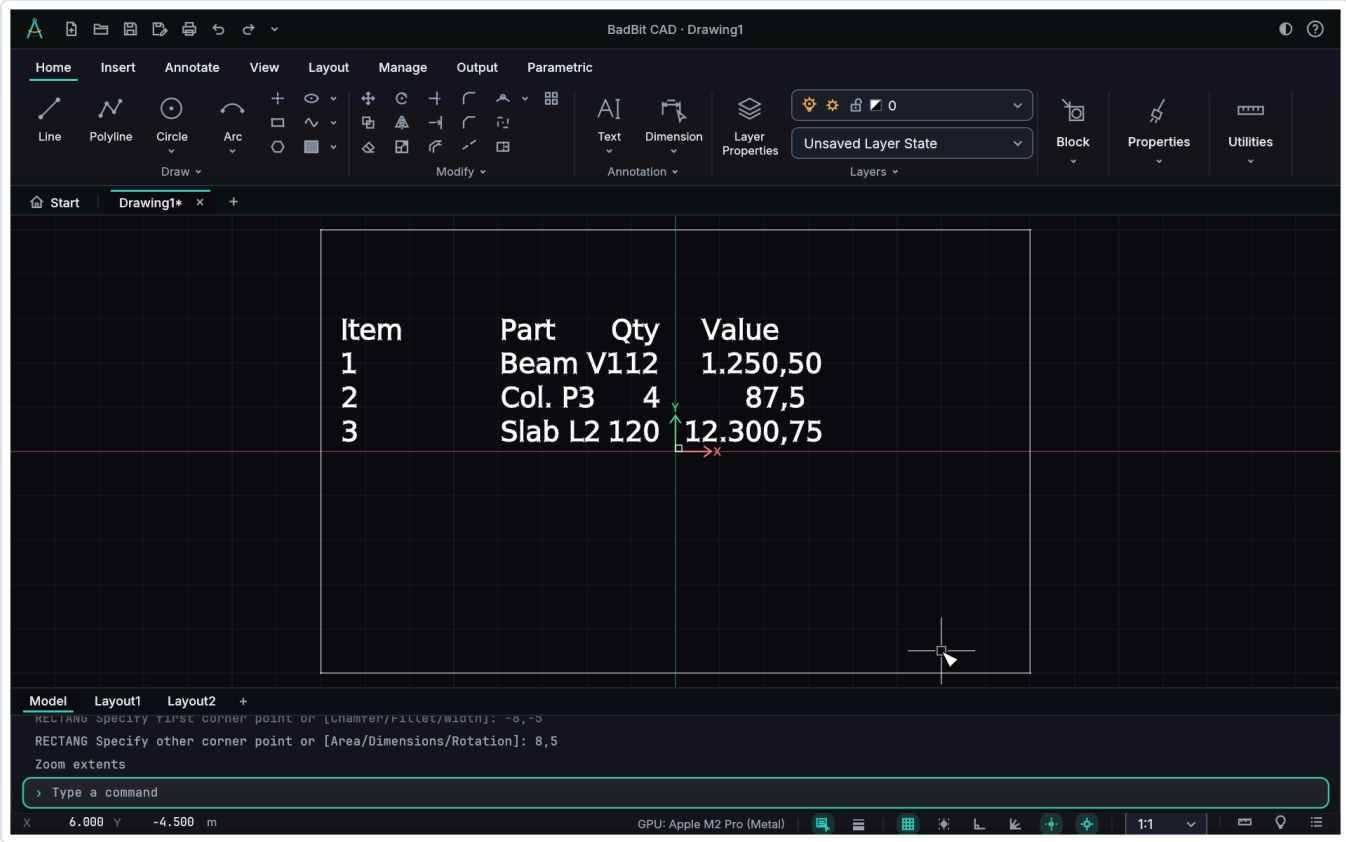
The Paragraph dialog: tabs, space before and after and the paragraph line spacing.

Tabs and the ruler

A tab (`Tab` in the text) takes what follows to the next stop of the paragraph; with no stops, there is one every four text heights. As in AutoCAD, a stop can be:

- **left**: the text starts at the stop;
- **center**: the text is centered on it;
- **right**: the text ends at it;
- **decimal**: the decimal separator of the number (the comma or the period, whichever comes first) sits at the stop; with no separator, like right.

While the MTEXT is being edited, the **tab ruler** appears above it, with a mark at every text height, the **indents** of the paragraph at the cursor (the upper triangle is the first-line indent; the lower one, the left indent) and the stops. The button to the left of the ruler shows the type of new stops and changes at each click (left, center, right, decimal). A click on the ruler creates a stop of that type there; dragging a stop moves it; dragging a stop off the ruler deletes it. Everything applies to the selected paragraphs and is undone with `Ctrl`+`Z` inside the editor.



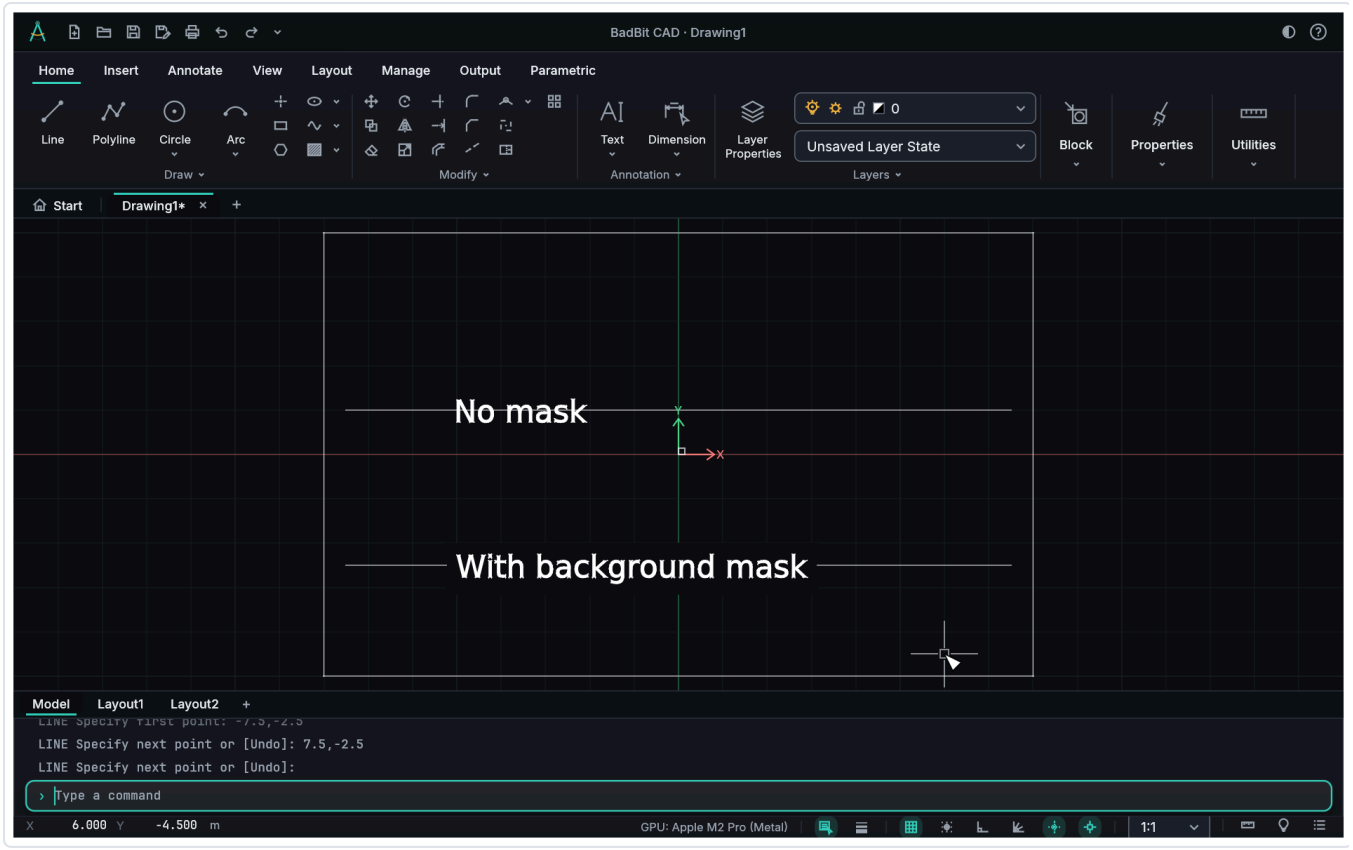
A simple table with tabs: the description on the left, the quantity on the right and the value with its comma aligned on the decimal stop.

Background mask

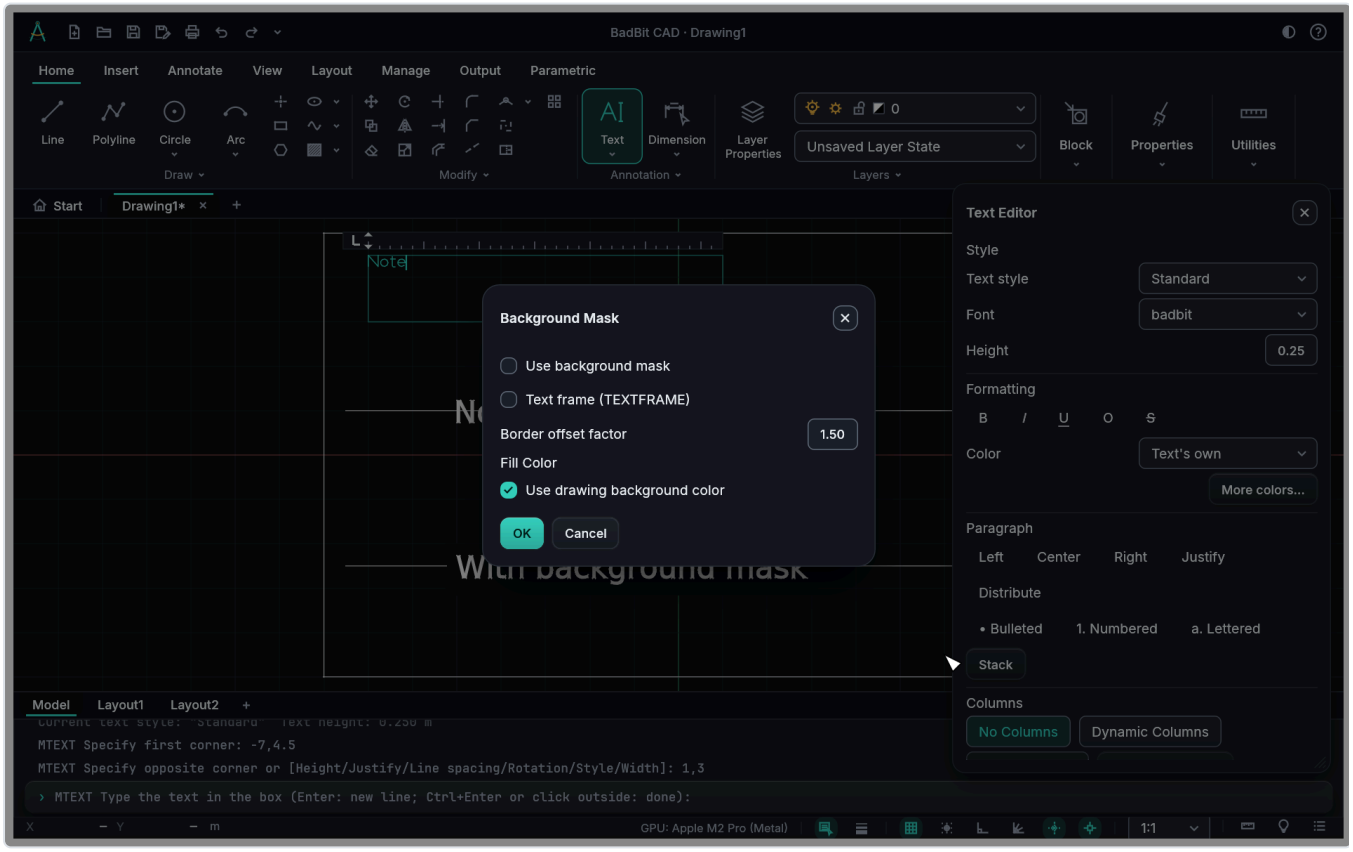
The **background mask** (AutoCAD's BACKGROUNDMASK) paints the MTEXT rectangle and hides what was drawn beneath it: a line that passes through the text disappears behind the letters. The **Background Mask...** button on the palette opens the dialog:

- **Use background mask**: turns it on and off;
- **Border offset factor**: from 1 to 5 (the default is 1.5); 1 is flush with the text block, and each additional unit widens the mask by one text height on each side (1.5 widens it by half a height);
- **Fill Color**: **Use drawing background color** (on screen, the background color of the drawing area; on a sheet, the white of the paper) or a color.

The mask appears on screen (GPU and CPU) and in plots, with the letters on top of it. In Properties, **Background Mask** turns it on and off. It goes into the `.bbd`, and into AutoCAD DXF and DWG from 2004 onward (earlier versions have no mask).



Two lines passing beneath two texts: without a mask, the line crosses the letters; with the background mask, it disappears behind the text.



The Background Mask dialog: use, border offset factor and fill color.

TEXT (DTEXT, alias DT)

Single-line text, like the AutoCAD TEXT:

1. **Start point** of the text (the justification point, if it is not Left);
2. **Height** (typed or pointed from the point; `Enter` repeats the last one);
3. **Rotation angle** (`Enter` repeats the last one);
4. **The text**, typed **directly in the drawing**, as in AutoCAD: the cursor blinks in the text itself, the arrow keys, `Home` and `End` move through it (with `Shift`, they select), `Backspace` and `Delete` erase, `Ctrl` + `A` (`Cmd` on the Mac) selects everything, and copy, cut and paste work. Here the space is a letter; it does not confirm. `Enter` creates the line and moves down to the next one; `Enter` on an empty line ends; a click outside the text also confirms the line. A U undoes all the lines of the command.

Options before the point:

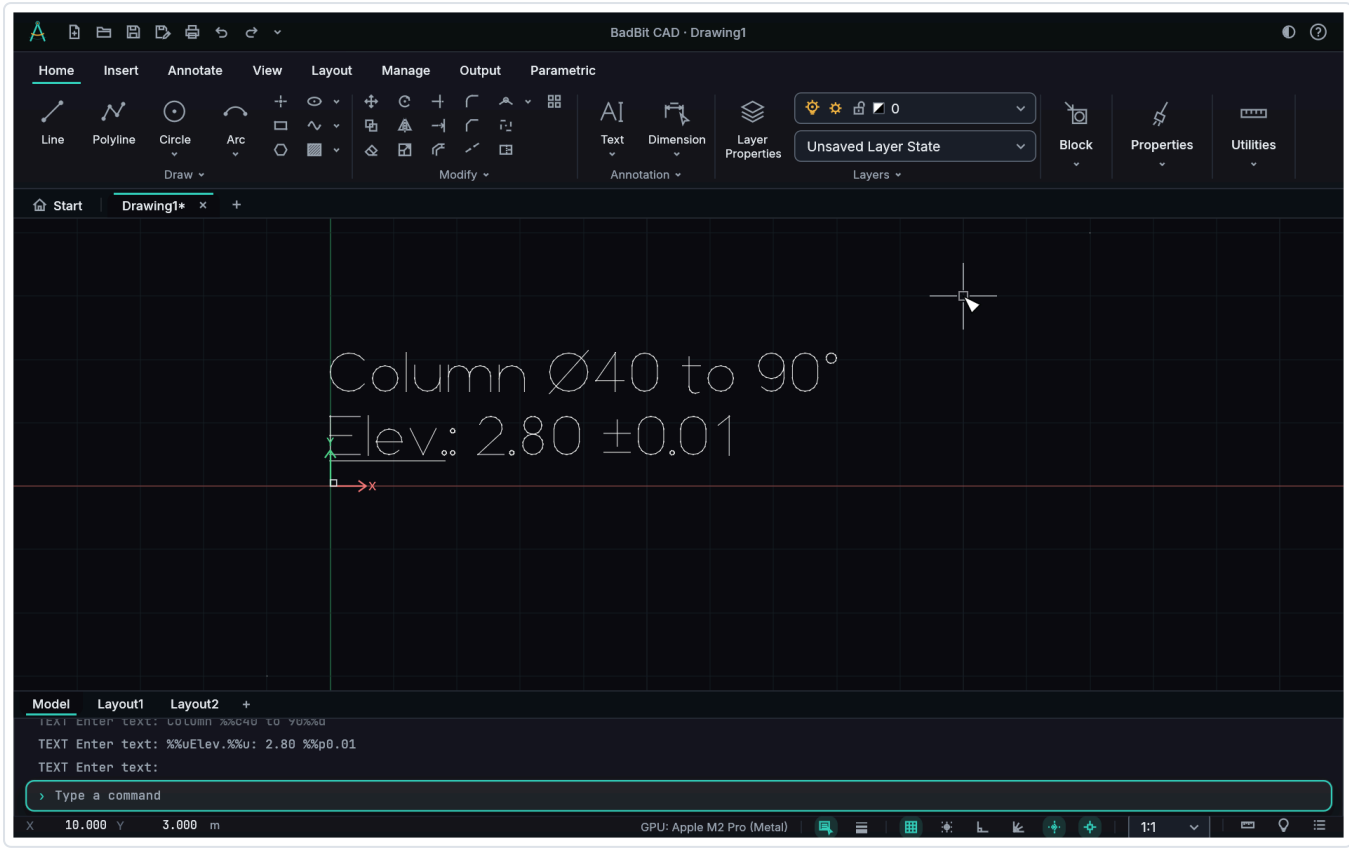
- **Justify** (`J`): Left, Center, Right, Middle and the nine points of the box (TL, TC, TR, ML, MC, MR, BL, BC, BR: top, middle and bottom; left, center and right). With **Align**, the text goes between two points and the height changes to fit; with **Fit**, also between two points, but with the given height and the width adjusted.
- **Style** (`S`): the name of the text style (`?` lists the styles in the drawing, with the font, height, width factor and obliquing angle of each one); it becomes the current one. With a fixed-height style, TEXT does not ask for the height: the rotation comes after the point.

The height, rotation and justification are remembered for the next text. `Enter` instead of the start point **continues below the last text**, as in AutoCAD: the command goes straight to typing, one line below, with the same height, rotation and justification.

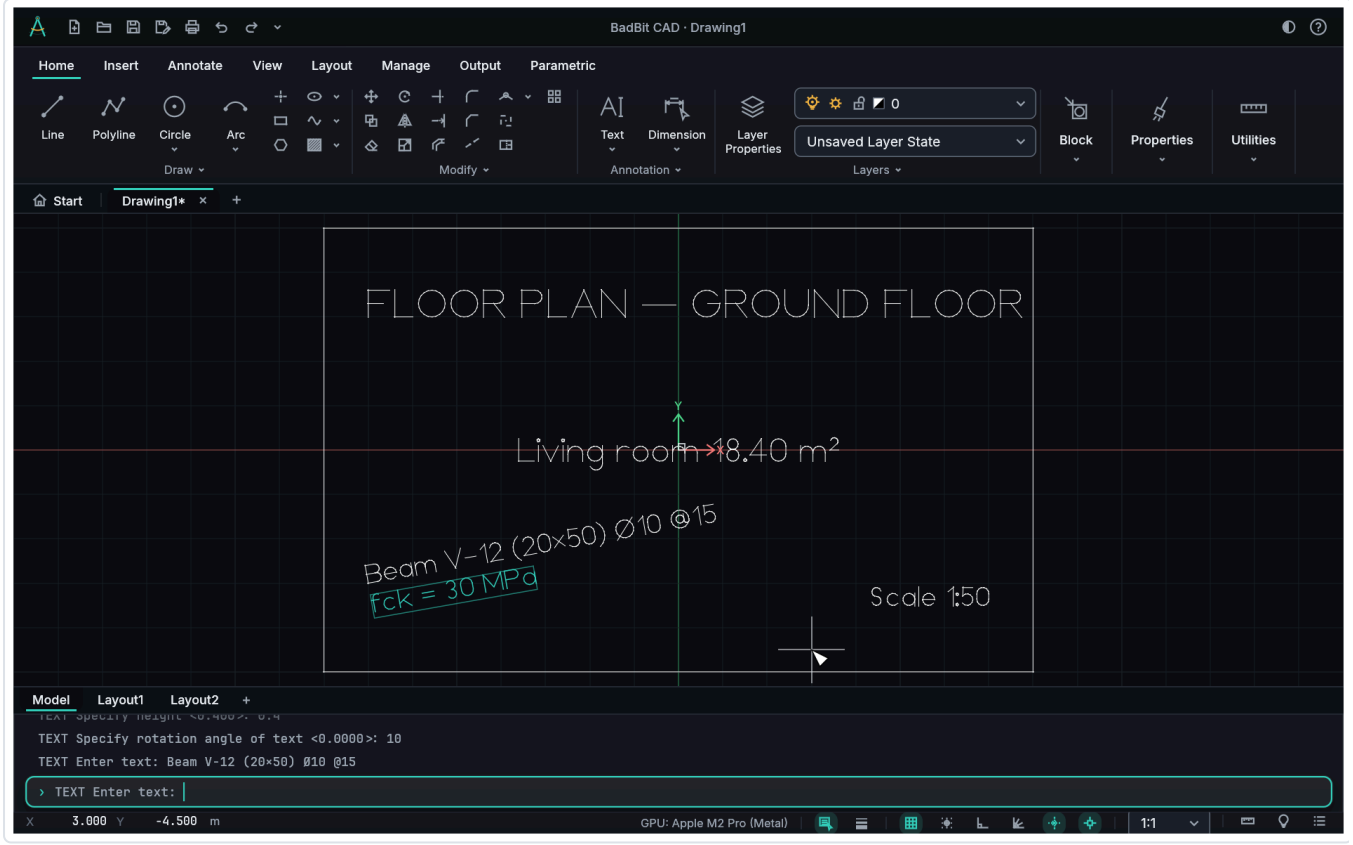
AutoCAD **control codes**, typed in the text (in TEXT, MTEXT and TEXTEDIT):

Code	Becomes
`%%d`	° (degree)
`%%p`	± (plus/minus)
`%%c`	⌀ (diameter)
`%%%`	%
`%%nnn`	the character with code `nnn` (for example, `%%065` is A)
`%%u`	turns underline on and off (TEXT only)
`%%o`	turns overline on and off (TEXT only)

The symbols replace the code when the line is confirmed; underline and overline appear while you type.



TEXT with `%%c`, `%%d`, `%%p` and `%%u`: the diameter, the degree, the plus/minus and the underline.



TEXT: texts with different justifications and a line being typed, which appears in the drawing.

Text has an insertion snap (the justification point). It is selected by clicking inside its box, goes into a window when the whole letters are inside (the box of the letters, not of the font) and is caught by a crossing that touches the outline of any letter, as in AutoCAD. MOVE, COPY, ROTATE, SCALE, ARRAY and STRETCH (by the point) work on it. MIRROR follows the MIRRTEXT variable (see MIRROR). The grip is at the justification point (and at the second point of Align and Fit). The properties palette and Quick Properties show the contents, the style, the justification, the height, the rotation, the width factor, the obliquing and whether the text is upside down; the palette changes the style (the list of styles in the drawing), the obliquing (up to 85° each way), the width factor and upside down, as in AutoCAD. Changing a text's style brings in the width factor and obliquing of the new style and, if it has a fixed height, the height as well. Changing the justification there leaves the text in place and moves the point, as in AutoCAD.

Single-line text also takes part in parametric drawing, as in AutoCAD: geometric constraints use its **baseline** (parallel, perpendicular, collinear, horizontal, vertical) and its **text point** (the justification point; coincident, fix). The text does not stretch: held by the baseline, it rotates around the point; held by the point, it moves along without rotating (see the parametric drawing chapter).

On the ribbon, Single Line Text is under the small arrow of the **Text** button on the **Annotation** panel of the Home tab.

STYLE (alias ST)

Text styles, as in AutoCAD: each text and each multiline text uses a style, which gives the **font**, the **height** (0 lets TEXT ask; greater than 0 fixes the height of new texts), the **width factor** (below 1 narrows the letters, above 1 widens them; from 0.01 to 100), the **obliquing angle** (the slant, up to 85° each way) and the **Upside down** and **Backwards** effects (single-line text only, as in AutoCAD). Every drawing has the **Standard** style, which cannot be renamed or deleted; an old drawing, from before styles existed, opens with it.

The **font** is BadBit CAD's own stroke font (`badbit`) or a TrueType/OpenType font installed on the computer: BadBit CAD looks in the system font folders (on macOS, Windows and Linux) and in the `fonts` folder next to the program. For TrueType fonts, the **Font Style** chooses **Regular**, **Bold**, **Italic** or **Bold Italic** (only those the family has); for the stroke font it is disabled. Height, width factor, obliquing angle, upside down and backwards work the same way in both.

A drawing made on another computer may use a font that is not installed on this one. As in AutoCAD, BadBit CAD draws those texts with a similar font (a common sans-serif one, such as Arial; with no TrueType at all, the stroke font) and warns **once** on the command line: `Font "Font X" not found: using "Arial" instead.` The name of the original font stays stored in the style and in the file: opened again on a computer that has the font, the text is shown in it again. The Text Style dialog shows the missing font in the list and says which one is being used.

The command opens the **Text Style** dialog:

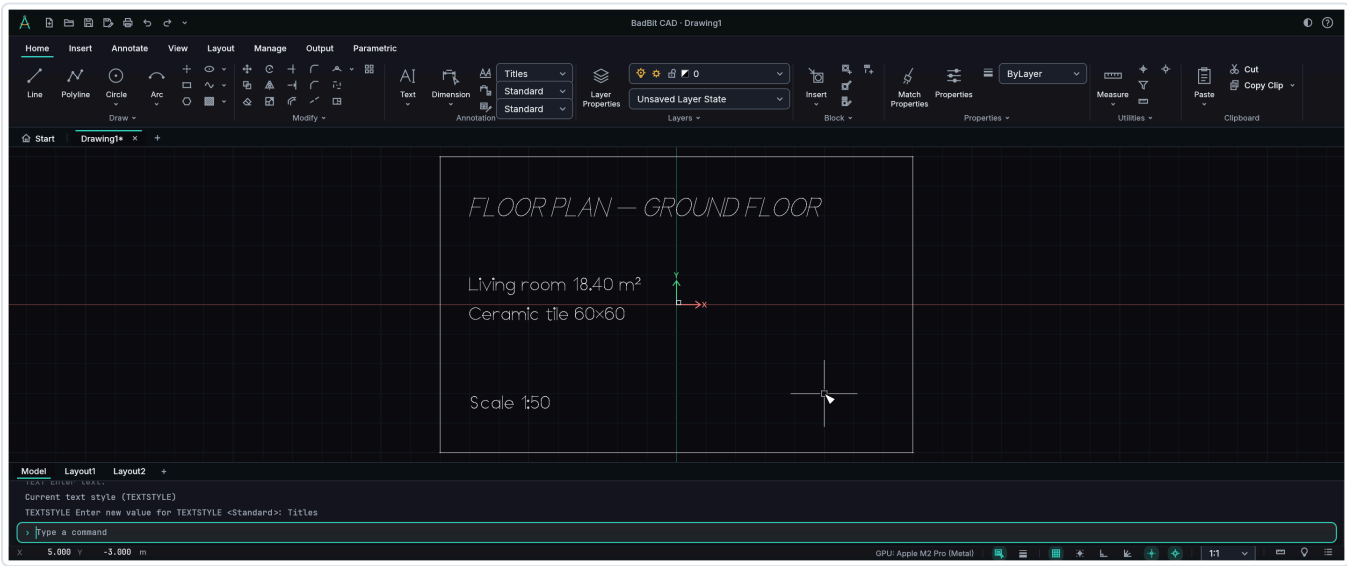
- on the left, the list of the drawing's **Styles**; a click marks the style and shows its values (right-clicking a name gives **Rename**). At the top, the current style;

- in the middle, the **Font** (the **Font Name**, with the stroke font and the installed TrueType fonts in alphabetical order, and the **Font Style**), the **Size** (the height) and the **Effects**, with the **Preview** below, which changes as you make adjustments (filled, for TrueType);
- on the right, **Set Current** (the marked style becomes the style of the next texts, the **TEXTSTYLE** variable), **New...** (asks for the name, suggesting `style1`, `style2`..., and creates it from the marked one) and **Delete**. Delete does not remove Standard, the current style or a style used by any text, and the dialog says why;
- **Apply** saves the changes to the style and **updates the texts that use the style** (the font, the width factor and the obliquing; the fixed height and the effects apply to new texts) and the dimensions whose dimension style uses that text style, all in a single undo step. Closing the dialog, or marking another style, with changes still to apply asks whether to save them.

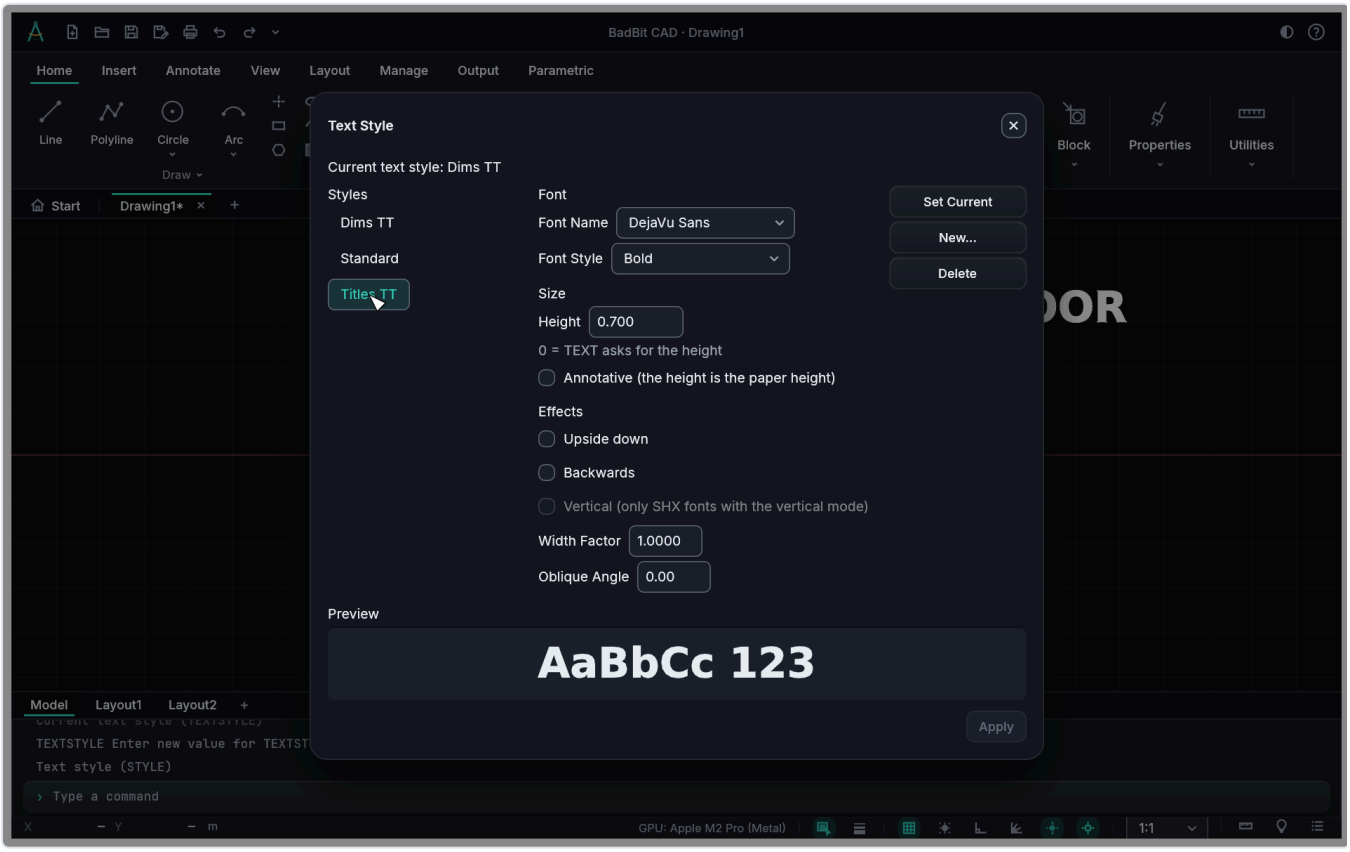
Creating, changing and deleting a style are each an undo step (`U` or `Ctrl` + `Z`). The current style belongs to the drawing and is saved in the file, as in AutoCAD; it can also be changed with the **TEXTSTYLE** variable (`TEXTSTYLE` and the name) or with the **Current text style** list on the **Annotation** panel of the ribbon, next to the Text button. With a narrow window, the panel becomes compact and the list goes away; STYLE remains on the open panel (the small arrow of the **Annotation** title).



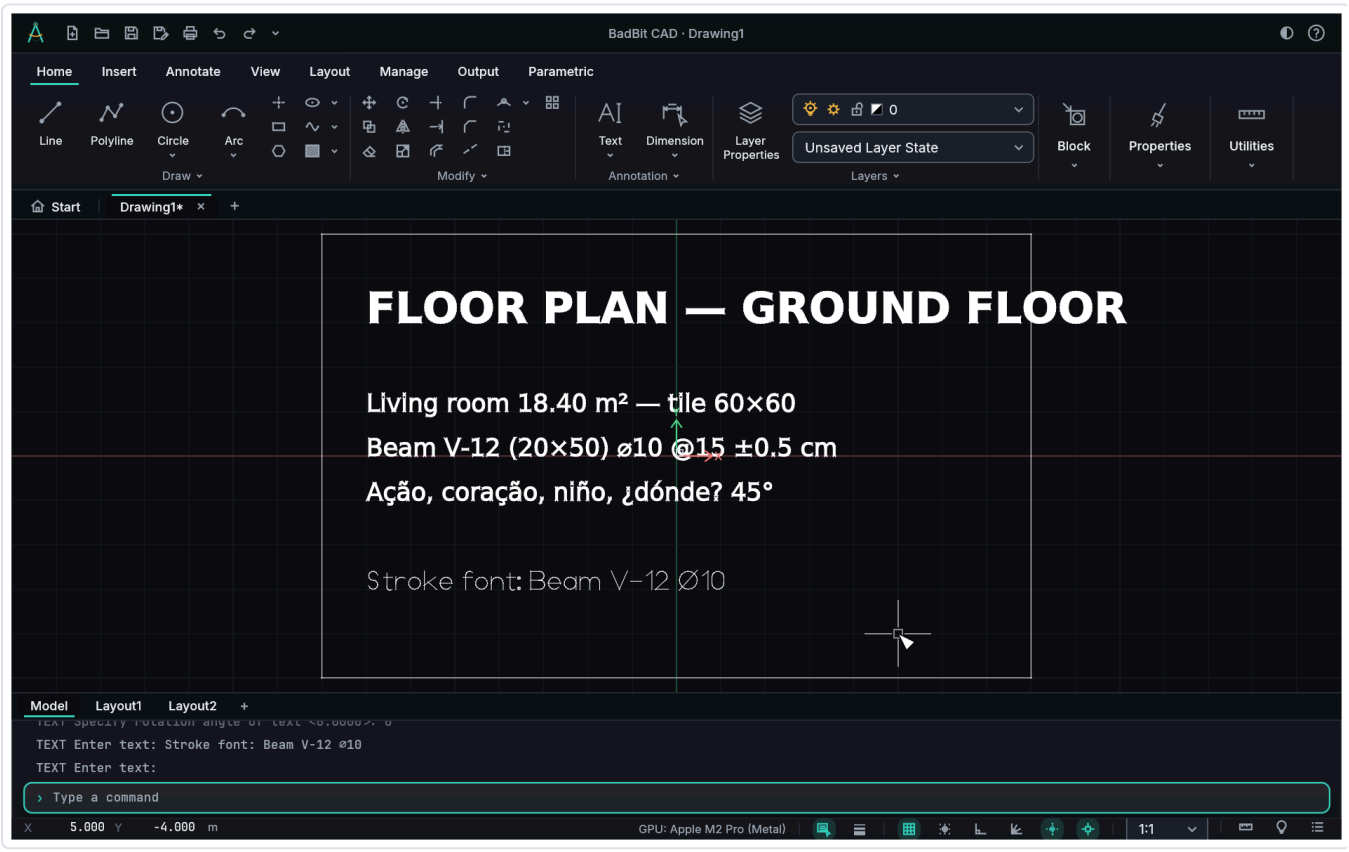
The Text Style dialog with the Titles style marked: fixed height, width factor 0.8 and 15° obliquing, and the preview.



Texts in three styles: Titles (narrow and slanted), Notes and Standard; on the ribbon, the current style list.



The Titles TT style with the DejaVu Sans font in Bold: the preview comes out filled.



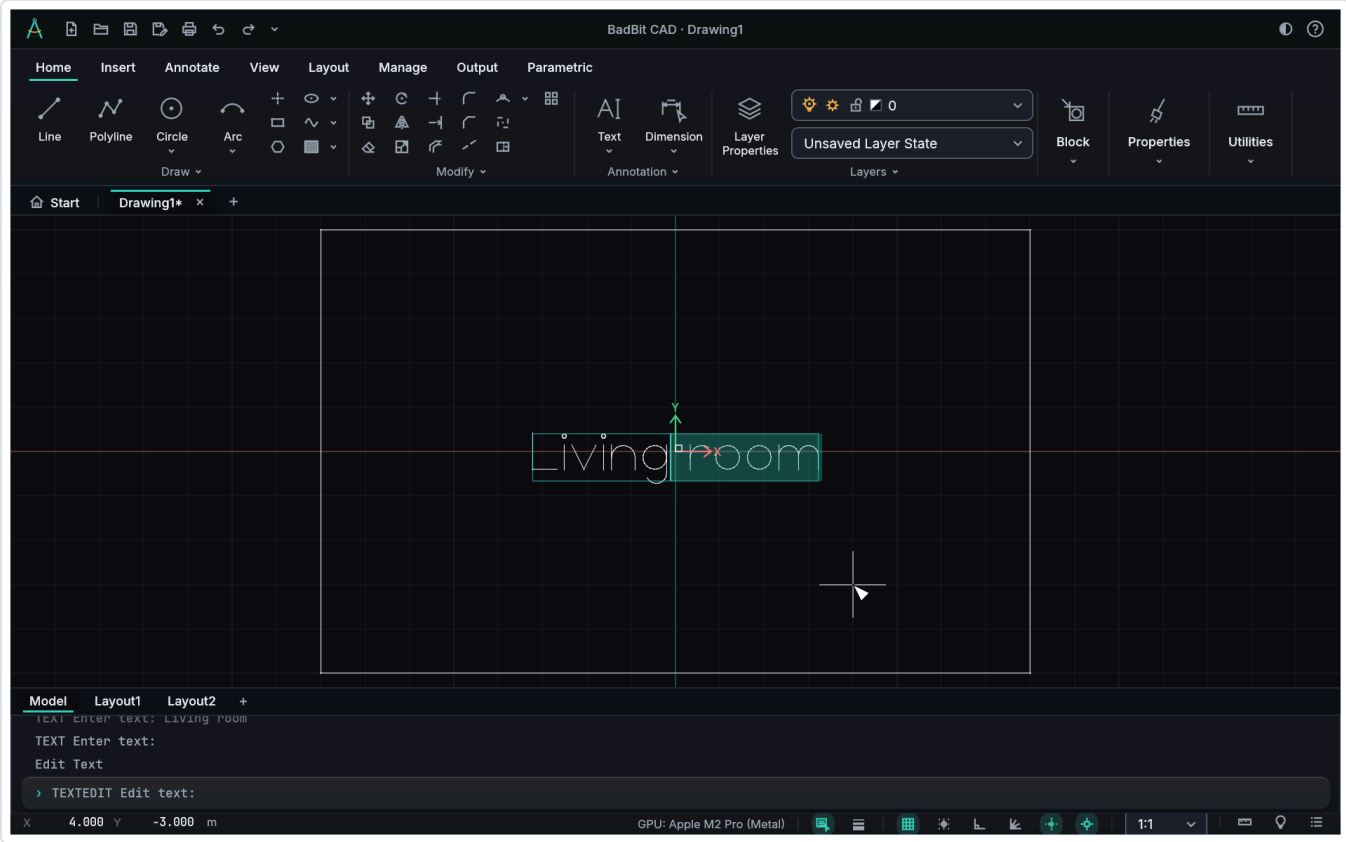
Filled TrueType texts (DejaVu Sans regular and bold) next to the stroke one, with accents and symbols.

-STYLE (alias `-ST`) does the same through the command line, as in AutoCAD: the style name (a new name creates the style from the current one; `?` lists), the font, the height, the width factor, the obliquing angle, **Backwards?** and **Upside down?** [Yes (`Y`)/No (`N`)]. `Enter` accepts the value between `< >`; at the end, the style becomes the current one. The font is typed by the family name (`Arial`), by the family with the style at the end (`Arial Bold`, `Arial Bold Italic`; in Portuguese, `Arial Negrito Itálico` also works) or by the file (`arialbd.ttf`), as in AutoCAD; `badbit` is the stroke font. A font installed while BadBit CAD is open works without restarting: when the name is not found, -STYLE searches the font folders again. A font that is not installed is refused with the message **font not found** and the prompt continues.

A TEXT with a TrueType font is selected by clicking inside its box, like any text, and has the **Insertion** snap at its point. **EXPLODE** turns a TrueType TEXT into the outlines of its letters, as closed polylines (like TXTEXP from the AutoCAD Express Tools; the inside of the "O" becomes a separate polyline); one with the stroke font does not explode, like TEXT in AutoCAD. MTEXT explodes, as in AutoCAD, into one TEXT per piece of each line, with the font, height and color of the piece (unformatted text gives one TEXT per line); underlines and fraction bars become lines. A justified paragraph comes out without the stretched spaces, each line aligned by the alignment point of the MTEXT, and a distributed one comes out centered, also as in AutoCAD.

TEXTEDIT (alias ED, DDEDIT) and double-click

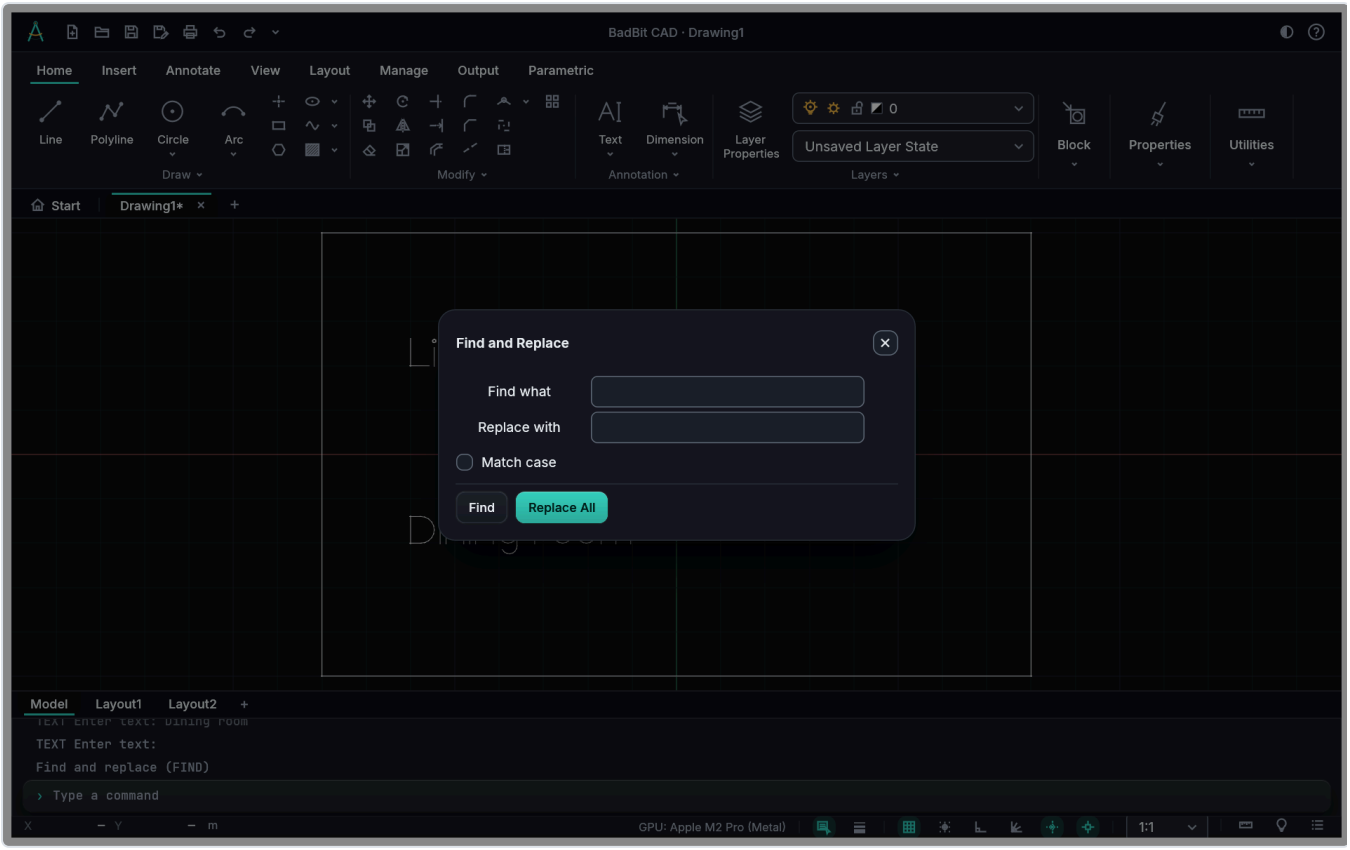
To change what is written, **double-click** the text (with no command active), or use TEXTEDIT and click it. Editing happens **in the drawing itself**, as in AutoCAD: the text comes in fully selected (typing replaces everything) and the cursor blinks in it; the same keys as in TEXT apply here, and a click inside the text puts the cursor there. `Enter` confirms, and the command asks for another text (`Enter` again ends). `Esc` cancels: the text goes back to what it was. A click outside the text also confirms. Each edited text is a single undo step; **Undo** (`U`, or `D`), at the command prompt, removes the last edit. With a text already selected, TEXTEDIT goes straight into it.



Editing in the drawing: the text box, the selection (the last letters) and the cursor.

Find and replace (FIND)

The **FIND** command opens the find and replace dialog, like the AutoCAD one: the string to find, the replacement text and whether case matters. **Find** selects the TEXTs and MTEXTs that contain the string and frames all of them on the screen; **Replace All** replaces the string in all of them at once, in a single undo step. In formatted MTEXT, the search is in the running text (formatting codes do not count) and the replacement keeps the word's formatting. Accents count (searching for "acao" does not find "ação").

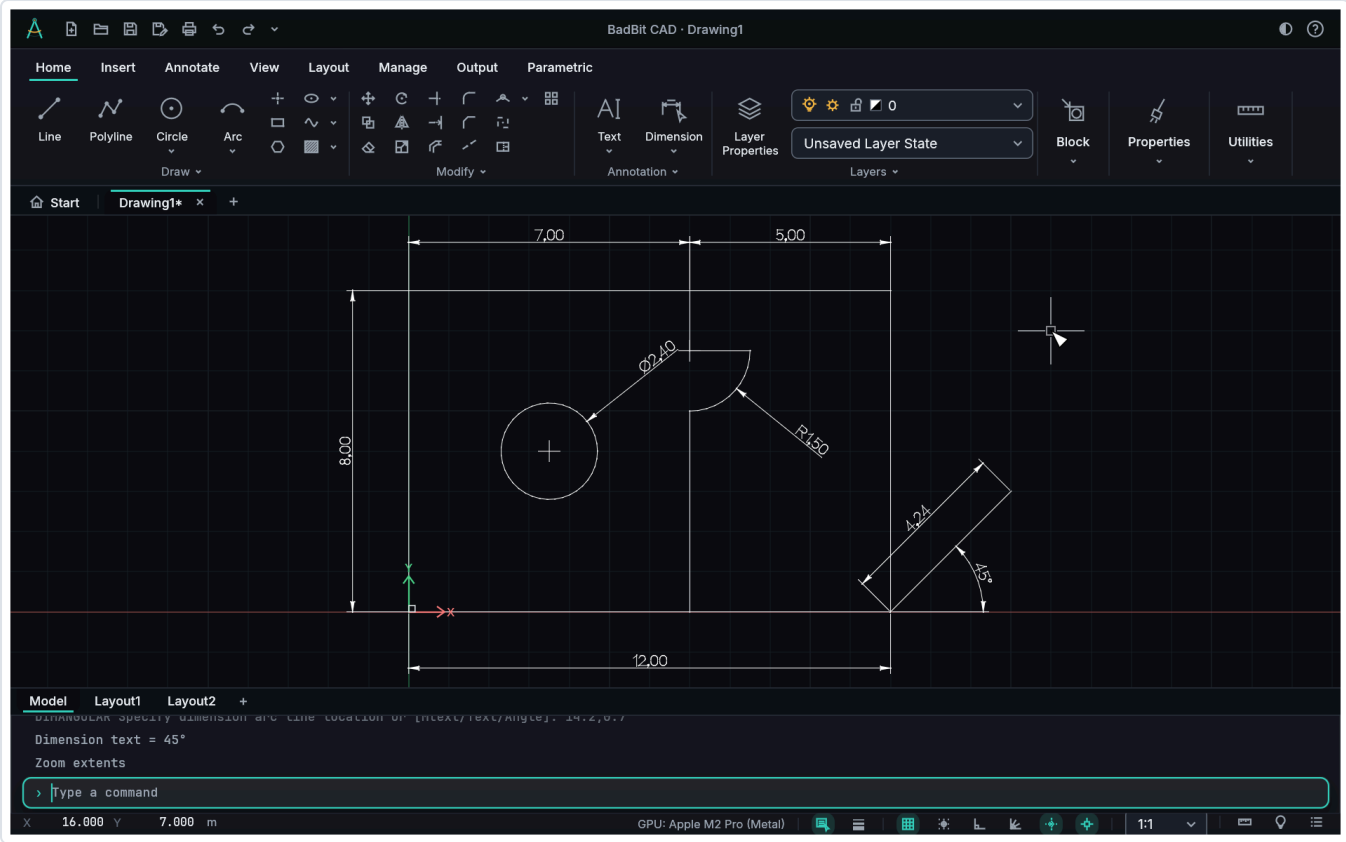


The FIND dialog: the string, the replacement and the options.

Dimensions

BadBit CAD dimensions follow those of AutoCAD: the same commands, aliases, prompts and options (the option letters are the AutoCAD ones: `Mtext`, `Text`, `Angle`…; in the Portuguese and Spanish interface they appear in parentheses after the translated word, as in `Texto (T)`). A dimension is **a single object**, which stores the definition points (the origins of the extension lines, a point on the dimension line and, if you moved it, the location of the text), the dimension style and the text. Everything you see is derived from that each time it is drawn: the **extension lines** (which start slightly away from the origin and go slightly past the dimension line), the **dimension line**, the **arrowheads** and the **text** with the measurement, in the text style of the dimension style, centered above the line and aligned with it. Moving, copying, rotating, scaling and mirroring the dimension move its points, and the measurement follows (scaling changes the measurement; the text and arrowheads stay the size set by the style, as in AutoCAD). When the text does not fit between the extension lines, it goes outside, past the second one; when the arrowheads do not fit, they flip outside.

The drawing is in meters: with the **Standard** style, `3` is written `3,00`. During each command, the dimension appears in the **preview** following the cursor until the location is clicked; afterwards, the command line states its text (`Dimension text = 3,00`). Each dimension is an undo step (in DIMCONTINUE, DIMBASELINE and DIM, the whole command is a single step, and the **Undo** option (`U`) removes the last dimension of the command).



A dimensioned floor plan: horizontal linear dimensions (with a continued one at the top), the vertical one, the diameter of the table, the radius of the door and, on the ramp, the aligned and the angular ones.

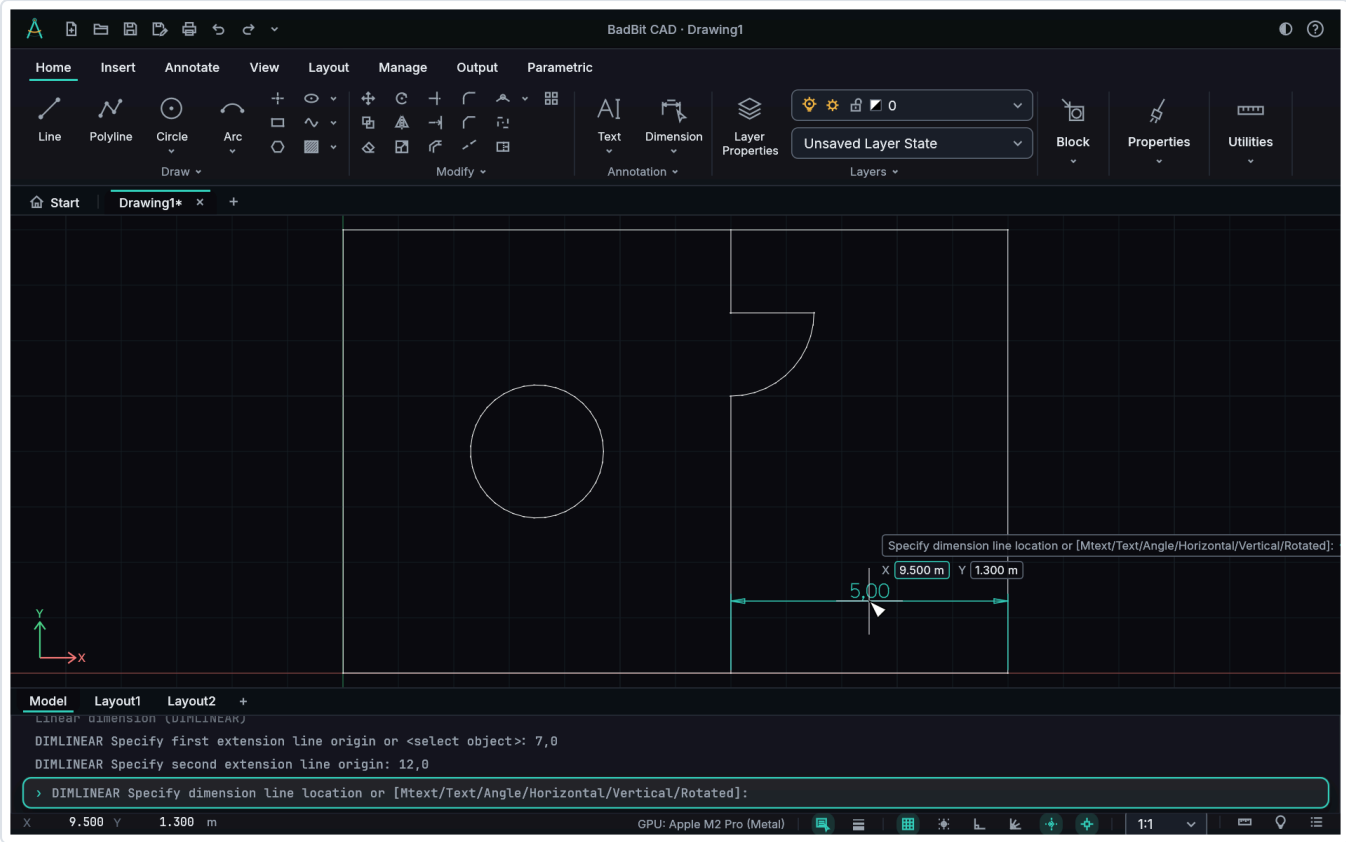
DIMLINEAR (alias DLI)

The linear dimension: horizontal, vertical or rotated.

1. **First extension line origin** or `Enter` to **select the object** (a line, an arc, a circle or a polyline segment gives both origins; on a circle, the ends of the diameter in the direction of the dimension);
2. **Second extension line origin**;
3. **Dimension line location**. As in AutoCAD, the cursor position chooses the direction: above or below the origins, the dimension is horizontal (it measures Δx); to the side, vertical (it measures Δy).

At the location prompt, these options apply:

- **Mtext** (`M`) and **Text** (`T`): the dimension text. `<>` inside it is the measurement (`<> typ.` gives `3,00 typ.`); `Enter` keeps the measurement; a text without `<>` replaces the measurement; a single space leaves the dimension with no text. The codes `%%c` (⌀), `%%d` (°) and `%%p` (±) work;
- **Angle** (`A`): the angle of the text;
- **Horizontal** (`H`) and **Vertical** (`V`): fix the direction, wherever the cursor is;
- **Rotated** (`R`): the angle of the dimension line (the dimension measures in its direction).



DIMLINEAR after the two origins: the dimension follows the cursor until the location is clicked.

DIMALIGNED (alias DAL)

The aligned dimension: the same prompts as DIMLINEAR (with `Enter` to select the object), but the dimension line stays parallel to the two origins and measures the distance between them. Options: **Mtext** (`M`), **Text** (`T`) and **Angle** (`A`).

DIMRADIUS (alias DRA) and DIMDIAMETER (alias DDI)

Select arc or circle (or an arc segment of a polyline) and the **dimension line location**, with the **Mtext**, **Text** and **Angle** options. The measurement comes out with the AutoCAD prefix: `R` for the radius (`R1,50`) and `⌀` for the diameter (`⌀2,40`). With the cursor outside the curve, the leader runs from the arrowhead, on the curve, to the text, and the center gets the center mark (its size is the style's); inside, the line passes through the center (the diameter has both arrowheads).

DIMANGULAR (alias DAN)

Select arc, circle, line, or <specify vertex>:

- **two lines** (or straight polyline segments): the angle between them. The two lines form four sectors; the one on the side where you place the dimension arc is used (the same pair gives 30° or 150°);
- **an arc**: its angle, with the vertex at the center;
- **a circle**: the clicked point is the first endpoint; then, the second;
- `Enter`: the three points, **vertex**, **first** and **second endpoint**; the side of the dimension arc chooses the angle or what is left to 360°.

Then the **dimension arc line location**, with **Mtext**, **Text** and **Angle**. The angle comes out in degrees (`90°`), at the style's angular precision.

DIMARC (alias DAR)

The length of an arc (or of an arc segment of a polyline): **select the arc** and the **dimension location**. The extension lines point toward the center and the dimension arc follows the arc; the measurement comes with the arc symbol in front (`⌒1,57` on an arc of radius 1 and 90°, which measures π/2).

At the location prompt, besides **Mtext**, **Text** and **Angle**:

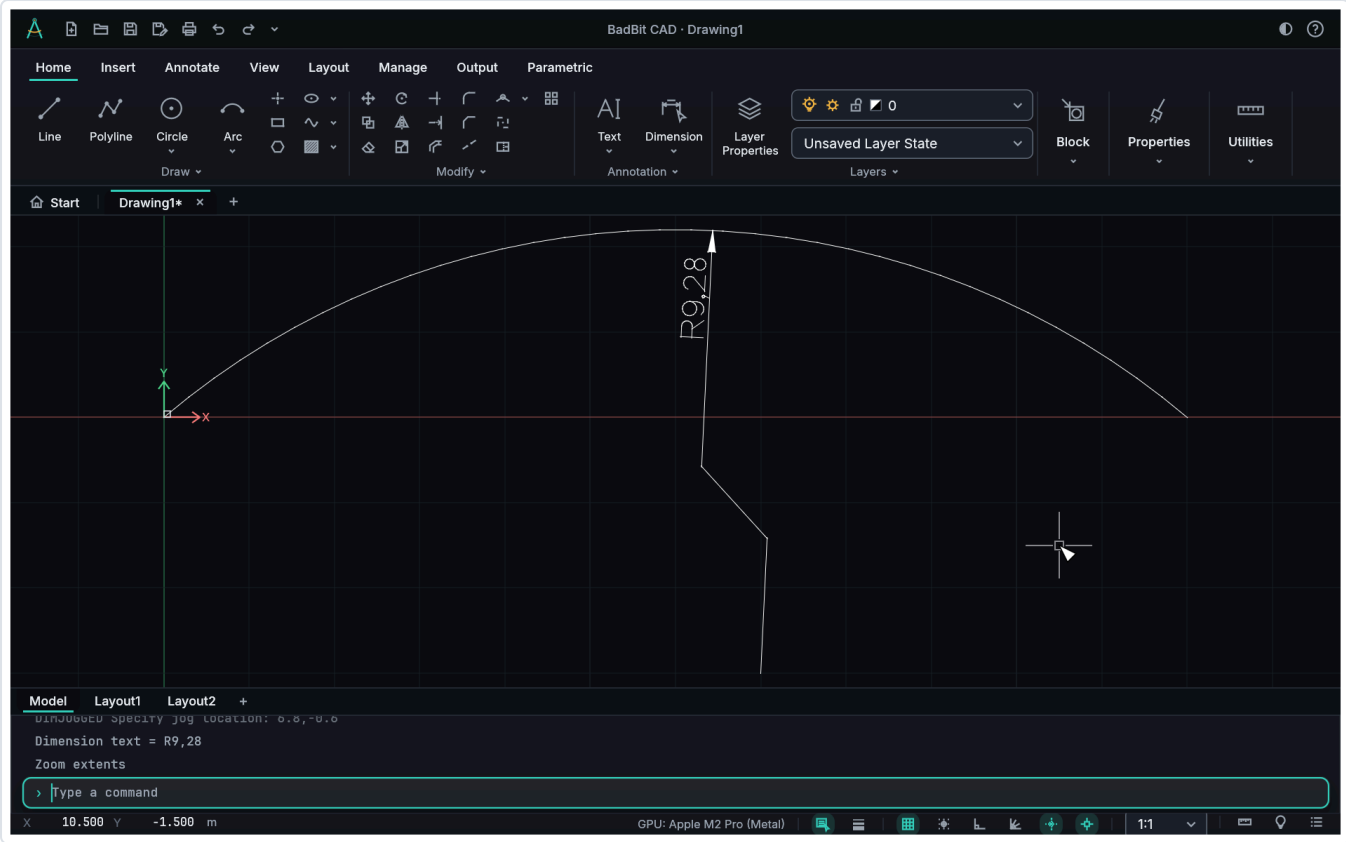
- **Partial** (`P`): dimensions only a portion of the arc. Click the **first** and **second point of the portion** (both on the arc, or brought to it) and return to the dimension location; the measurement is that of the portion;
- **Leader** (`L`): a leader with an arrowhead runs from the text to the arc, as in AutoCAD (useful when the dimension is far away). The arrowhead is the style's leader arrowhead (DIMLDRBLK). With the leader on, the option becomes **No leader** (`N`), which removes it.

DIMJOGGED (alias DJO, JOG)

The jogged radius, for arcs whose center is far away (off the sheet, or too far for the leader to reach it), as in AutoCAD:

1. **Select arc or circle**;
2. **Center location override**: the point the leader starts from, instead of the real center;
3. **Dimension line location** (with **Mtext**, **Text** and **Angle**): the direction of the leader and the location of the text;
4. **Jog location**: where the leader makes its zigzag.

The measurement is the real radius (`R9,28`), with the `R` prefix. The jog angle is the style's (**Radius jog angle**, DIMJOGANG, 45° in Standard). The grips move the center override, the text and the jog. In the Properties palette, the type appears as **Jogged radius**.



DIMJOGGED on an arc with a distant center: the leader starts from the center override, jogs and reaches the arc.

DIMORDINATE (alias DOR)

The coordinate of a point measured from the drawing origin: **the feature location** and **the leader endpoint**. As in AutoCAD, the direction of the leader chooses what is measured: going up or down, the X coordinate (X datum); to the side, the Y. **Xdatum** (`X`) and **Ydatum** (`Y`) fix it; **Mtext**, **Text** and **Angle** also work. When the leader endpoint is not aligned with the point, the leader makes the AutoCAD jog.

DIMCONTINUE (alias DCO) and DIMBASELINE (alias DBA)

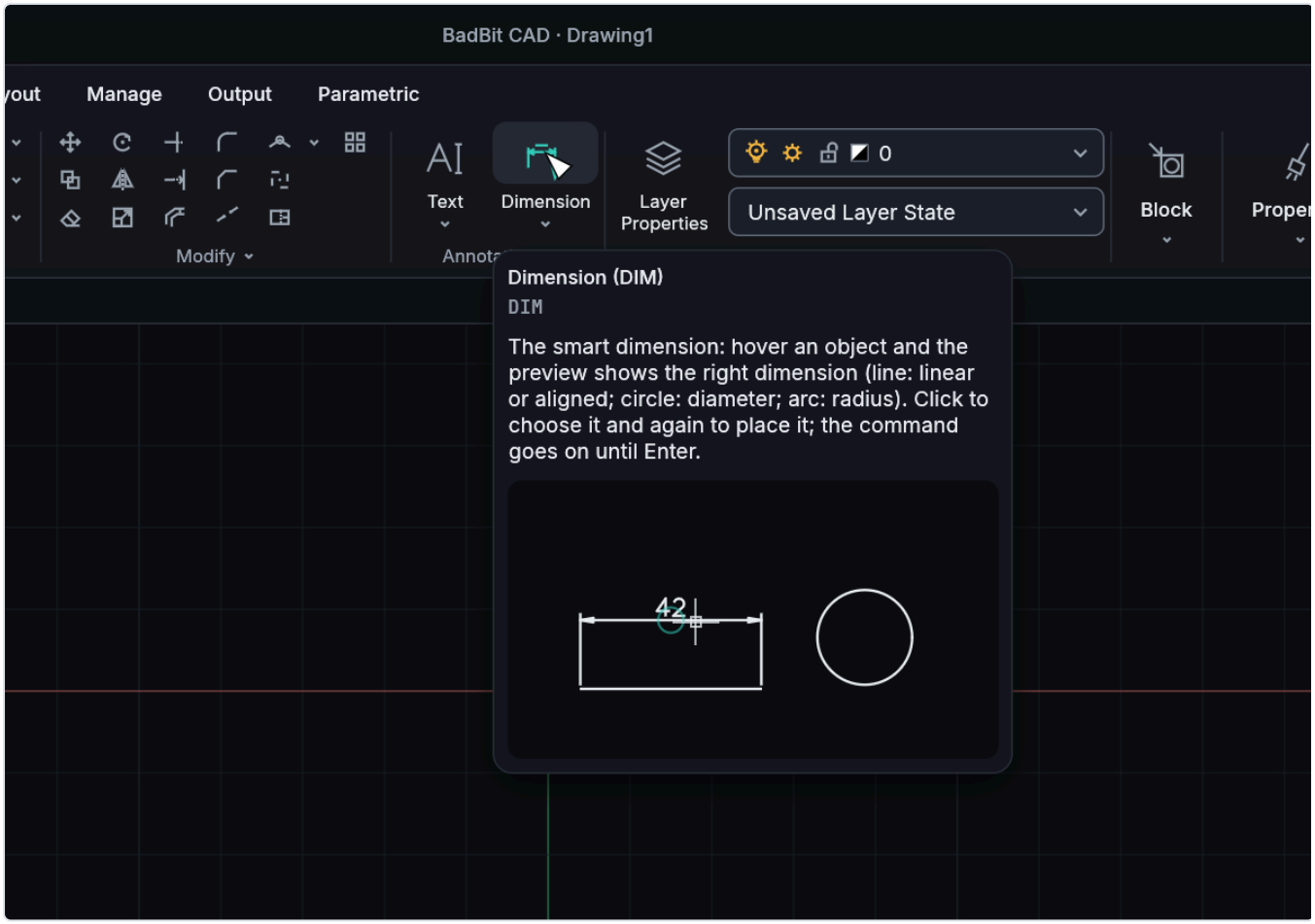
Chains of dimensions from the **last dimension** created (linear, aligned, angular or ordinate) or from a chosen one (**Select**, `S`, or `Enter`). Each point gives a new dimension:

- in **DIMCONTINUE**, the new one starts from the second extension line of the previous one, on the same dimension line;
- in **DIMBASELINE**, all start from the same origin (the first extension line of the base), each dimension line offset from the previous one by the style's **baseline spacing** (DIMDLI).

When choosing the starting dimension, the extension line nearest the click is the origin. **Undo** (`U`) removes the last one; `Enter` goes back to asking for the starting dimension, and another `Enter` ends.

DIM (the smart dimension)

It is the large **Dimension** button on the Annotation panel. Move the cursor over an object and the preview already shows the right type of dimension: on a **line**, the aligned one (with the cursor away from it, the horizontal or vertical one); on a **circle**, the diameter; on an **arc**, the radius. Click the object and then the location. On a line, clicking another line makes the **angular** dimension between the two; on a circle or arc, **Radius** (`R`) and **Diameter** (`D`) switch the type; **text aNgle** (`N`), **Mtext** and **Text** also work. With no object under the click, it is the first origin of a linear dimension. DIM keeps asking for dimensions until `Enter`; at the first prompt, **Angular** (`A`), **Baseline** (`B`), **Continue** (`C`) and **Ordinate** (`O`) go to those commands, and **Undo** (`U`) removes the last dimension.



The tooltip of the Dimension button, with the animation: the preview on the line and then on the circle.

The small arrow of the **Dimension** button lists the types (Linear, Aligned, Angular, Arc Length, Radius, Diameter, Ordinate and Jogged); the open Annotation panel (the small arrow of the title) has STYLE, DIMSTYLE, MLEADERSTYLE, MLEADER, MLEADEREDIT, QLEADER, QDIM, TOLERANCE, DIMCONTINUE and DIMBASELINE. Each button has a tooltip with the animation of the command.

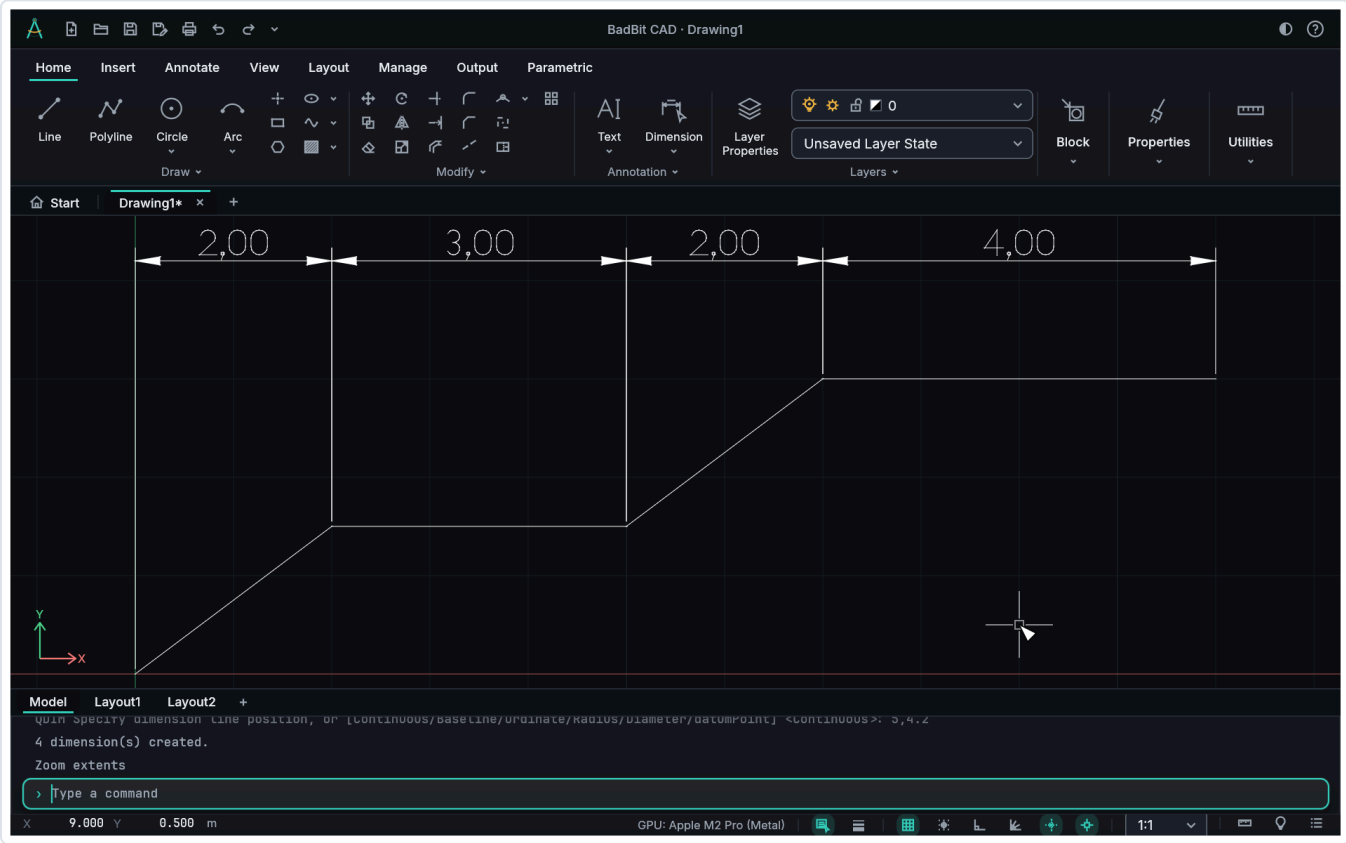
QDIM (quick dimensions)

Dimensions a whole selection at once, like the AutoCAD QDIM. **Select geometry to dimension** (lines, polylines, arcs and circles; a window works) and `Enter`. The points that count are the ends of lines and arcs, the vertices of polylines and the centers of circles; points at the same position along the

dimension are entered only once. Then the **dimension line position**, with the preview of all the dimensions following the cursor (as with linear dimensions, above or below gives horizontal ones; to the side, vertical ones):

- **Continuous** (`C`, the default): one dimension between each pair of neighboring points, in series;
- **Baseline** (`B`): all from the first point (or from the datum point), each one offset from the previous one by the style's baseline spacing;
- **Ordinate** (`O`): the coordinate of each point from the datum point (without it, from the drawing origin);
- **Radius** (`R`) and **Diameter** (`D`): one dimension on each arc and circle in the selection;
- **datumPoint** (`P`): the point from which the baseline and the ordinates start.

The chosen mode is remembered for the next QDIM, as in AutoCAD. All the dimensions go into a single undo step, and the command line says how many were created. QDIM dimensions are associative (with DIMASSOC 2): each one is linked to the points it measured.



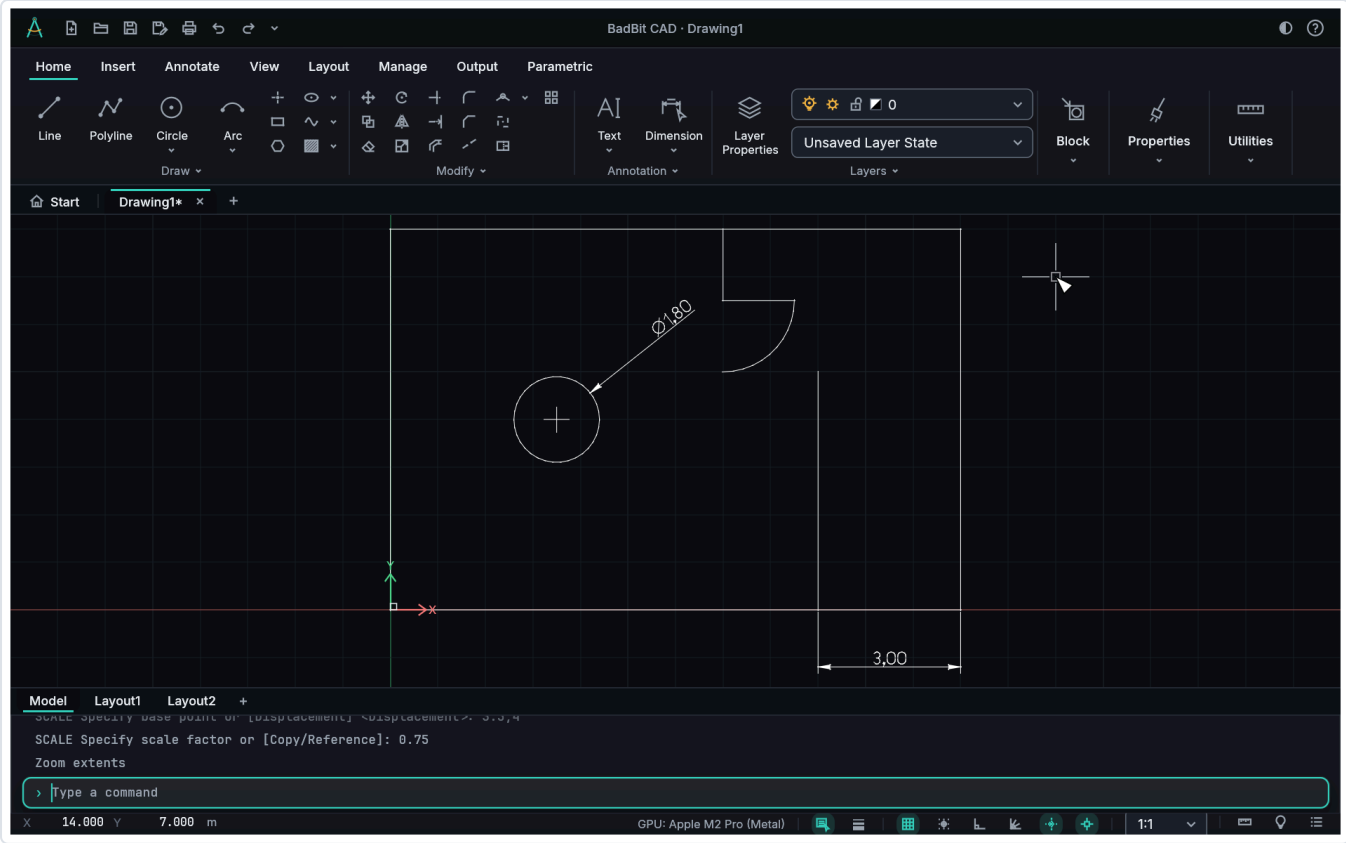
QDIM on a polyline: four dimensions in series at once, each one linked to a vertex.

Associative dimensions (DIMASSOC)

As in AutoCAD (DIMASSOC 2, the default), a dimension made **on the points of an object** stays linked to it: the end or midpoint of a line, the vertex of a polyline, the center, the quadrant or a point on the curve of a circle or arc, the intersection of two objects, a point (POINT) and the insertion point of a text or block. **Only through object snaps, as in AutoCAD**: what gets linked is a point that came from an **object snap** (endpoint, midpoint, center, quadrant, intersection, node, insertion, nearest) and the **object selected** in the command (the `Enter` of DIMLINEAR and DIMALIGNED, the object click in DIM, the arc or circle of the radius, diameter and DIMJOGGED, the lines of the angular dimension, the

objects of QDIM). A **typed point** (even if it falls exactly on the end of a line), a **free click** (with object snap off, `F3`) and the **grid snap** (`F9`) **do not link**: the dimension stays loose. With one point from an object snap and the other typed, the dimension is **partially** associative. In continued and baseline dimensions, the origin that comes from the previous dimension is linked if it was linked there. When the object **changes** (MOVE, ROTATE, SCALE, MIRROR, STRETCH, grips, the Properties palette…), the dimension is rebuilt with the new points, **in the same undo step**: a U brings back the object and the dimension together. The dimension keeps the same offset from the object, and the measurement follows.

Any point on the object (the **Nearest** snap, `NEA`): a dimension to the middle of a wall, away from the ends and the midpoint, is linked too, on a line, an arc, a circle, an ellipse or a polyline segment. The point keeps the **proportion** at which it lies on the object (on a polyline segment, the proportion along the segment; on a circle, the angle): moving, rotating or scaling the object takes the point along; stretching the object keeps the point at the same proportion (3 m from the start of a 10 m wall stretched to 20 m, it goes to 6 m), as in AutoCAD. Stretching only the dimension along the object keeps the link at the new point. In automation, a point given by coordinates that falls exactly on an object, without being an endpoint, midpoint or intersection, links the same way (in the dimension operations and in DIMREASSOCIATE with points; in a command point, only with the Nearest snap on in OSMODE).



The partition moved and the table scaled: the linear dimension changed to 3,00 and the diameter to 1,80 on their own.

What breaks the link:

- moving **only the dimension** (moving, rotating or stretching the dimension without the object): the point that left the object is released, as in AutoCAD; the one that still falls on the object stays linked. Copying the dimension together with the object links the copy to the copy of the object;

- **erasing the object**: the dimension stays where it is, loose;
- **DIMDISASSOCIATE** (alias `DDA`): **select the dimensions** and `Enter`; the command line says how many were released.

DIMREASSOCIATE (alias `DRE`) links again (or links a loose dimension for the first time). **Select the dimensions** and `Enter`; for each one, an **X** marks the definition point being asked for, and the prompt is that of the point: **first extension line origin**, **second** and the **next definition point** (on a radius or diameter, **the arc or circle**). Click the point on the object **with an object snap** (a typed point does not link) or `Enter` to skip to the next one. At the first prompt, **Select object** (`S`) links a linear or aligned dimension to both ends of a line at once.

The **DIMASSOC** variable chooses how new dimensions are created: **2** associative (the default), **1** loose (not linked, but still a single object) and **0** exploded (the parts of the dimension, lines, arrowheads and text, as separate objects). The **Properties** palette shows the **Associative** row: **Yes**, **Partial** (only some points linked) or **No**; LIST also tells.

In DXF and DWG the link is stored the way AutoCAD stores it (the dimension's DIMASSOC object), in any version (DWG 2000 to 2018, DXF 2000 to 2018): opening the file in AutoCAD and moving the object, the dimension follows. An associative dimension from an AutoCAD file opens **linked** in BadBit CAD and stays linked when saved in another version. A **Nearest** point is stored as AutoCAD's Nearest snap, with its position on the object; on a spline it stays in BadBit CAD only (it is saved loose in the file).

Editing dimensions: grips, DIMEDIT and DIMTEDIT

With the dimension selected, the **grips** are at the origins, at the ends of the dimension line and on the text. The grip of an origin changes what is measured; the one at an end of the dimension line moves the line away or closer; the text grip takes the text to where you click, and the dimension line goes along (AutoCAD's DIMTMOVE 0). On a radius or diameter, the grip rotates the leader around the center. STRETCH stretches the dimension by the points inside the crossing window. EXPLODE turns the dimension into its parts: lines, arcs, the arrowheads (polylines whose width tapers to the tip, shown filled on screen) and the text.

DIMEDIT (alias `DED`): **[Home/New/Rotate/Oblique] <Home>** and then the objects. **Home** returns the text to the style's location and rotation; **New** replaces the text (`<>` is the measurement; `Enter` goes back to the measurement); **Rotate** rotates the text; **Oblique** slants the extension lines of linear and aligned dimensions to the given angle (`Enter`: normal again).

DIMTEDIT (alias `DIMTED`): **select dimension** and the **new location for the text**, which follows the cursor; or **Left** (`L`), **Right** (`R`) and **Center** (`C`) along the dimension line, **Home** (`H`) and **Angle** (`A`).

The **Properties** palette and **Quick Properties** show the dimension **type**, the **dimension style** (the list changes the style), the **measurement**, the **text override**, the **text height**, the **arrow size**, the **extension line offset**, the **text gap**, the **overall scale** (DIMSCALE), whether the dimension **has overrides**, whether it is **associative** and whether it is **annotative** (with its scales). Changing the height, arrowhead, offset, gap or scale there puts an **override on the chosen dimensions only** (the style does not change; see Style overrides, below). LIST shows the type, the style, the text, the measurement and whether the dimension is associative.

DIMBREAK interrupts the lines of the dimension (the dimension line and the extension lines) where other objects cross, as in AutoCAD, so the dimension does not clash with the drawing. **Select dimension to add/remove break or [Multiple]**; then, **select object to break dimension or [Auto/Manual/Remove] <Auto>**:

- clicking an object breaks only where it crosses;
- **Auto** (`A`, or `Enter`): wherever any object crosses;
- **Manual** (`M`): the **first** and **second break point**, brought to the nearest line of the dimension;
- **Remove** (`R`): removes all the breaks of the dimension.

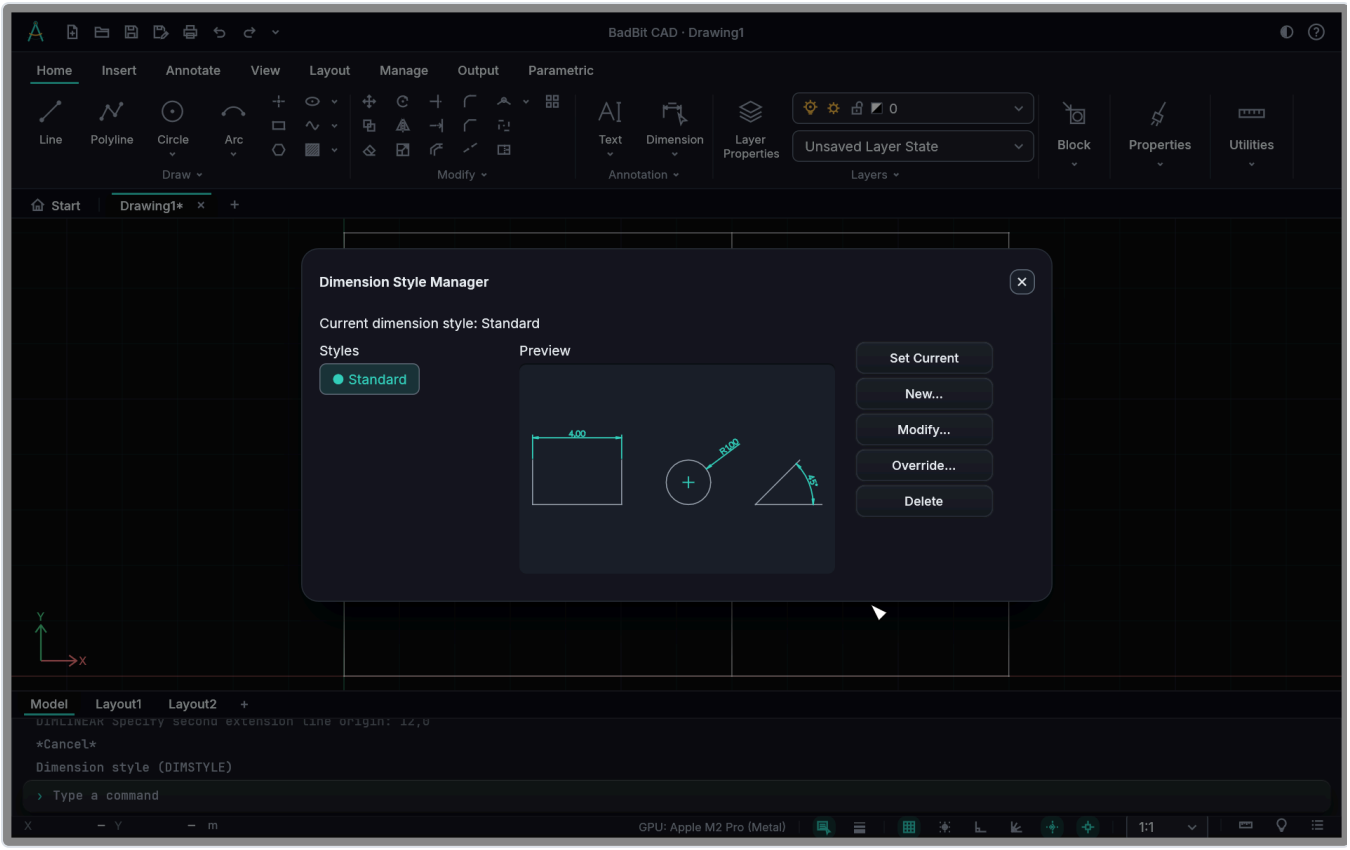
With **Multiple**, choose the dimensions, `Enter` and **[Break/Remove] <Break>**. The size of each break is the style's **Break size** (0.375 in Standard). Everything in a single undo step; dimensions on a locked layer are left out.

DIMSPACE adjusts the spacing between parallel linear or aligned dimensions, as in AutoCAD: **select base dimension**, then **the dimensions to space** and `Enter`, and **enter value or [Auto] <Auto>**. Each dimension goes to the given distance from the previous one, starting from the base; **Auto** uses twice the style's text height, and **0** aligns all of them on the dimension line of the base.

DIMSTYLE (alias D, DST, DDIM)

The **Dimension Style Manager**, like the AutoCAD one: the current style at the top, the list of the drawing's styles (the current one marked with ●), the preview of the marked style and the buttons:

- **Set Current**: the next dimensions use the marked style (the **DIMSTYLE** variable); a double-click in the list does the same;
- **New...**: the name (it suggests `Copy of` and the marked one) and **Continue**: the style is created as a copy of the marked one and the Modify dialog opens on it;
- **Modify...**: the style dialog;
- **Override...**: the overrides of the current style (see Style overrides, below);
- **Delete**: does not remove Standard, the current style or a style used by any dimension, and says why.

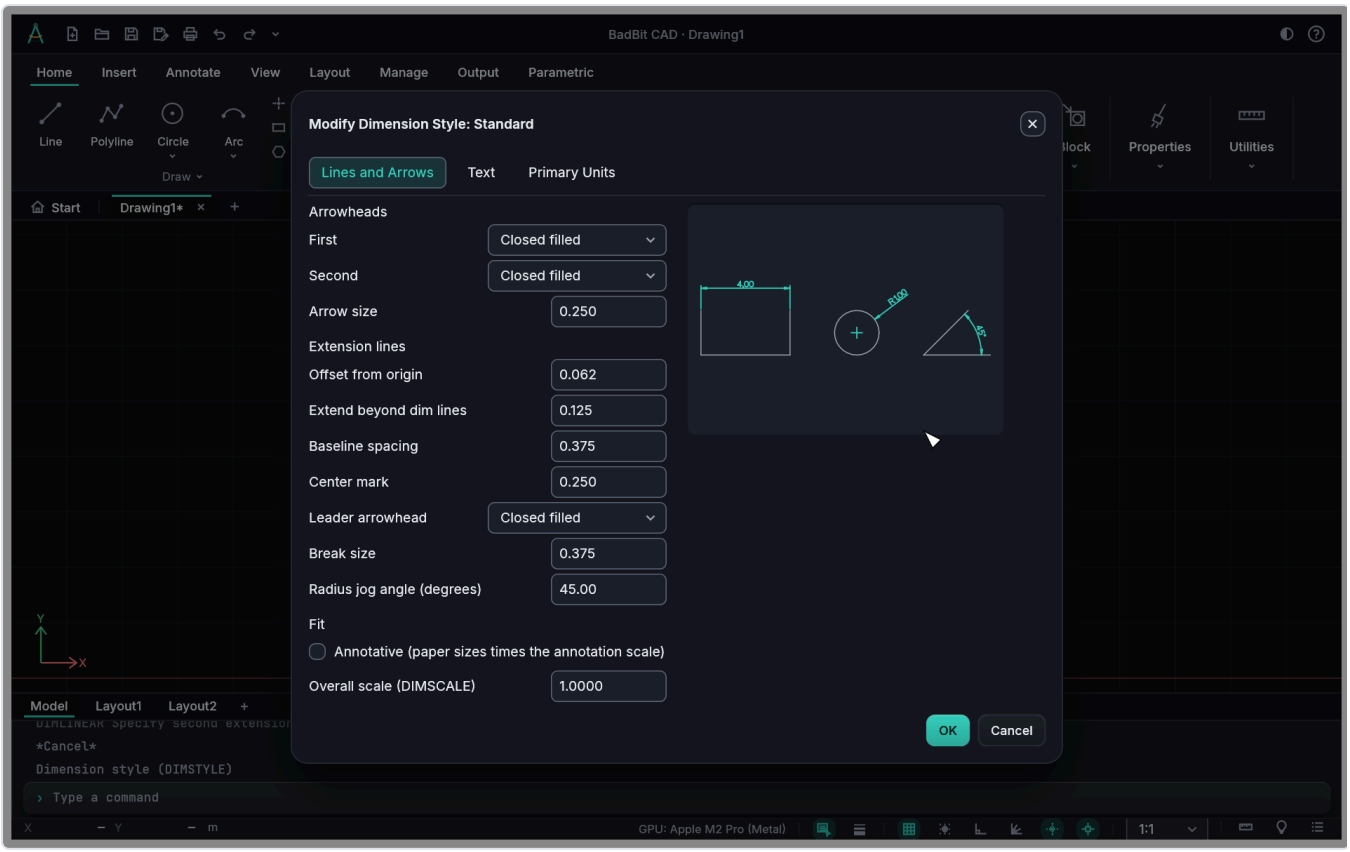


The Dimension Style Manager with Standard and the preview.

The **Modify Dimension Style** dialog has three tabs, with the preview beside them:

- **Lines and Arrows**: the first and second arrowheads (**Closed filled**, **Open**, **Architectural tick**, **Dot** or **None**), the arrow size, the offset of the extension line from the origin, the extension beyond the dimension line, the baseline spacing, the center mark size, the **leader arrowhead** (that of QLEADER, LEADER and the DIMARC leader, DIMLDRBLK), the **break size** (DIMBREAK), the **radius jog angle** (DIMJOGGED, in degrees) and, under Fit, **Annotative** (see Annotative objects, below) and the **overall scale** (DIMSCALE), which multiplies all the sizes;
- **Text**: the text style (a text style with a fixed height controls the height), the text height, the distance between the text and the dimension line, the vertical position (**Above** the line or **Centered**, with the line interrupted) and the alignment (**Aligned with dimension line** or **Horizontal**);
- **Primary Units**: the format (**Decimal**, **Scientific**, **Engineering**, **Architectural** and **Fractional**; in the last three, as in AutoCAD, the drawing number counts as inches), the precision (decimal places, or the denominator of fractions), the decimal separator, the **prefix** and the **suffix**, the **measurement scale factor** (DIMLFAC: with 100, a drawing in meters is dimensioned in centimeters), the suppression of leading and trailing zeros, and the format and precision of angles.

OK saves the style and redraws all of its dimensions, in a single undo step (the command line says how many changed); **Cancel** leaves things as they were.



The Modify Dimension Style dialog, on the Lines and Arrows tab.

Standard is AutoCAD's ISO-25 for a drawing in meters: text of 0.25 (the same as TEXTSIZE), closed filled arrowheads of 0.25, offset from the origin of 0.0625, extension of 0.125, text distance of 0.0625, baseline spacing of 0.375, two decimal places with a comma, text above the line and aligned with it. Unlike ISO-25, Standard does not suppress trailing zeros (`3,00`, as dimensions in meters are written in Brazil); the option is on the Primary Units tab. The current style belongs to the drawing and is saved in the file; it can also be changed with the **Current dimension style** list on the Annotation panel (with a wide window) or with the **DIMSTYLE** variable in SETVAR. Changing a text style also redraws the dimensions of the dimension styles that use it, and a text style used by a dimension style cannot be deleted.

Style overrides: Override..., variables, DIMOVERRIDE and -DIMSTYLE

As in AutoCAD, you can change a value of the style **without changing the style**: the change is kept as an **override**. There are three scopes:

- **On the current style**: new dimensions are created with the override; existing ones do not change. This is done with the **Override...** button of DIMSTYLE (it opens the Modify dialog with the title *Overrides of style...*; OK saves only what differs from the style, and the command line says how many overrides were kept) or by typing the **dimension variable** (DIMTXT, DIMASZ, DIMDEC... see the System variables chapter), directly or through SETVAR. With overrides, the manager shows *<style overrides> on the current style*. They are saved with the drawing (in DWG and DXF, in the dimension variables of the header, as AutoCAD writes them) and come back when it is opened; the dimensions made with them keep their own and do not change when the file is reopened;

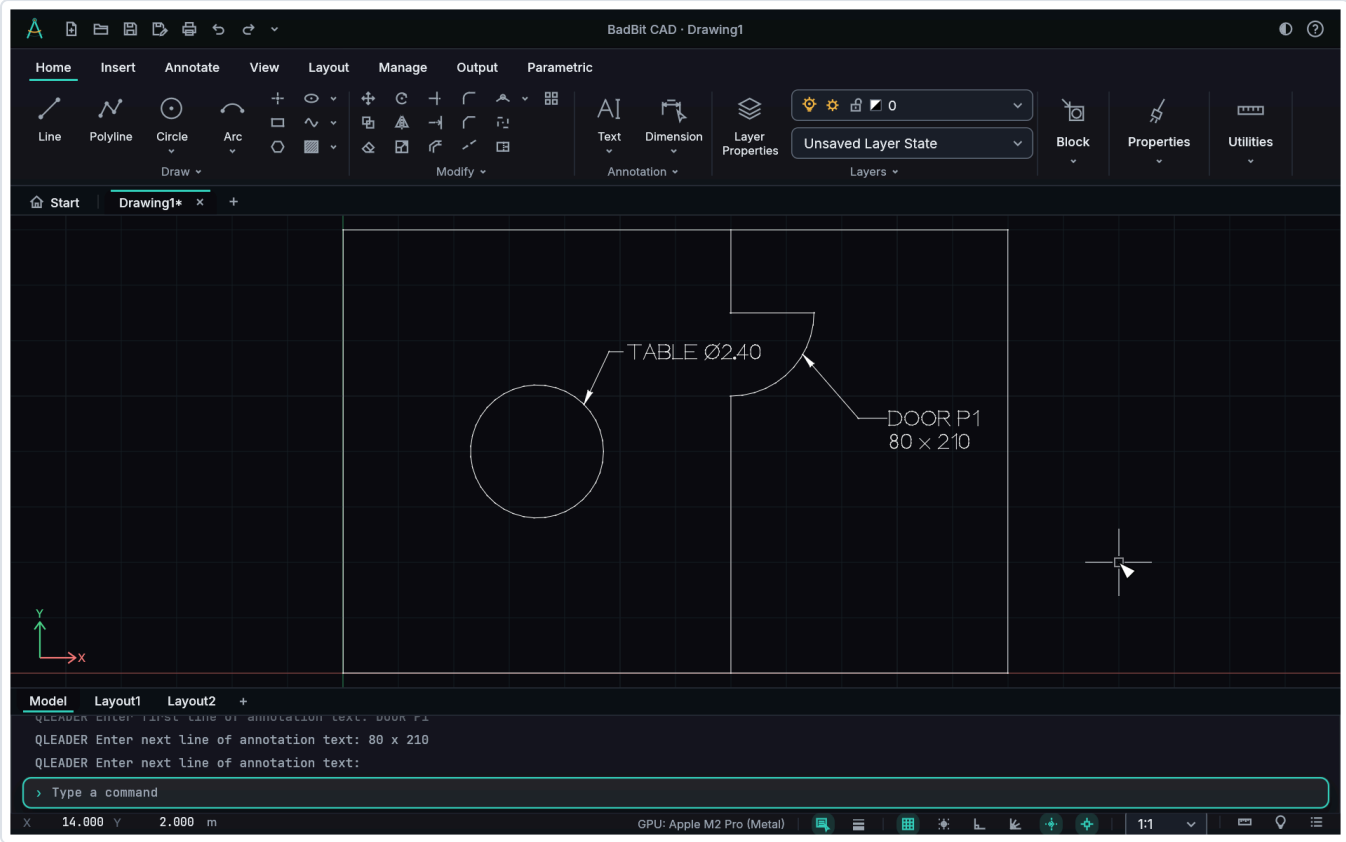
- **On chosen dimensions**: through the **Properties** palette (text height, arrowhead, offset, gap, overall scale) or through **DIMOVERRIDE** (alias `DOV`): **enter the name of the dimension variable to override**, the **new value**, another variable or `Enter`, and then **select the dimensions**. **Clear overrides** (`C`) at the first prompt removes all the overrides from the chosen dimensions;
- A dimension's override **remains in effect** when its style changes: the rest follows the style, the overridden value stays.

-DIMSTYLE is the dimension style through the command line, as in AutoCAD:
[Save/Restore/STatus/Variables/Apply/?] <Restore>:

- **Save**: saves the current style **with its overrides** as a new style (the name), which becomes current and without overrides; if the name already exists, it asks whether to redefine it;
- **Restore**: makes the typed style current, or that of a dimension (`Enter` and select the dimension);
- **STatus**: lists all the dimension variables of the current style (with the overrides), each one with its value and what it does;
- **Variables**: the same list for the typed style or that of a dimension, without changing the current one;
- **Apply**: passes the current style, with its overrides, to the chosen dimensions;
- **?**: lists the styles in the drawing.

Leaders (MLEADER and QLEADER)

Note leaders follow those of AutoCAD: the **multileader** (MLEADER), a single object with the arrowhead, the lines, the landing and the text; and the **classic leader** (QLEADER and LEADER), the arrowhead and the lines, with the text beside them as another object. The sizes come from the multileader style (MLEADERSTYLE) for the multileader and from the dimension style (the leader arrowhead and the arrow size) for the classic one. The closed filled arrowhead comes out filled on screen, like that of dimensions. **EXPLODE** of the classic leader gives one polyline per line and the pieces of the arrowhead, each one once; that of the multileader gives the lines, the arrowhead, the landing and the text.



A multileader on the table (MLEADER) and a classic leader with two lines of text on the door (QLEADER).

MLEADER (alias MLD)

1. **Specify leader arrowhead location or [leader Landing first]**;
2. **Specify leader landing location**: the leader follows the cursor; the landing and the text go to the side of the cursor (right or left);
3. **The text**, typed in the drawing itself, as in MTEXT: `Enter` breaks the line and `Ctrl`+`Enter` (`Cmd`+`Enter` on the Mac) confirms. Empty text leaves the leader without text.

With **leader Landing first** (`L`), the order is reversed: the landing and then the arrowhead (**leader arrowHead first**, `H`, switches back). If the style asks for more than two **leader points**, MLEADER asks for the middle points before the landing (`Enter` ends sooner). The codes `%%c`, `%%d` and `%%p` work in the text. The Properties palette shows the **multileader style** (the list changes it), the text, how many **leader lines** and the arrow size; the grips move the tip of each line and the landing (the text goes along).

MLEADEREDIT (alias MLE)

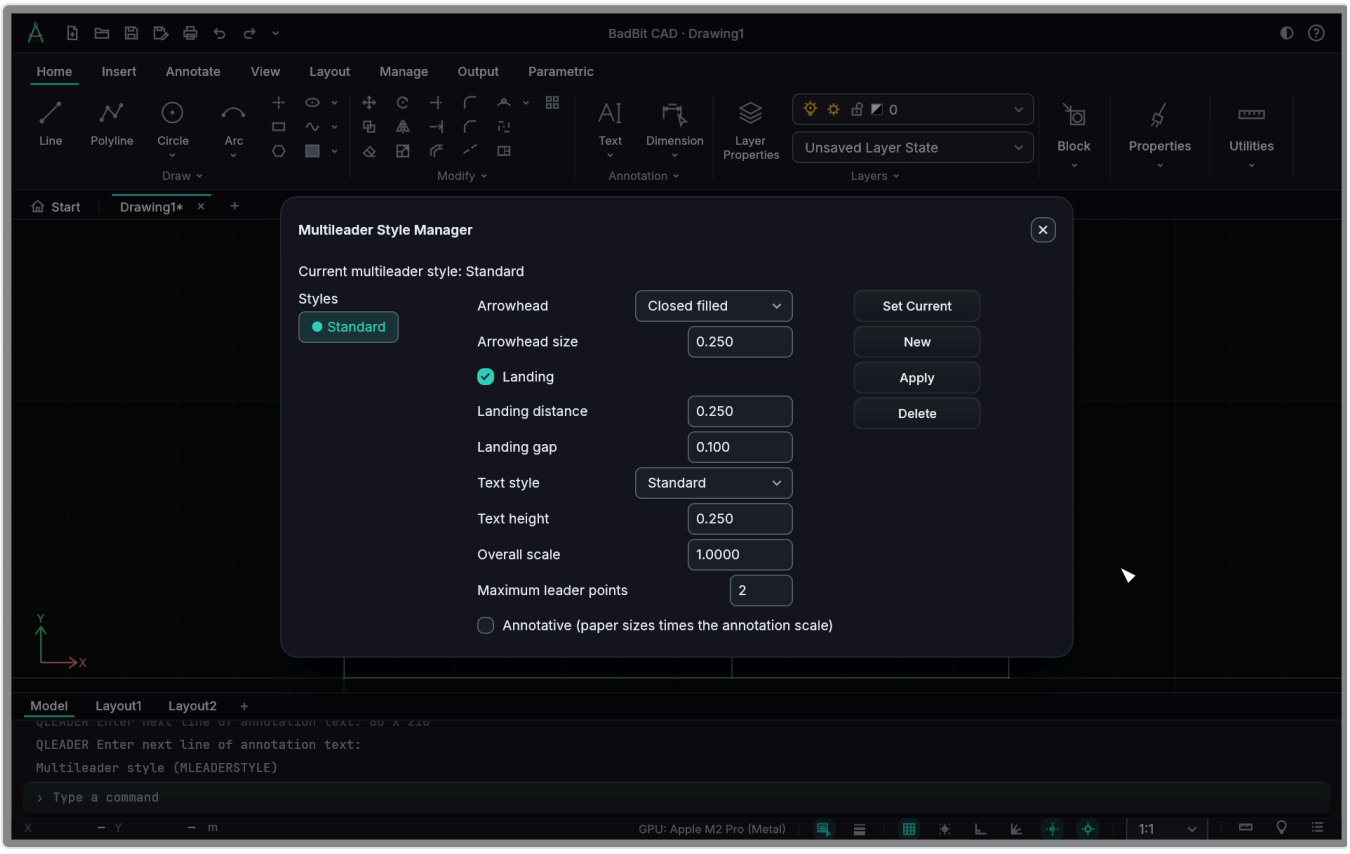
More lines on the same multileader, all reaching the same landing: **select a multileader** and **specify leader arrowhead location or [Remove leaders]**, as many times as you like, and `Enter`. **Remove leaders** (`R`) asks for **the leaders to remove** (click near each one); the last line cannot be removed (the multileader needs at least one).

MLEADERSTYLE (alias MLS)

The **Multileader Style Manager**: the current style at the top, the list of styles and, beside it, the marked style for editing: the **arrowhead** (the same as for dimensions) and the **arrowhead size**, **Landing**, the **landing distance**, the **landing gap** (between the landing and the text), the **text style** and the **text height**, the **overall scale**, the **maximum leader points** (2: a single straight line to the landing) and **Annotative**. The buttons:

- **Set Current**: the next leaders use the marked style;
- **New**: the name of the new style, which is created as a copy of the marked one (it suggests `Copy of` and the name);
- **Apply**: saves the changes to the marked style and rebuilds all of its leaders, in a single undo step (the command line says how many);
- **Delete**: does not remove Standard, the current style or a style used by any leader.

Standard follows AutoCAD's in a drawing in meters: closed filled arrowhead of 0.25, landing of 0.25, gap of 0.1 and text of 0.25.



The Multileader Style Manager with Standard.

QLEADER (alias LE) and LEADER (alias LEAD)

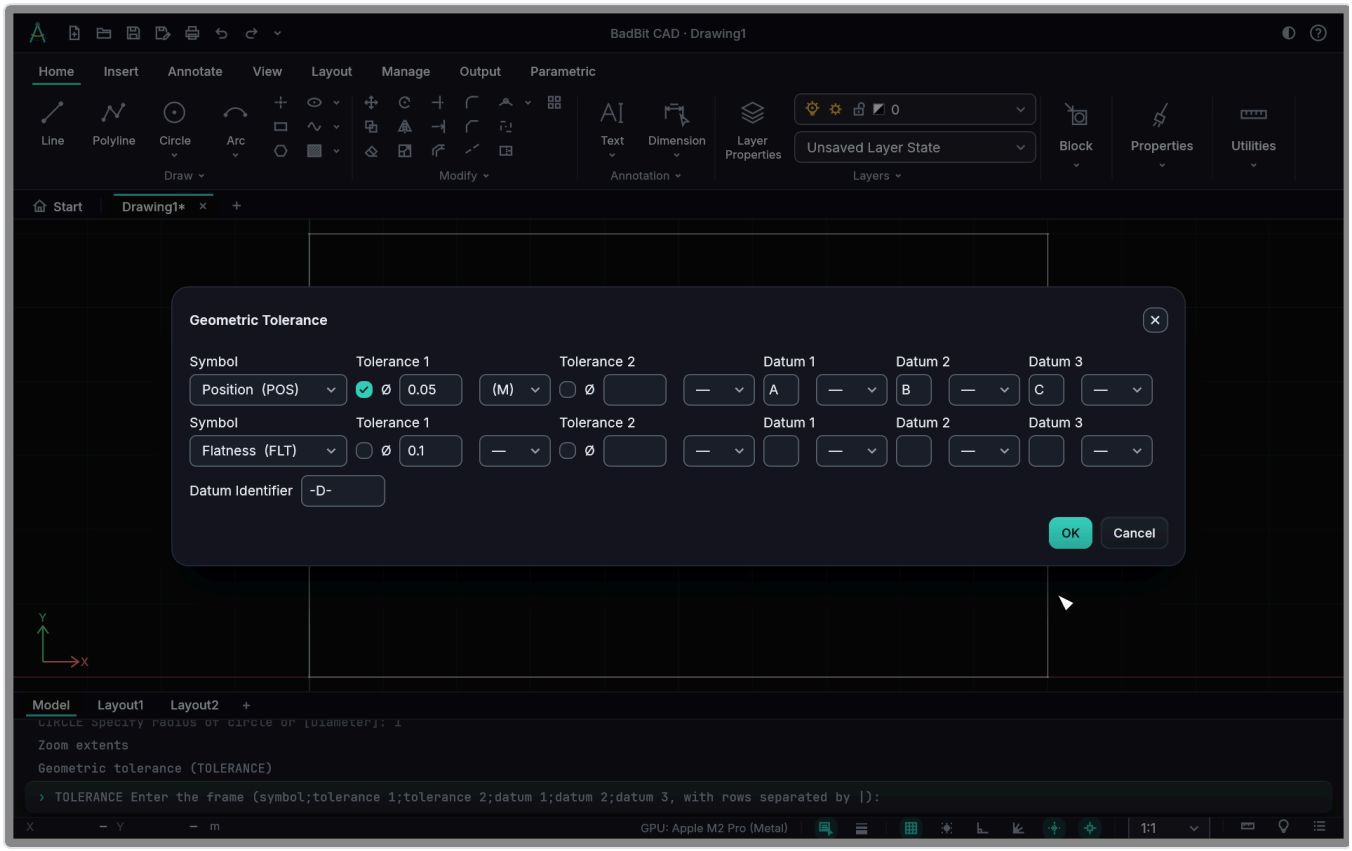
QLEADER: **specify first leader point** (the arrowhead tip) and **the next point**, up to three points (the third one already ends; `Enter` after the second one also does). Then **the text width** (`Enter`: 0, no automatic wrapping) and the lines of the text, one per prompt (**the first line**, **the next line**...), until an `Enter` on an empty line. The leader and the text (an MTEXT, placed against the end of the leader) go into a single undo step.

LEADER: **the leader start point** and the **next points**, with no limit, with **Undo** (`U`), which removes the last point, and **Annotation** (`A`, or `Enter`), which goes to the text, line by line as in QLEADER (without asking for the width).

Geometric tolerance (TOLERANCE, alias TOL)

The **feature control frame**, like the AutoCAD one: up to two rows, each one with the characteristic **symbol**, two **tolerances** (with the diameter ⌀ and the material condition) and up to three **datum references**, and the **datum identifier** in a cell below. The symbols are letters of the BadBit CAD stroke font, so the frame looks the same on screen, in print and at any zoom. The text height and the gap are those of the current dimension style (DIMTXT and DIMGAP).

TOLERANCE opens the **Geometric Tolerance** dialog: in each row, the **Symbol** list (the name and the code, such as *Position (POS)*), **Tolerance 1** and **Tolerance 2** (the **Ø** box puts the diameter before the value; the list next to it is the material condition: **(M)**, **(L)** or **(S)**) and **Datum 1 to 3** (the letter and the condition); below, the **Datum Identifier**. **OK** asks for the **tolerance location** (the middle of the left side of the first row), with the frame following the cursor; **Cancel** leaves the command.



The Geometric Tolerance dialog: position with ⌀0,05 Ⓜ and datums A, B and C; flatness of 0,1; the identifier -D-.

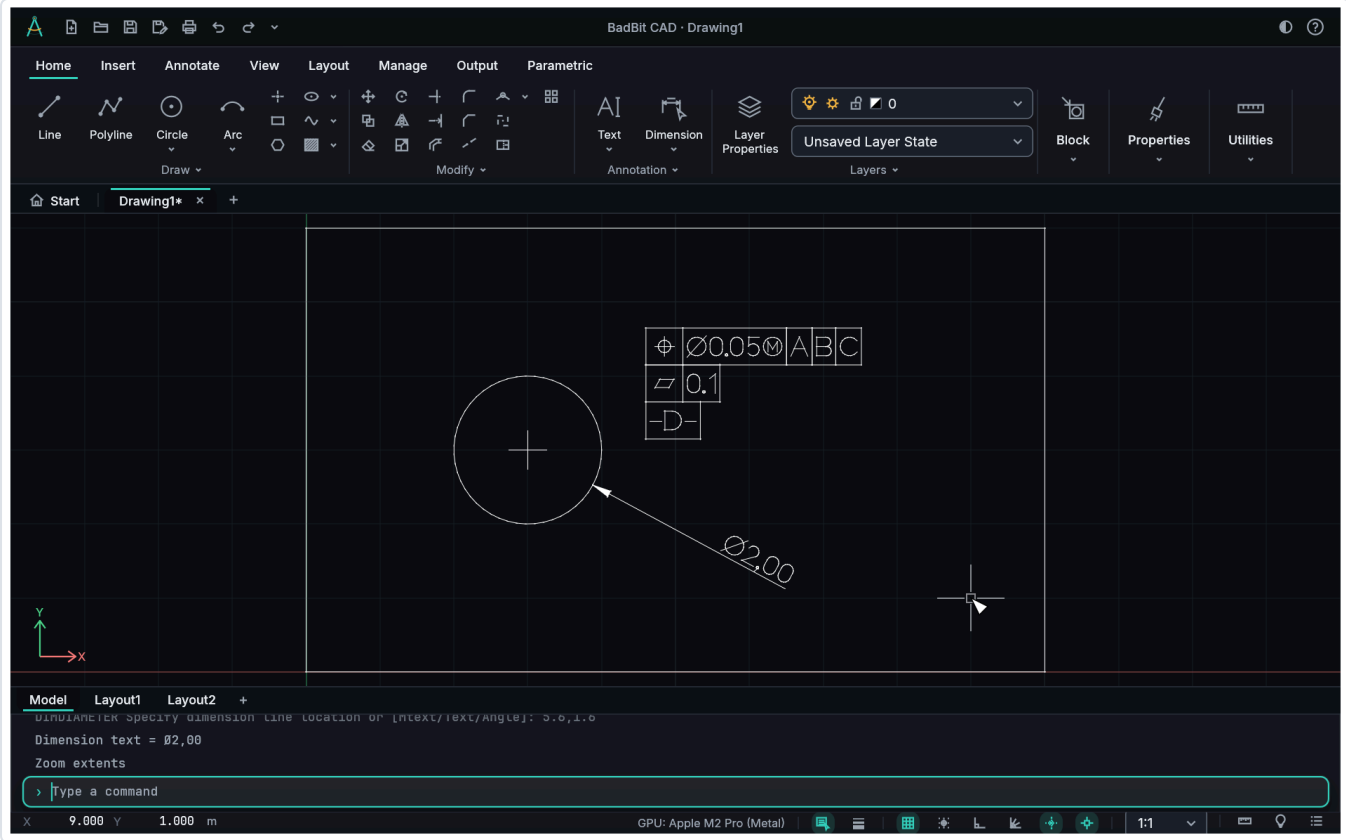
OK sends the frame to the command in the **typed form** of TOLERANCE, a line of text: the cells separated by `;`, in the order **symbol;tolerance 1;tolerance 2;datum 1;datum 2;datum 3**, the two rows separated by `|` and the identifier as `ID=`. The frame in the figure is:

```
POS;%%c0,05(M);;A;B;C|FLT;0,1|ID=-D-
```

An empty cell (`;;`) does not appear. The symbol codes, in the order of the list in the dialog:

Code	Characteristic	Code	Characteristic
POS	Position	CIR	Circularity
CON	Concentricity	STR	Straightness
SYM	Symmetry	PRS	Profile of a surface
PAR	Parallelism	PRL	Profile of a line
PER	Perpendicularity	RUN	Circular runout
ANG	Angularity	TRU	Total runout
CYL	Cylindricity		
FLT	Flatness		

In values and datums, `%%c` (or `DIA` at the start) becomes the diameter ⌀, and modifiers in parentheses become the circled symbols: `(M)` maximum material, `(L)` least material, `(S)` regardless of feature size, `(P)` projected zone, `(F)` free state, `(T)` tangent plane and `(U)` unequal zone. More than two rows are not allowed. The Properties palette shows the position, the text height, the contents of the frame and whether it is annotative.



The frame placed next to the hole, with the diameter dimensioned.

Center marks and centerlines (CENTERMARK, CENTERLINE)

The **center marks** and **centerlines** of AutoCAD 2017 onward: **associative** objects that mark the center of holes and the axis of symmetry between two lines. They are on the **Annotate** tab, **Centerlines** panel (the **Center Mark** and **Centerline** buttons).

CENTERMARK: **select circle or arc to add center mark**. Each click on a circle, an arc or an arc segment of a polyline creates the mark (the cross at the center and the four **extensions**, which extend past the curve by CENTEREXE), and the command continues until `Enter`; all the marks of the command are a single undo step. The size of each dash of the cross is CENTERCROSSSIZE and the gap to the extensions, CENTERCROSSGAP: with the `x` (such as `0.1x`, the default), they are proportional to the diameter; without it, lengths; `ByLineType` takes the last dash and the last space of the linetype. On a small circle, whose radius does not reach half the cross plus the gap, the gap disappears. With CENTERMARKEXE 0, new marks come out with the cross only.

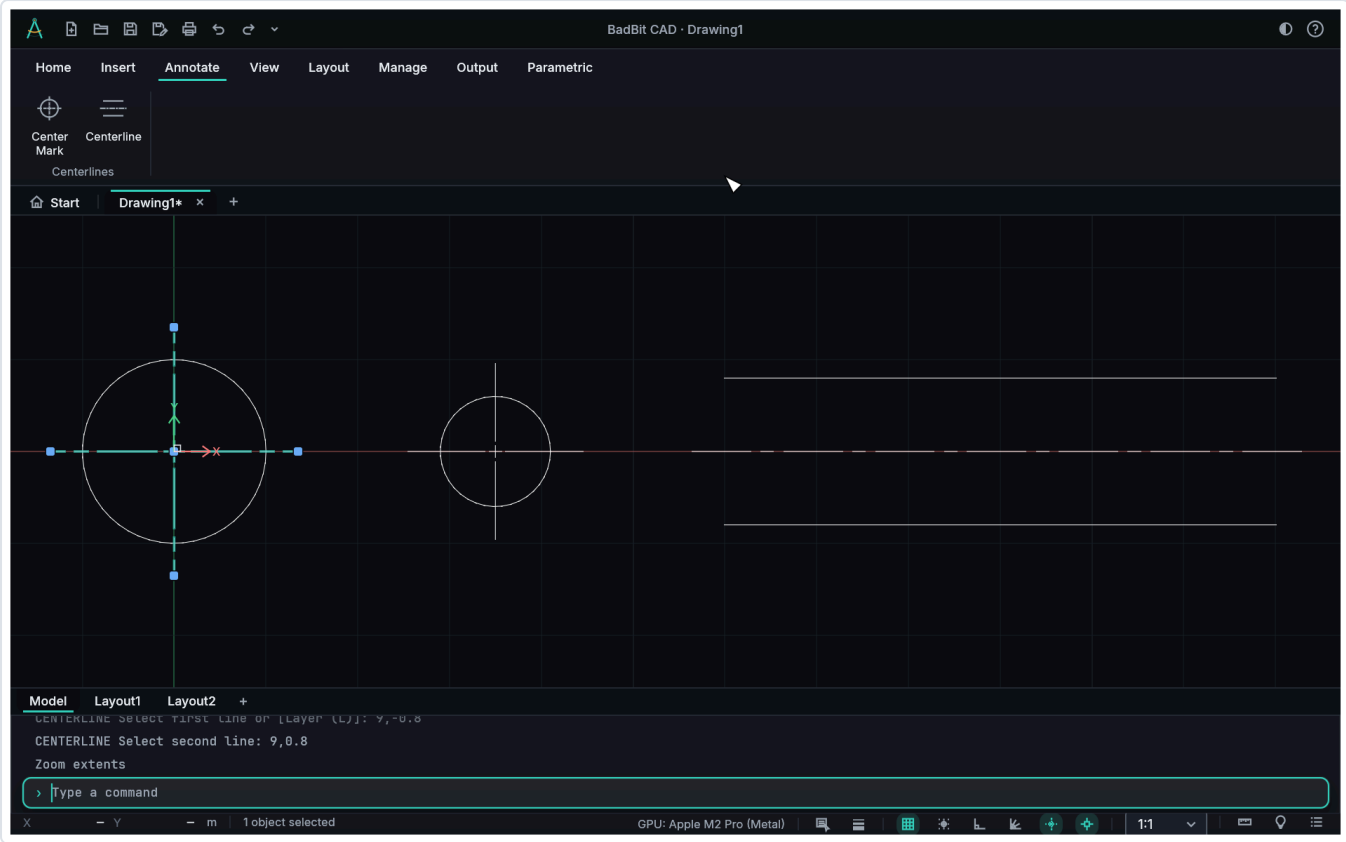
CENTERLINE: **select first line** and **second line** (lines or straight polyline segments). If they are parallel, the centerline passes through the middle of the corresponding ends; if not, it follows the **bisector** of the angle chosen by the clicks (click near the parts of the lines between which the axis should lie), from the middle of the ends on one side to that of the ends on the other. Both ends extend by CENTEREXE.

In both commands, the **Layer** option (`L`) asks for the name of the layer of new marks and lines (CENTERLAYER; **Use current** or `.` returns to the current layer); a layer that does not exist is created along with the mark. Marks and lines come out with the linetype of CENTERLTYPE (the default, CENTER2, is loaded from the BadBit CAD library if the drawing does not have it) and the linetype scale of CENTERLTSCALE. The variables belong to the drawing (they go into the file and into undo) and apply only to new marks.

Associative: moving, rotating, scaling or stretching the circle, arc or lines takes the mark or centerline along, in the same undo step (also through grips and Properties). A mark moved on its own (without the circle) is released; moving both together keeps the link; copying both links the copy of the mark to the copy of the circle. Erasing the circle leaves the mark loose, where it was.

- **CENTERDISASSOCIATE**: **select center objects**; they are released and stay where they are.
- **CENTERREASSOCIATE**: **select center mark or centerline** and then the circle or arc (the mark goes to its center) or the two lines.
- **CENTERRESET**: returns the extensions of the chosen marks and lines to the current CENTEREXE (a length set by grips stays).

Grips: on a mark, the **center** grip moves the mark (hovering over it, the menu has **Change extension length**, which changes the four extensions together) and the grips at the **ends of the extensions** fix the length of that extension (it no longer follows the size of the circle). On a centerline, the **midpoint** grip moves it, the **base** grips change the length (the extension goes along) and the **end** grips change only how far the extension goes. The **Properties** palette shows the center and the radius, the **Cross size** and the **Cross gap** (type `0.1x` or a length), **Show extensions**, the **Extension**, the rotation and whether it is **Associative**; for the line, the start, the end, the length and the start and end extensions. EXPLODE turns the mark and the line into their dashes (lines).



The Annotate tab with the Centerlines panel, the center marks on two holes and the centerline of a shaft; the selected mark of the large hole shows the grips of the center and of the extensions.

In **DXF** and **DWG**, the mark and the line go as an insertion of an anonymous block with the dashes: AutoCAD shows the right geometry, but as a block reference, without associativity (AutoCAD's associative structure is not public). The data to rebuild the mark goes along, and BadBit CAD recognizes it on the way back, linked to the objects again, even if the insertion was moved or the circle changed in AutoCAD. Marks made in AutoCAD open as an insertion of their anonymous block.

Annotative objects and the annotation scale (CANNOSCALE)

As in AutoCAD, texts, dimensions, leaders and tolerances can be **annotative**: they store the **paper** size and appear in the drawing at the paper size **times the current annotation scale**. A text of 2.5 mm on paper (0.0025 in a drawing in meters) appears at 0.125 at 1:50 and at 0.25 at 1:100, without needing a style for each scale.

- To create annotative objects, check **Annotative** in the style: in STYLE (the text style; the height becomes the paper height), in DIMSTYLE (Lines and Arrows tab, under Fit; it also applies to tolerances) and in MLEADERSTYLE. Whatever is created with the style is annotative, at the current scale;
- The **current annotation scale** is the **CANNOSCALE** list on the status bar (1:1, 1:2, 1:5, 1:10, 1:20, 1:25, 1:50, 1:75, 1:100, 1:200, 1:250, 1:500, 1:1000, 2:1 and 10:1) or the **CANNOSCALE** variable (type `1:50`, `1:100`, `2:1`...). It is saved in the drawing;
- When the scale changes, the annotative objects that **support** the new scale start appearing at it; the others stay as they were. The command line says how many changed, and the change is an undo step;

- With a positive **ANNOAUTOSCALE** (1 to 4), the new scale **is added to all** annotative objects when CANNOSCALE changes, and all of them start appearing at it. Zero or negative (the default is −4) turns it off.

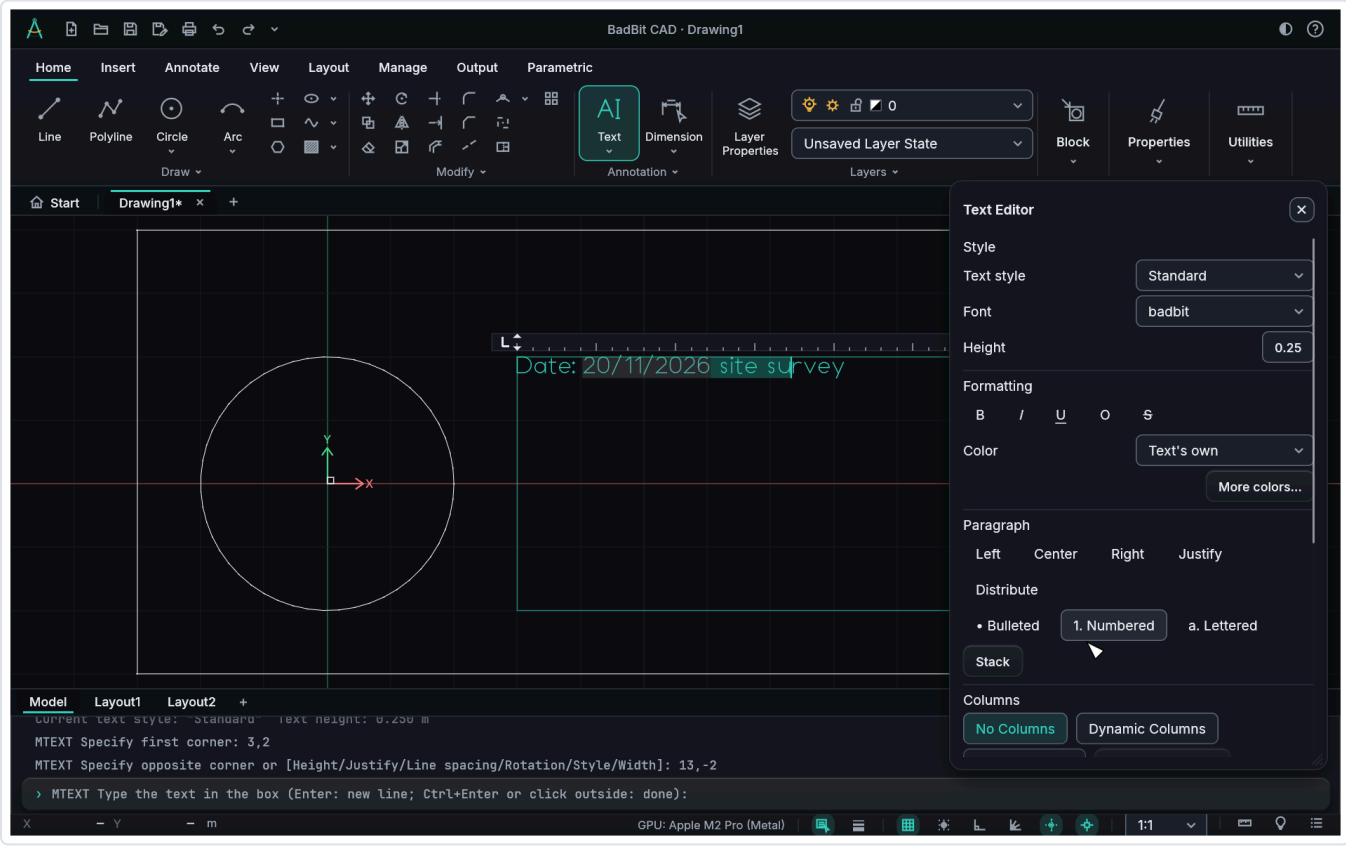
The Properties palette shows the **Annotative** row: **No**, or **Yes** with the scales the object supports (`Yes (1:50, 1:100)`).

Selecting text with the mouse in the editor

In the drawing's text editor (MTEXT, TEXT, TEXTEDIT and table cells being edited), the mouse selects as in AutoCAD:

- **press and drag** inside the text: the cursor goes to where the button was pressed and the selection follows the mouse until the button is released (it does not open the drawing's selection window);
- **Shift+click**: extends the selection from the cursor to the clicked point;
- **double-click**: selects the word (consecutive letters and digits; on a space, a punctuation mark, a fraction or a field, just that);
- **triple-click**: selects the whole paragraph (without the break that ends it).

A single click still puts the cursor at the point, and a click outside the text confirms. With the selection made, the **Text Editor** bar changes its formatting (bold, color, height...), and Ctrl+C, Ctrl+X and Ctrl+V copy, cut and paste.



The word "medição" selected by dragging the mouse; before it, a date field (a single item for the cursor).

Fields (FIELD)

A **field** is a piece of text that is calculated on its own, as in AutoCAD: the date the drawing was created or saved, the current date and time, the file name, or the area and length of an object. In the editor it behaves like **a single letter**: the arrow keys and Backspace go over it as a whole.

Inserting a field

- **In the text editor**: **Ctrl+F** (or the **Field...** button on the Text Editor bar, in the Symbol group) opens the **Field** dialog and the field goes in at the cursor. It works in MTEXT, in **single-line TEXT** (TEXT/DTEXT, also in TEXTEDIT) and in **table cells**.
- **With the FIELD command**: with the editor open, it does the same; without the editor, after OK it asks for the **start point** and creates an MTEXT with the field, at the current MTEXT height, style and alignment (Esc cancels).

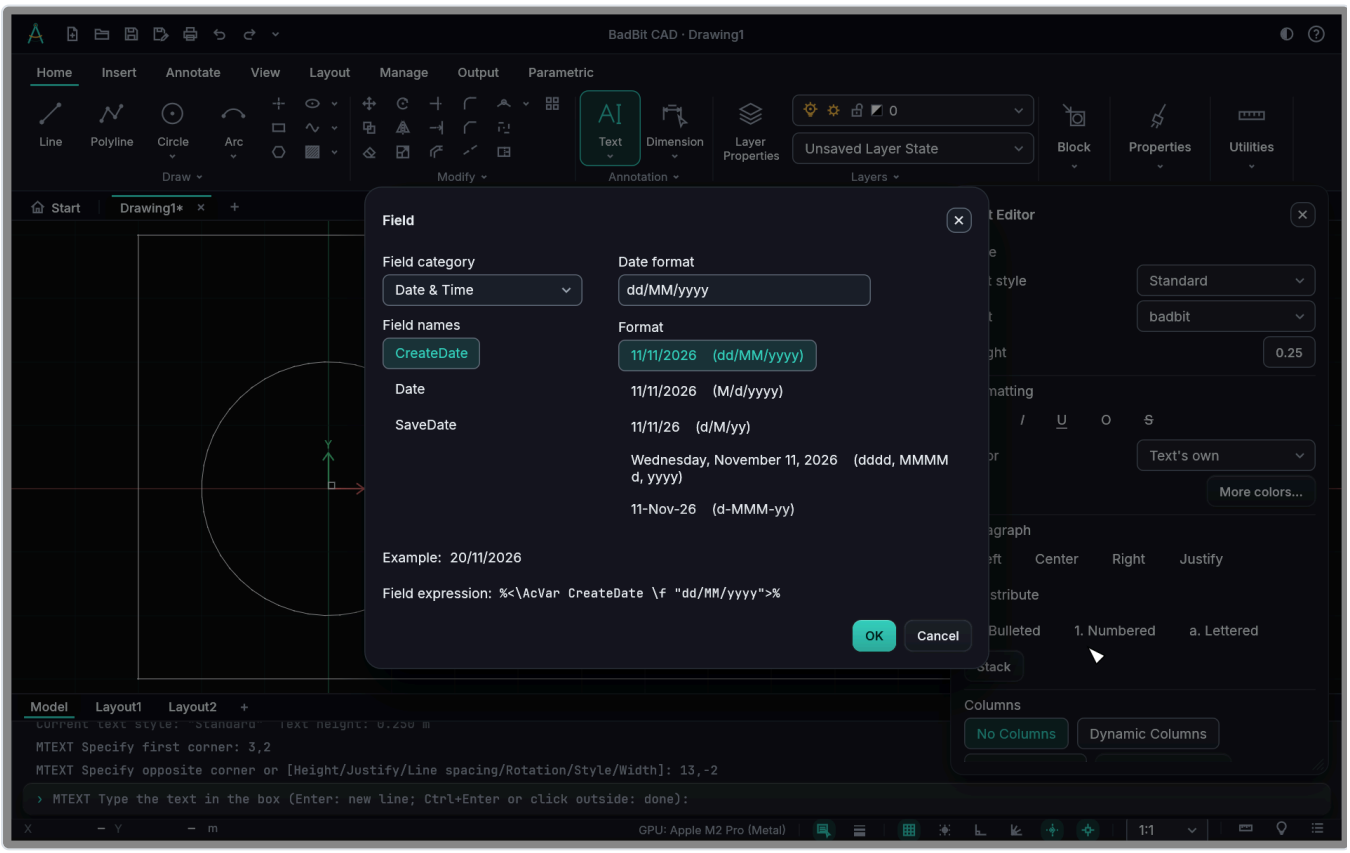
The **Field** dialog has, on the left, the **Field category** and the **Field names**; on the right, the format; below, the **Example** (the value calculated now) and the **Field expression** (the AutoCAD code, such as `%<\AcVar CreateDate \f "dd/MM/yyyy">%`).

Category	Name	What it shows	Format
Date & Time	CreateDate	the date the drawing was created	list of formats or a typed one (`dd/MM/yyyy`, `M/d/yyyy`, `h:mm tt`, `dddd, MMMM d, yyyy`...)
Date & Time	SaveDate	the date of the last save	same
Date & Time	Date	the current date and time	same
Document	Filename	the file name	filename only, path only, path and filename; with or without the extension; the case (uppercase, lowercase, first capital, title case)
Objects	Object	the area or length of an object	**Select object** (the dialog disappears until the object is clicked), the **Property** (Area or Length), the **Units** (decimal or scientific), the **Precision** and the **Conversion factor**

The area and length number comes out with the **drawing's decimal separator**, the same as the dimensions (the DIMDSEP of the current dimension style, the comma in Standard), not with the interface language's: the dialog writes the comma in the code as `%ds44`, as AutoCAD does when the comma is chosen, and AutoCAD shows the same value. A field without `%ds` (from an AutoCAD file) comes out with the period, as there.

In date formats, `d`/`dd` is the day, `M`/`MM` the month (`MMM` and `MMMM` the name), `yy`/`yyyy` the year, `h`/`hh` the 12-hour hour and `H`/`HH` the 24-hour one, `mm` the minutes, `ss` the seconds and `tt` AM/PM; text in single quotes comes out as is. The **names of months and weekdays** (`MMMM`, `MMM`, `dddd`, `ddd`) and `tt` come out **in the interface language**, as AutoCAD does with the Windows language: in Portuguese, `dddd, d 'de' MMMM 'de' yyyy` gives "quarta-feira, 11 de novembro de 2026" (`MMM` gives "nov", `ddd` gives "qua"); in English, "Wednesday, 11 de November de 2026"; in

Spanish, "miércoles, 11 de noviembre de 2026" (and `tt` gives "a. m."/"p. m."). The value stays as it was calculated: changing the language does not change the text until the next calculation (UPDATEFIELD or FIELDEVAL). Area and length come out in the **drawing unit** (the area squared); the conversion factor multiplies the value (0.0001 gives hectares from m², for example).



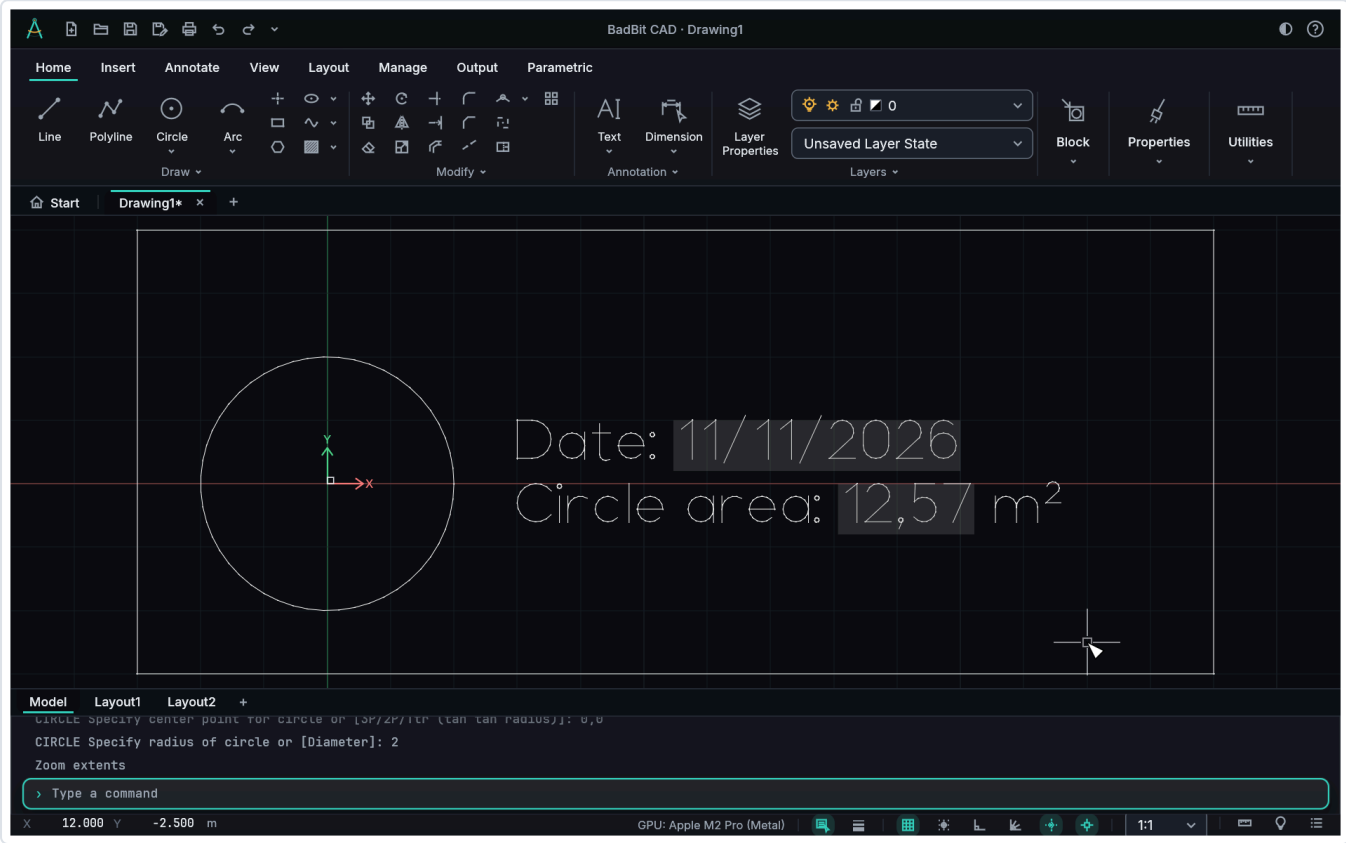
The Field dialog opened with Ctrl+F in the editor: Date & Time, CreateDate, with the list of formats.

Updating fields: UPDATEFIELD, FIELDEVAL and FIELDDISPLAY

- **UPDATEFIELD** asks for objects and recalculates their fields; the command line says how many changed (an undo step). Area and length fields stay **linked to the object**: after scaling the circle, UPDATEFIELD shows the new area; with the object erased, the field shows `####`. A field not yet calculated shows `----`.
- **FIELDEVAL** (0 to 31, default 31) says when fields are recalculated on their own, as in AutoCAD: the sum of **1 on open** (the `.bbd`, DXF and DWG), **2 on save** (before writing the `.bbd`, DXF or DWG), **4 on plot** (PLOT, printing and the preview), **8 on eTransmit** and **16 on regen** (REGEN). BadBit CAD has no eTransmit: the 8 is only stored. With 0, only UPDATEFIELD recalculates. Each recalculation that changes a value is an undo step (UPDATEFIELD).
- **Fields inside blocks**: when the recalculation changes the text of a block item, the block is redefined. All the blocks that changed are redefined at once, **from the inside out**: a block that contains another one that was also redefined receives the new items of the inner one, and the blocks that contain them, the insertions and the arrays receive the rebuilt items, each one only once and in the same undo step. An insertion (or array) whose own text also changed gets the new text and the new items.
- **FIELDDISPLAY** (0 or 1, default 1): with 1, fields appear with a **gray background**, on screen only (never in plots or in the file); with 0, without background.

FIELDDISPLAY and FIELDEVAL are in the **profile** (Variables section; see *Options and profiles*): they apply to all drawings and go along when the profile is exported. In AutoCAD FIELDEVAL is stored in the drawing; in BadBit CAD it follows the profile.

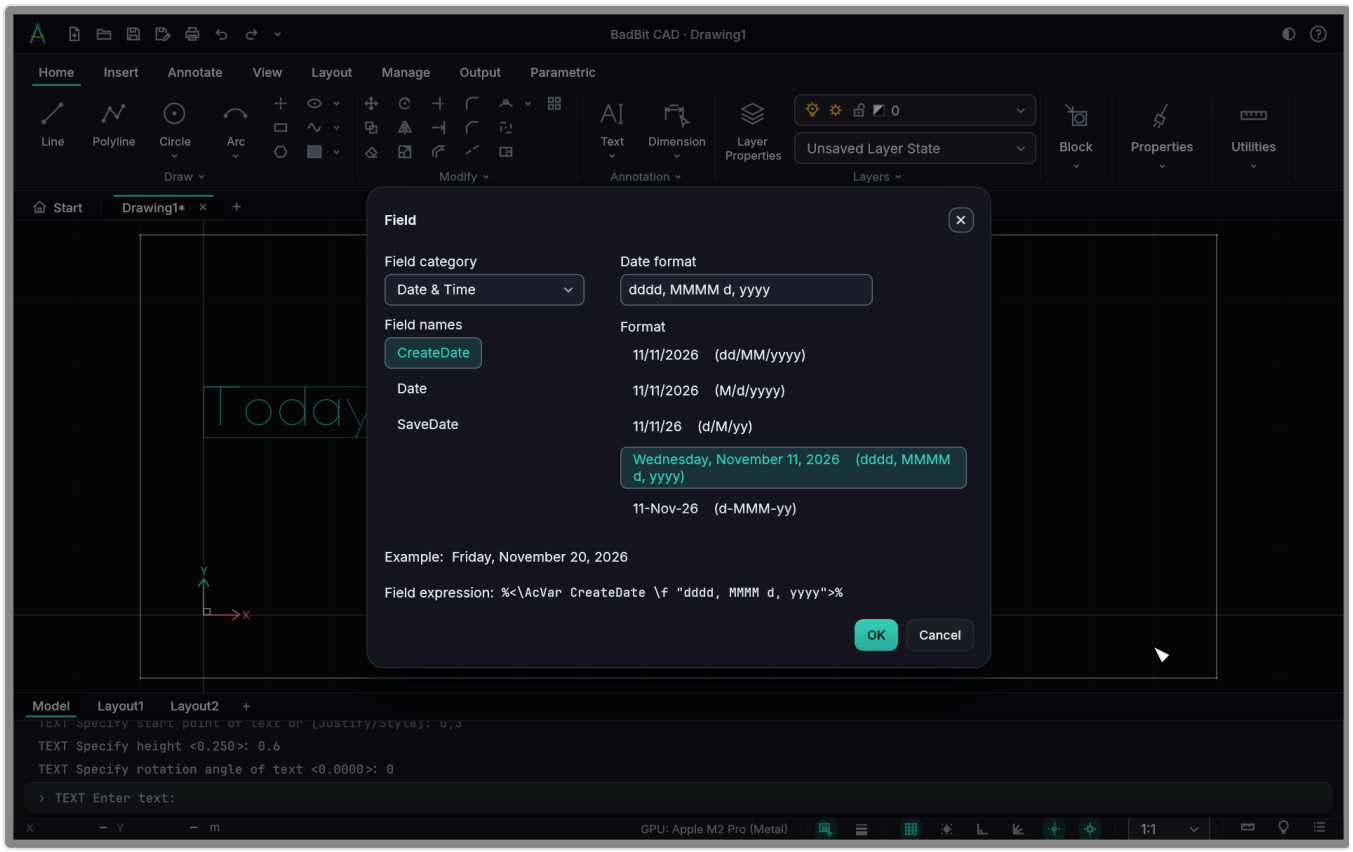
The drawing dates are **TDCREATE** (creation) and **TDUPDATE** (last save), in the computer's local time: they come from DWG and DXF and go into them, and they are kept in the `.bbd` preferences (a new drawing starts with the time it was opened, and each save updates TDUPDATE).



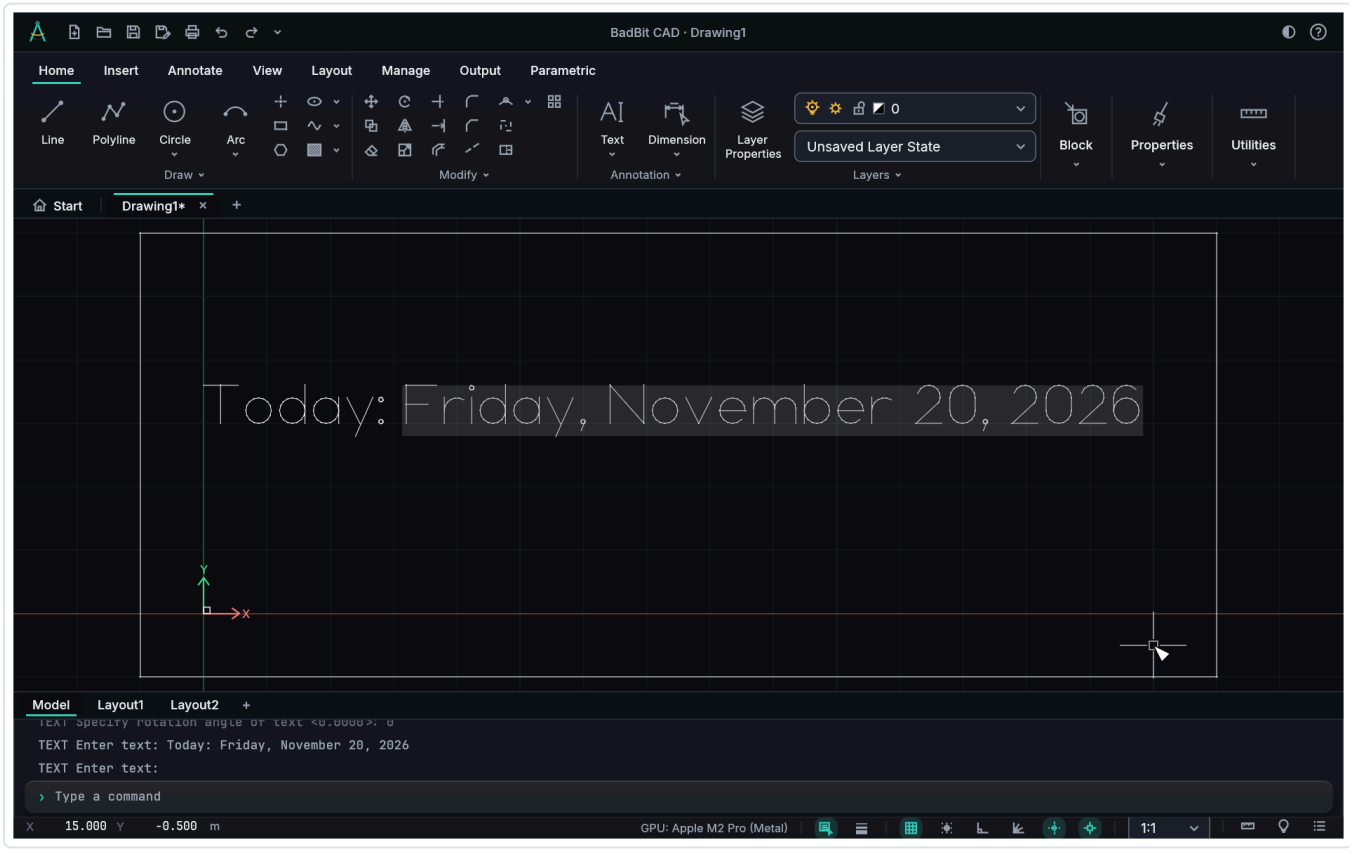
An MTEXT with the date and the area of the circle of radius 2 (12,57, with the comma of the Standard dimension style), with the gray background of FIELDDISPLAY.

Field in single-line TEXT

In TEXT (TEXT/DTEXT) the field goes in with **Ctrl+F** with the editor open, as in MTEXT: the text shows the value, with the gray background of FIELDDISPLAY (also while typing), and the field is a single letter for the cursor. TEXTEDIT opens the text with the fields in place; UPDATEFIELD and FIELDEVAL recalculate them.



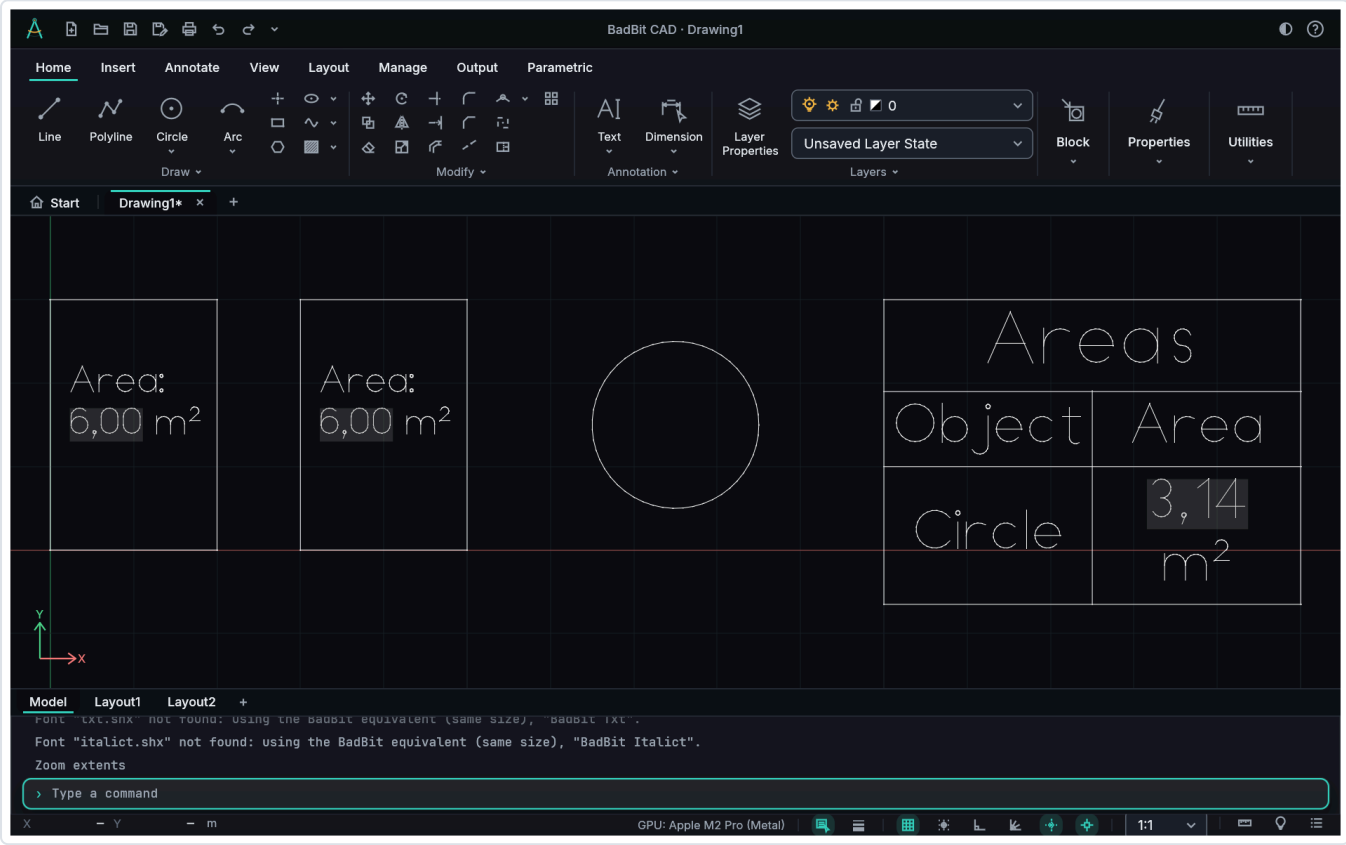
Ctrl+F in single-line TEXT: in the list of formats, the names of the day and month in Portuguese.



The TEXT "Today: " with the CreateDate field in the `dddd, MMMM d, yyyy` format, with the gray background.

Fields in blocks and tables

- **Inside a block**: an area or length field in a block's text can point to **another object of the same block** (as in AutoCAD). Draw the object and the text with the field (Ctrl+F, Objects, Select object), and make the **BLOCK** with both: the field stays linked to the object inside the definition. Each insertion shows the value of the definition (the block's object, in the drawing unit, without the insertion's scale). UPDATEFIELD (choosing an insertion, or with all) and FIELDEVAL recalculate the definition and all insertions change together. In **BEDIT** the field points to the object in the block editor, and goes back to the block on saving; **EXPLODE** links the field of the exploded text to the exploded object.
- **In a table cell**: Ctrl+F in the cell editor puts the field in it (for example, the area of a polyline in the drawing); UPDATEFIELD of the table (or FIELDEVAL) recalculates the cells.
- The gray background of FIELDDISPLAY also appears on fields in insertions and cells. A field in a block that points to an object **outside** the block (in the drawing) also works, and comes back linked to it when the DXF or DWG is opened.



Two insertions of the SALA block (a 2 × 3 rectangle and the MTEXT with its area: 6.00 m²) and a table with the area of the circle of radius 1 (3.14 m²) in a cell.

Fields in files

In **DXF** and **DWG**, MTEXT carries the **displayed value** in the text and the **code** in the FIELD object of its extension dictionary (`ACAD_FIELD` → `TEXT`), the way AutoCAD stores it: AutoCAD opens the file with the fields working, and BadBit CAD reads back the same code (the area field comes back linked to the object read). The `.bbd` keeps the code and the last value. Fields that BadBit CAD does not yet calculate (other AutoCAD ones) are kept and saved unchanged, with the value they came with.

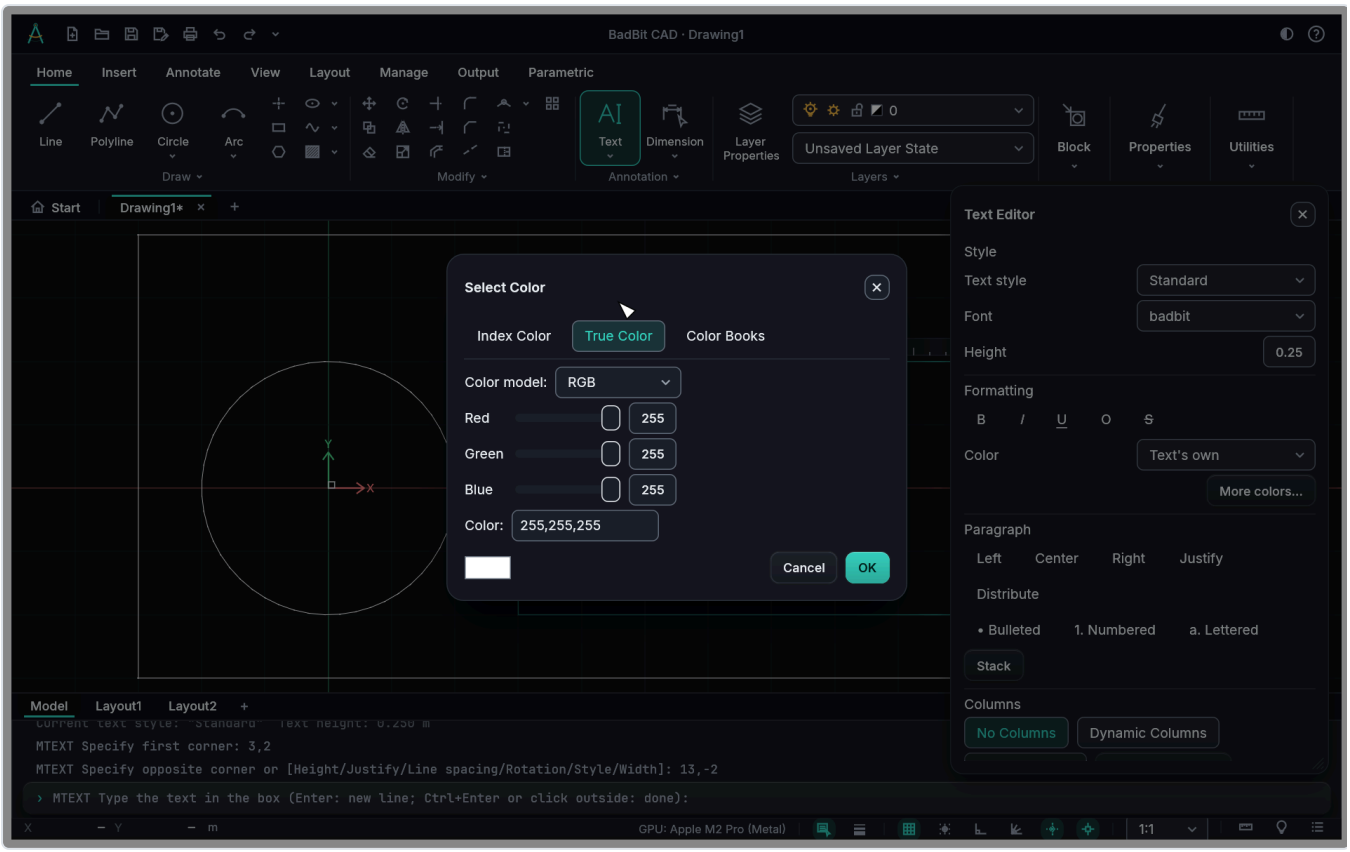
- **Single-line TEXT** with a field goes the same way: the value in the text and the root FIELD with the plain text (`Today: %<\_FldIdx 0>%`) and the inner fields; it comes back with the same fields (in DXF R12, which has no FIELD, only the value is written).
- In a **block**, the field's object (the 331 of the FIELD) is the entity of the same block, and reading brings the field back linked to the block's object.
- In an AutoCAD **table cell** (ACAD_TABLE, from 2004 onward), the field goes and comes back the way AutoCAD stores it: the cell points (the 344) to the root FIELD, which belongs to the table, without the cell text (the displayed value goes in the cell value, from 2007 onward); reading brings the field back linked to the object read, in DXF and DWG — also in a table edited in AutoCAD. In 2000 (with no AutoCAD table) the field comes back through BadBit CAD's data, linked to the object in the same way.

The color of a piece of text: the Select Color dialog

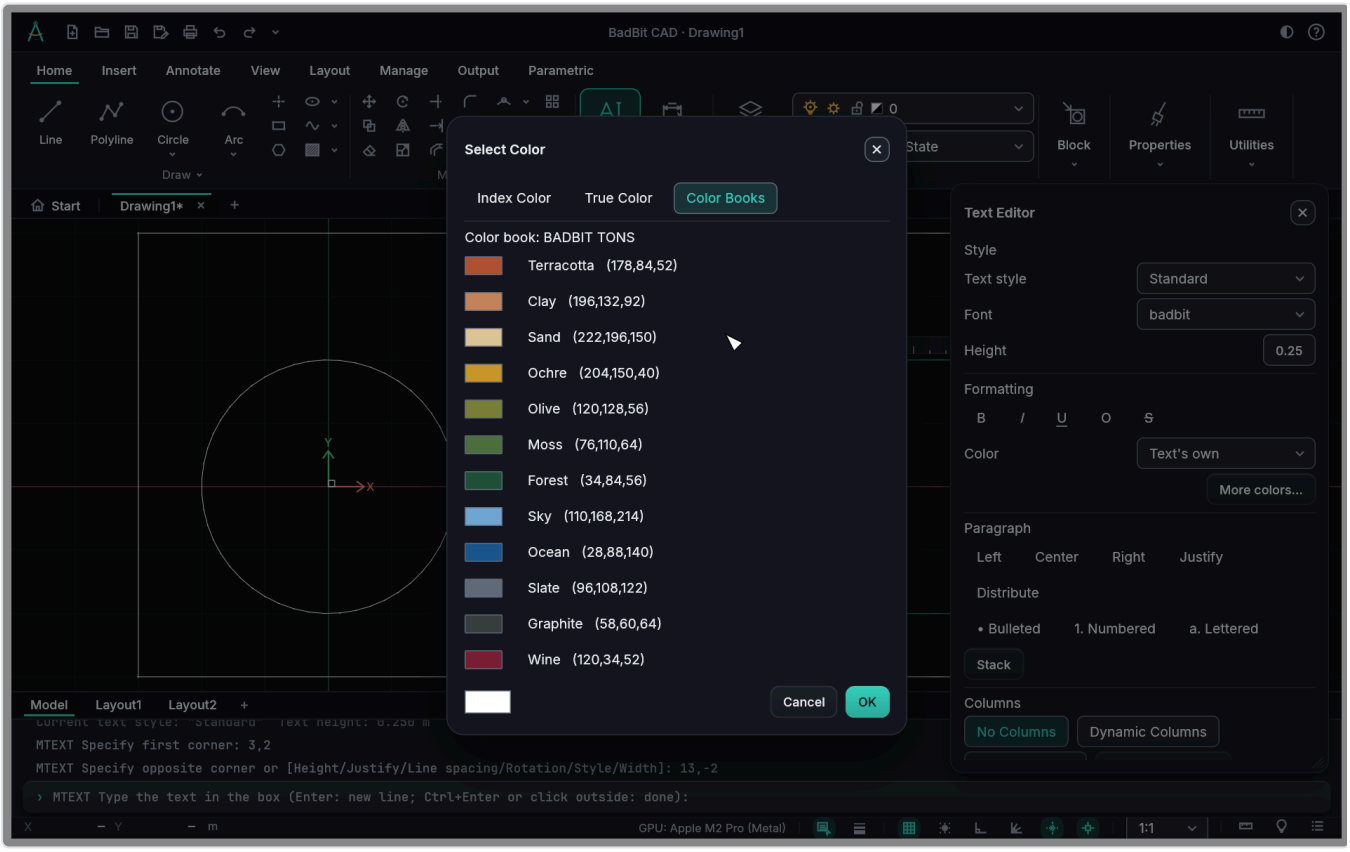
On the **Text Editor** bar, next to the **Color** list (the basic colors, ByLayer and ByBlock), the **More colors...** button opens the **Select Color** dialog with the three AutoCAD tabs; the chosen color goes to the selection (or to what is typed next):

- **Index Color**: the grid of AutoCAD's 255 colors, the nine standard ones and the grays, the **ByLayer** and **ByBlock** buttons and the **Color** field (the number, or BYLAYER and BYBLOCK). Double-clicking a color confirms.
- **True Color**: the **Color model RGB** (red, green and blue from 0 to 255) or **HSL** (hue from 0 to 360, saturation and luminance from 0 to 100), and the **Color** field with `R,G,B`.
- **Color Books**: BadBit CAD's own book, **BADBIT TONS**, with twelve named tones (Terracotta, Clay, Sand, Ochre, Olive, Moss, Forest, Sky, Ocean, Slate, Graphite and Wine). The formatted text of MTEXT stores only the index color or the true color, so a book color goes into the piece of text through its RGB (in AutoCAD it is the same: the book name is kept only in the color of the whole object).

The Color list on the bar shows the true color chosen at the end (`RGB 178,84,52`).



The True Color tab, in the RGB model.



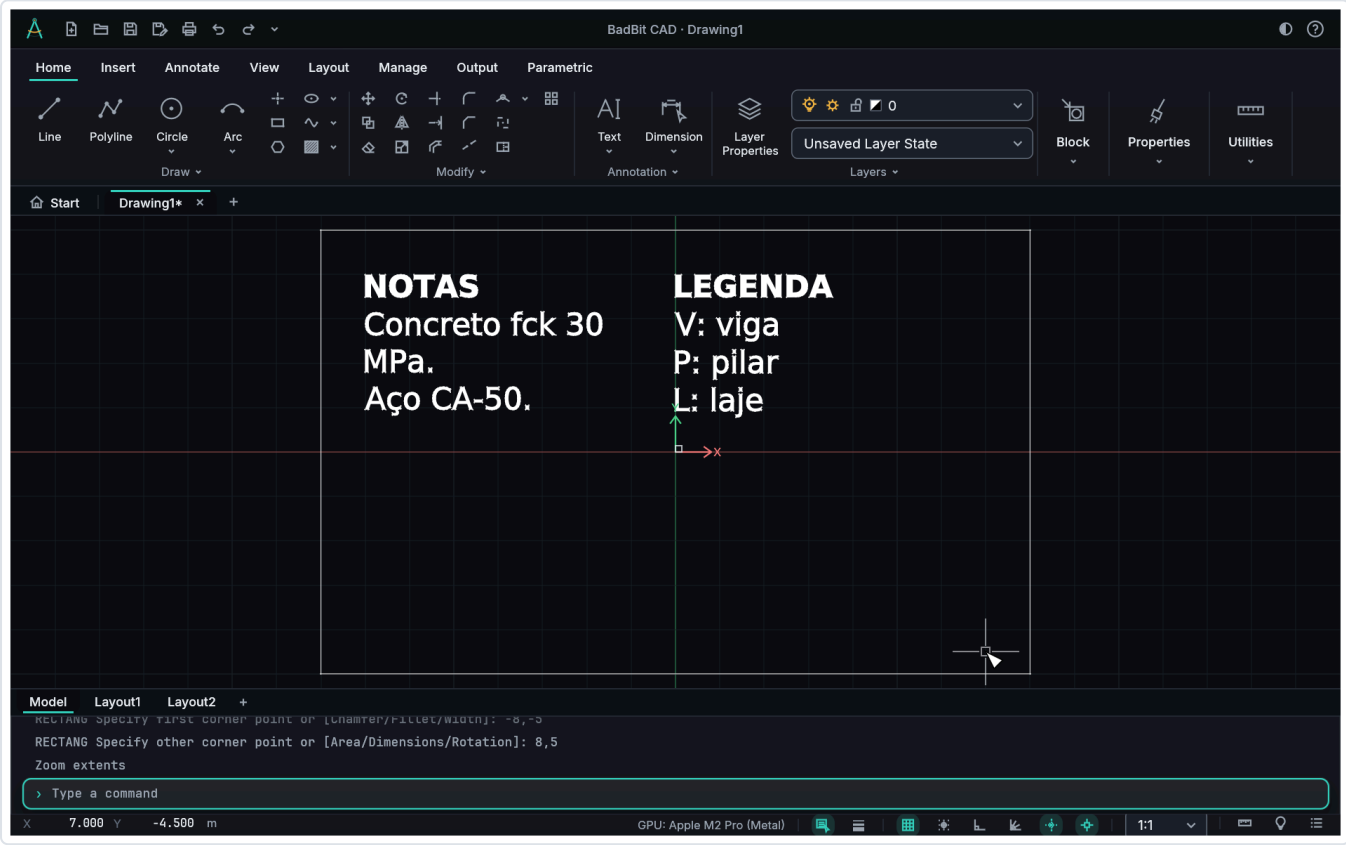
The BadBit CAD color book.

MTEXT: column break, ruler indents, reversed flow and mask in draw order

Column break (Alt+Enter)

In MTEXT with columns, **Alt+Enter** in the editor (or the **Insert Column Break (Alt+Enter)** button in the Columns part of the Text Editor palette) inserts a **column break**: the rest of the text starts at the top of the next column, even if it would still fit in the current one. It is the `\N` code of the AutoCAD MTEXT, written and read as `\N` in the `.bbd`, DXF and DWG of all versions (before AutoCAD 2018, the column that ends at the break carries the `\N` at the end of its MTEXT). As in AutoCAD:

- without columns, the button is disabled and Alt+Enter does nothing; a `\N` coming from a file without columns breaks the paragraph like an Enter;
- in the last static column, the break only starts a new paragraph (there is no next column);
- in dynamic columns, the break opens a new column;
- Backspace right after it removes it, and so does the editor's Ctrl+Z.



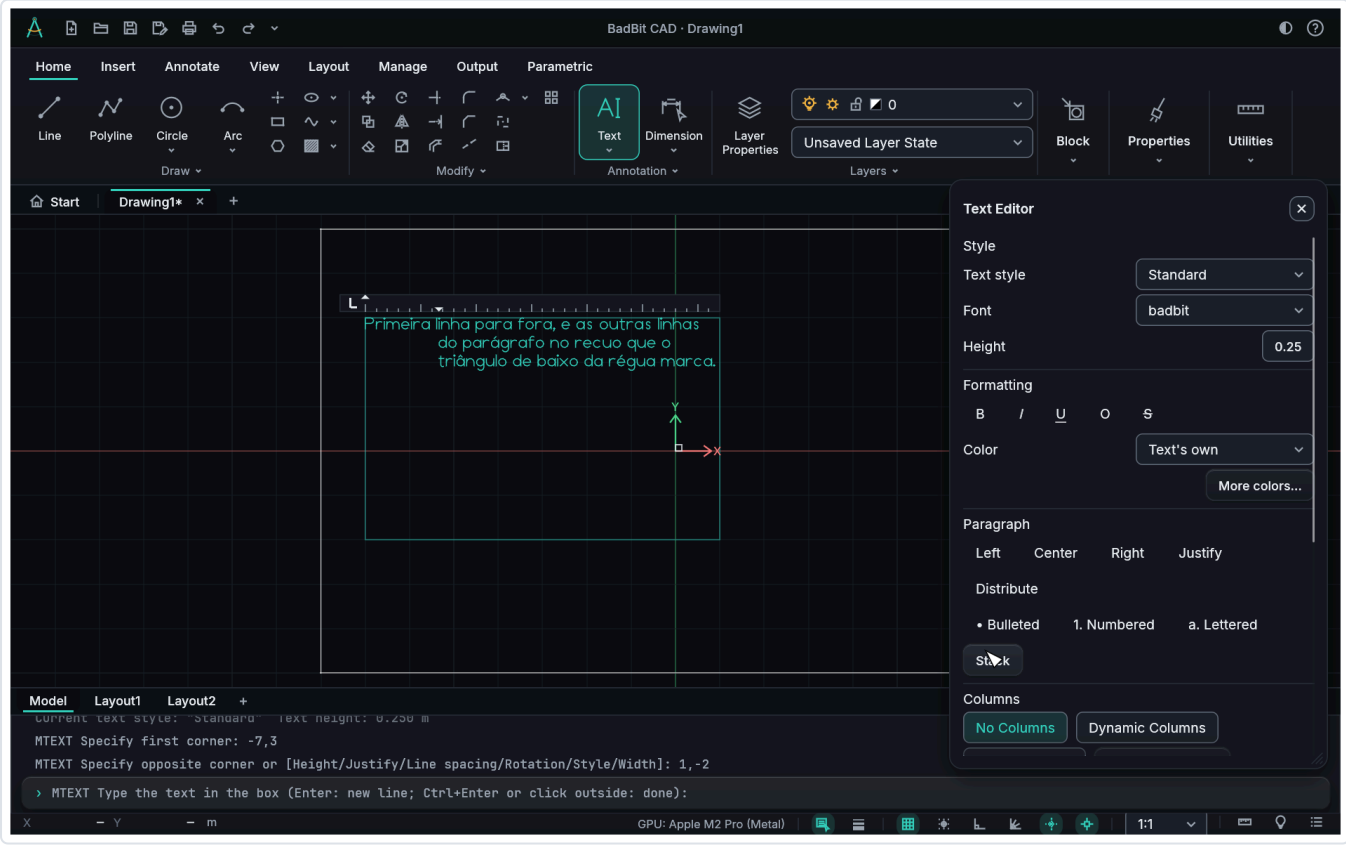
Two static columns: the break after "Aço CA-50." takes LEGENDA to the top of the second column.

Ruler indents

On the ruler above the text being edited, the two triangles on the left are the indents of the paragraph at the cursor (or of the selected paragraphs), as on the AutoCAD ruler:

- the **upper** one is the **first line** indent: dragging it changes only that line;
- the **lower** one is the **paragraph** indent (the other lines): dragging it moves the other lines and leaves the first one where it was; with the first one farther out, it is the **hanging indent** of lists.

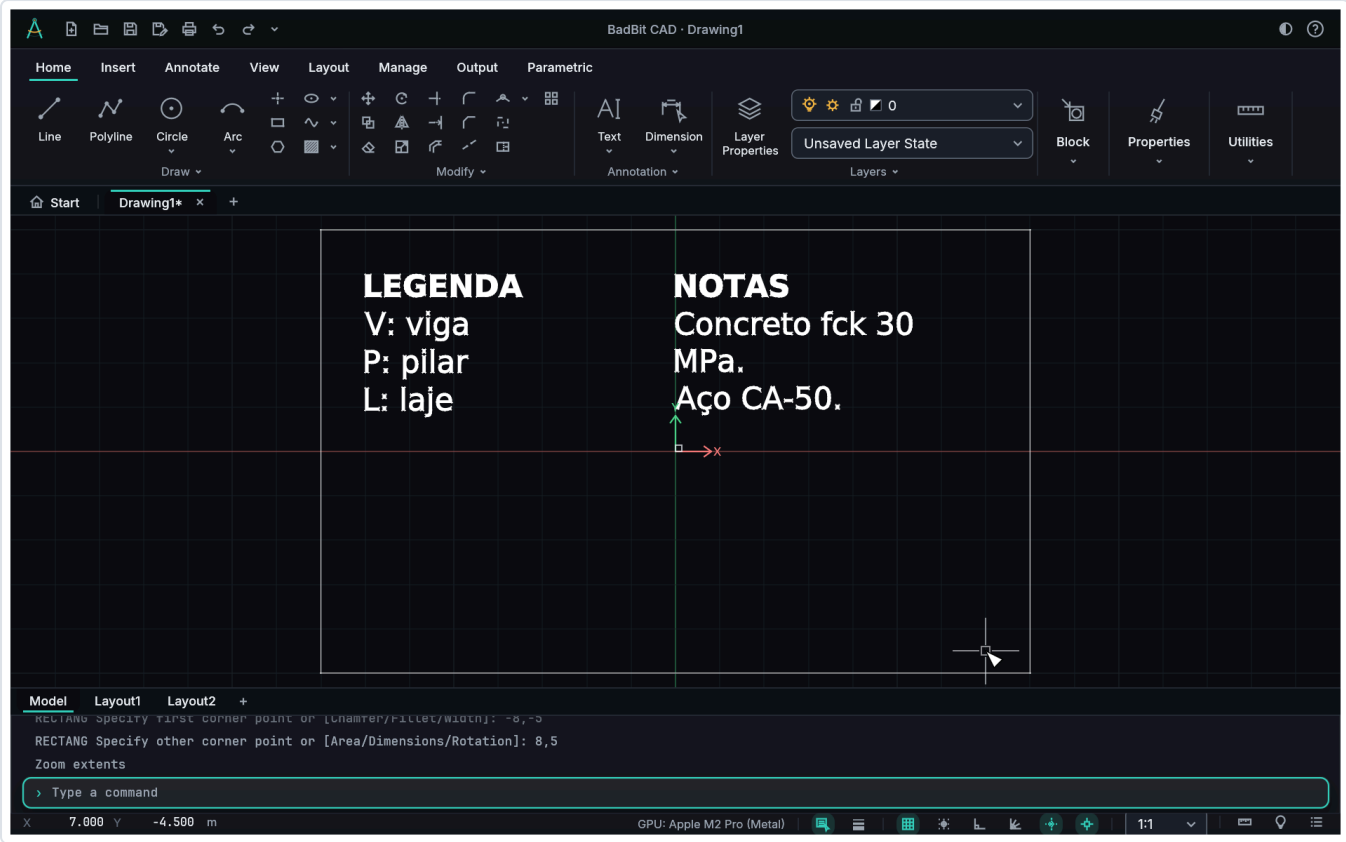
The value goes, in text heights, into the paragraph's `\pi…,l…;` code, rounded to 0.01 height; the editor's Ctrl+Z undoes it. Clicking a triangle without dragging does not create a tab stop.



The lower triangle dragged to the right: the first line stays farther out (hanging indent).

Reversed column flow

The AutoCAD MTEXT stores the **reversed flow** of the columns (code 78; the API's `ColumnFlowReversed` property, for languages written from right to left): the **first column is on the right** and the text flows to the ones on the left. BadBit CAD reads, draws and saves it that way (the block stays in the same place and with the same width). In the **Column Settings** dialog, the **Reversed flow (first column on the right)** box turns it on and off (AutoCAD does not have this box: only the file and programming change the flow).



The same text as in the column break, with reversed flow: NOTAS in the right column.

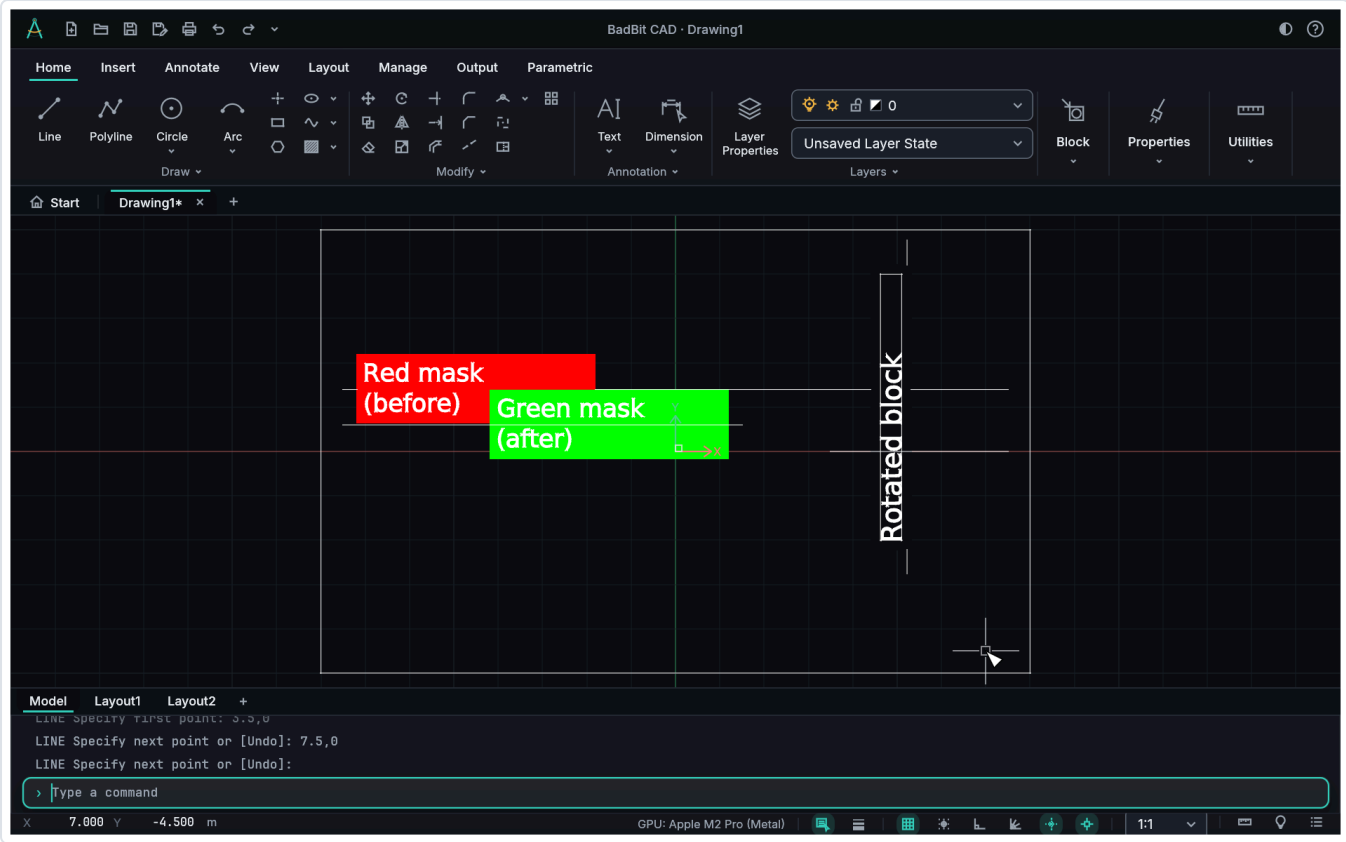
The background mask in draw order

The background mask hides only what comes **before** it in the **draw order** (the order in which the objects were created; for an opened file, the file's order); what comes **after** stays on top, as in AutoCAD:

- a line drawn before the text disappears beneath it; one drawn after it passes over the mask;
- two masks that cross: the one of the text created later stays on top of the other;
- a hatch sent to the back stays beneath all masks, and one brought to the front, on top;
- inside a **block**, the masked text hides the items that come before it in the block and, with the insertion, what came before it in the drawing; the insertion can have scale and rotation;
- masked text on a layer that is off or frozen hides nothing.

The screen (GPU and CPU) and plots show the same. The **text frame** (AutoCAD's TEXTFRAME, the **Text frame (TEXTFRAME)** box in the Background Mask dialog) outlines the text block, with or without the background, and also comes out in plots.

In **AutoCAD 2000 DXF and DWG**, MTEXT has nowhere to store the mask (it was introduced in AutoCAD 2005, the 2004 format): the file is written without it and the save report warns under **Simplified or converted** (`MTEXT BACKGROUNDMASK`). From 2004 onward, it goes in full.



The upper line came before the texts and disappears beneath the masks; the middle one came after and passes over them. The green mask, created after the red one, stays on top of it. On the right, a block rotated 90° with the masked text and the frame: it hides the line of the block itself and the one above, and the horizontal line created afterwards passes over it.

AutoCAD SHX fonts

SHX fonts are AutoCAD's stroke fonts (txt, simplex, romans, romand, romanc, italic, italict, isocp, monotxt and the others): each letter is a handful of lines and arcs, with no fill. BadBit CAD **reads `.shx` files** — the regular fonts, the Unicode ones and the *big fonts* of Asian languages — and draws texts with them as AutoCAD does:

- the **text height is the height of the capitals** of the SHX (in TrueType, that of the font's A);
- the **width** of each letter is the font's, and the width factor, the obliquing angle, **Backwards**, **Upside down**, underline and overline (`%%u`, `%%o`) and the symbols `%%c` (⌀), `%%d` (°) and `%%p` (±) work the same way (in non-Unicode SHX fonts, at AutoCAD codes 127, 128 and 129);
- a letter the font does not have comes out in the BadBit CAD stroke font, in its place;
- the **text box is that of the drawn strokes**, as in AutoCAD: right-justified text ends at the last stroke (not at the space after the last letter), centered text sits in the middle of the strokes, Align and Fit fit up to the last stroke, **Middle** sits at the middle of the height of the strokes, and zoom extents and window selection use the box of the strokes. Spaces at the end of the text count (right-justified text with trailing spaces ends before the point, as in AutoCAD); leading ones do not;
- the symbols `%%c`, `%%d` and `%%p` of a text coming from DXF or DWG come out drawn (the code stays stored in the text and goes back unchanged on saving);
- the style can have a **big font** along with it, as in the AutoCAD -STYLE: `txt.shx,extfont.shx`; letters that the main font does not have come from the big font;

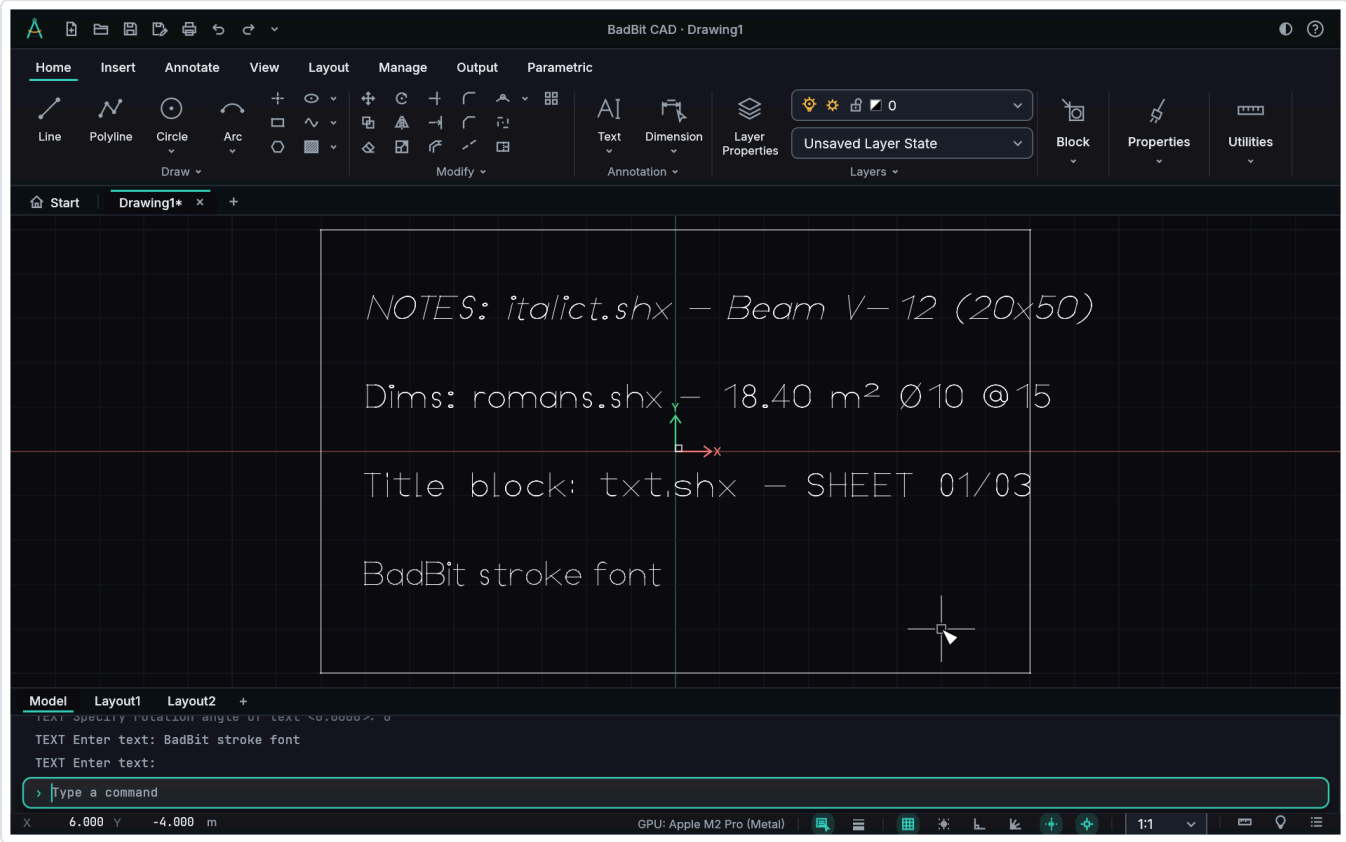
- **Asian** big fonts (Japanese, simplified and traditional Chinese, Korean) store each letter by its code in the region's code page — Shift-JIS, GBK, Big5 or KSC —, not by Unicode: BadBit CAD looks the letter up by that code. The code page comes from the drawing (the `$DWGCODEPAGE` of the DXF or DWG: ANSI_932, ANSI_936, ANSI_950 or ANSI_949); in a drawing with a Latin code page, the one indicated by the name of known big fonts applies (`extfont`, `bigfont` → Japanese; `gbcbig` → simplified Chinese; `chineset` → traditional Chinese; `whgtxt` and the `whg*` ones → Korean) or the ranges in the file (Shift-JIS ones are recognized on their own);
- **Vertical**: in SHX fonts that have the vertical mode, STYLE lets you check **Vertical** (it is grayed out for the others, as in AutoCAD; in -STYLE, the **Vertical?** prompt comes after **Upside down?**, only for these fonts) and single-line TEXT comes out with the letters one below the other, starting at the text point. MTEXT has no vertical mode, as in AutoCAD.

BadBit CAD does not come with the Autodesk fonts (they belong to Autodesk and come with the AutoCAD license). It looks for the files in the **font search path** (in the Options dialog, **Files** part; see *Options and profiles* in the interface chapter): the default is the font folders of AutoCAD and DWG TrueView installed on the computer and their user support folders — on the Mac, the `Fonts/shx` and `Support` inside `AutoCAD 20xx.app` and the user `Support` in `~/Library/Application Support/Autodesk/AutoCAD 20xx`; on Windows, the `Fonts` and `Support` of `Arquivos de Programas\Autodesk\AutoCAD 20xx` and the user `Support` in `%APPDATA%\Autodesk`. The **folder of the open drawing** always applies — an SHX next to the file is found on its own. The folders are searched once, when the program opens, without slowing down the screen; a font is read the first time a text uses it.

An SHX that is not on the computer

When the style's SHX is not in any folder, BadBit CAD does as AutoCAD does: it uses the **FONTALT** font (a variable stored in the profile; the default is `simplex.shx`), if it is in one of the folders. If not even that one is there — or with `FONTALT` set to `.` (period) —, it uses the **BadBit CAD equivalent**: a single-stroke font of our own, with the **same metrics** as the missing SHX (the capital height, the descender and the width of each letter), so the text takes up the same space as in AutoCAD and still fits in the same frames and title blocks. There are equivalents for txt, monotxt, simplex, romans, romand, romanc, romant, complex, italic, italicc, italict, isocp and isoct (and their variants); for other SHX fonts, the simplex one.

The command line warns **once** per font (`Font "italict.shx" not found: using the BadBit CAD equivalent (same size), "BadBit Italict".`), and the DXF and DWG **Import report** says the same at the top: `Font italict not found: using the BadBit CAD equivalent (same size).` (with FONTALT found: `using simplex.shx (FONTALT)`). The SHX name stays stored in the style and goes back when the DXF and DWG are saved: opened on a computer that has the font, the text comes out in it.

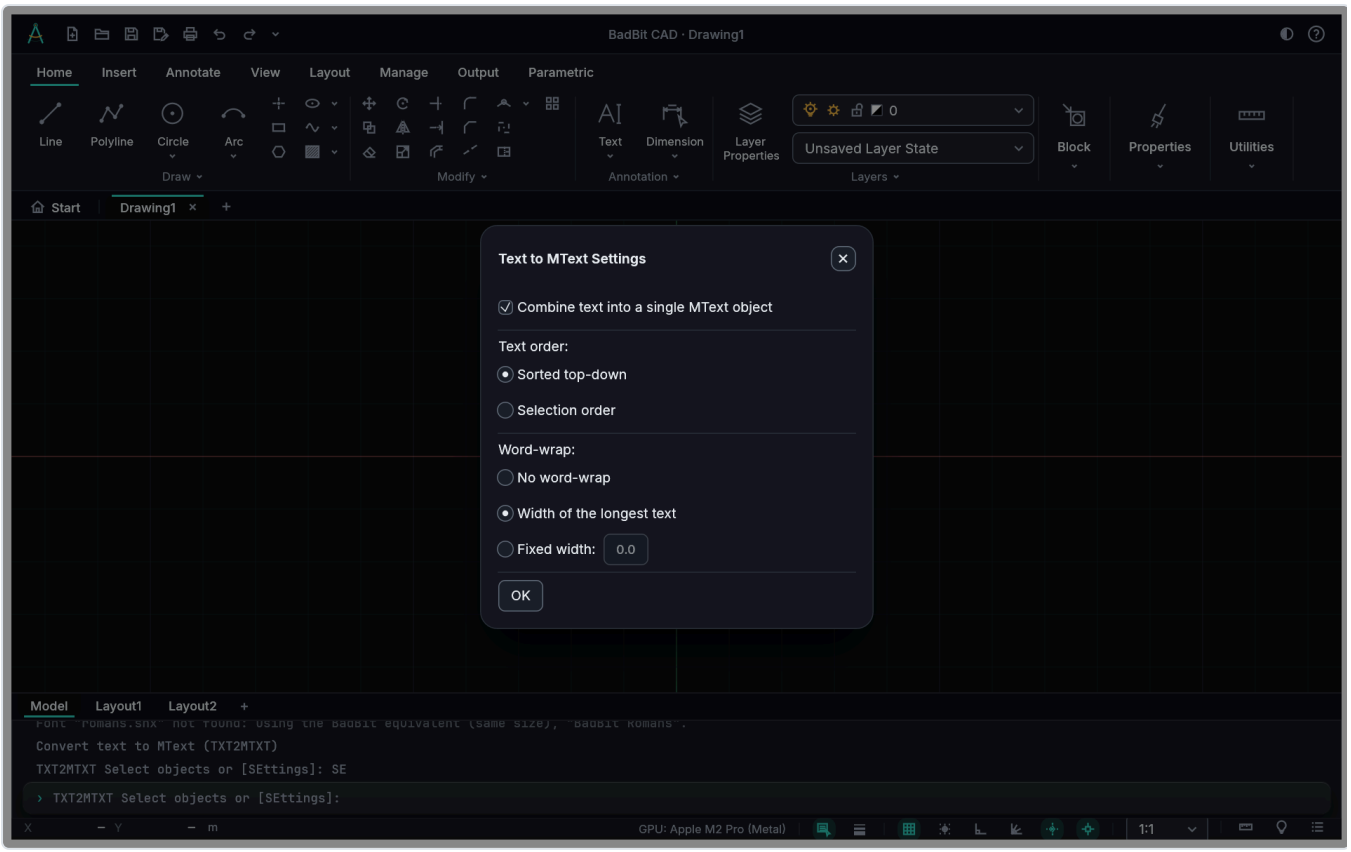


Three styles with SHX fonts that are not on the computer (italict, romans and txt), drawn with the BadBit CAD equivalents of the same size; below, the BadBit CAD stroke font.

In STYLE, the font list also includes the SHX fonts found in the search folders; a missing SHX appears in the list under its stored name. In **-STYLE**, type the SHX file (`romans.shx`, or `txt.shx,extfont.shx` with the big font): an SHX that is not on the computer is accepted (texts use FONTALT or the equivalent, with the warning). The BadBit CAD stroke font goes into DXF and DWG as `romans.shx` (which every AutoCAD has) and comes back as itself when opened in BadBit CAD.

Texts into MTEXT: TXT2MTXT

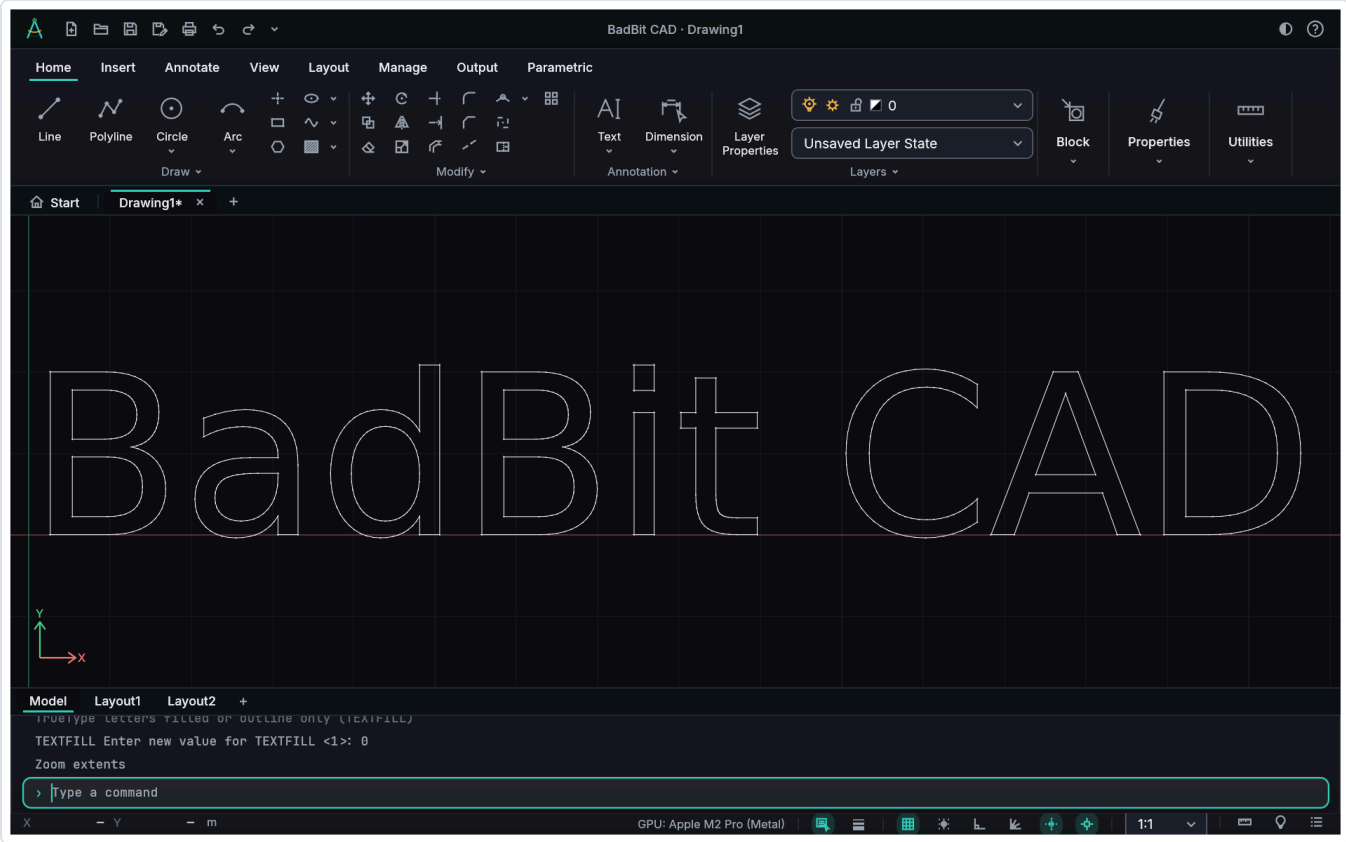
Converts TEXTs (and MTEXTs) into a single MTEXT (one paragraph per text) or into one MTEXT per text. The MTEXT goes at the top corner of the first one, with its rotation, height, style and font, on its layer; texts of another height keep their own height. At the **Select objects** prompt, the **SEttings** option (`SE`) opens the dialog: **Combine text into a single MText object**, the **order** (sorted top-down, or in selection order) and the **word-wrap** (no word-wrap, by the width of the longest text or a fixed width). The choices are kept in the profile. It is on the open Annotation panel (Home tab).



The Text to MText Settings dialog.

Outline-only TrueType letters: TEXTFILL

The **TEXTFILL** variable (in the profile, as in the AutoCAD registry) shows the letters of TrueType fonts filled (1, the default) or as outlines only (0). It applies on screen, in plots and in PDFs; EXPLODE of a TrueType text still gives the outlines.



A TrueType text with TEXTFILL 0: only the outline of the letters.

Scaling texts in place: SCALETEXT and TSCALE

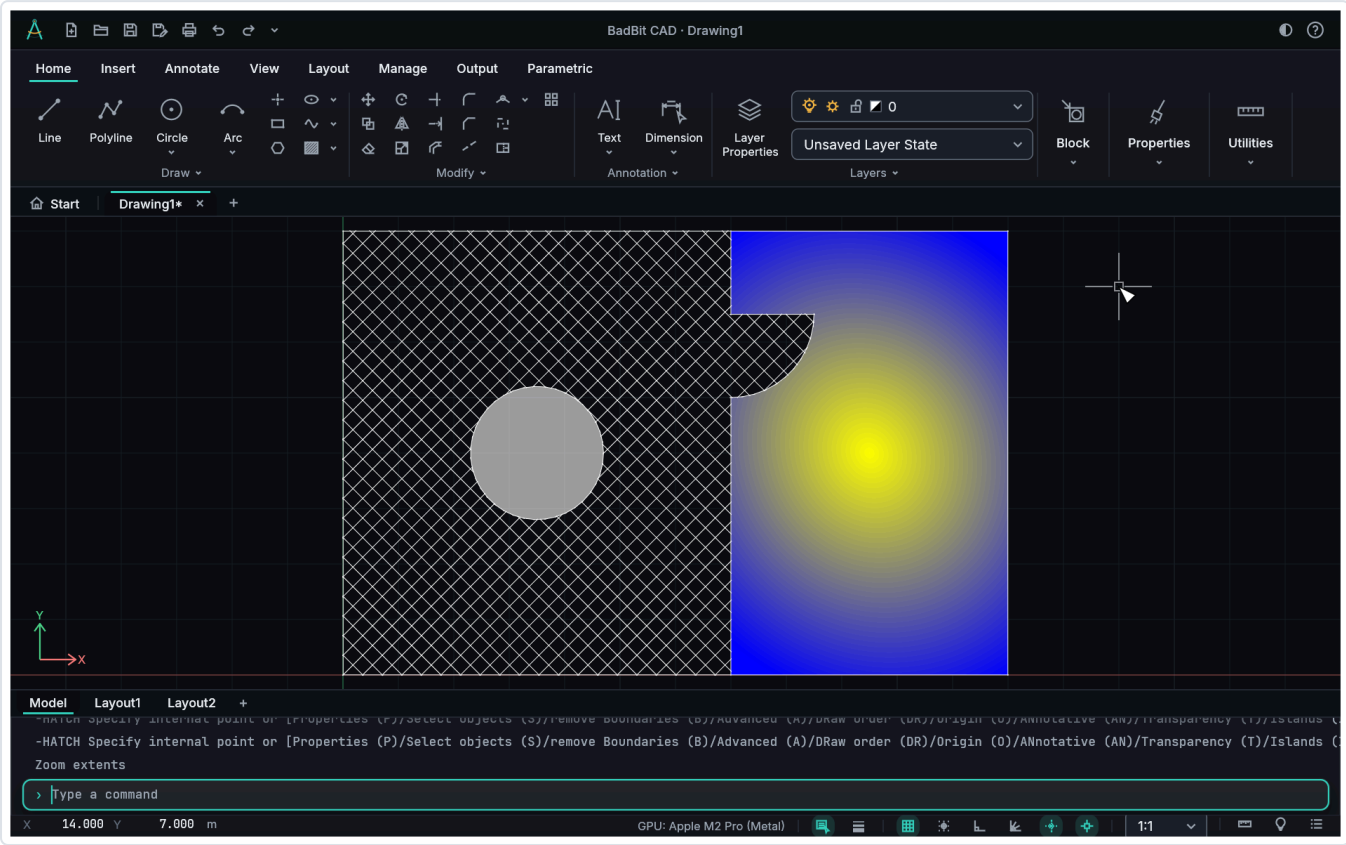
SCALE enlarges the text and also moves it away from the base point. **SCALETEXT** changes only the size: each text, MTEXT, attribute definition and attribute (of the chosen insertions) grows from a point of its own, without leaving its place. After choosing the objects (or with them chosen beforehand):

1. **Enter a base point option for scaling [Existing/Left/Center/Middle/Right/TL/TC/TR/ML/MC/MR/BL/BC/BR] <Existing>: Existing** uses the justification point of each text; the others use that point of the text box, and the justification of the text does not change. In single-line text, Left, Align and Fit count as BL.
2. **Specify new model height or [Paper height/Match object/Scale factor] <2.5>:** the model height changes only non-annotative texts; **Paper height** changes only annotative ones (the height in the model becomes the paper height times the annotation scale shown); **Match object** takes the height of another clicked text (only those of the same kind, annotative or not, change); **Scale factor** multiplies all of them (with **Reference (R)**: the reference length and the new one).

TSCALE is the Express Tools one: "Specify justification to use as base point for scaling [Existing/Start/Center/...] <Existing>:", then "[Scale (factor)/Height] <Scale>:" and the **Scale factor** (default 1.2) or the **Final height**, which applies to all the chosen texts, annotative or not. Both commands remember the last answers and are a single undo step; texts on a locked layer are left out.

Hatches and gradients

A **hatch** fills a closed area with a pattern of lines (concrete, sand, bricks, steel…), with a solid color, or with a gradient. It is a single object: it is selected with a click inside the area or on the edge, and it moves, copies, rotates, scales, and mirrors like the others. The **Hatch** button is on the **Draw** panel of the Home tab; its small arrow gives **Gradient** and **Boundary** (BOUNDARY).



The living room with ANSI37 (the table is left out, it is an island), the bedroom with a spherical gradient, and the solid table, half transparent.

HATCH (aliases H, BH)

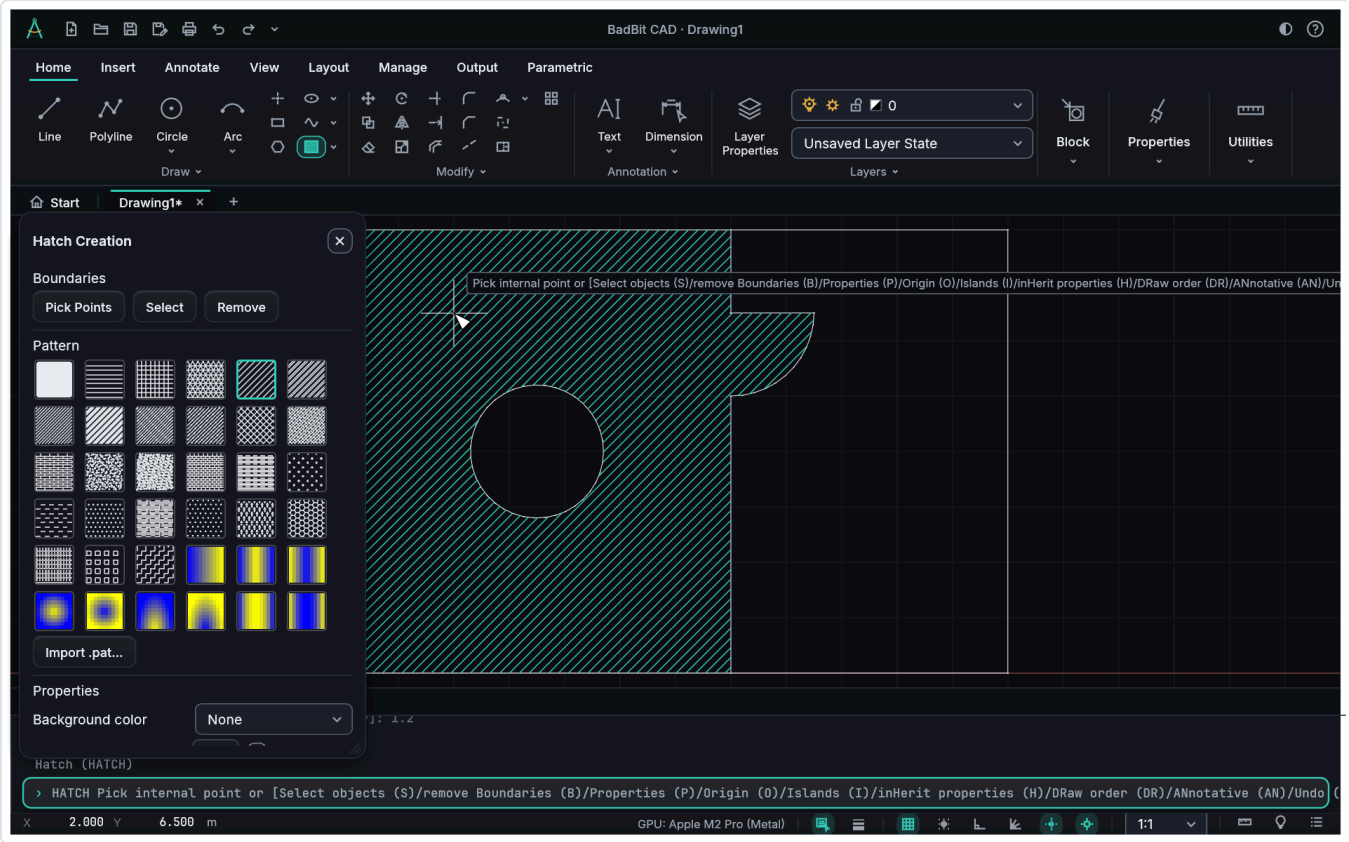
When it starts, the **Hatch Creation** panel opens next to the drawing (it is the equivalent of AutoCAD's contextual tab, as a palette that does not get in the way of clicking in the drawing) and the command line asks:

```
Pick internal point or [Select objects (S)/remove Boundaries (B)/Properties (P)/Origin (O)/Islands (I)/inHerit properties (H)/DRaw order (DR)/ANnotative (AN)/Undo (U)/seTtings (T)]:
```

- **Internal point** (the click): the closed area around the point, formed by any objects that intersect (lines, arcs, circles, polylines, ellipses, splines, blocks). When you **move the cursor** over an area, its hatch appears in the preview, as in AutoCAD. The objects inside the area (a column, a pillar) become **islands**. Each click adds an area; `Enter` creates the hatch with all of them.
- **Select objects** (`S`): uses the outline of closed objects (closed polyline, circle, closed ellipse and spline, region), by clicking or by window. `K` goes back to internal points.

- **remove Boundaries** (`B`): click the edge of a chosen area to remove it. `A` goes back to adding.
- **Properties** (`P`): the pattern name (`?` lists the patterns; `S` solid; `U` user defined, with the spacing and the double crosshatch; `G` gradient), the **scale**, and the **angle**.
- **Origin** (`O`): the point where the pattern starts (to align bricks with a corner, for example).
- **Islands** (`I`): island detection (HPISLAND): **Normal** alternates from outside to inside (the island's island is hatched again), **Outer** hatches only up to the first islands, and **Ignore** goes over them.
- **inHerit properties** (`H`): choose a hatch in the drawing; pattern, scale, angle, islands, transparency, and background color carry over to the next one.
- **Undo** (`U`): removes the last area or object chosen.
- **seTtings** (`T`): shows the panel again, if it was closed.

A point outside any closed area: the notice *Valid hatch boundary not found* and the command continues. A point between two **nearly coincident** curves (closer together than the drawing's tolerance, a billionth of its size, along a stretch, like two splines drawn one over the other): the area between them has no defined boundary and the hatch is not created, with the notice *Hatch not created at that point: two curves there are nearly coincident*; the area would come out wrong. Move the curves apart or make one coincide with the other. The area found **does not depend on the zoom**: the internal point considers every visible object of the drawing (those on layers that are on), on the screen or not. The objects in view only speed up the search; an area that does not fit in the view is searched for in the neighborhood of the point, growing up to the drawing extents. The preview under the cursor shows only the areas that fit in the view (the others appear on the click). In a huge drawing (more than 60 thousand objects in view, or in the neighborhood of the point), the preview under the cursor is turned off and the search stops at that size. A point whose area contains another area already chosen is refused, with the notice *That point is not added*: with the hatch's even-odd fill, the inner area would be left empty (pick only the outer one, which already takes the islands, or undo the inner one with `U`). A point in an island of a chosen area (a room inside the house, with the area around the house already chosen) is added normally. Through automation, each refused point appears in the command result, with the notice.



HATCH with the cursor over the living room: the preview shows the hatch, with the table left out.

The Hatch Creation panel

- **Boundaries**: Pick Points, Select, Remove.
- **Pattern**: the thumbnails of the patterns (SOLID first) and of the gradients; **Import .pat...** adds the patterns from an AutoCAD `.pat` file.
- **Properties**: the **background color** (behind the pattern lines) or the **gradient colors** (setting the second one to None makes a one-color gradient, which goes to the chosen tint), **Centered**, the **transparency** (0 to 90%), the **angle**, and the **scale**.
- **Origin**: Set Origin (asks for the point) and Origin 0,0.
- **Options**: **Associative**, **Annotative**, island detection, draw order (behind or in front of the boundary), and Inherit Properties.
- **Close Hatch Creation**: finishes the command by creating the hatch (the same as `Enter`).

The color of the hatch lines is the object's color (ByLayer, like the others, or the one chosen beforehand).

-HATCH (aliases -H, -BH)

Hatching from the command line only, without the panel: `Specify internal point or [Properties (P)/Select objects (S)/remove Boundaries (B)/Advanced (A)/DRaw order (DR)/Origin (O)/ANnotative (AN)/Transparency (T)/Islands (I)/inHerit properties (H)/Undo (U)]`. **Advanced** gives the island style and the associativity. Example: `-h`, `p`, `ar-conc`, `2`, `0`, click the area, `Enter`.

GRADIENT (alias GD)

The same thing with the **gradient** fill already chosen: LINEAR, CYLINDER, INVCYLINDER, SPHERICAL, INVSPHERICAL, HEMISPHERICAL, INVHEMISPHERICAL, CURVED, and INVCURVED (AutoCAD's names), between two colors or from one color to the tint, with an angle and centered or offset.

Patterns

BadBit CAD's library (its own file in AutoCAD's `.pat` format, written from scratch) has the commonly used patterns: **SOLID**, **LINE**, **NET**, **NET3**, **ANSI31** to **ANSI38**, **AR-B816**, **AR-CONC**, **AR-SAND**, **BRICK**, **CLAY**, **CROSS**, **DASH**, **DOTS**, **EARTH**, **GRASS**, **GRAVEL**, **HONEY**, **INSUL**, **SQUARE**, and **ZIGZAG**. The measurements are in meters, designed for floor plans (ANSI31 has lines 12.5 cm apart at scale 1): the **scale** adjusts it to the drawing.

Patterns imported from a `.pat` (from AutoCAD or another source) are added to the list. The pattern definition goes **inside** the hatch in the `.bbd` file: the drawing opens the same way on another machine, without the `.pat`. A scale too small for the area (more than a million lines, AutoCAD's HPMAXLINES) is rejected with the notice *The pattern is too dense for this area*.

Associative hatch

With **Associative** on (the default, HPASSOC), the hatch follows the boundary: stretching a wall, moving a pillar, or scaling the circle rebuilds the hatch in the same undo step. If the boundary stops being closed (or one of its objects is erased), the hatch stays where it is and stops being associative, as in AutoCAD. Copying the boundary together with the hatch leaves the copy associated with the copies.

HATCHEDIT (alias HE, double-click)

Double-clicking a hatch opens its editing, with the same panel: each change applies to the chosen hatches. From the command line: `[DIsassociate (DI)/Style (S)/Properties (P)/DRaw order (DR)/Origin (O)/ANnotative (AN)/Transparency (T)]`; each option changes only that. The **Properties** palette (`Ctrl` + `1`) also shows and changes the pattern, the scale, the angle, the transparency, and the islands, with the painted **area**. **MATCHPROP** carries the properties of a hatch to others (the Hatch box in the Settings).

Selection, grips, and editing

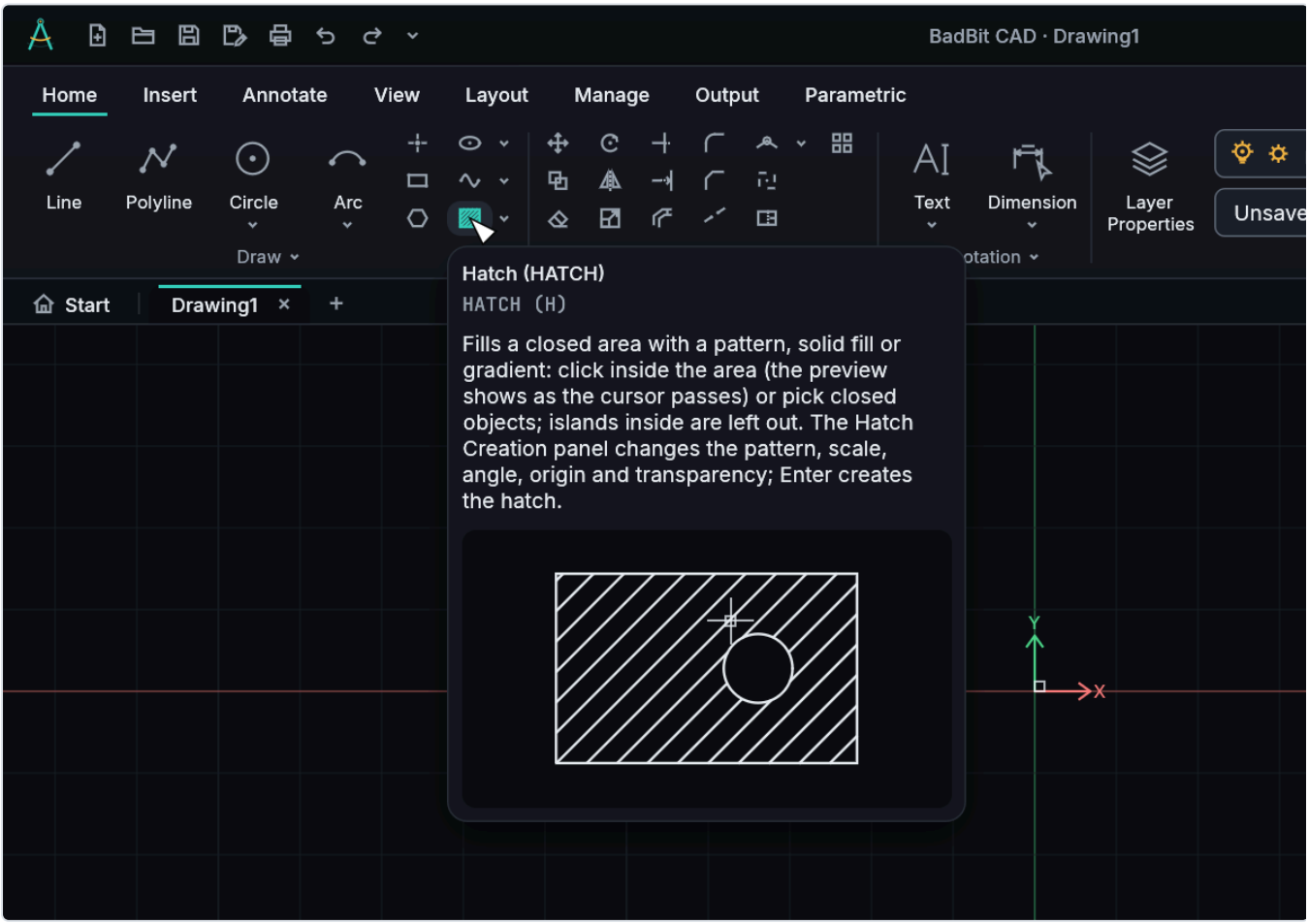
- A click **inside** the painted area or on the **edge** selects the hatch (a line on top of it still wins). A crossing window picks the hatch it touches.
- **Grips**: the **middle** one moves the hatch; a non-associative hatch also shows the vertices of the boundaries, which can be dragged (and the hatch is rebuilt).
- **STRETCH** stretches the boundaries the window catches. **EXPLODE** turns the hatch into the pattern lines (solid and gradient hatches, into the boundaries).
- **Snaps** do not stick to hatches (AutoCAD's OSNAPHATCH 0).

Draw order

The hatch stays **beneath the lines** (HPDRAWORDER *Behind boundary*): walls and dimensions appear on top of it. **DRAWORDER** (alias DR), with the options **Above objects** (`A`), **Under objects** (`U`), **Front** (`F`), and **Back** (`B`), brings the chosen objects in front of everything, behind everything, or right above/below the reference objects (asked for after `A` and `U`); **HATCHTOBACK** sends all hatches to the back. Draw order applies to any object (hatches, images, lines), goes into DXF and DWG, and comes back from them (see the images chapter).

Fill on the screen

The fill comes out the same on the GPU and on the CPU-only path (a computer without a graphics card), without slowing down the drawing: thousands of hatches go into the stored frame, and only what changes is redrawn. **Wide polylines**, the DONUT's **donut**, and the **filled arrowheads of dimensions** also come out filled.

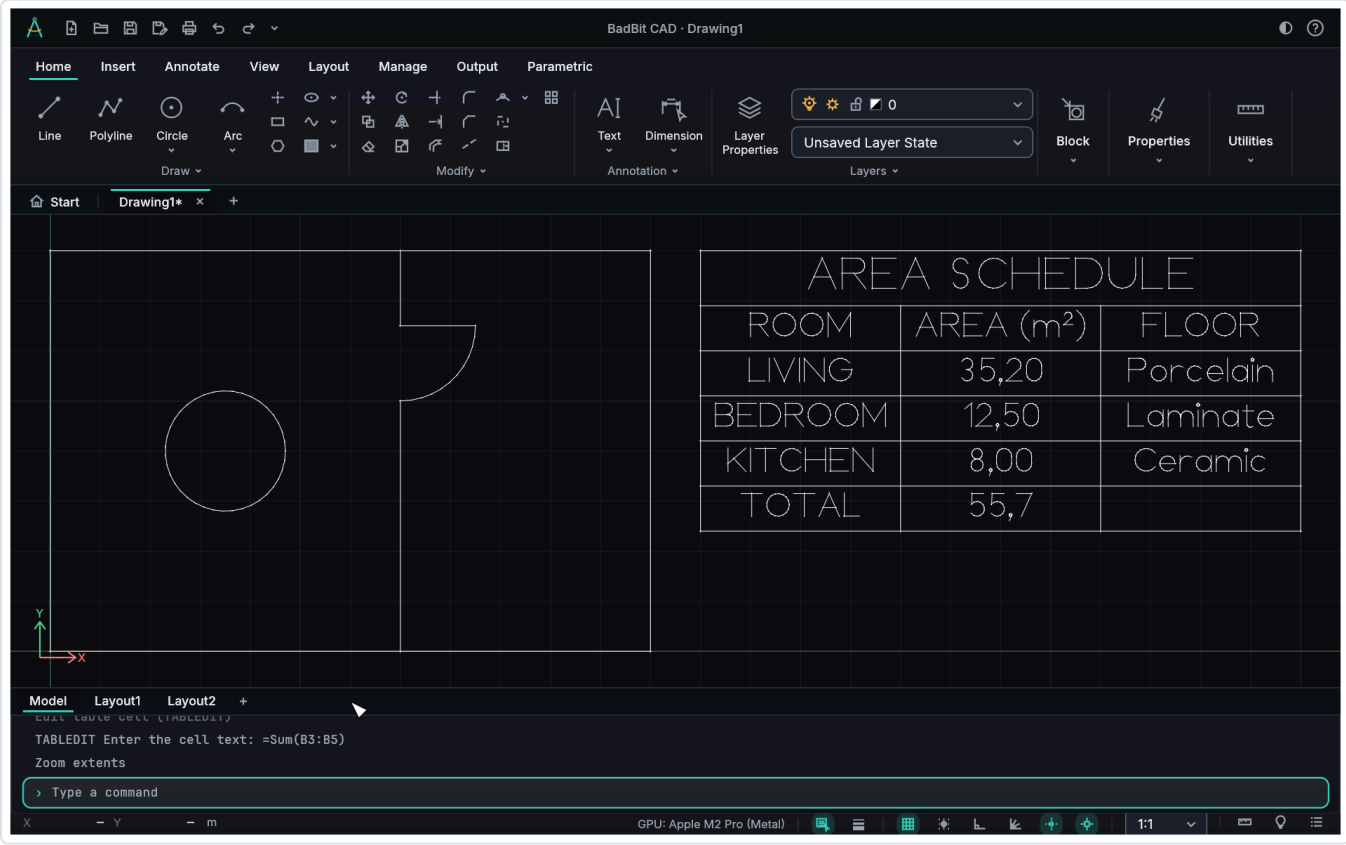


The extended tooltip of the Hatch button, with the internal point animation.

Tables

Tables follow AutoCAD's: a single object with the rows and columns, the text of each cell (multiline, with the same font as MTEXT), the alignment, merged cells, the borders and background color of each cell, the three cell styles (**Title**, **Header**, and **Data**), the number format, and **formulas**, which recalculate by themselves when a cell changes. The row grows to fit the text. The table style (TABLESTYLE) gives the starting sizes, colors, and borders.

Table is on the **Annotation** panel of the Home tab (open the panel's list); TABLESTYLE is in the same list.



An area schedule made with -TABLE and typed into the cells; the total is the formula `=Sum(B3:B5)`.

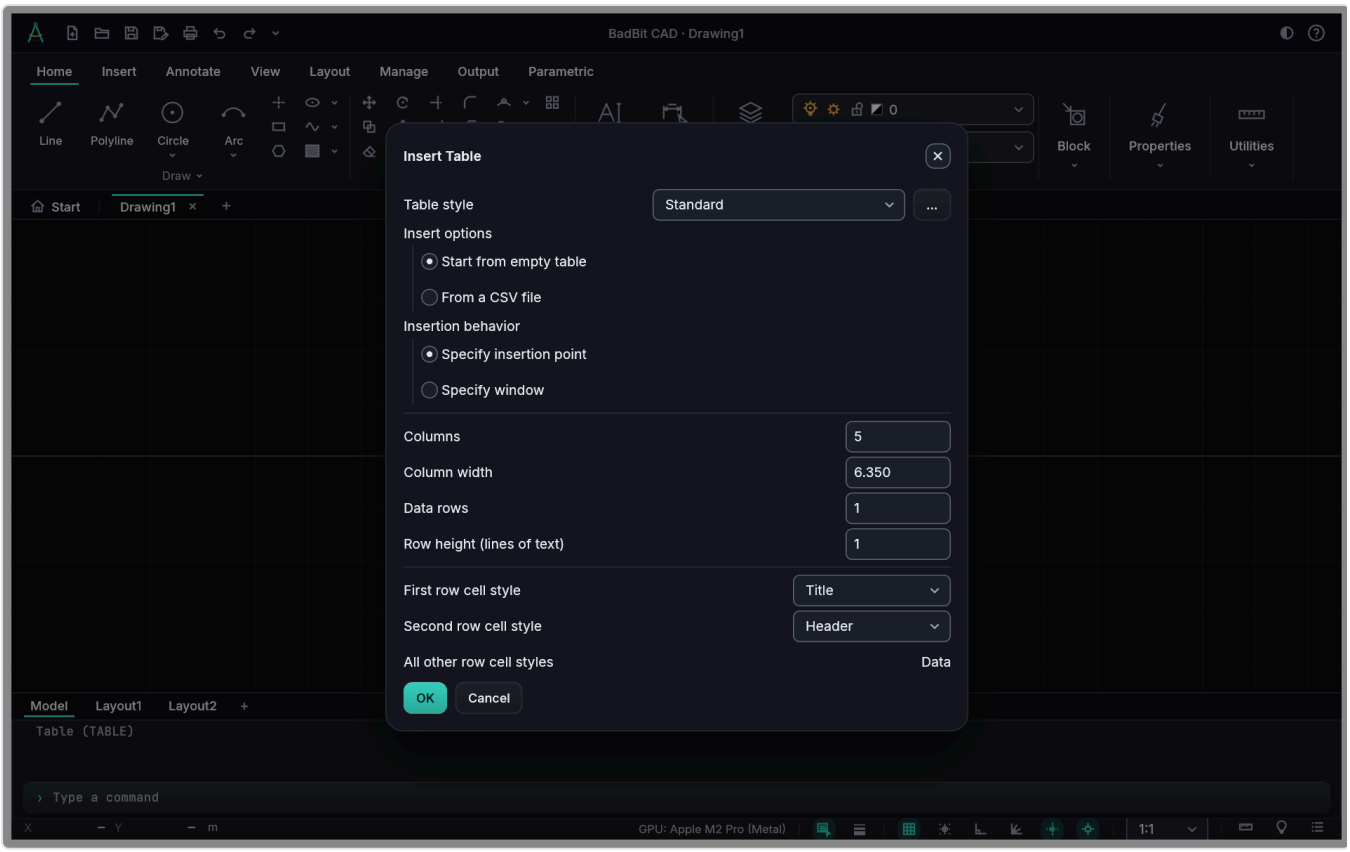
TABLE (alias TB)

Opens the **Insert Table** window:

- **Table style**: the list of the drawing's styles (the `...` button opens TABLESTYLE); the one chosen becomes current;
- **Insert options**: two option buttons, as in AutoCAD: **Start from empty table** or **From a CSV file**; with CSV selected, the **File** field appears with the **Browse...** button (AutoCAD does it through the data link with Excel; here, CSV is enough: comma, semicolon, or tab, with quotes). From the CSV, each line of the file becomes a row of the table, the first one with the Header style;

- **Insertion behavior** (also option buttons): **Specify insertion point** (the top left corner; the table follows the cursor) or **Specify window** (the two corners: the columns divide the width and as many data rows fit as the height allows);
- **Columns** and **Column width**, **Data rows** and **Row height** (in lines of text);
- **First row cell style** and **Second row cell style** (Title, Header, or Data); the others are Data. With Title and Header, the table has the data rows plus these two; the title is created merged across the full width.

The window remembers what was chosen. After OK, **specify the insertion point** (or the corners of the window). The empty table comes in with the title already being edited in the drawing itself.



The Insert Table window with the defaults: five columns, one data row, title, and header.

-TABLE (alias -TB)

The same table from the command line: **the number of columns**, **the number of data rows**, and **the insertion point or [Style (S)/Width (W)/Height (H)]**. **Style** asks for the table style name, **Width** for the column width, and **Height** for the row height in lines of text. `Enter` accepts the value between `<>`.

Typing in the cells (TABLEDIT and double-click)

The cell text is typed in the drawing itself, with the blinking cursor, as in MTEXT. The keys, as in AutoCAD:

- `Tab` confirms and goes to the next cell (on the last one, it creates a new row); `Shift` + `Tab` goes back;
- `Enter` confirms and moves down; `Alt` + `Enter` breaks the line inside the cell;

- the arrows move within the text and, at its end, move to the neighboring cell (up and down, in single-line text);
- `Ctrl` + `Enter` (`Cmd` + `Enter` on the Mac) confirms and finishes; `Esc` abandons the cell being edited.

Each cell typed is an undo step. **TABLEDIT** asks to **select a table cell**; **double-clicking** a cell does the same directly.

Formulas

A cell that starts with `=` is a formula, as in AutoCAD. Cells are referenced by column (letters) and row (number; the first row, the title's, is 1): `A1`, `B3`, `$C$2`. These work:

- the operations `+ - * /` and `^`, with parentheses: `=B3*2`, `=(B3+B4)/2`;
- the functions over ranges and cells, separated by commas: `=Sum(B3:B5)`, `=Average(B3:B5)`, `=Count(B3:B5)`, `=Min(...)`, `=Max(...)`.

An empty cell counts as 0 in operations and is not included in functions (the average of 1, 2, and an empty one is 1.5); text that is not a number is also left out of functions. An error (division by zero, text in an operation, circular reference, unreadable formula, reference to a deleted cell) appears as `#####`.

Inserting or deleting rows and columns adjusts the references like AutoCAD: what comes after shifts, a range grows when the new row goes into the middle of it and shrinks when one of its rows is removed; a reference to a deleted cell becomes `#REF` (and the cell shows `#####`).

Number format

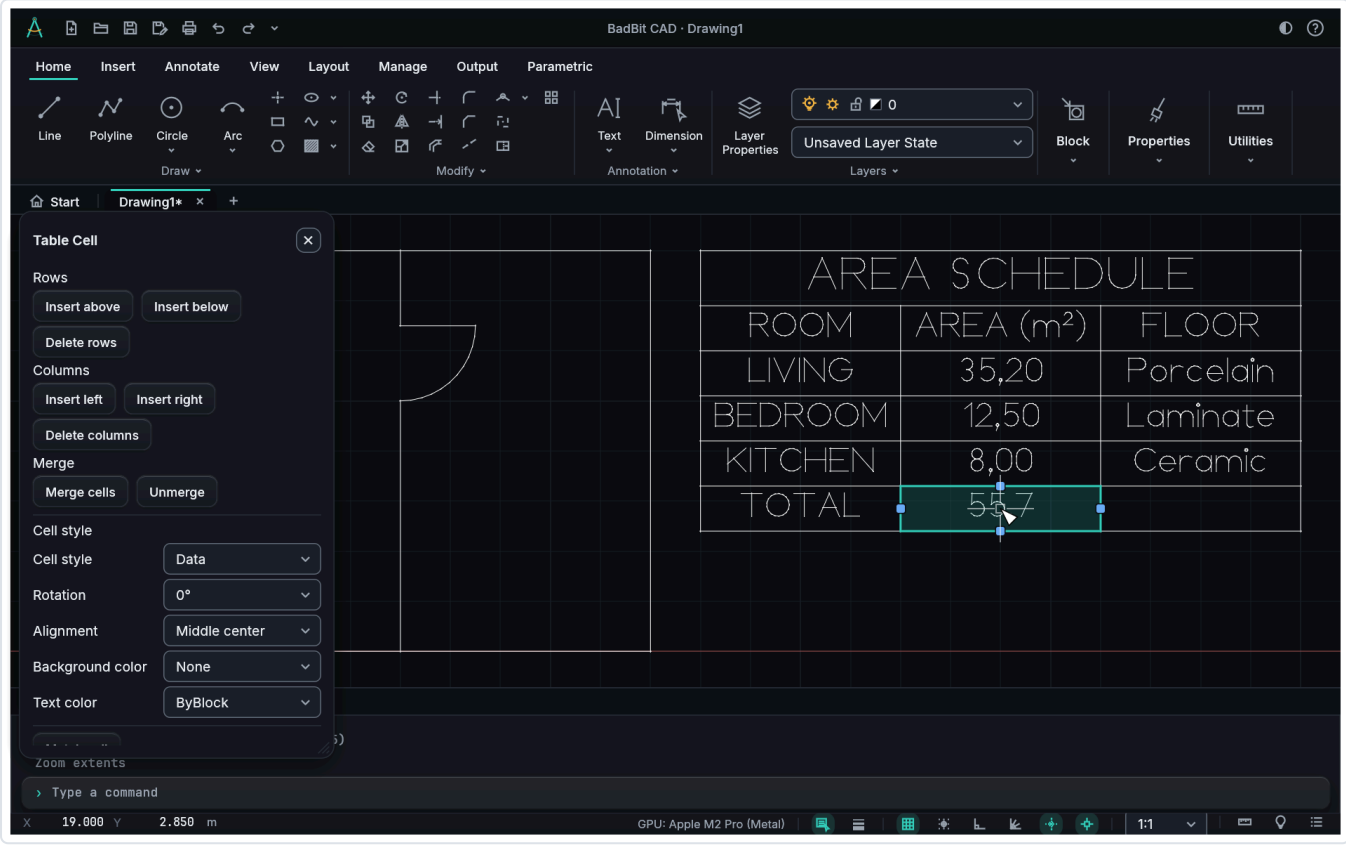
Each cell style (and each cell, if you want) has the data type: **General** (what was typed; the result of formulas with up to six decimal places, without trailing zeros), **Text**, **Decimal number**, **Whole number**, **Percentage**, **Currency** (with the symbol, `R$` by default, and thousands separated), **Angle** (in degrees), and quantities with a unit (the value in meters, from the drawing, written in `cm`, `mm`, `m²`...), with the **decimal places**. For numeric types, the typed number also comes out in the format (`1.5` becomes `1.50` with two decimal places). Comma or period both work when typing.

Choosing cells

With no command, **clicking inside a cell** selects it (a click on the edge selects the whole table, as in AutoCAD); `Shift`+click extends to another cell. The selection shows:

- the cell's **side grips**: the top and bottom ones change the row height, the side ones the column width (click the grip, click the new place, or drag);
- the **Table Cell** palette (in place of AutoCAD's contextual tab, which today's ribbon does not switch by context).

With cells chosen, the arrows and `Tab` move (`Shift`+arrow extends), `F2` edits, starting to type a number or a formula replaces the content (letters still go to the command line, for commands), `Delete` clears the content, `Ctrl`+`C` copies as tab-separated text (pastes straight into a spreadsheet), and `Ctrl`+`V` pastes what came from a spreadsheet starting at the chosen cell. `Esc` releases the cells.



The total cell chosen: the highlight, the side grips, and the Table Cell palette.

The Table Cell palette

- **Rows**: **Insert above**, **Insert below** (with the height, style, and format of the chosen row), and **Delete rows**;
- **Columns**: **Insert left**, **Insert right**, and **Delete columns** (the merged title follows);
- **Merge cells** (the content of the first cell that has any is kept) and **Unmerge**;
- **Cell style**: the cell style (Title, Header, or Data), the **alignment** (AutoCAD's nine), the **background color**, and the **text color**;
- **Borders**: the color and the lineweight, and the buttons **All**, **Outside**, **Inside**, **Top**, **Bottom**, **Left**, **Right**, **No border**, and **From style** (goes back to the style's). Border lineweight appears with LWDISPLAY on, like that of lines;
- **Data format**: the type, the decimal places, and the symbol, and **Apply format** (or **Style format**);
- **Formula**: **Sum**, **Average**, and **Count**. With a range chosen, the formula goes in the cell after it (below the column, or to the right of the row); in a single cell, it goes into editing already with the range of the numbers above (or to the left), like the AutoSum of spreadsheets; **Clear contents**;
- **TABLEEXPORT**, to write the table to CSV.

Properties (`Ctrl` + `1`) shows the table (the style, which the list changes, the rows, the columns, the width, which stretches the columns, and the height) and, with cells chosen, the **Cell** group: the cell style, the alignment, the column width, the row height, the format, the contents, and the shown value.

Editing the whole table

Selected by its edge, the table has AutoCAD's grips: the insertion point corner (moves), the corner at the end of the width (stretches the columns evenly), the one at the end of the height (stretches the rows), the opposite corner (both), and a grip at the top of each column division (the width of that column). MOVE, COPY, ROTATE, SCALE (the widths, heights, texts, and margins grow together), and MIRROR (the table goes to the other side and stays readable, like text with MIRRTEXT 0) work as on other objects; STRETCH takes the table along when the window catches the insertion point. Snaps pick the cell corners (Endpoint), the midpoints of the borders, and the insertion point (Insertion). **EXPLODE** breaks the table up into lines (the borders, with the color and lineweight of each one), multiline texts, and solid hatches (the backgrounds).

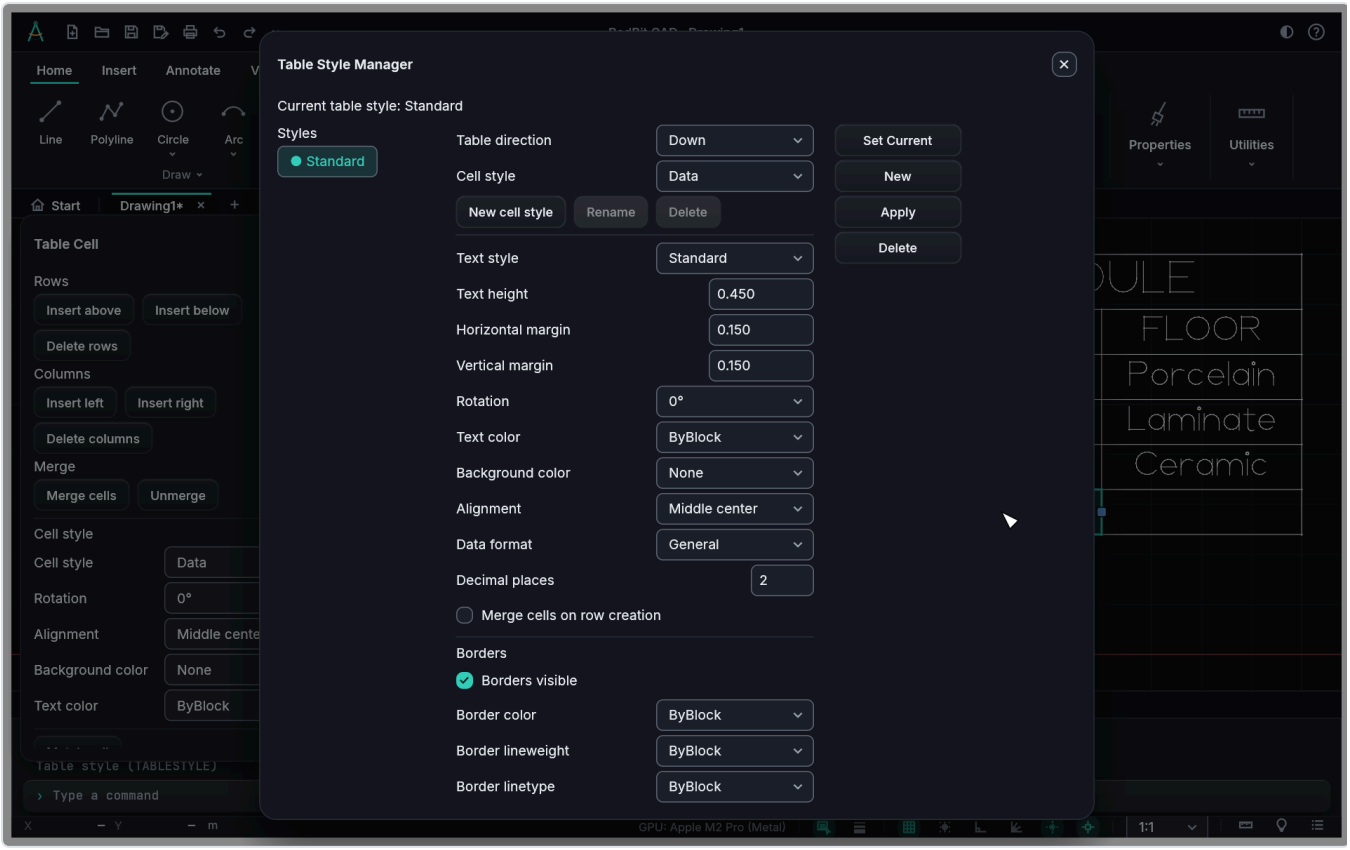
TABLEEXPORT

Select a table (or choose one beforehand): the table goes to a **CSV** file with the shown values (formulas already calculated and numbers in the cells' format), separated by commas, the way AutoCAD writes it. With FILEDIA 0, the file name goes on the command line.

TABLESTYLE (alias TS)

The **Table Style Manager**: the current style at the top, the list of styles and, next to it, the selected style: the **table direction** (**Down**, with the title at the top, or **Up**, with the title at the bottom and the insertion point at the bottom corner) and, for each **cell style** (Title, Header, Data), the **text style**, the **text height** (the text style's fixed height, if any, applies instead), the horizontal and vertical **margins**, the **text color**, the **background color**, the **alignment**, the **data format** and the decimal places, **Merge cells on row creation** (the title's), and the **borders** (visible, the color, and the lineweight). The buttons **Set Current**, **New** (a copy of the selected one), **Apply** (saves and rebuilds the style's tables in a single undo step), and **Delete** (not Standard, not the current one, not a style used by any table). Changing a text style used by a table style changes its tables along with it.

Standard follows metric AutoCAD's in a drawing in meters (a tenth of the millimeters, like dimensions): title of 0.6, header and data of 0.45, margins of 0.15, all centered and with all borders.



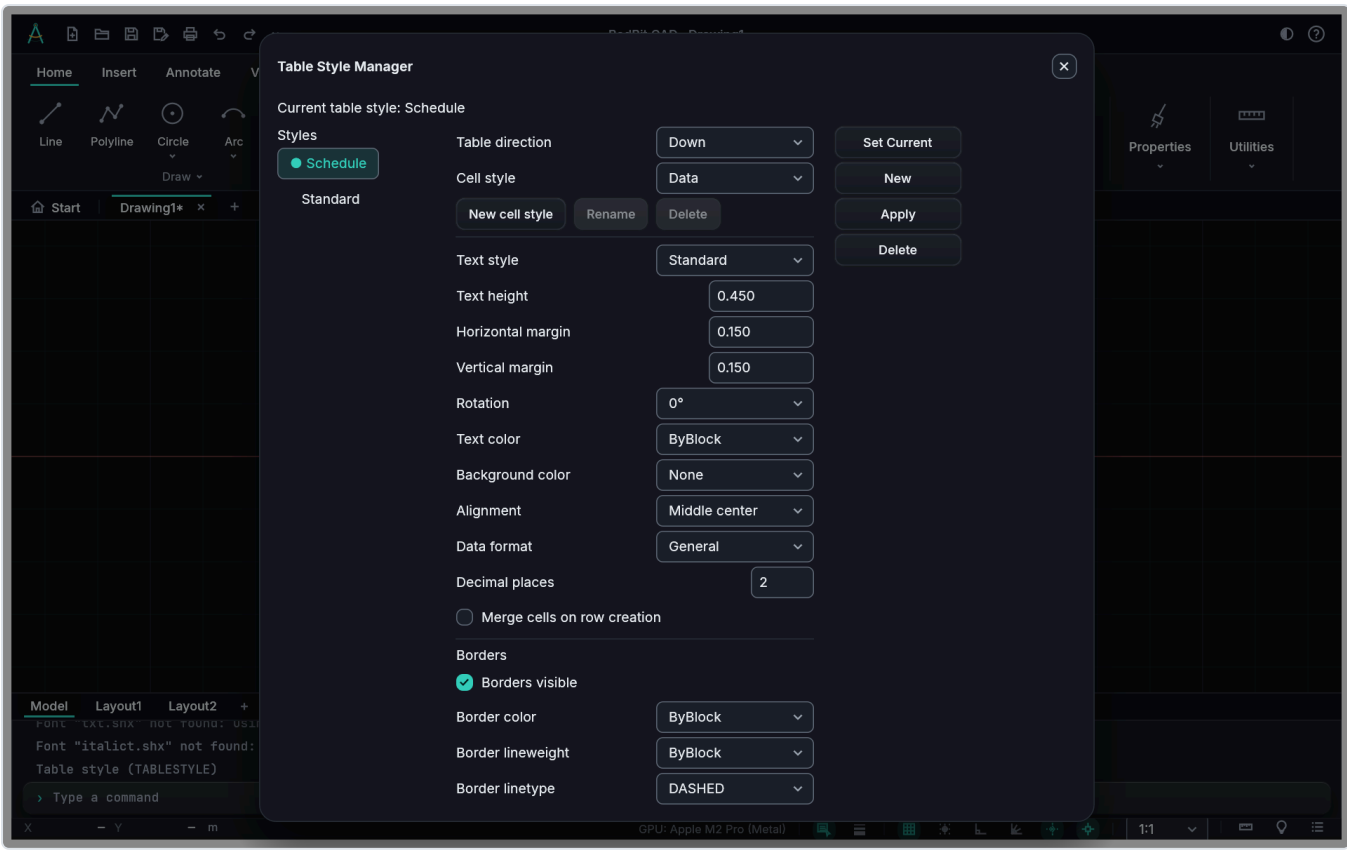
The Table Style Manager with Standard and the Data cell style.

User cell styles

Besides **Title**, **Header**, and **Data**, the table style can have named cell styles, like AutoCAD's **Cell styles** list. In TABLESTYLE, below the **Cell style** list:

- **New cell style**: type the name and click **OK** (or `Enter`); the new style starts the same as the cell style shown (AutoCAD's "Start with") and is already chosen in the list. Repeated names (case-insensitive) and the names of AutoCAD's three are not allowed;
- **Rename** and **Delete**: only on user styles; Title, Header, and Data cannot be renamed or deleted.

Nothing changes in the tables until **Apply**. The cells of a deleted cell style go back to **Data** when the table style is applied, in the same undo step. User cell styles appear in the **Cell style** list of the Table Cell palette and of Properties. When saving to DXF/DWG they go to the file's table style (from 2007 on, in the list of cell styles that AutoCAD reads) and come back when opening; see *DXF and DWG files*.



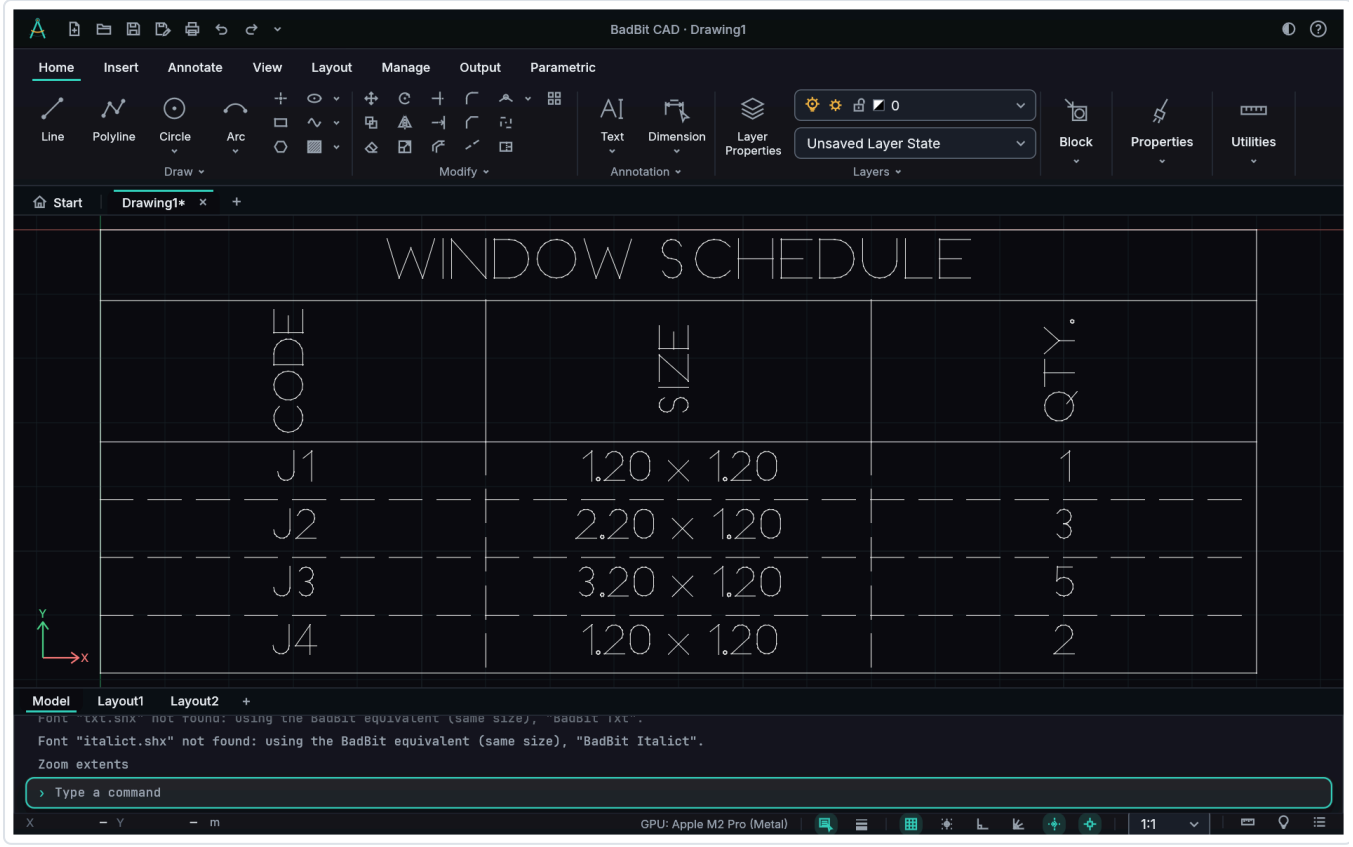
TABLESTYLE with the cell style buttons, the rotation, and the border linetype.

Rotated text in the cell

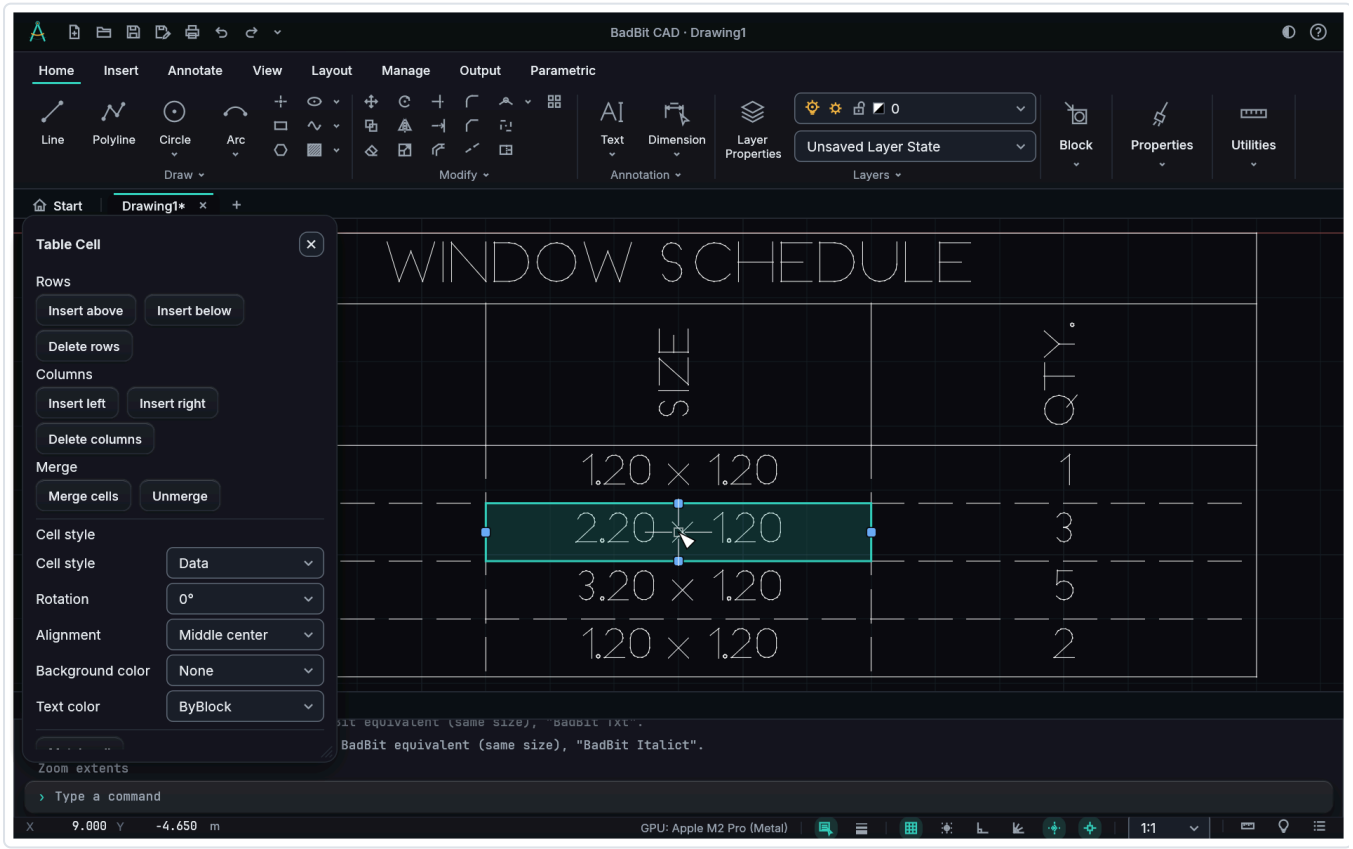
The **Rotation** of the cell text is 0°, 90°, 180°, or 270° (counterclockwise, from the table direction), in the cell style (TABLESTYLE) or in each cell (the Table Cell palette and the Cell group of Properties). Upright text (90° and 270°) does not wrap: the row grows until its length fits, and the cell alignment puts the text in place. At 180° the height is that of the lying-down text.

Border linetype

Borders have a **linetype**, in addition to the color and lineweight: **ByBlock** (the table's), **ByLayer** (the table layer's), **Continuous**, and the linetypes loaded in the drawing (LINETYPE loads them). In the Table Cell palette, choose the **Border linetype** before the borders button (All, Outside, Inside…); in TABLESTYLE, it applies to the four borders of the cell style. The border comes out dashed on the screen and in plotting (at the global LTSCALE scale times the table's), and **EXPLODE** gives lines with the same linetype.



The header rotated 90° (the row grew for the text) and the inner borders of the data dashed (DASHED).

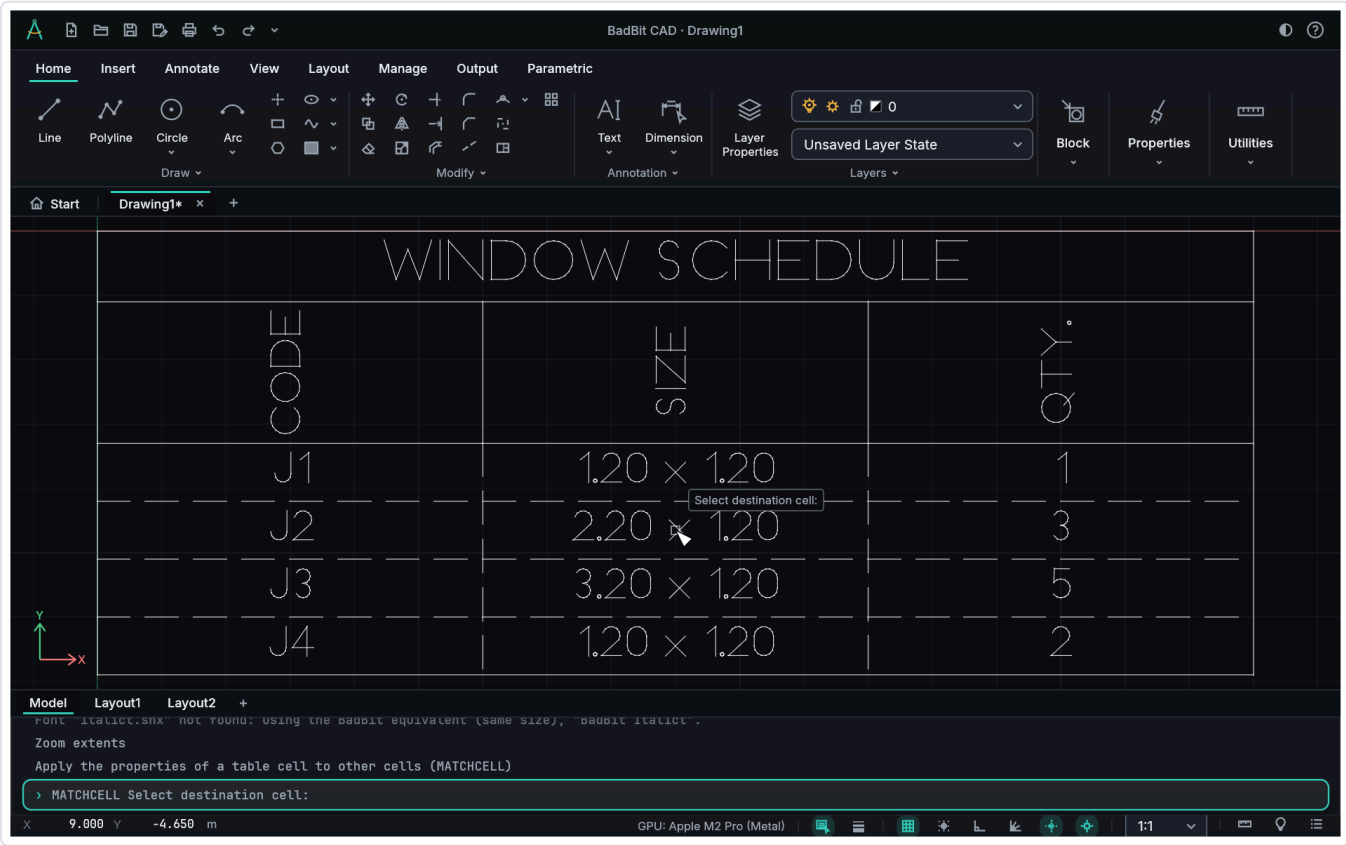


The palette with Rotation, Border linetype, and the Match cell button.

MATCHCELL and MATCHPROP in tables

MATCHCELL copies the formatting of one cell to others, as in AutoCAD: **Select source cell** and then **Select destination cell** as many times as you want; `Enter` or `Esc` finishes. The cell style (a user style goes by name; in a table without it, the cell becomes Data), the alignment, the text height, color, and rotation, the background, the borders, and the format are copied; the content is not. Each destination cell is an undo step. With a cell already chosen, it is the source: that is the **Match cell** button of the Table Cell palette.

MATCHPROP (brush) with a cell chosen does the same as MATCHCELL. Between whole tables, the **Table** box in the **Settings** carries the source's table style over to the destinations.



MATCHCELL with the chosen cell as the source, asking for the destination.

CTABLESTYLE

The current table style is the **CTABLESTYLE** variable: SETVAR (or the variable name on the command line) shows and changes it, and the name must be that of a table style in the drawing. The **Current table style** list on the Annotation panel (below those for text style and dimension style) changes the same style, as do TABLESTYLE and -TABLE (Style option).

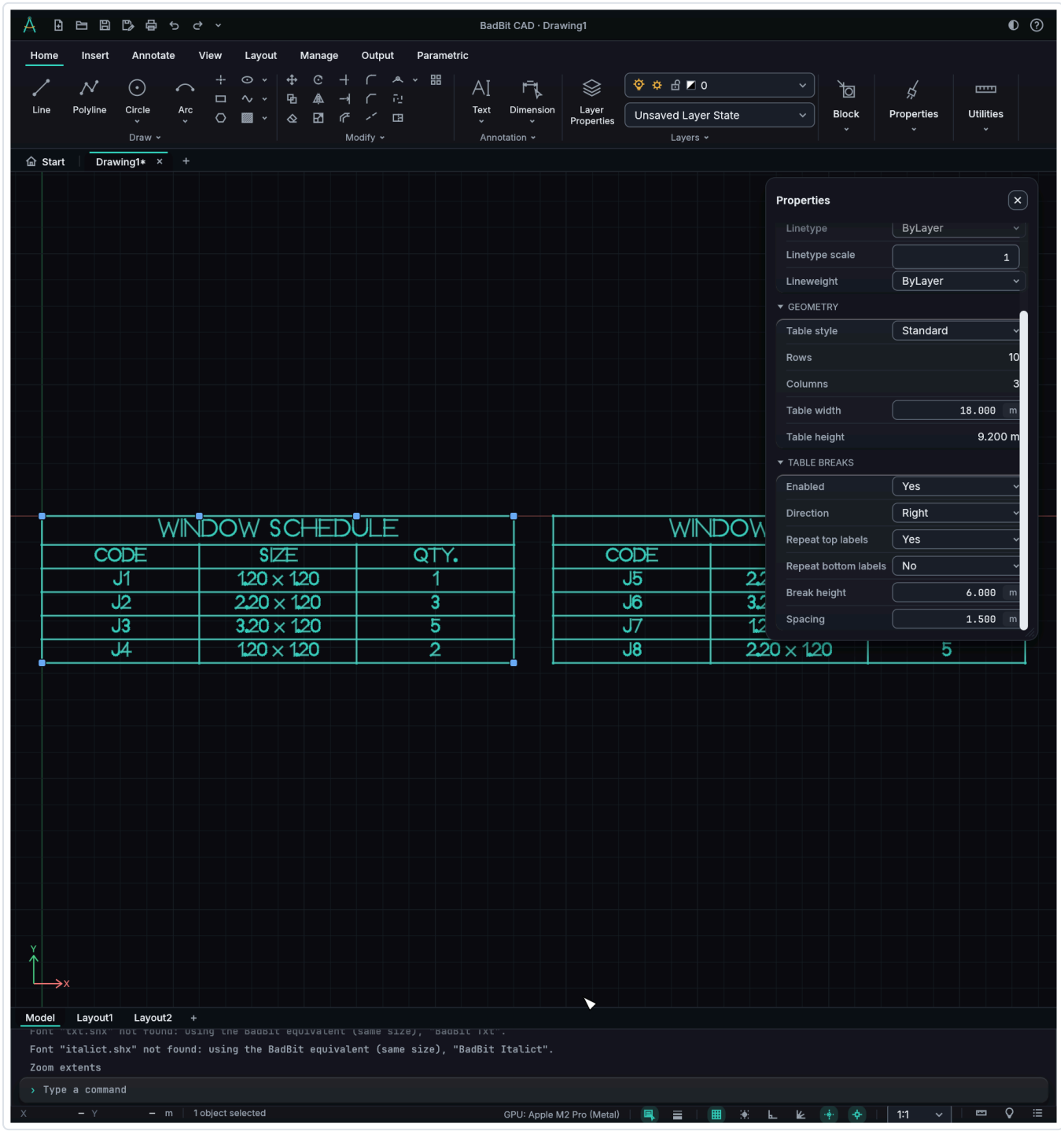
Table breaks

A long table can be broken into parts, like AutoCAD's **Table breaks** properties. Select the table by its edge and, in Properties (`Ctrl` + `1`), in the **Table breaks** group:

- **Enabled**: Yes breaks the table (with no height yet, the break height starts at half the table);

- **Direction**: **Right**, **Left**, or **Down**, where the parts go;
- **Repeat top labels**: the title and the header appear in each part;
- **Repeat bottom labels**: the header rows at the end of the table appear in each part;
- **Break height**: the maximum height of each part (rows that do not fit go to the next one; a merged cell is never cut);
- **Spacing**: the space between the parts.

With the table broken, the bottom grip of the first part changes the break height (the corner one also changes the width). The cells of the other parts are chosen with a click, with the side grips; selection, snaps, the table's bounding box, SCALE (the height and spacing grow together), EXPLODE, and the block written to DXF/DWG see all the parts; in DWG from 2010 on the break also goes as AutoCAD's.

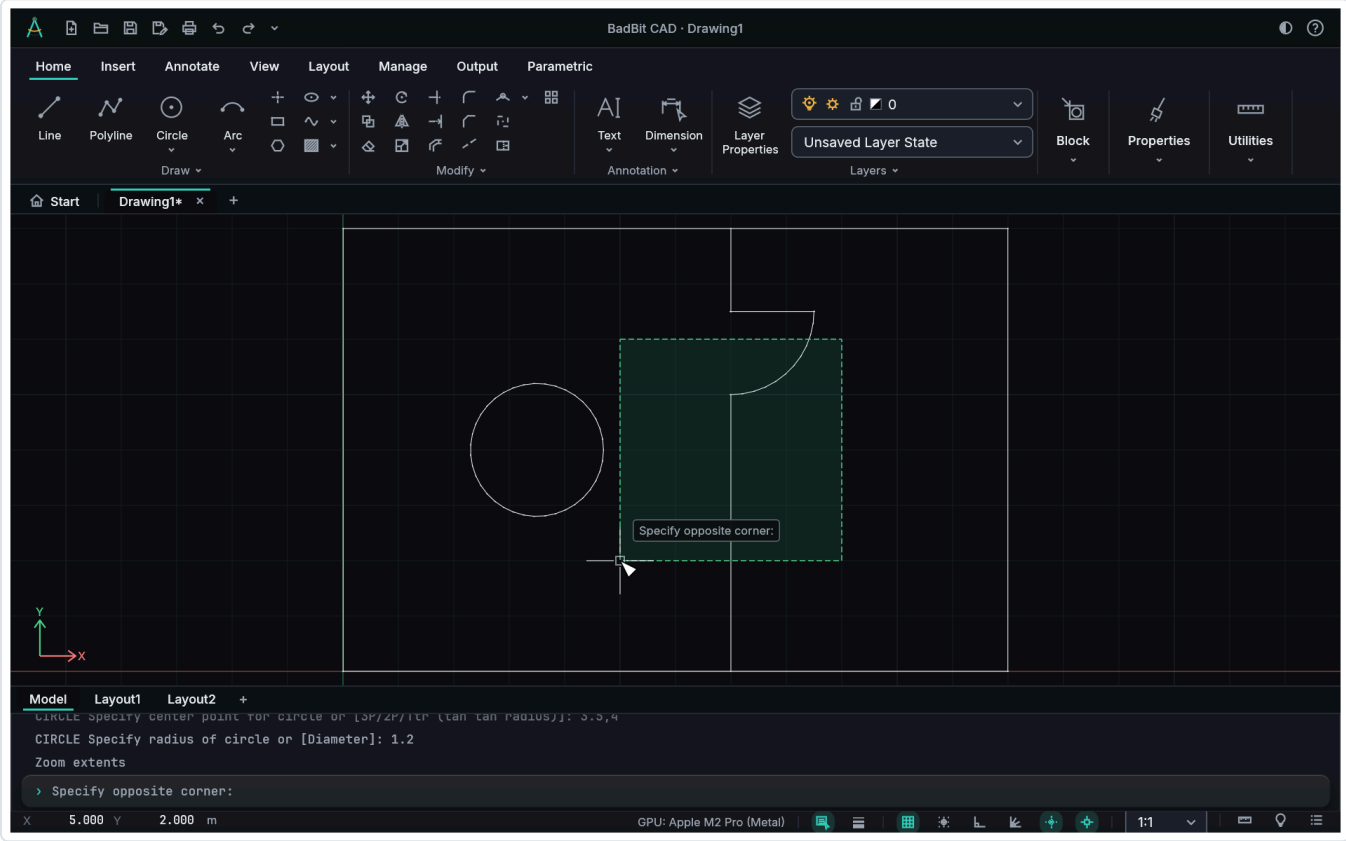


Eight data rows in two parts to the right, with the title and header repeated, and the Table breaks group.

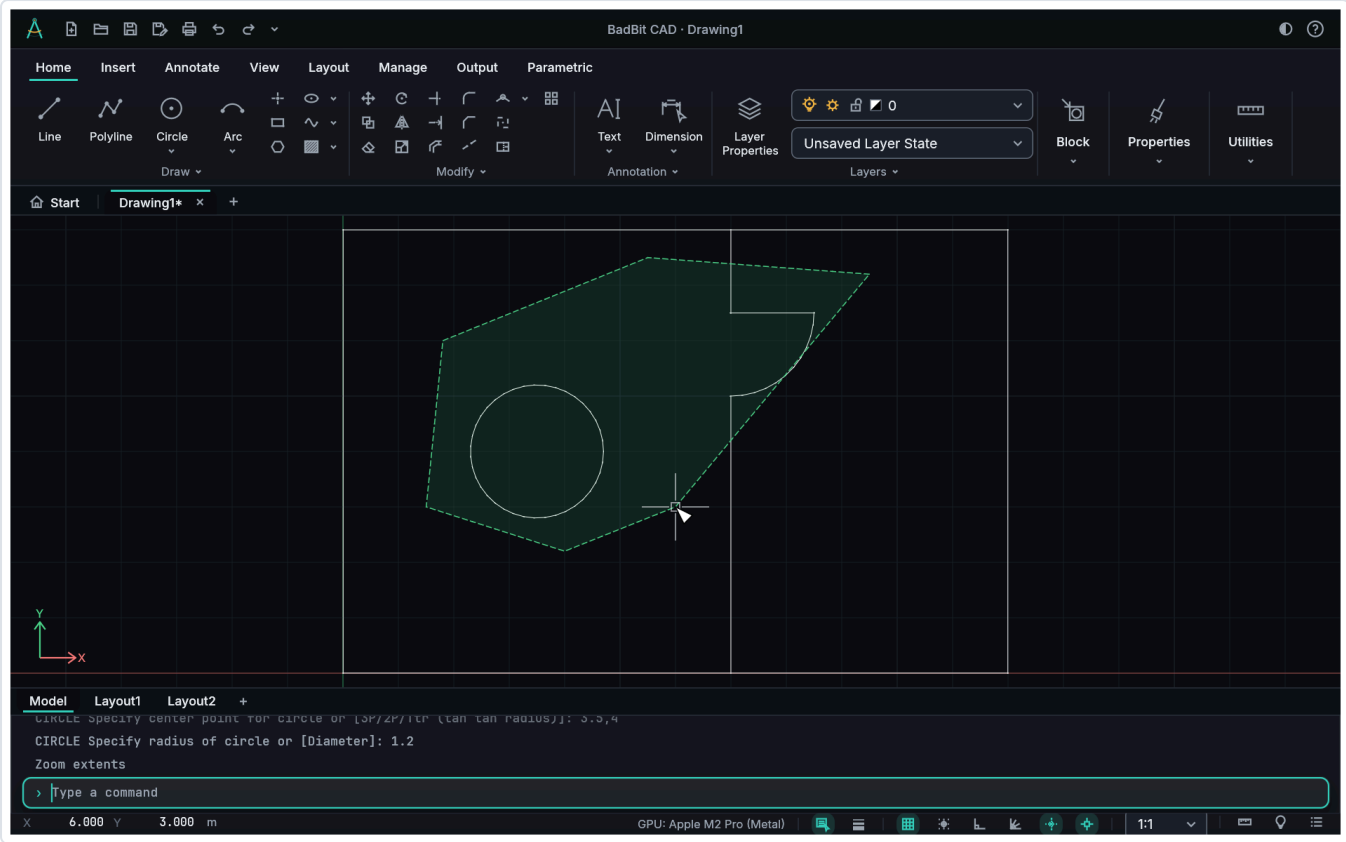
Selection and editing

Selecting

How	Selects
Click on the object	The nearest object inside the cursor box (its size is PICKBOX).
Window: click in an empty spot and click again to the right (blue)	Only what is entirely inside.
Crossing: click in an empty spot and click again to the left (green, dashed)	Whatever is inside or touches it.
Lasso: press in an empty spot and drag around, holding the button down	To the right, the window of the outline (blue); to the left, the crossing (green). Dragging in a straight line gives the rectangular window between the ends; once the path curves and becomes a lasso, it stays a lasso until you release the button.
`Ctrl` + `A`	Everything that is visible.
`Shift` +click or `Shift` +window	Removes from the selection.

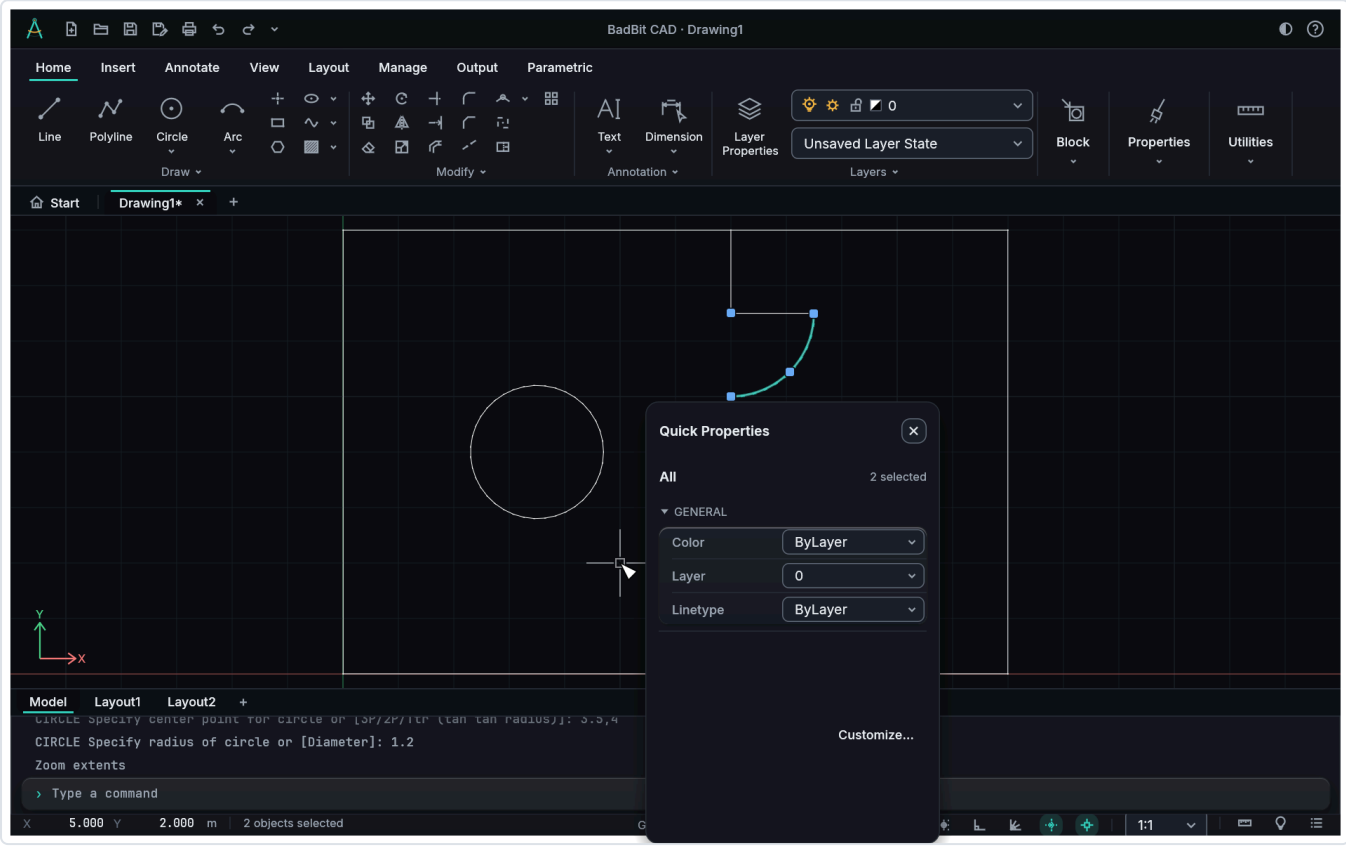


Crossing: from right to left, it picks whatever it touches.



Lasso dragged to the left: a crossing along the drawn outline.

Selected objects are highlighted in the accent color. You can select before the command (and then call MOVE, COPY or ERASE) or after it (the command asks "Select objects:" and `Enter` ends the selection). `Esc` clears the selection.

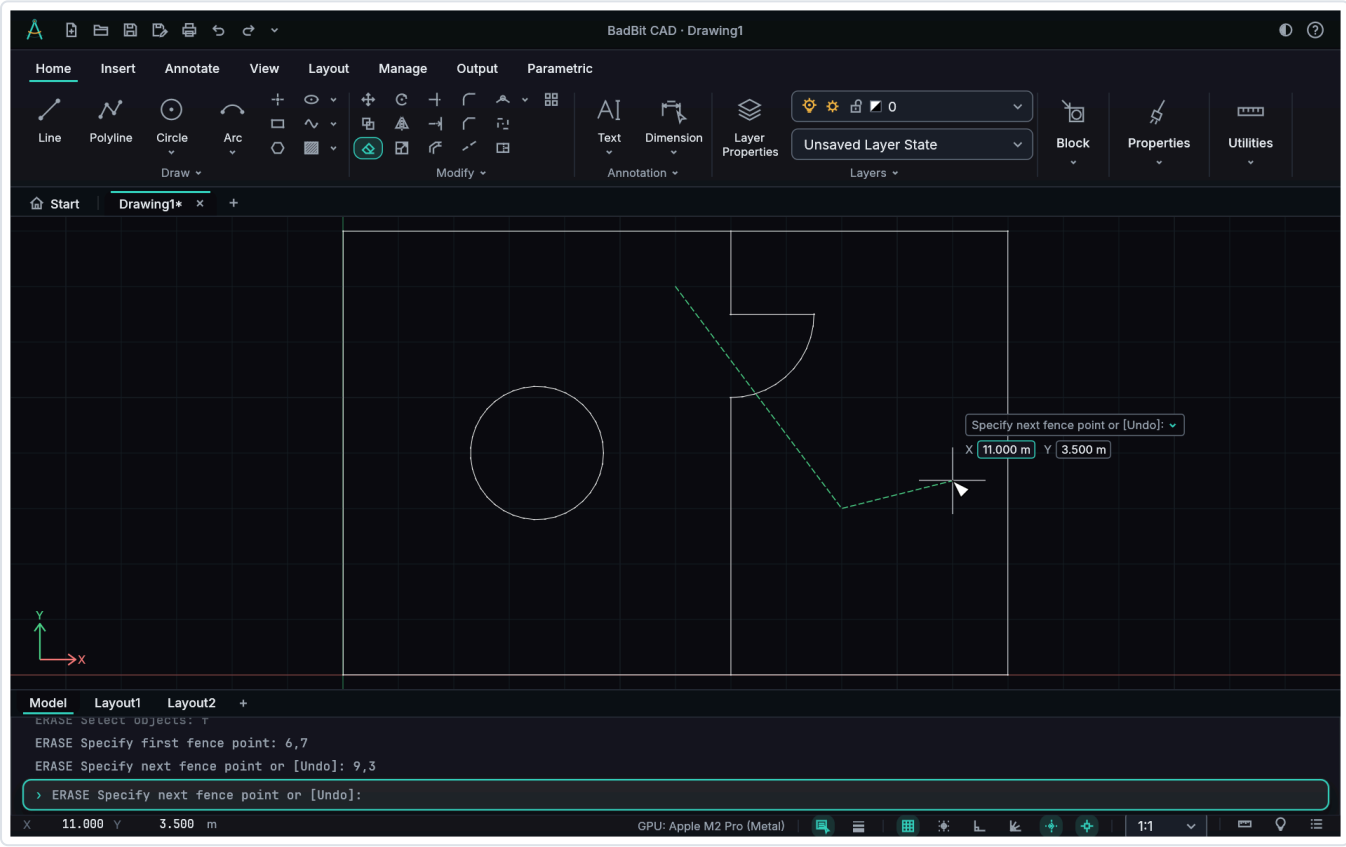


Selected objects, highlighted.

"Select objects" options

When a command asks "Select objects:", you can type the AutoCAD options (the letters are the AutoCAD ones, in English; the Portuguese words also work):

Option	What it does
ALL (`ALL`)	Everything that is visible (layers that are off or frozen are left out).
Previous (`P`)	The last selection used by a command.
Last (`L`)	The last object drawn.
Window (`W`) and **Crossing** (`C`)	Two corners: window or crossing, whatever the direction. **BOX** (`BOX`) decides by the direction.
Fence (`F`)	A line of points: picks whatever it crosses. `Enter` ends; `U` removes the last point.
WPolygon (`WP`) and **CPolygon** (`CP`)	A polygon of points: what is entirely inside (window) or what touches it (crossing).
Remove (`R`) and **Add** (`A`)	The next clicks remove from the selection ("Remove objects:"), until Add.
Undo (`U`)	Undoes the last pick.
SIngle (`SI`)	The first pick already ends the selection.



Fence: picks the lines that the line of points crosses.

A typed coordinate counts as a click. STRETCH accepts **CPolygon** in addition to the crossing window.

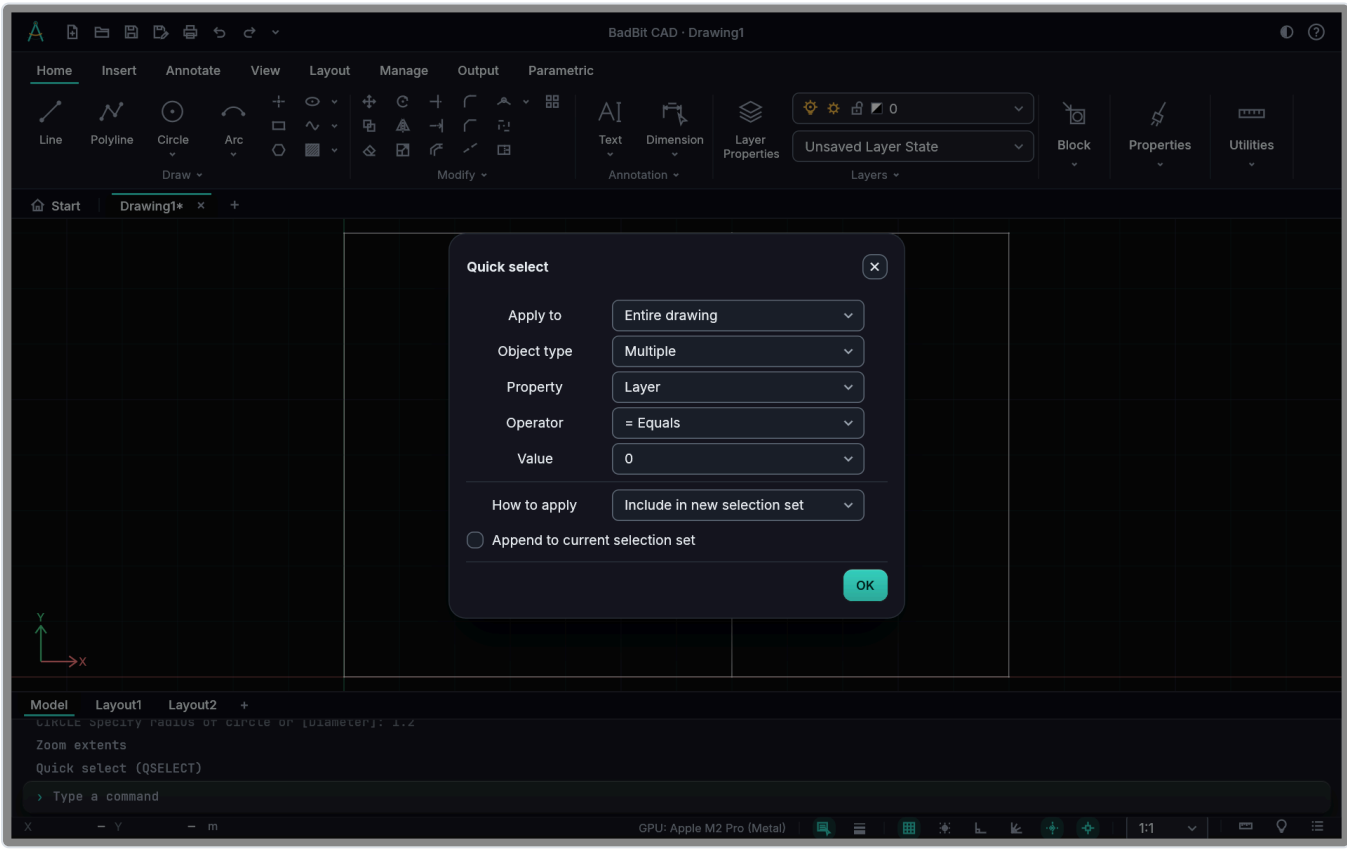
Overlapping objects

With **selection cycling** on (`Ctrl` + `W`, or the SELECTIONCYCLING variable), clicking where there is more than one object opens a list of them (type and layer): click the one you want. `Esc` closes it without choosing. With it off, the click picks the nearest one.

Without the list, as in older AutoCAD: hold `Shift` + `Space` and click; each click in the same spot swaps the object for the next overlapping one, from top to bottom, and the command line shows `<Cycle on>` and the type and layer of the current one. This also works at a command's "Select objects"; `Enter` or releasing the keys accepts.

SELECT, SELECTSIMILAR and QSELECT

- **SELECT**: only asks "Select objects:"; whatever was picked stays selected for the next command (and becomes the Previous selection).
- **SELECTSIMILAR**: picks all objects of the same type and on the same layer as the selected ones (with SELECTSIMILARMODE you can also compare color, linetype, lineweight and text style).
- **QSELECT** (quick select): a dialog picks by property, in the entire drawing or in the current selection: object type, property (layer, color, linetype, lineweight, length, area, radius, text height and contents), operator (equals, not equal, greater than, less than or select all) and value. In texts, `*` and `?` are wildcards. You can include or exclude what passes and append it to the current selection.

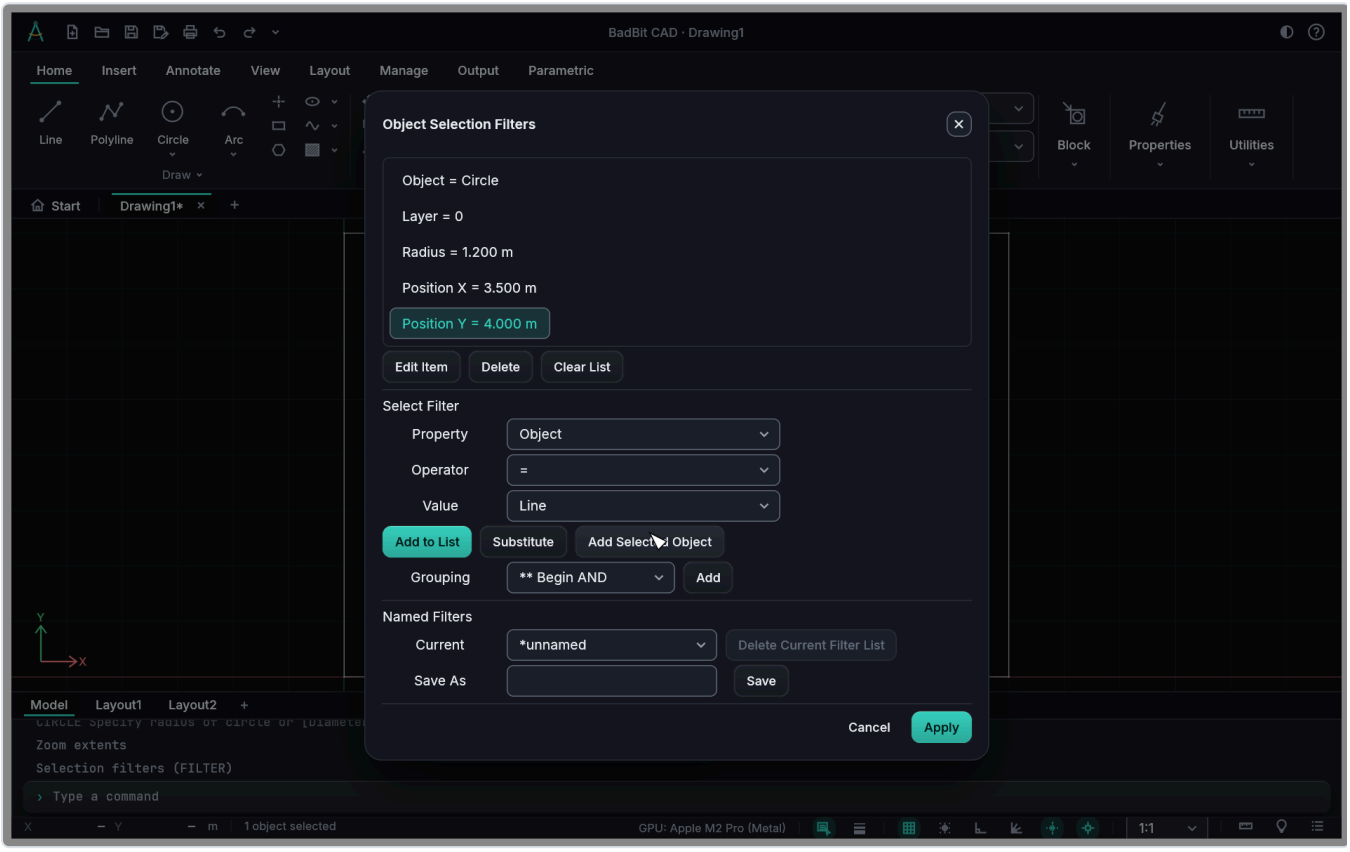


QSELECT: the object type, the property, the operator and the value.

FILTER (selection filters, alias FI)

The **Object Selection Filters** dialog, like the AutoCAD one, builds a list of conditions; only what passes it goes into the selection:

- **Select Filter**: the **property** (object, layer, color, linetype, lineweight, radius, length, area, X and Y position, text value, height and style, block name, attribute tag and value, rotation), the **operator** (`=`, `≠`, `<`, `≤`, `>`, `≥` and `*`, any value) and the **value** (from the list or typed; in texts, `*` and `?` are wildcards). **Add to List** puts the test after the selected item; **Substitute** replaces the selected item; **Edit Item** brings the item back to the fields.
- **Add Selected Object**: the tests that describe the object (type, the object's own layer, color, linetype and lineweight, radius, block name, text and position). With nothing selected, the dialog disappears and the command line asks for the object.
- **Grouping operators**: **Begin AND**, **Begin OR**, **Begin XOR** and **Begin NOT**, each one closed by its **End**. Tests outside a group apply together (AND). NOT takes one item, XOR two (it passes if exactly one passes), AND and OR at least one. A badly closed group is reported on Apply.
- **Named Filters**: **Save As** stores the list under a name **in the drawing** (it goes into the `.bbd` file and also into **DXF and DWG**, in the drawing dictionary, entry `BADBIT_FILTERS`, from which it comes back on opening) **and in the profile** (it applies to other drawings); the **Current** list brings back a saved filter; **Delete Current Filter List** removes it from both.
- **Apply**: outside a command, it opens the selection (like SELECT) with the filter: pick the objects (window, `ALL`...) and Enter; whatever passed stays selected. Inside a command that asks for objects, type `'FILTER` (transparent): on Apply, the command line says "Applying filter to selection"; the first Enter leaves the filtered selection and returns to the command's prompt, with the chosen objects.



FILTER: the list built by Add Selected Object (a circle) and the Select Filter fields.

Selection variables

Variable	Default	What it does
PICKADD	2	0: each pick replaces the selection (`Shift` adds). 1: adds (`Shift` removes). 2: like 1, and SELECT leaves the objects selected.
PICKFIRST	1	0: commands do not use what was selected before them.
PICKBOX	4	Size of the cursor box, from 0 to 50.
PICKDRAG	2	1: dragging makes the rectangular window (as before the lasso).
PICKAUTO	5	Without the 4 (for example, 1), dragging makes the rectangular window instead of the lasso.
SELECTIONCYCLING	0	2: the list of overlapping objects (the same as `Ctrl` + `W`).
SELECTSIMILARMODE	130	Sum of what similar objects must have in common: 1 color, 2 layer, 4 linetype, 16 lineweight, 64 text style.
PICKSTYLE	1	0: no group selection. 1: a selectable group member brings the whole group. 2: an associative hatch brings its boundary. 3: both. `Ctrl` + `Shift` + `A` turns groups on and off.

To change any variable, type its name on the command line, or **SETVAR** and the name (`?` lists all of them with their current values).

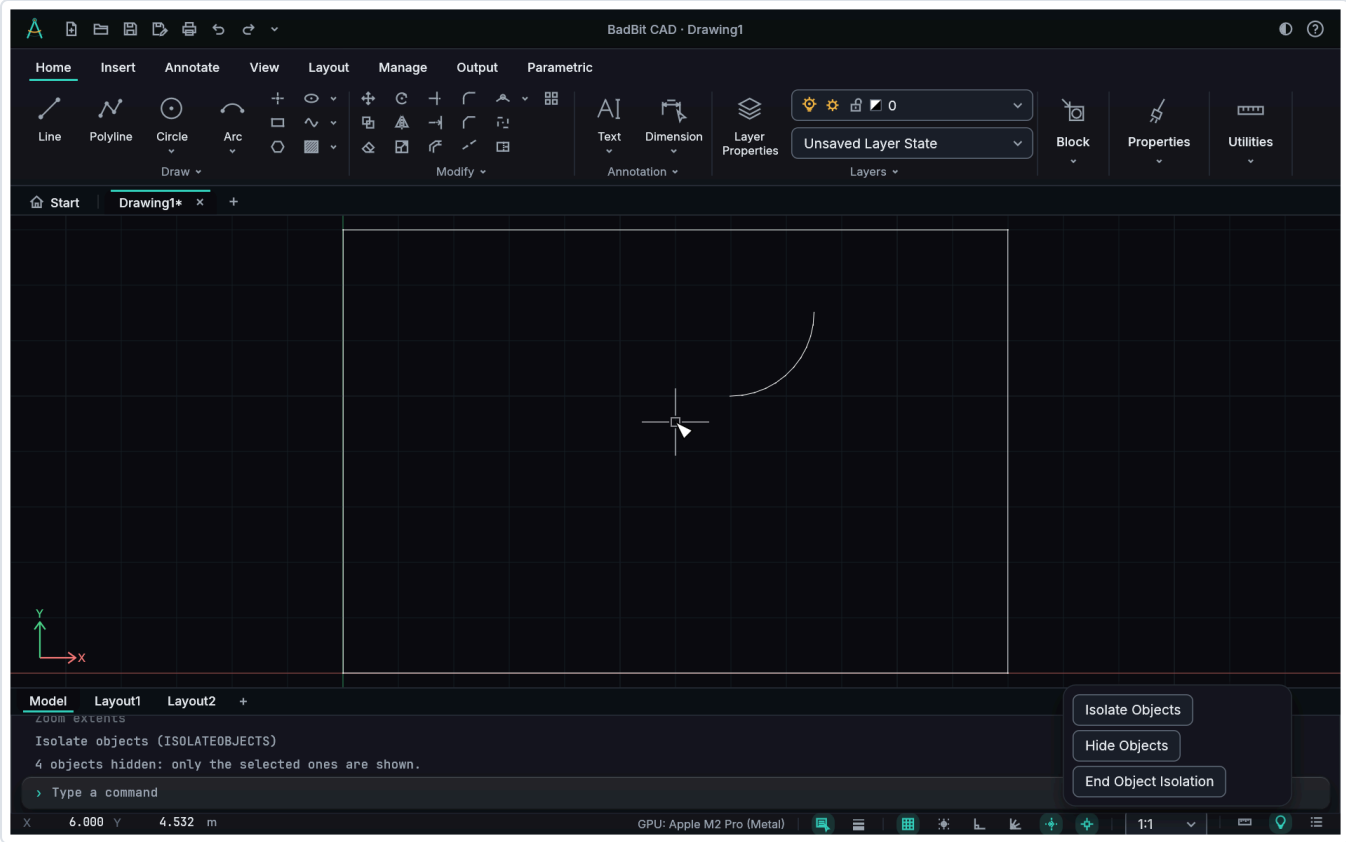
Isolating and hiding objects (ISOLATEOBJECTS, HIDEOBJECTS, UNISOLATEOBJECTS)

To work with only some objects without touching the layers, as in AutoCAD:

Command	What it does
ISOLATEOBJECTS	Hides all objects in the current space (model or layout) except the chosen ones.
HIDEOBJECTS	Hides the chosen objects.
UNISOLATEOBJECTS	Shows again all objects hidden by the two commands above.

With objects already selected, ISOLATEOBJECTS and HIDEOBJECTS take effect immediately; without a selection, they ask `Select objects:` and `Enter` ends (with nothing chosen, it cancels). Isolating again, with objects already hidden, also hides the ones not chosen now; hidden objects stay hidden until UNISOLATEOBJECTS. Objects drawn afterwards appear normally.

Hiding is **display only**: no property of the object changes (layer, color, linetype), the drawing is not modified and undo does not undo it. A hidden object **does not appear** (on the graphics card, and in drawing by the CPU only), **cannot be selected** (not by a click, a window, `ALL` or `Ctrl` + `A`), **does not snap** and **does not come out in plots** or PDFs. A hidden text with a background mask no longer covers what lies beneath it.



ISOLATEOBJECTS with the outline and the door chosen: the rest disappears, the Isolate objects button on the status bar (the light bulb) lights up and clicking it opens the menu.

The **Isolate objects** button on the status bar (the light bulb, between Customize and Units) stays lit while there are hidden objects. Clicking it opens the menu **Isolate Objects**, **Hide Objects** and **End Object Isolation** (the last one only with hidden objects). The same **Isolate** submenu is in the right-click menu with objects selected (see *Right-click menu*, below).

The **OBJECTISOLATIONMODE** variable (in the profile, as in AutoCAD) says whether hidden objects stay hidden after the drawing is closed: **0** (the default), only in this session (when the drawing is opened again, everything appears); **1**, they are saved in the `.bbd` and the drawing opens with them hidden (UNISOLATEOBJECTS brings them back). DXF and DWG do not carry the hidden state: the AutoCAD help and the open DWG specification do not say how AutoCAD stores it, so saving as DXF or DWG saves all objects visible.

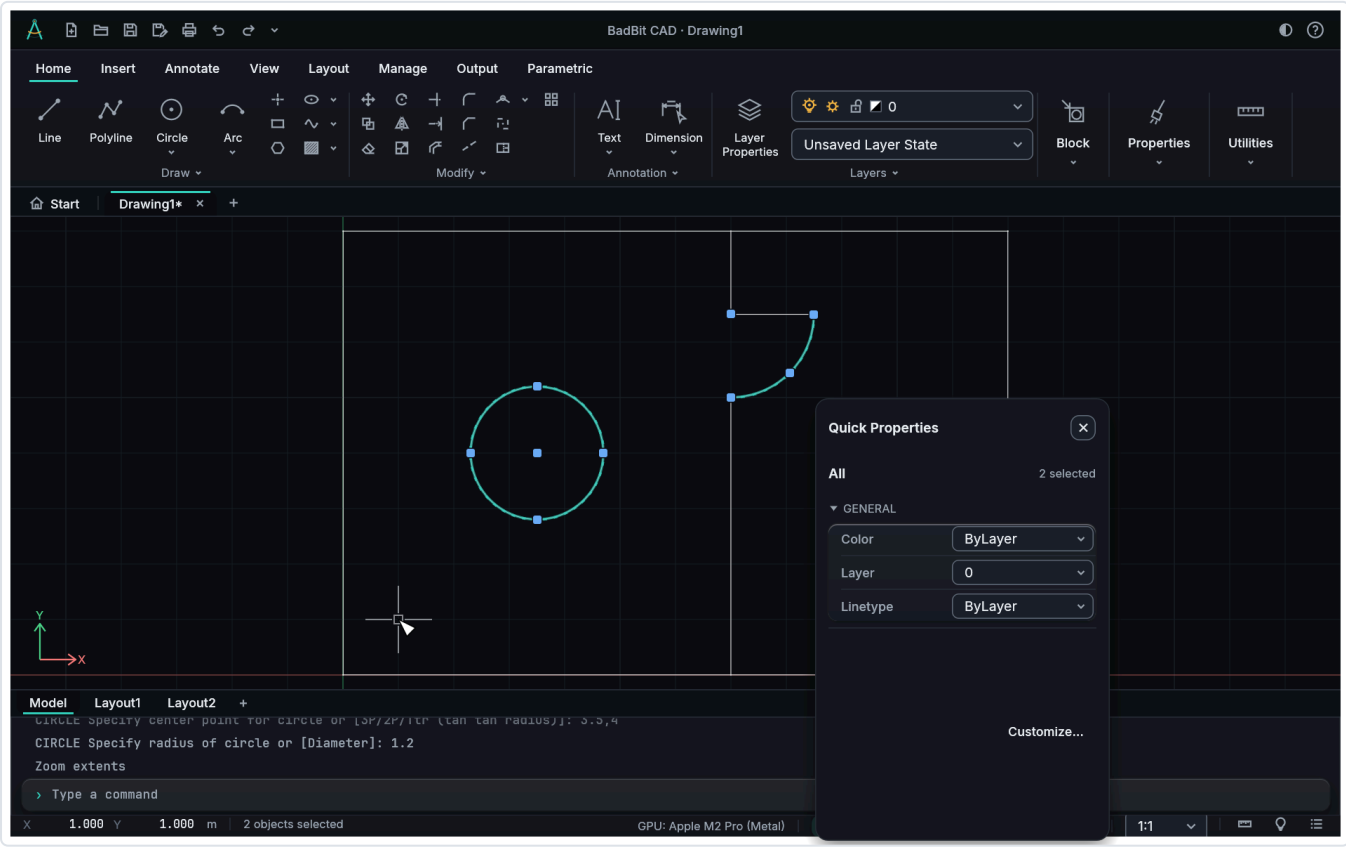
Right-click menu

With objects selected and no command active, right-clicking in the drawing opens the edit menu, like the AutoCAD one: **Repeat** the last command, the **Clipboard** submenu (Cut, Copy, Copy with Base Point, Paste, Paste as Block and Paste to Original Coordinates, with their shortcuts) and the **Isolate** submenu (Isolate Objects, Hide Objects and End Object Isolation). Without a selection, the right button is still `Enter`; on a block insertion selected on its own, it opens the insertion menu (Edit Reference In-Place, Block Editor...). `Esc` or a click outside closes the menu.

Grips

With objects selected and no command active, the **grips** appear: small blue squares at the key points of each object (up to 100 selected objects, as in AutoCAD).

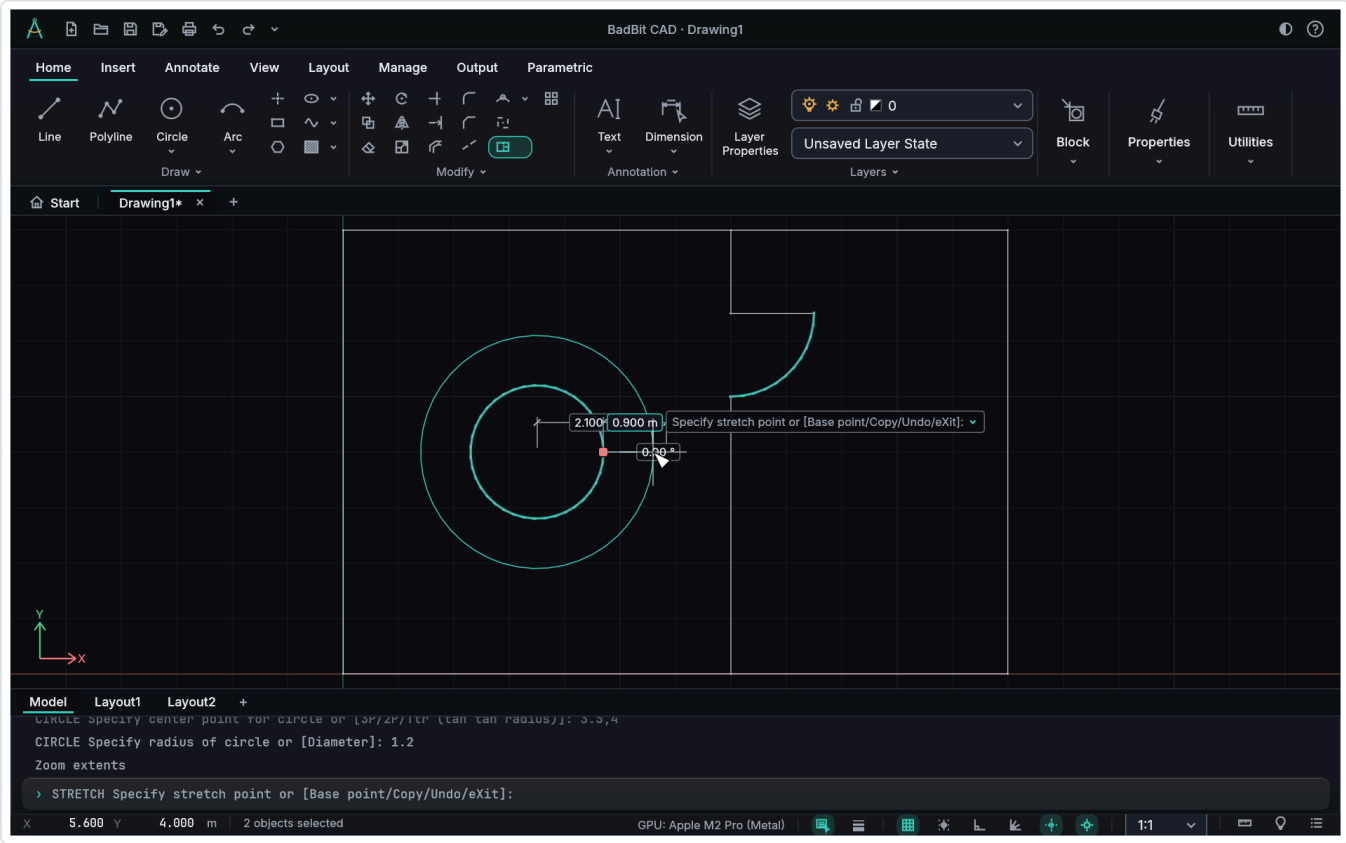
Object	Grips
Line	the two endpoints and the midpoint
Circle	the center and the four quadrants
Arc	the two endpoints, the midpoint and the center
Polyline	the vertices and the midpoint of each segment
Point	the point itself
Ellipse	the center and the ends of the axes (on an arc, also the two endpoints)
Spline	the fit points (moving one rebuilds the curve) or the control vertices
Construction line	the root (moves) and two points that rotate the line around it
Ray	the start (moves) and the through point (rotates)



The table and the door selected, with their grips.

Click a grip to make it hot (it turns red): it follows the cursor, with object snaps, Ortho and polar, until the next click or a typed point. You can also drag, as in AutoCAD: press on the grip, drag and release where it should go. An endpoint moves; the midpoint of a line and the center move the whole object; a quadrant changes the radius of a circle; the end of an ellipse axis changes the size of that axis (the

endpoints of an elliptical arc change where it starts and ends); the midpoint of an arc changes the curve; the midpoint of a polyline segment moves the segment. Grips in the same place, such as the corner of two lines, move together. While the grip moves, dimensions show the length of the segments next to it (and the radius, on a circle or arc).



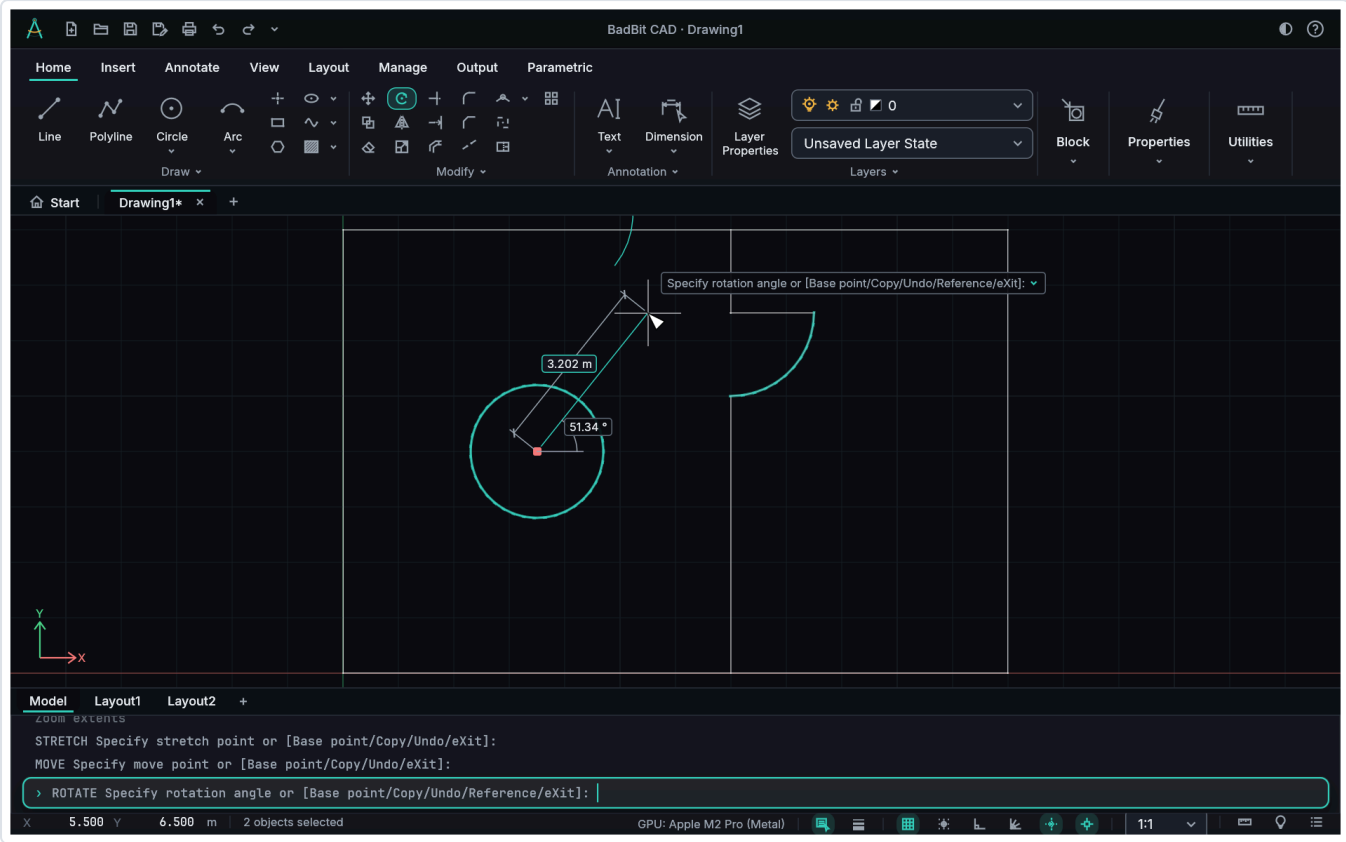
The table's quadrant hot: the circle grows to the cursor.

Grip modes

With the grip hot, `Enter` (or Space, or the right button) changes the mode, in the AutoCAD order: **Stretch**, **Move**, **Rotate**, **Scale** and **Mirror**. The mode can also be typed (`ST`, `MO`, `RO`, `SC`, `MI`). Move, Rotate, Scale and Mirror apply to the whole selection, starting from the hot grip:

Option	What it does
Base point (`B`)	Another base point, instead of the grip.
Copy (`C`)	Each point makes a copy and the grips stay hot (`Enter` ends). Holding `Ctrl` while clicking also copies.
Undo (`U`)	Removes the last copy.
Reference (`R`, in Rotate and Scale)	The current angle or length (typed or by two points), as in ROTATE and SCALE.
eXit (`X`)	Releases the grips.

In Rotate, a number is the angle; in Scale, the factor. `Esc` releases the grips. The selection remains afterwards, and everything done with the grips is a single undo step.



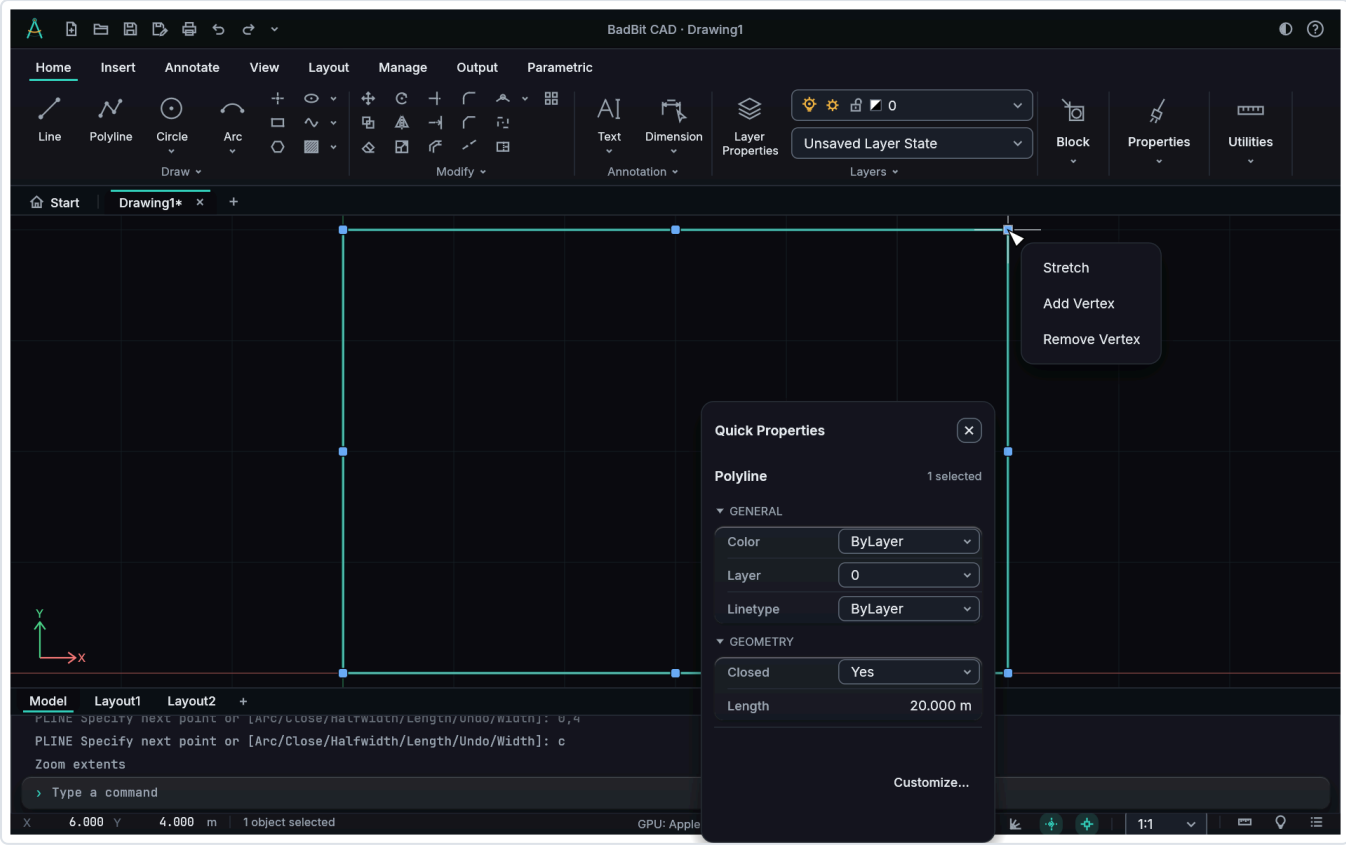
Rotate mode: the selection rotates around the hot grip to the cursor.

Several grips together: `Shift`+click makes each grip hot (or cold); then a click (without Shift) on one of them edits all of them, each moving by the same amount.

Grip menu

Resting the cursor half a second over a grip, without pressing the button, brings up its menu, as in AutoCAD (pressing and holding the grip stretches: then the menu does not appear). Move the cursor to the option and click:

Grip	Options
Polyline vertex	Stretch, **Add Vertex** (a new vertex after this one, at the clicked point), **Remove Vertex**
Midpoint of a polyline segment	Stretch, Add Vertex, **Convert to Arc** (the segment passes through the clicked point) or **Convert to Line**
Endpoint of a line or arc	Stretch, **Lengthen** (the endpoint moves only along the direction of the line, or along the arc's circle)
Midpoint of an arc	Stretch, **Radius**



The polyline vertex menu.

Welding by grip

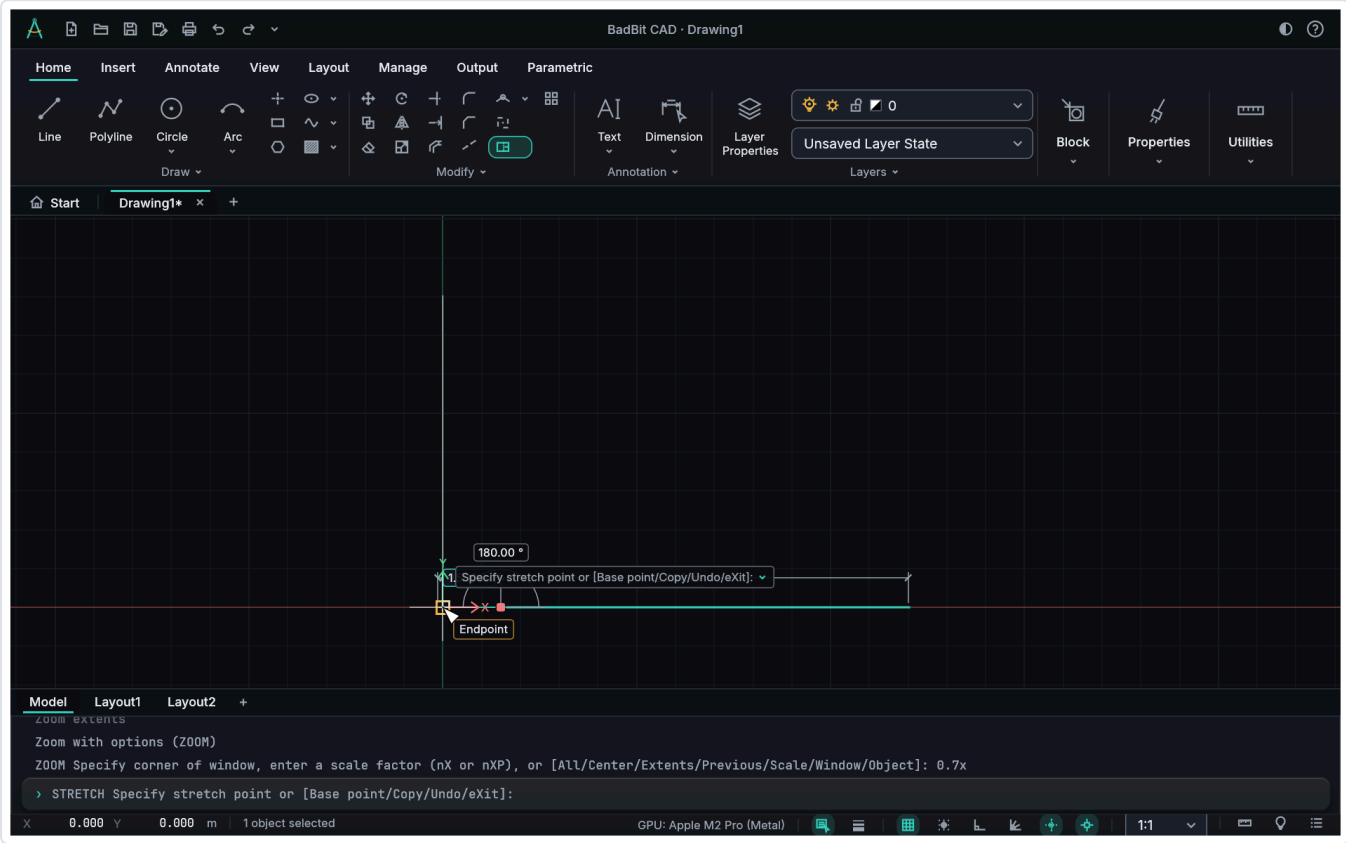
A BadBit CAD feature (AutoCAD does not have it). With no command, with the object selected, stretch the **endpoint** grip of a line, arc or open polyline (by dragging, or by clicking the grip and then the destination) and release it exactly on the endpoint of another such object: the endpoint snap takes you there. Instead of just touching, the two become a single object:

The two	Become
Lines in the same direction	one line
Lines at an angle (a 90° corner or any other)	a polyline with the two segments, in order
Consecutive arcs of the same circle	one arc (or the circle, if they close the loop)
A line or arc at the end of a polyline	the polyline, with the extra segment (the arc goes in as a curved segment)
Two polylines, end to end	one polyline; if the other end also meets (the loop closes), it becomes closed

The welded object keeps the properties of the two and stays selected, and the command line reports it (for example, "2 objects welded into 1 polyline"). It is all a single undo step: `Ctrl` + `Z` brings the two pieces back as they were before the drag.

Only objects with the same properties (layer, color, linetype, linetype scale and lineweight) are welded; with different properties, the endpoint just touches, as before. Also not welded: the grip copy (the **Copy** option or `Ctrl` on the click), the several hot grips of `Shift`, or an endpoint released where two others already meet (it cannot tell which one to weld).

To turn it on and off, use the checkable item **Weld when dragging a grip** in the expanded Modify panel (**Modify** ▾) or the `GRIPWELD` variable (1 on, the default; 0 off), which is stored in the user profile.



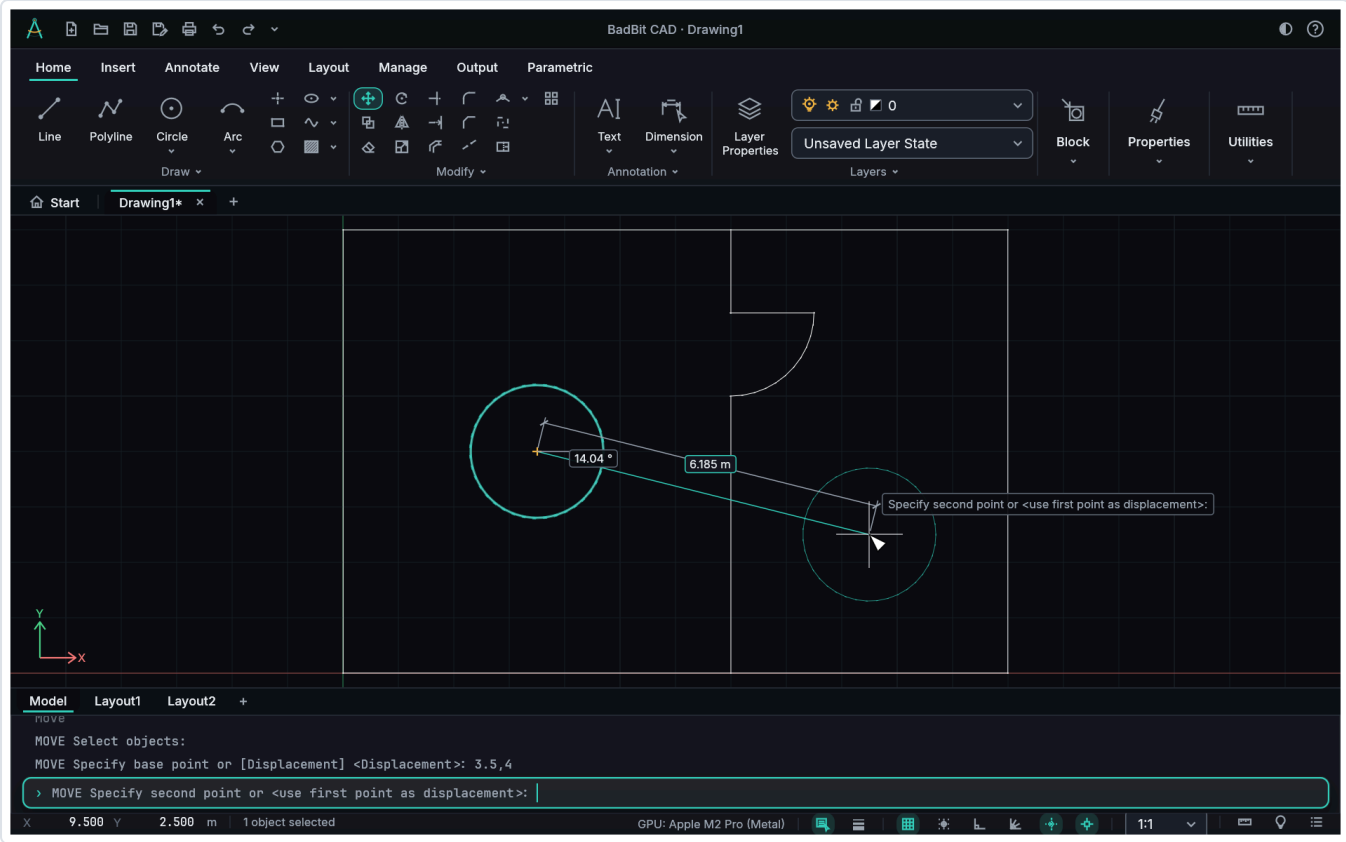
The end of the bottom wall dragged to the end of the left wall (the endpoint snap marks the point); on release, the two become a polyline.

MOVE (alias M)

Moves objects from a base point to a second point. While the second point is being chosen, the selection follows the cursor.

As in AutoCAD, there are two shortcuts for moving by a vector:

- **Displacement** (`D`, or `Enter` at the base point): type the vector, `x,y` (for example, `3,4` moves 3 in x and 4 in y). The last displacement appears between `< >` and `Enter` repeats it.
- `Enter` at the second point uses the **first point as displacement**: the base point `2,5` moves 2 in x and 5 in y.



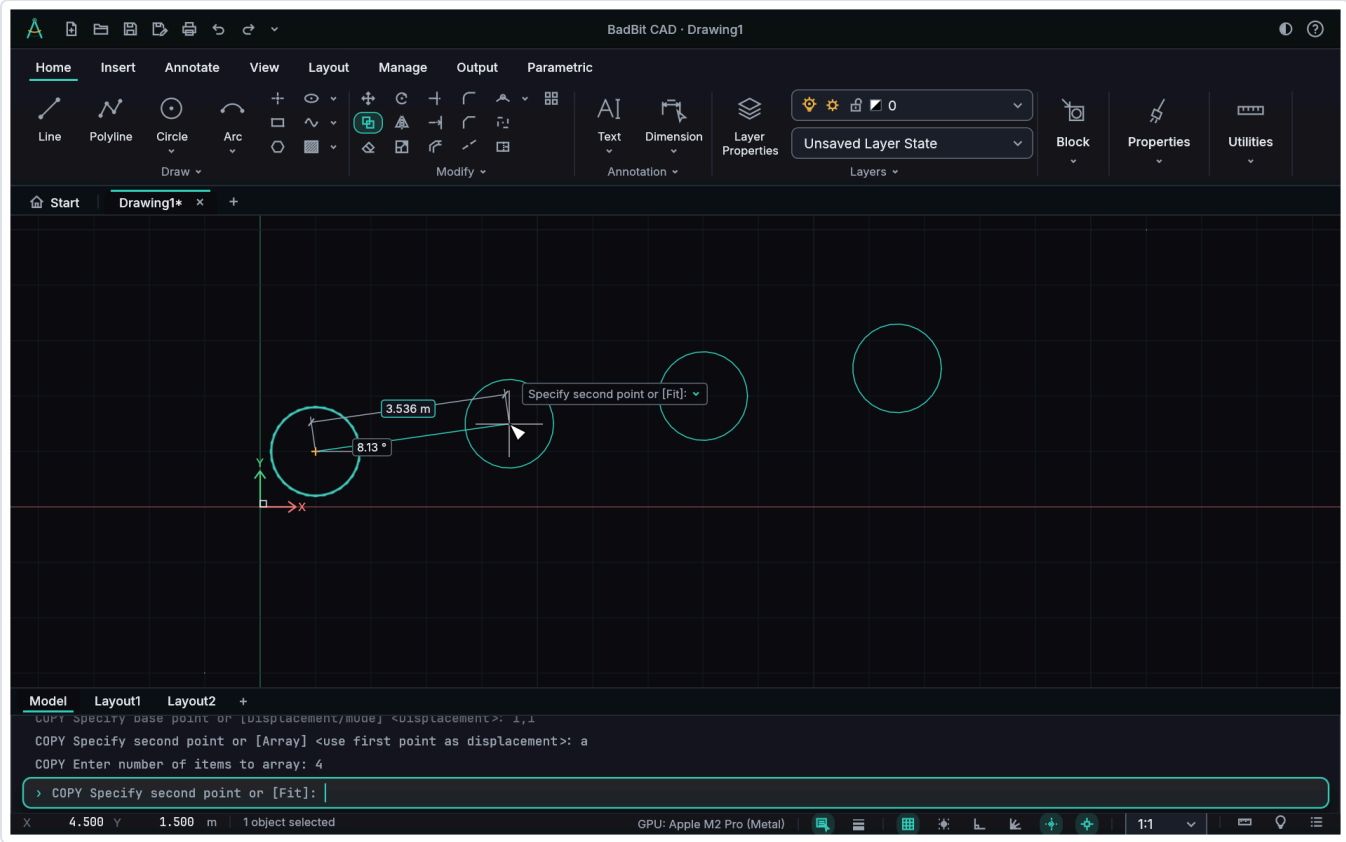
MOVE: the selection dragged to the cursor.

COPY (aliases CO, CP)

Like MOVE, but leaves the original. At the start, the command line shows the copy mode, as in AutoCAD ("Current settings: Copy mode = Multiple"). MOVE's **Displacement** and first point as displacement also apply; they make a single copy and end.

Option	What it does
mOde (`O` , at the base point)	**Multiple** (`M` , the default): each click creates a copy, until `Enter` or **Exit** (`E`). **Single** (`S`): one copy and it ends. The mode is remembered.
Array (`A` , at the second point)	Type how many items (the original counts) and the second point: the copies repeat at that distance, in a line. With **Fit** (`F`), the second point is that of the last item, and the others are divided equally between it and the original.
Undo (`U`)	Removes the last copy (or the whole last array) and keeps asking.

The whole COPY is a single undo step.

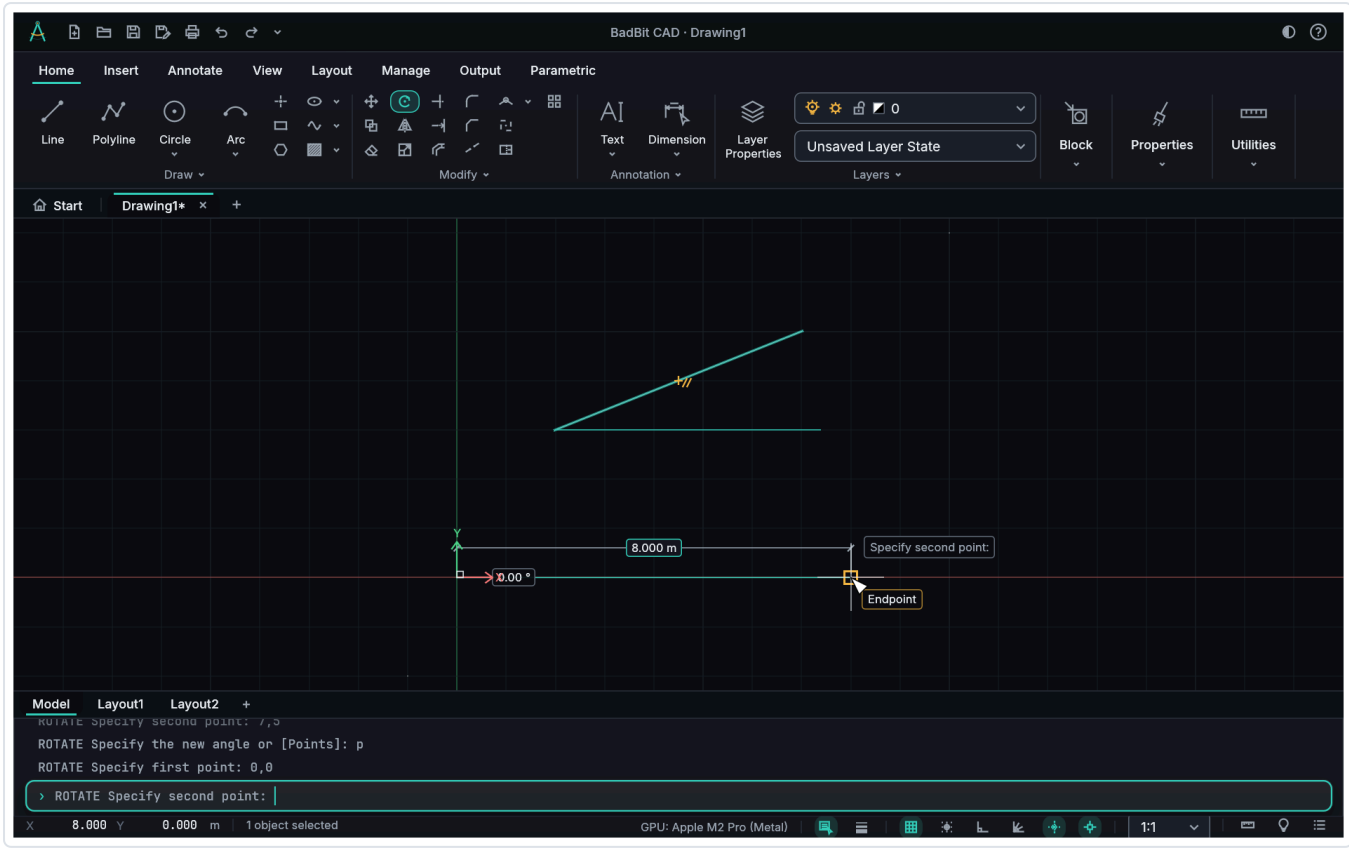


COPY with an Array of 4 items: the preview shows the three copies up to the cursor, at the same distance.

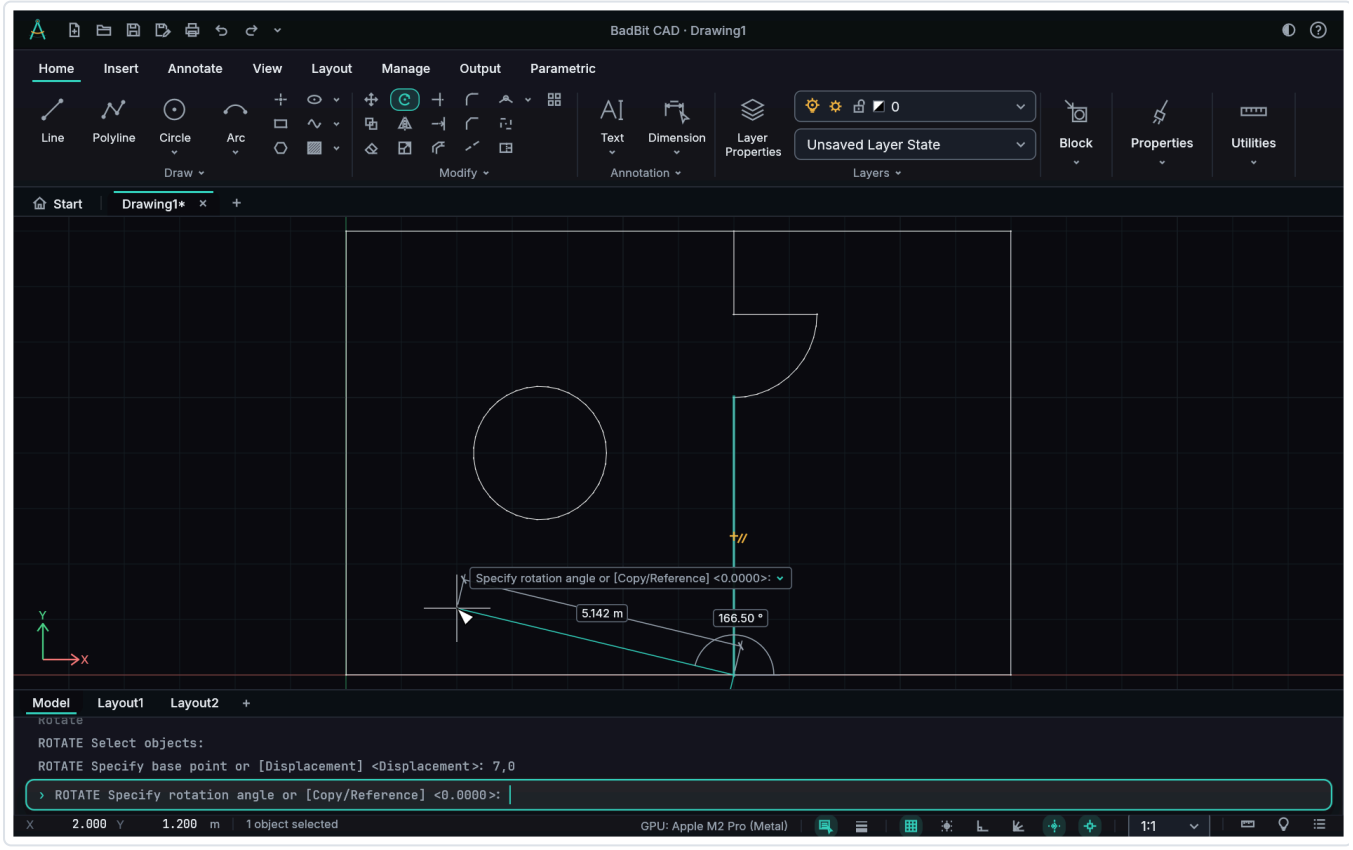
ROTATE (alias RO)

Rotates objects around a base point. After the base point, type the angle (in degrees, counterclockwise) or point with the cursor: the selection rotates along, in the preview.

- **Copy** (`C`): rotates a copy and leaves the original.
- **Reference** (`R`): enter the current reference angle and then the new angle; the object rotates by the difference. Useful for aligning something that is askew without knowing the exact angle. The reference angle can be typed or given by **two points** (the first click, or typed point, asks for the second point: for example, the two ends of the askew line). At the new angle, **Points** (`P`) also asks for two points (the ends of the line you want to align with); without the option, the angle is from the base point to the cursor.



ROTATE with Reference: the askew line, pointed to by its two ends, is aligned with the line below using the Points option.



ROTATE: the partition rotating around the base point, following the cursor.

SCALE (alias SC)

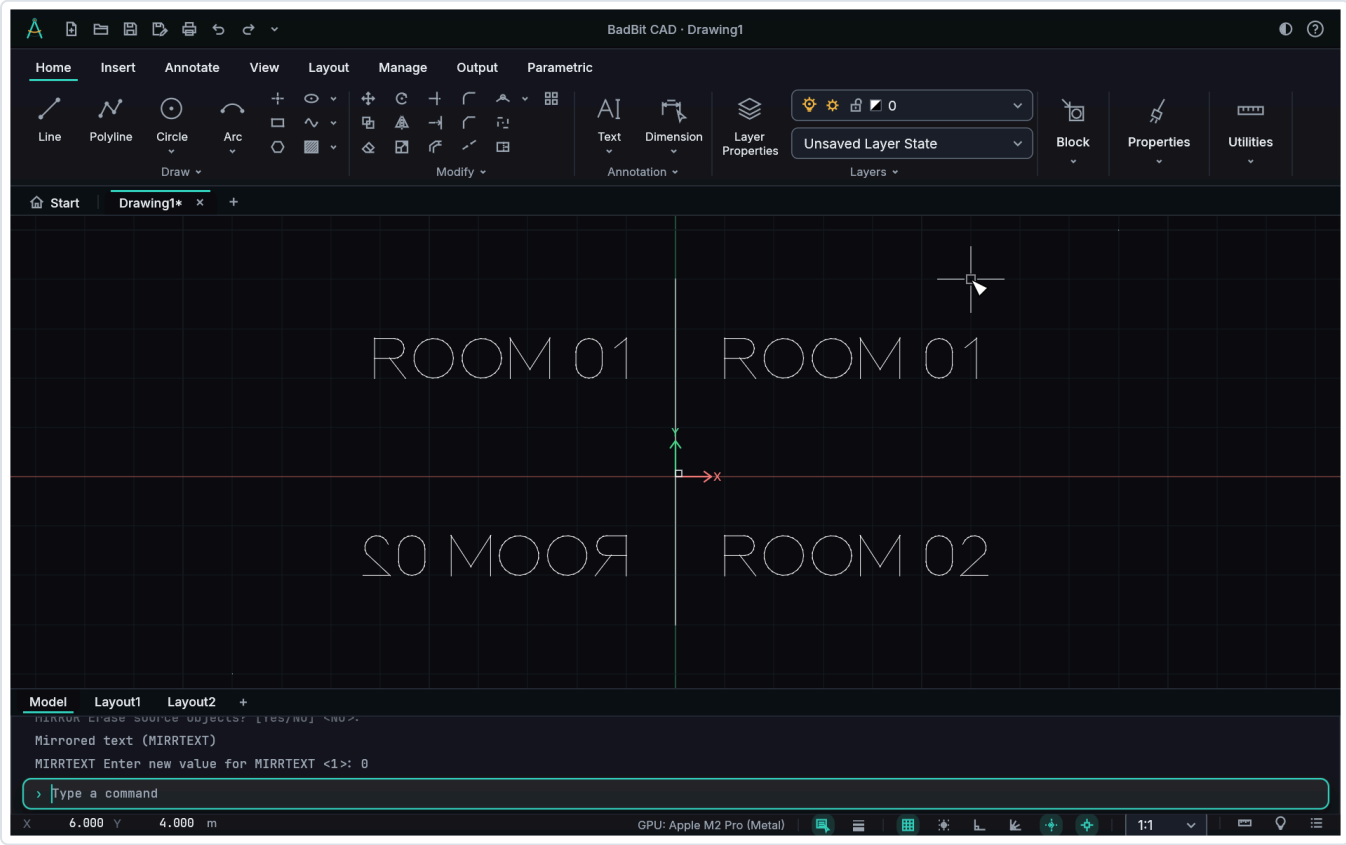
Enlarges or reduces objects from a base point. Type the factor (2 doubles, 0.5 halves) or point with the cursor: the distance from the base point to it is the factor.

- **Copy** (`C`): scales a copy and leaves the original.
- **Reference** (`R`): enter the current length and the new length; the factor is the ratio between them. Useful for giving a part an exact dimension. As in ROTATE, the reference length can be given by two points, and the new length accepts **Points** (`P`).

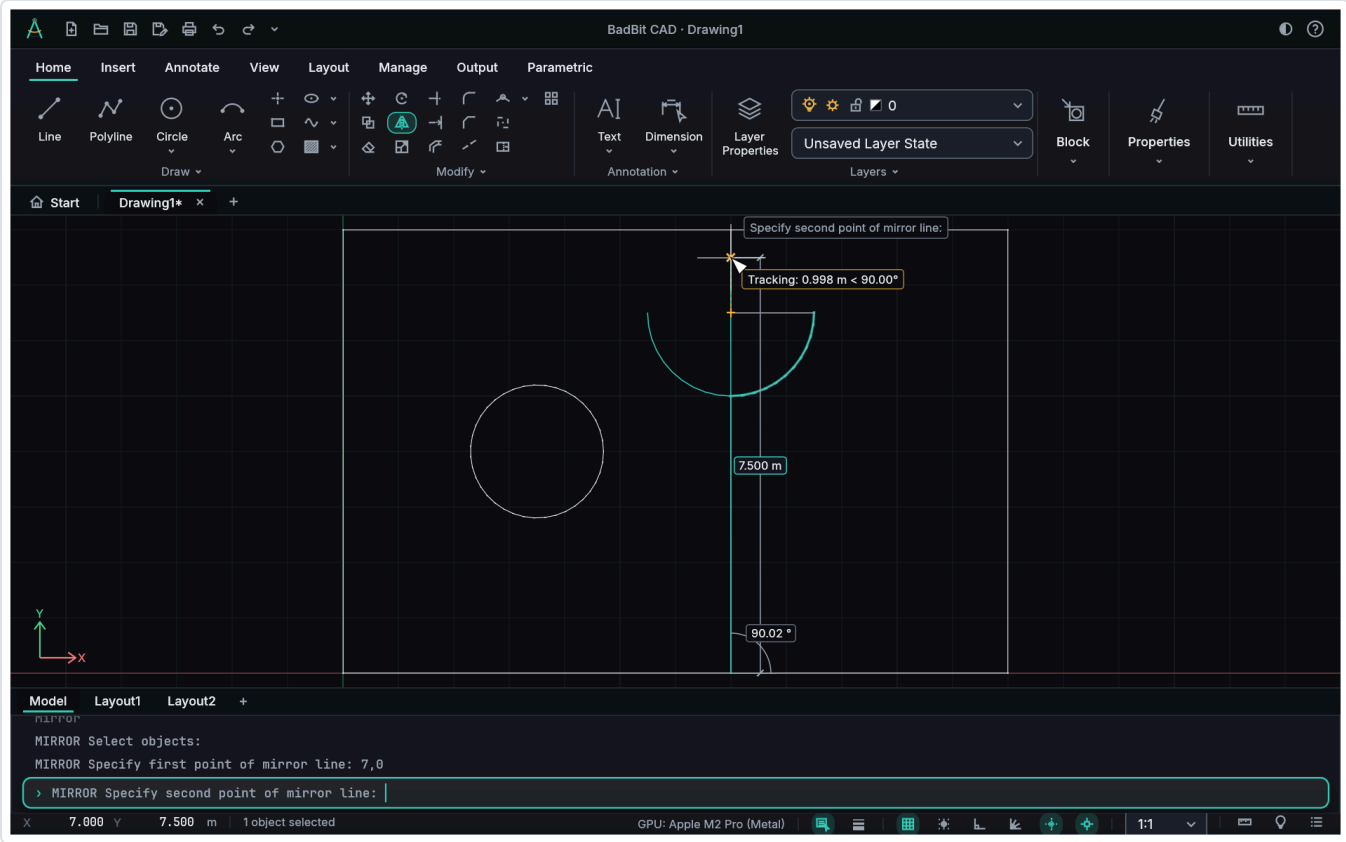
MIRROR (alias MI)

Mirrors objects about a line defined by two points. At the end, it asks "Erase source objects? [Yes/No] <No>": `Enter` keeps the original and creates the mirrored copy; `Y` leaves only the mirrored one. Arcs reverse their direction, as in a mirror.

Texts follow the **MIRRTEXT** variable, as in AutoCAD. With 0 (the default), the text goes to the mirrored place, but the letters stay readable: mirrored about a vertical line, it still reads from left to right (and the justification switches sides, Left becomes Right, as in AutoCAD); about a horizontal line, it stays upright. With 1, single-line text becomes its mirror image, letter by letter. Multiline text (MTEXT) always stays readable.



The two texts on the right mirrored about the vertical line: on top with MIRRTEXT 0 (readable), below with MIRRTEXT 1 (mirrored).



MIRROR: the door mirrored about the line through the corner of the partition.

The three appear in the Modify panel of the ribbon and each operation is a single undo step.

OFFSET (alias O)

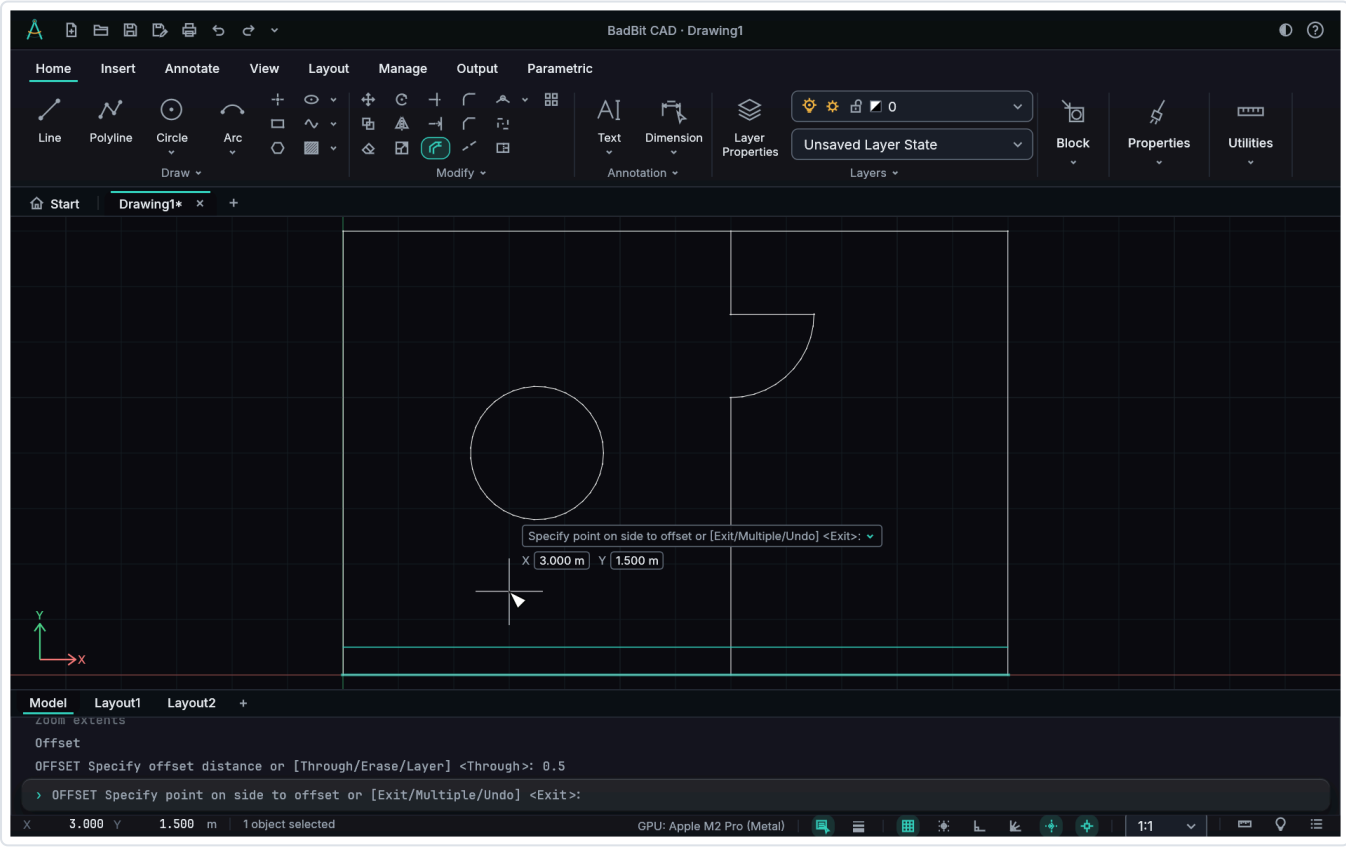
Creates parallel copies at a distance, such as double walls or outlines. Enter the distance (typed or by two points); then, repeatedly, click the object and the side where the parallel copy should go. `Enter` ends.

- **Through** (`T`): instead of a fixed distance, each parallel copy passes through the clicked point.
- **Erase** (`E`): erases the source object after creating the parallel copy.
- **Layer** (`L`): the parallel copy goes to the source layer (default) or to the current layer.
- **Multiple** (`M`, after choosing the side or the object): each click makes one more parallel copy, from the last one made, at the same distance (like the layers of a wall); `Enter` goes back to asking for the object.
- **Undo** (`U`), inside the command, removes the last parallel copy. After the command, a U removes all the ones it created.

On a polyline, segments that would come out reversed in an inward offset (the loops, on a short segment) are dropped, and their neighbors meet. Outer corners follow the OFFSETGAPTYPE variable: 0 extends the segments until they meet (the default), 1 rounds them with an arc whose radius is the distance, 2 chamfers.

The distance and the options are remembered for the next use; `Enter` at the first prompt repeats the last distance.

Object	Parallel copy
Line	A parallel line, on the side of the click.
Circle and arc	Same center, larger (outside) or smaller (inside) radius.
Polyline	Each segment parallel, with the corners joined where the parallels cross (or as OFFSETGAPTYPE asks, at outer corners).
Ellipse	The parallel curve (which is not an ellipse), as a spline, as in AutoCAD. Inward, down to the smallest radius of curvature.
Spline	Another spline, parallel, on the side of the click.
Construction line and ray	Another construction line (or ray), parallel, on the side of the click.



OFFSET of 0.5 m: the parallel copy of the outer wall appears on the cursor side before the click.

TRIM (alias TR) and EXTEND (alias EX)

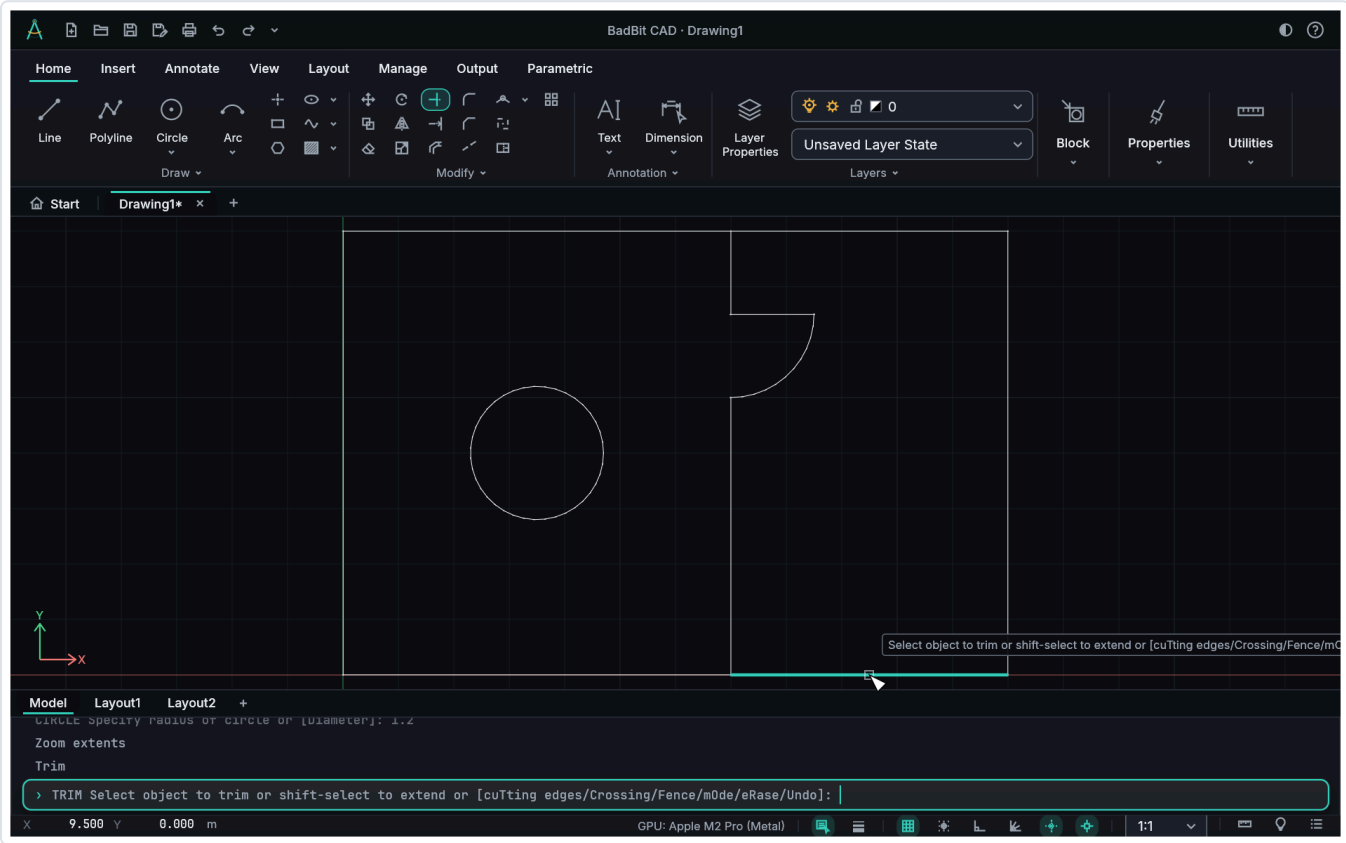
Both repeat until `Enter` and have the two AutoCAD modes (the **mOde** option, `0`, or the TRIMEXTENDMODE variable):

- **Quick** (the default): all objects in the drawing serve as cutting or boundary edges, without choosing them first. An object that crosses no other object is removed entirely by TRIM.
- **Standard**: the edges are chosen first ("Select cutting edges or <select all>:", with all the selection options; `Enter` with nothing uses all objects). The edges are highlighted, and only they cut or bound. An object that crosses no edge does not change.

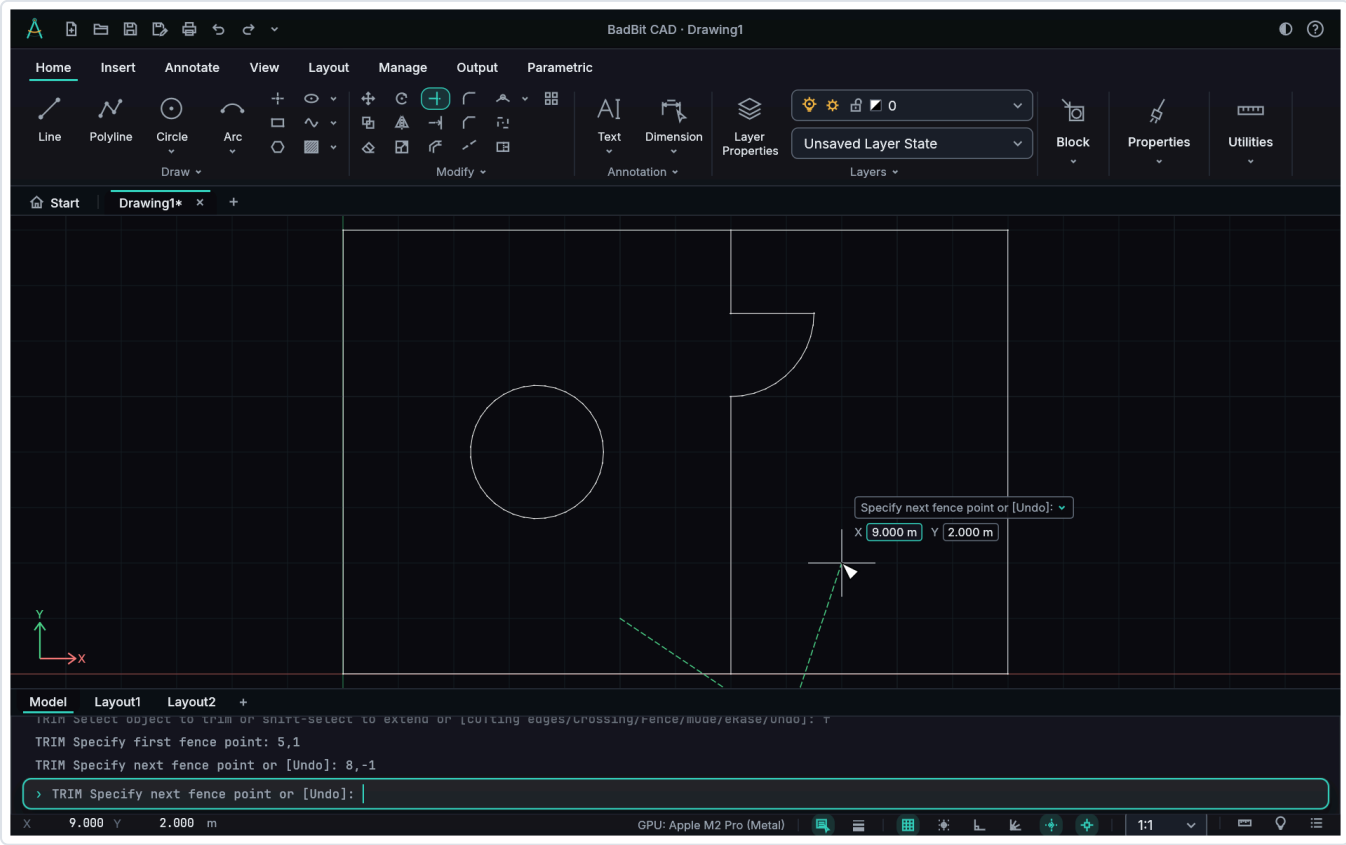
Then, at each click:

- **TRIM:** click the piece that should go. It disappears up to the neighboring intersections (or up to the end). On a circle, the arc on the other side remains; a closed polyline opens there.
- **EXTEND:** click near the end that should grow. Lines, open polylines and arcs go to the first object (or edge) ahead.
- With `Shift` held down, each one does the other: in TRIM, the click extends; in EXTEND, it trims.
- Before the click, the piece that goes (or comes in) is highlighted under the cursor.

Option	What it does
Fence (`F`)	A line of points: trims (or extends) at once everything it crosses, at the points where it crosses. In quick mode, **dragging** with the button held down makes a freehand fence.
Crossing (`C`)	Two corners: trims the pieces the window catches (in EXTEND, the ends closest to it).
cuTting edges (`T`) / **Boundary edges** (`B`)	In quick mode, chooses the edges (as in standard mode) from then on.
Edge (`E` , in standard mode)	**Extend** (`E`): edges count as extended, a line as an infinite straight line and an arc as the whole circle; **No extend** (`N`). This is the EDGEMODE variable.
eRase (`R` , in TRIM)	Chooses objects to erase (with the full selection) and goes back to trimming.
Undo (`U`)	Reverts the last click, fence or crossing. After the command, a U reverts everything it did.



TRIM: with the cursor on the bottom wall, the stretch up to the partition is highlighted; the click removes it.



TRIM with Fence: the line of points (dashed) passes through the bottom wall and the partition; Enter trims everything it crosses.

FILLET (alias F) and CHAMFER (alias CHA)

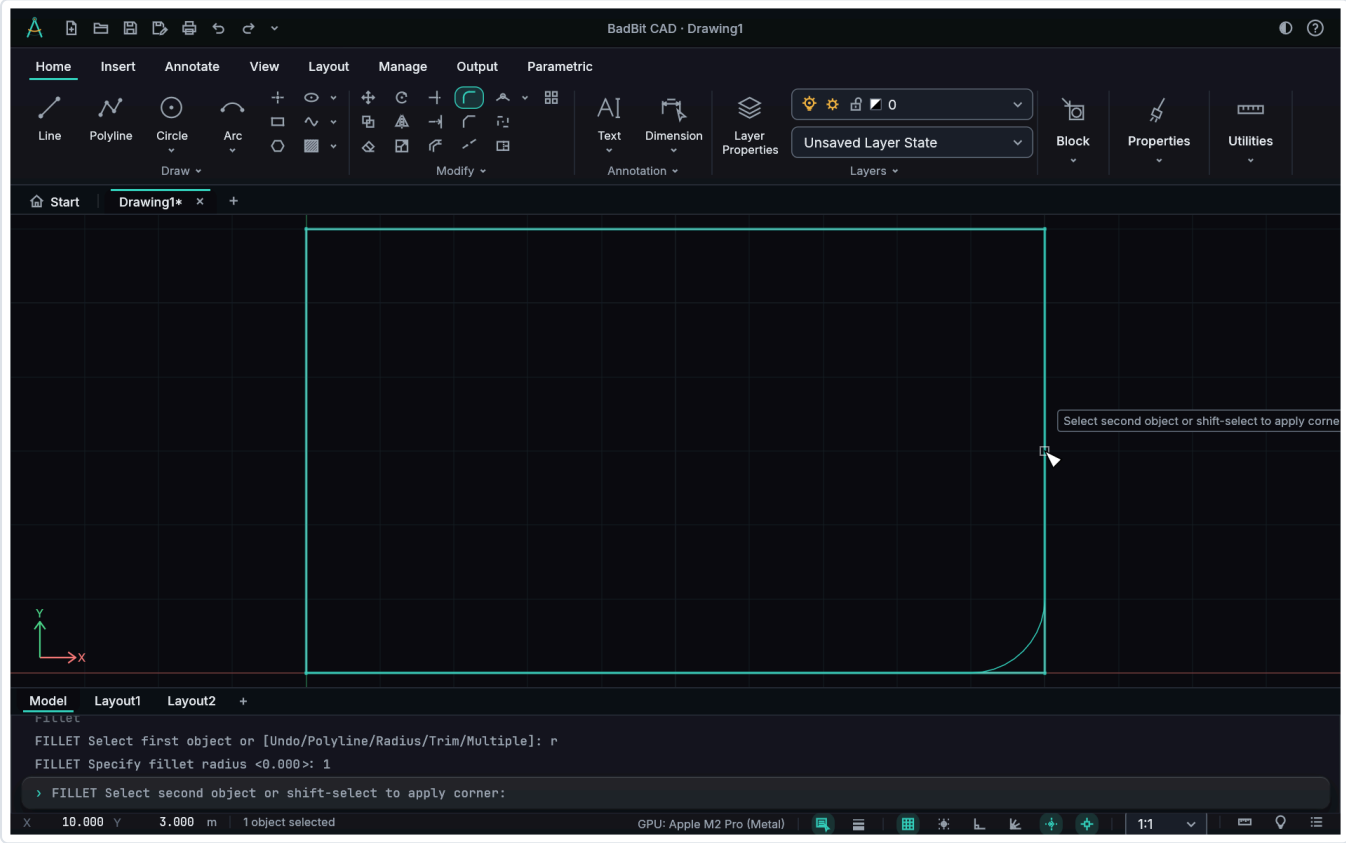
They round (FILLET) or cut straight (CHAMFER) the corner between two lines, or between two straight segments of a polyline. Click the first and the second, on the side that should remain: both are trimmed to the corner, and the arc or the chamfer segment goes in between them. Before the second click, the result appears in the preview.

On a polyline (the corner of a rectangle, for example), click the two neighboring segments: the corner goes into the polyline itself, as an arc or straight segment, and it remains a single object. The end of an open polyline also fillets with a line. Two parallel lines in FILLET get a semicircle at the end of the first one, as in AutoCAD.

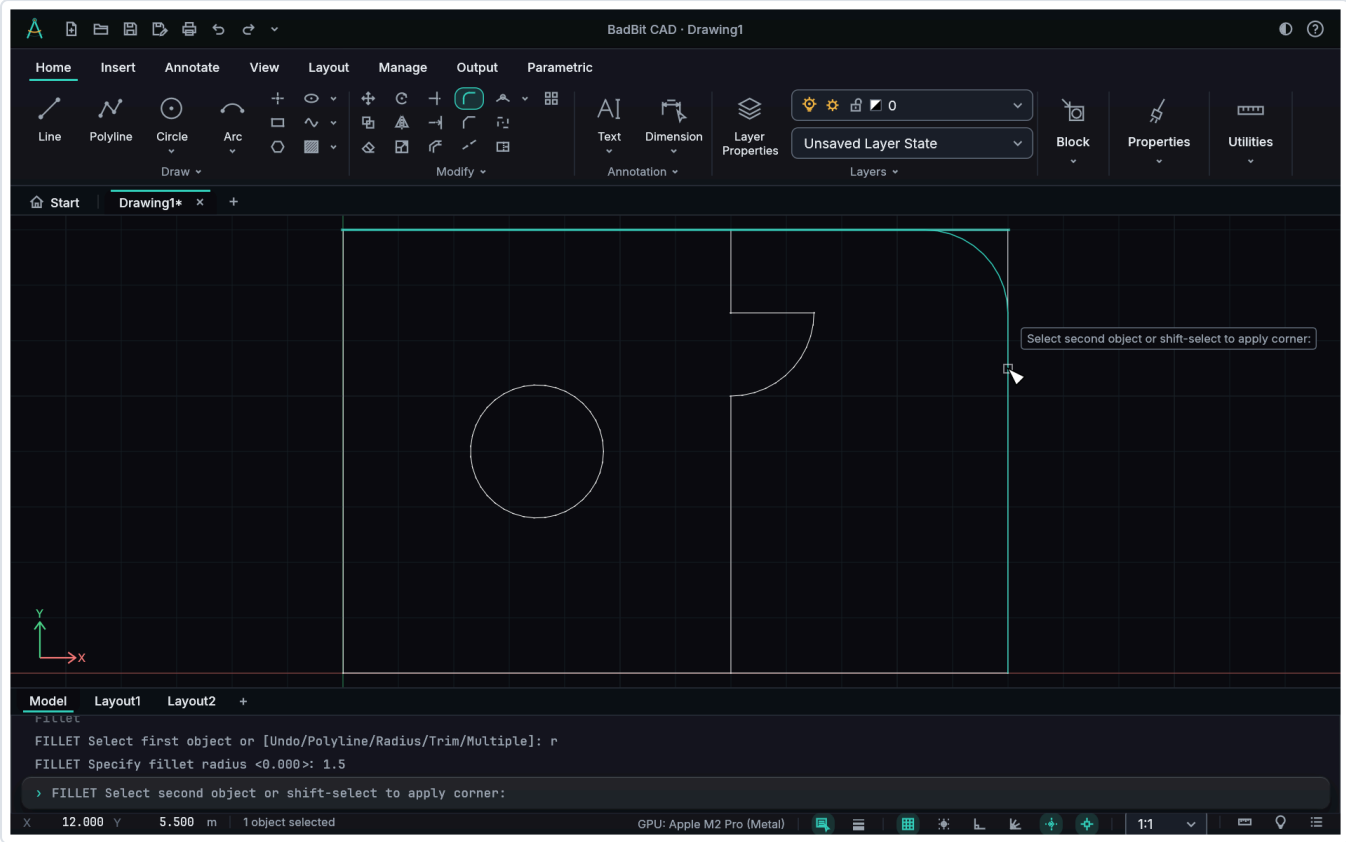
- **Polyline** (`P`): all corners of a polyline at once (those that do not fit the radius or the distances stay as they are).
- **Radius** (`R`, in FILLET): the radius of the arc. With radius 0, the lines just meet at a sharp corner (useful for closing corners that do not touch or that overshoot).
- **Distance** (`D`, in CHAMFER): the two distances, measured from the corner along each line.
- **Angle** (`A`, in CHAMFER): the length on the first line and the angle from it.
- **Trim** (`T`): with "No trim", the lines stay as they are and only the arc or segment is added.
- **Multiple** (`M`): repeats until `Enter`; within it, `U` reverts the last one.
- With `Shift` on the second click, the corner comes out sharp, without changing the stored radius.

The radius, the distances and the mode are remembered for the next use. The arc or chamfer goes to the layer of the lines, if both are on the same one; otherwise, to the current layer.

Arcs and circles (fillets tangent to curves) are not supported yet; they will come with the rest of the tangencies.



FILLET of radius 1 on the corner of a rectangle: the arc corner goes into the polyline itself.



FILLET of radius 1.5: with the top wall chosen, the rounded corner appears in the preview over the right wall.

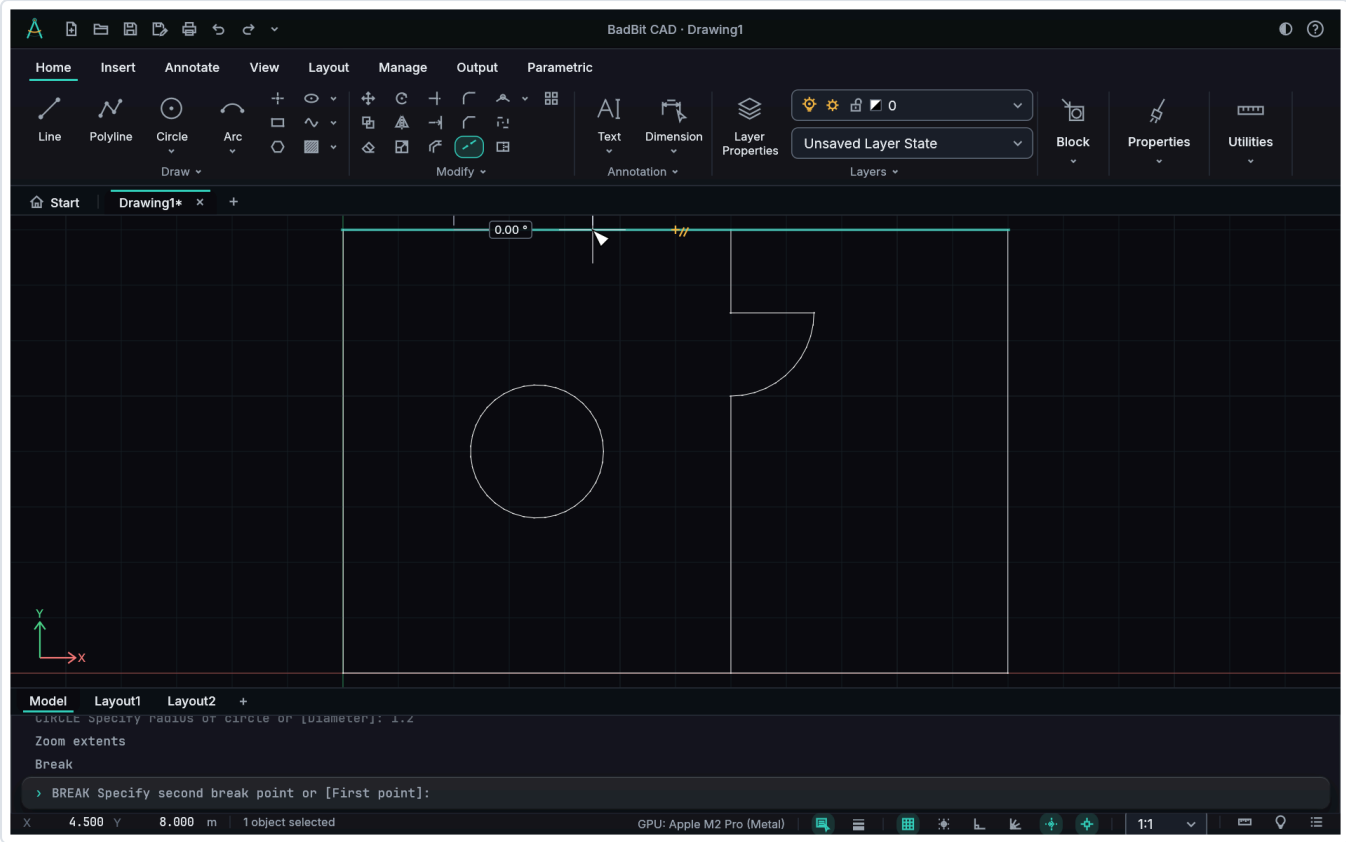
BREAK (alias BR)

Removes the piece of an object between two points. The click that picks the object is already the first point; then enter the second (with object snaps). With **First point** (`F`, or `P`), the first point is chosen again. Typing `@` at the second point just splits the object there, without removing anything.

On a circle, the arc from the first to the second point in the counterclockwise direction is removed, as in AutoCAD. A closed polyline opens.

Break at Point (BREAKATPOINT)

Splits an object at a single point, without removing anything: pick the object and click the break point. It is the button on the expanded Modify panel (**Modify** ▾), as in AutoCAD. A closed polyline split this way becomes an open one with its ends at the point; JOIN closes it again. It does not work on a circle (two points are needed).



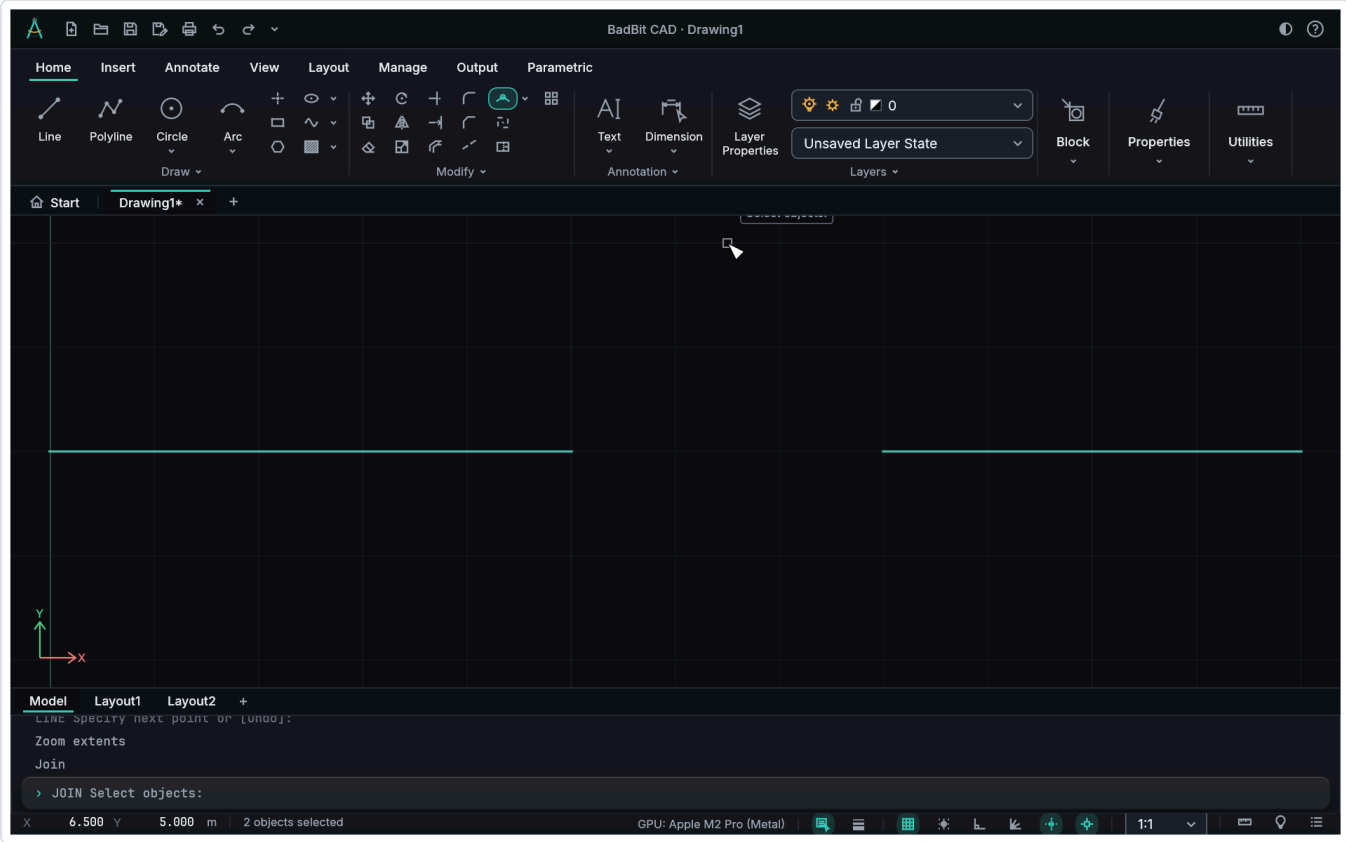
BREAK: the top wall picked at (2, 8); the second point defines the piece that is removed.

JOIN (alias J)

Joins objects into one. Select them and confirm with `Enter`:

- collinear lines become one line, even with a gap between them, as in AutoCAD: the new line runs from the farthest end of one to that of the other, covering the gap (non-collinear lines with a gap are not joined);
- arcs of the same circle (same center and radius) become one arc, also with a gap: the arc grows counterclockwise from the first one picked, as in AutoCAD (or becomes the circle, if they close the loop);
- lines and open polylines connected end to end become a polyline (closed, if the chain closes);
- arcs of the same ellipse that touch become an elliptical arc (or the whole ellipse);
- splines connected end to end become one spline (the reversed ones are flipped);
- a single open polyline with its ends at the same place (a closed one that was broken at a point) becomes closed again.

The first object picked is the source: the result keeps its identity (the properties and the layer).



Two collinear lines, with a gap between them, picked in JOIN: with Enter they become a single line.

Ends less than a millionth of a meter apart count as the same; at large coordinates (UTM), the tolerance grows along with them. Apart from collinear lines and arcs of the same circle, separate ends are not joined, as in AutoCAD: for them there is PEDIT (Join with fuzz distance) and Join Ends. REGION, when building the boundary, still joins only what touches.

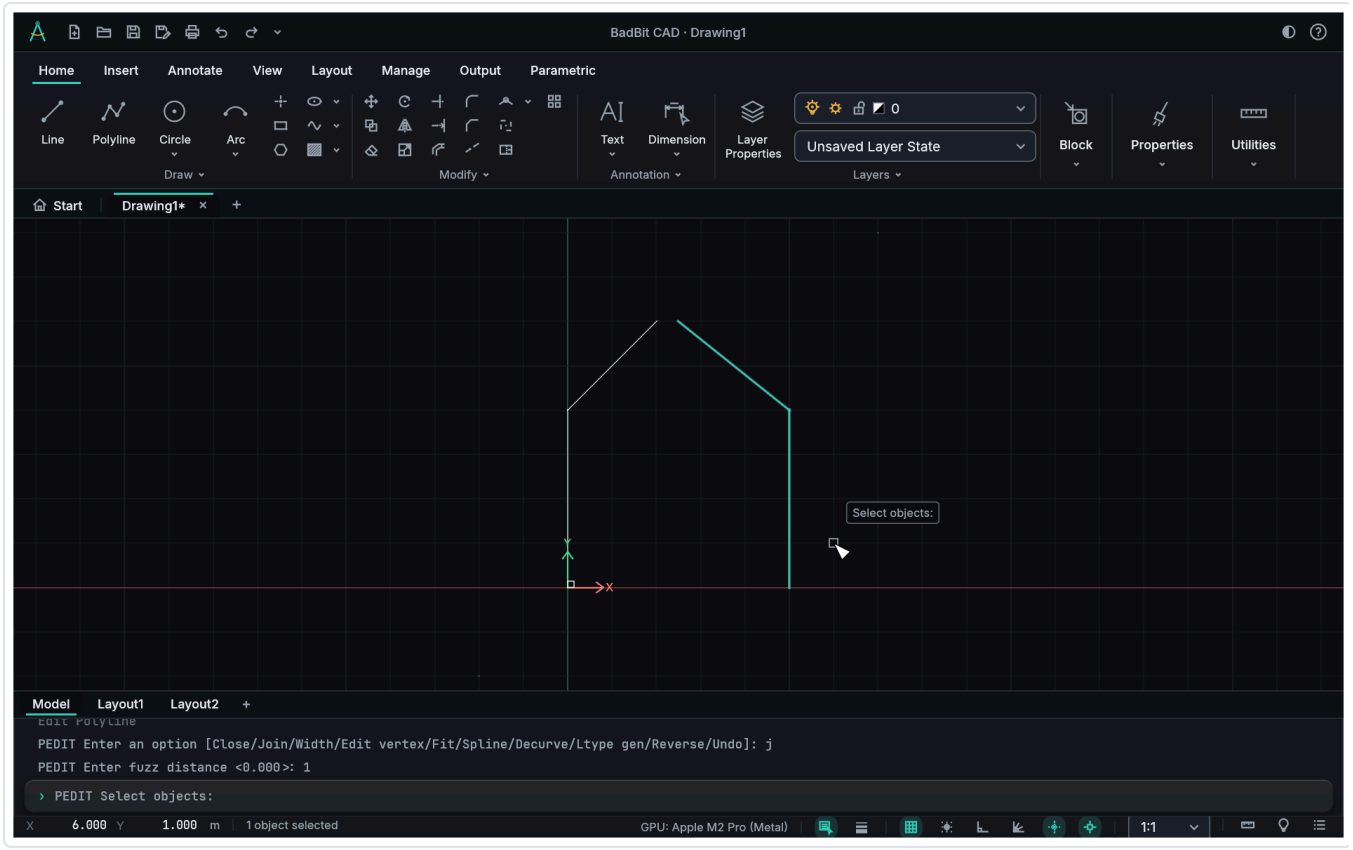
PEDIT (alias PE)

Edits polylines, like the AutoCAD PEDIT. Pick the polyline, or type **Multiple** (`M`) to pick several with the full selection. A line, an arc or a spline also works: the command asks whether they should be turned into a polyline (`Enter` accepts; a spline becomes straight segments). Then the options, repeating until `Enter`:

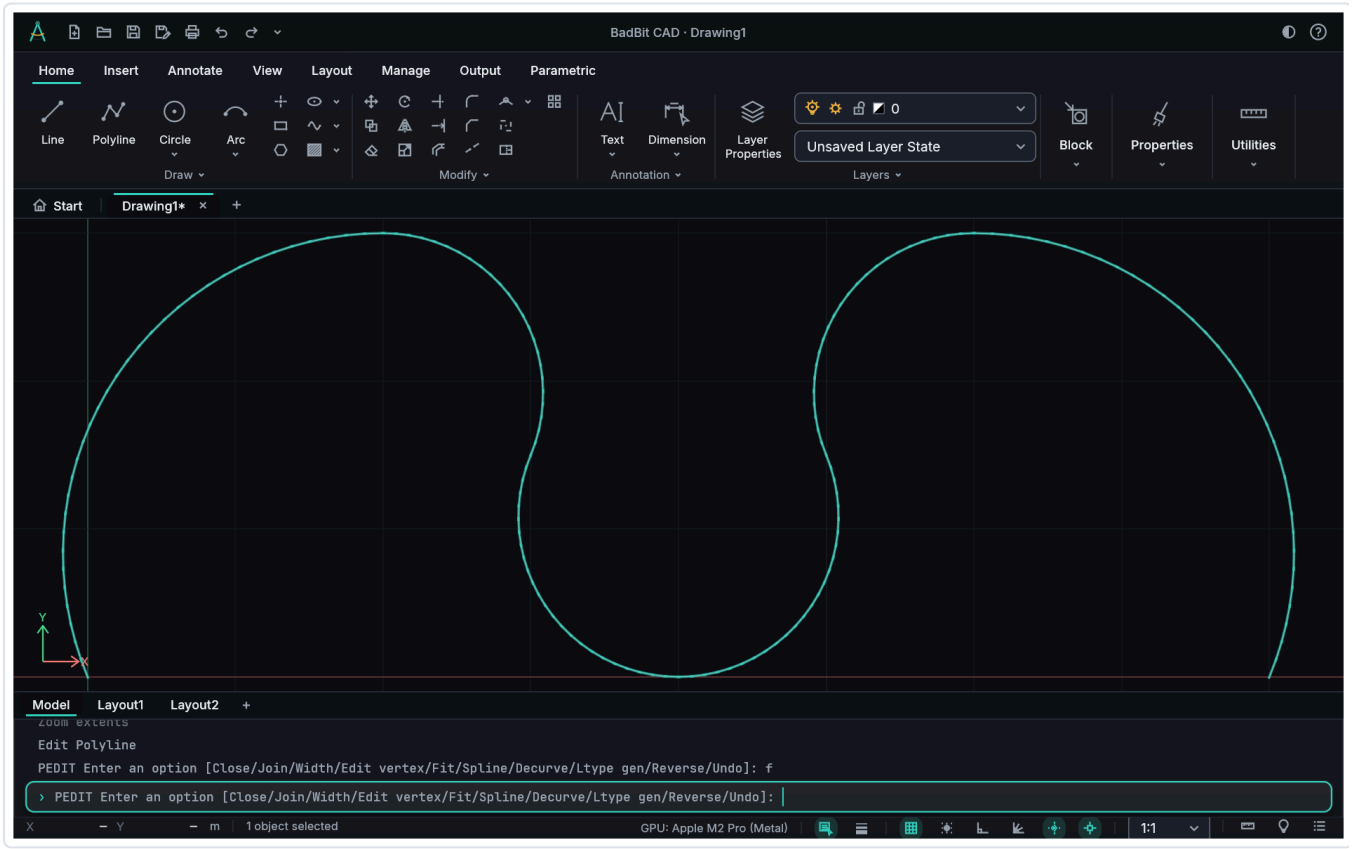
Option	What it does
Close (`C`) / **Open** (`O`)	Connects the last vertex to the first (needs three or more) / removes the closing segment.
Join (`J`)	Asks for the **fuzz distance** and the objects. Lines, arcs, open polylines and splines with an end within that distance are added; if the ends do not touch, a new segment covers the gap. In Multiple, the picked ones are joined to each other. The fuzz distance is remembered.
Width (`W`)	The same width on all segments.
Edit vertex (`E`)	An X marks the vertex; see below.
Fit (`F`)	Each segment becomes two arcs that pass through the vertices, with a smooth curve (tangents given in Edit vertex apply).
Spline (`S`)	The polyline becomes the B-spline that has the vertices as control points: it passes through the ends and approaches the others. SPLINETYPE chooses quadratic (5) or cubic (6, the default) and SPLINESEGS how many straight segments per span (8).
Decurve (`D`)	Returns to the vertices from before Fit or Spline, with straight segments (or just straightens the arcs).
Ltype gen (`L`)	On (`ON`): the linetype runs along the whole polyline, without restarting at each vertex; Off (`OFF`).
Reverse (`R`)	Reverses the direction of the polyline.
Undo (`U`)	Reverts the last option.

In **Edit vertex**: **Next** (`N` , or `Enter`) and **Previous** (`P`) move the X; **Break** (`B`) and **Straighten** (`S`) mark the vertex, move to another one and **Go** (`G`) removes what lies between the two (Break splits the polyline; Straighten replaces the path with a straight segment); **Insert** (`I`) adds a vertex after the marked one; **Move** (`M`) takes the marked one to another point; **Tangent** (`T`) gives the direction of the curve there, for Fit; **Width** (`W`) gives the starting and ending width of the next segment; **eXit** (`X`) returns to the options.

The whole command is a single undo step. The button is on the expanded Modify panel (**Modify ▾**), as in AutoCAD.



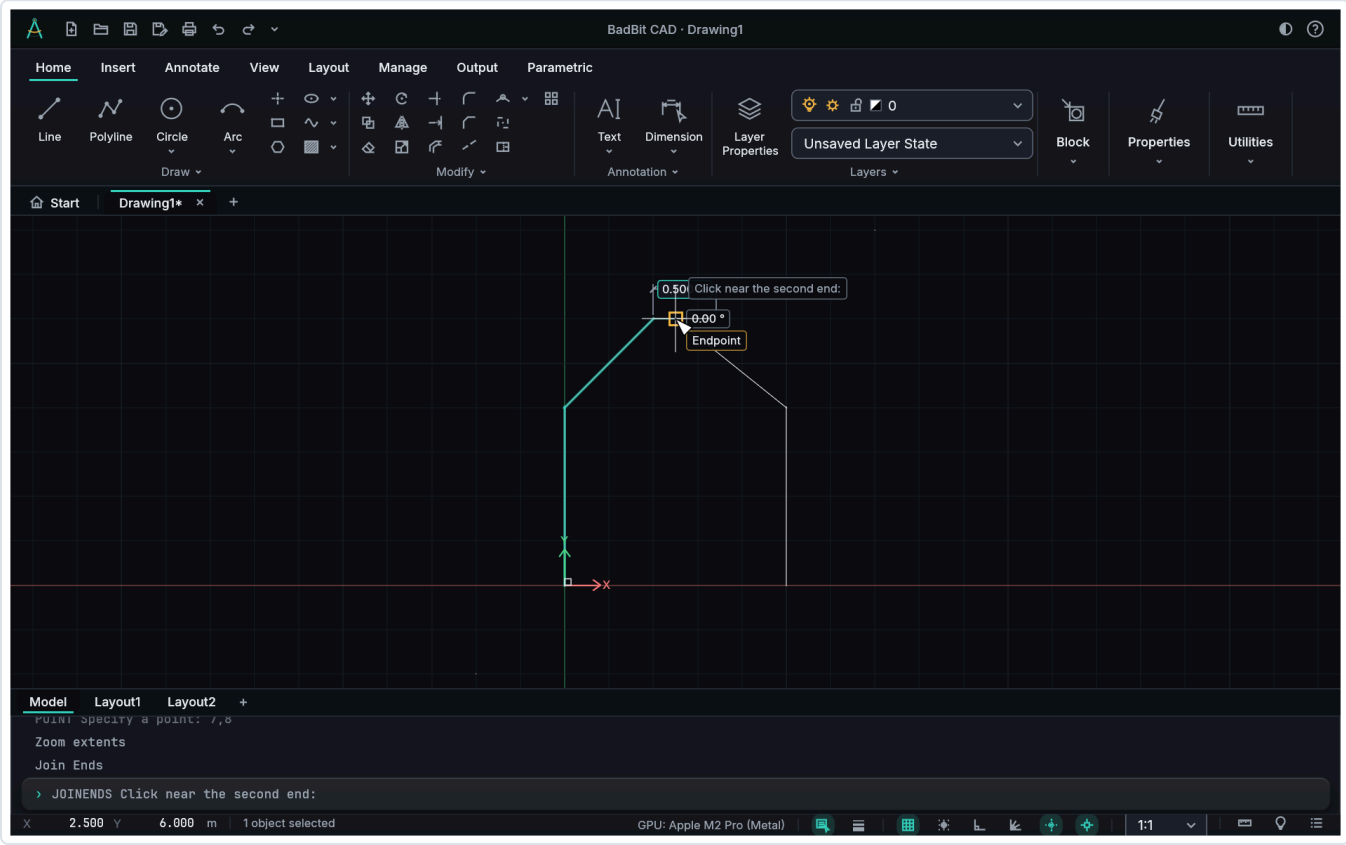
PEDIT: the polyline on the left picked, Join with a fuzz distance of 1 m and the one on the right being added; the 0.5 m gap gets a segment.



PEDIT Fit: the zigzag polyline becomes a smooth curve through the same vertices.

Join Ends (JOINENDS)

A BadBit CAD command that AutoCAD does not have: click near one end of a line or open polyline and near another end. The two get a segment between them and become a single polyline. If both ends belong to the same polyline, it closes. Object snaps apply during the command (the endpoint snap helps hit the end), and the preview shows the segment up to the cursor. It is under the small arrow of the **Join** button.



Join Ends: the end of the polyline on the left picked and the segment up to the one on the right.

EXPLODE (alias X)

Breaks a polyline into its segments (lines and arcs), an MTEXT into one TEXT per piece of each line (with its font, height and color; underlines become lines; a justified paragraph comes out without the stretched spaces), a block insertion into the block's objects, an associative array into the copies and a region into its edges, as in AutoCAD. A TEXT with a TrueType font becomes the outlines of the letters as closed polylines (like TXTEXP from Express Tools); one with a stroke font does not explode.

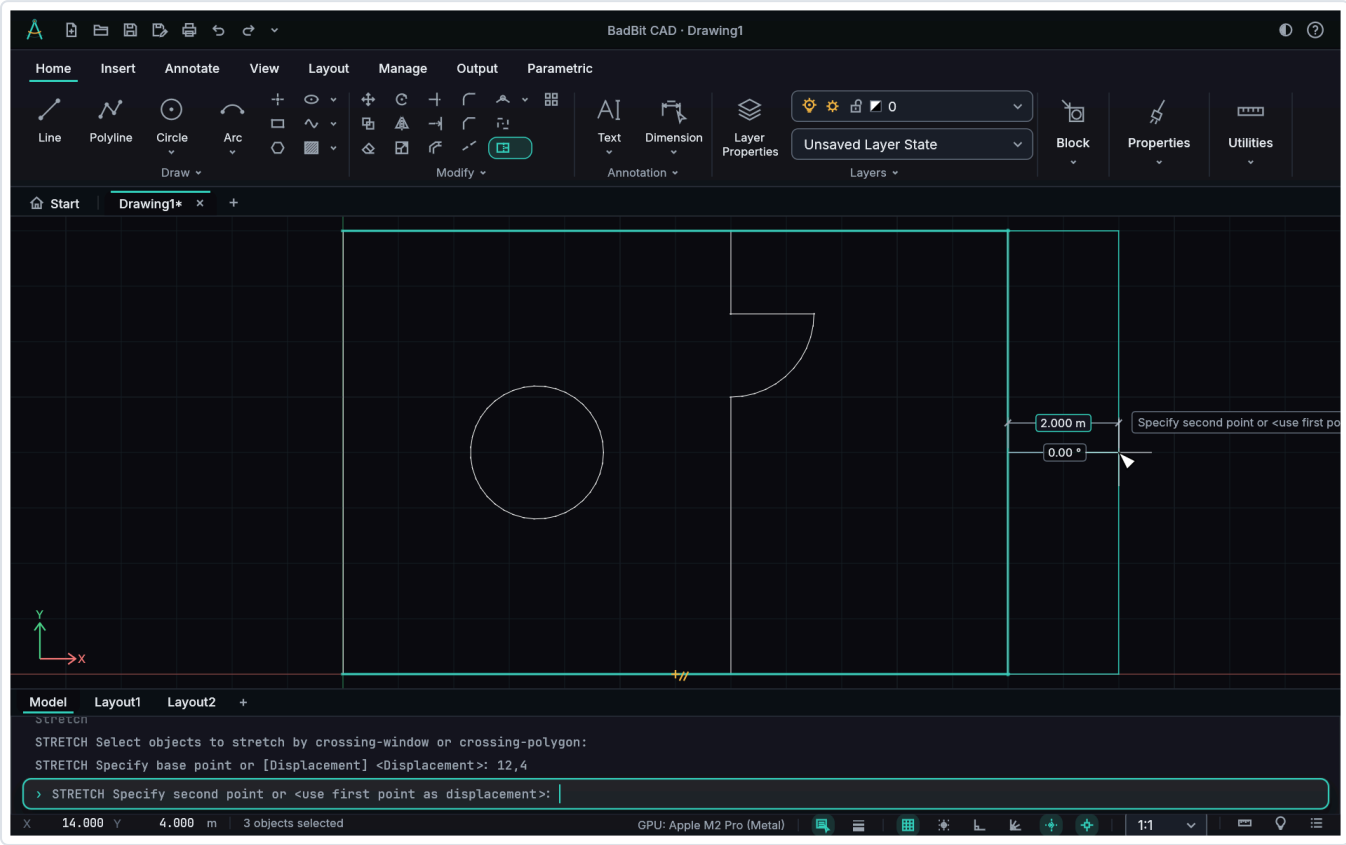
The three are in the Modify panel of the ribbon and each operation is a single undo step.

STRETCH (alias S)

Stretches objects by moving only part of them. Select with a **crossing window** (dragging from right to left) that catches the points that should move and confirm with `Enter`; the prompt reminds you of this, like the AutoCAD one. Clicking the objects is not enough: without a window, STRETCH warns and

asks for the window again (and a selection made before the command is set aside for the same reason); then enter the base point and the second point, as in MOVE (**Displacement** and the first point as displacement also apply). Before the click, the preview shows the result.

- Whatever is entirely inside the window moves as a whole.
- Of what crosses the window, only the points inside move: the ends of a line, the vertices of a polyline, the end of an arc (the arc passes through the new end, keeping the midpoint and the other end). A circle moves if its center is inside.



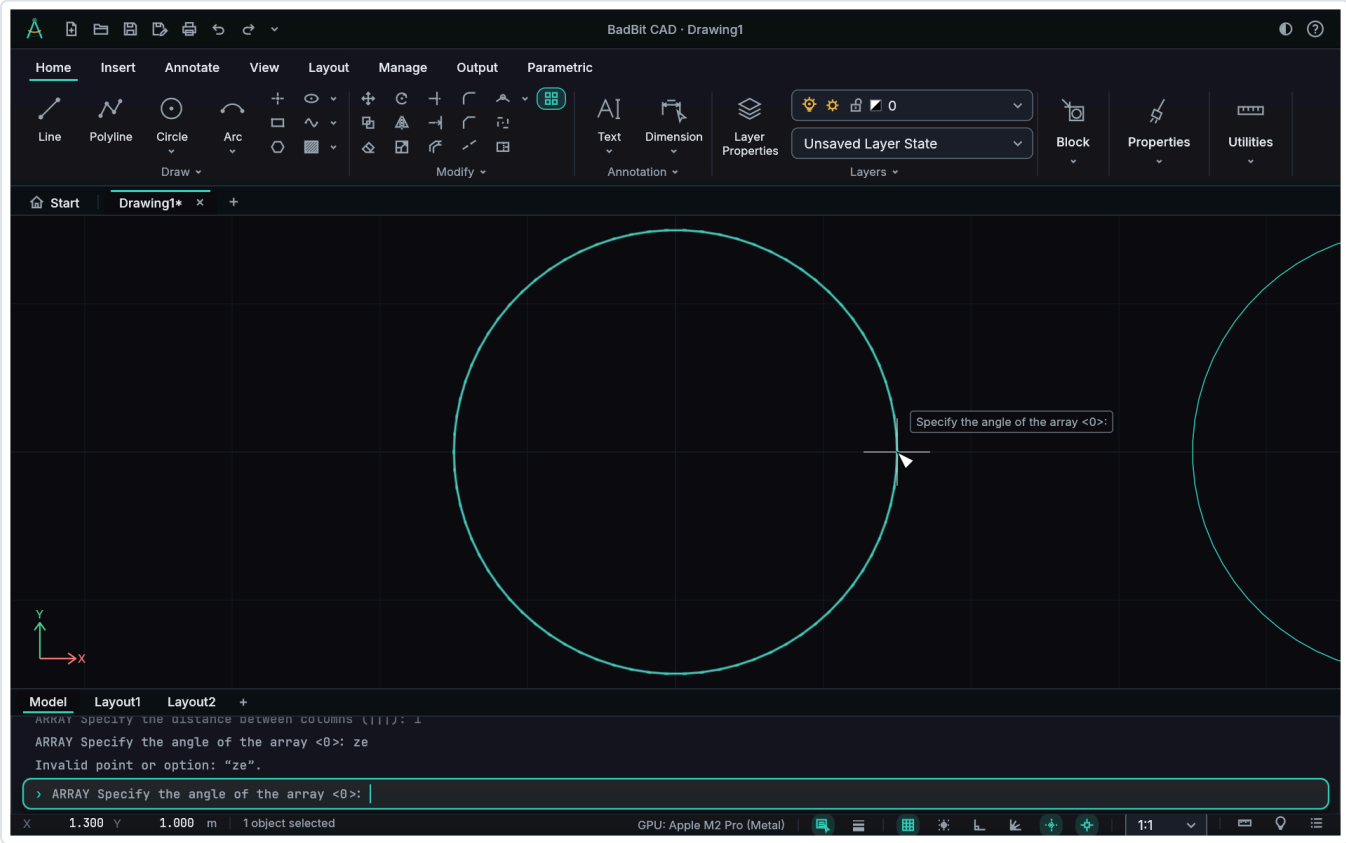
STRETCH: the window caught the right wall; it moves and the top and bottom walls stretch along.

ARRAY (alias AR)

Copies in a grid or around a center, like the command-line form of AutoCAD (-ARRAY). Select the objects and choose the type:

- **Rectangular** (`R`): number of rows, of columns, the distance between rows and between columns (negative for down or left) and the **angle of the grid** (`Enter`: 0; the preview rotates the grid with the cursor).
- **Polar** (`P`): the center (or first the **Base point** of the items, `B`), the number of items, the angle to fill (360 by default; positive is counterclockwise) and whether the items rotate along. `Enter` at the number of items asks for the **angle between items**, which decides how many fit in the angle to fill.
- **PAth** (`PA`): the curve (line, arc, circle, polyline, ellipse, spline), the number of items and whether they **align** to the curve (rotate with it). The original goes to the start of the curve and the copies are spread equally to the end (on a closed curve, all the way around it).

Before the last answer, the preview shows the copies. By default the array is **associative**: a single object, which **ARRAYEDIT** changes later (see *Blocks, associative arrays and regions*). **ASsociative** (`AS`), at the type prompt, switches to separate copies, which can be edited one by one. The whole array is a single undo step.



Rectangular ARRAY of 3 rows by 4 columns, 1 m apart.

LENGTHEN (alias LEN)

Changes the length of lines, arcs, open polylines, elliptical arcs and open splines at the end nearest the click. Clicking an object right at the start only **measures**: it shows the length (and the angle of an arc). The ways to change it:

Option	What it does
DElta (`DE`)	Adds a length (negative shortens) or, with **Angle** (`A`), an angle to the arc.
Percent (`P`)	The new length is the given percentage of the current one.
Total (`T`, or `Enter`)	The final length (or, with **Angle**, the final angle of the arc).
DYnamic (`DY`)	Drags the end: the preview shows the new object.

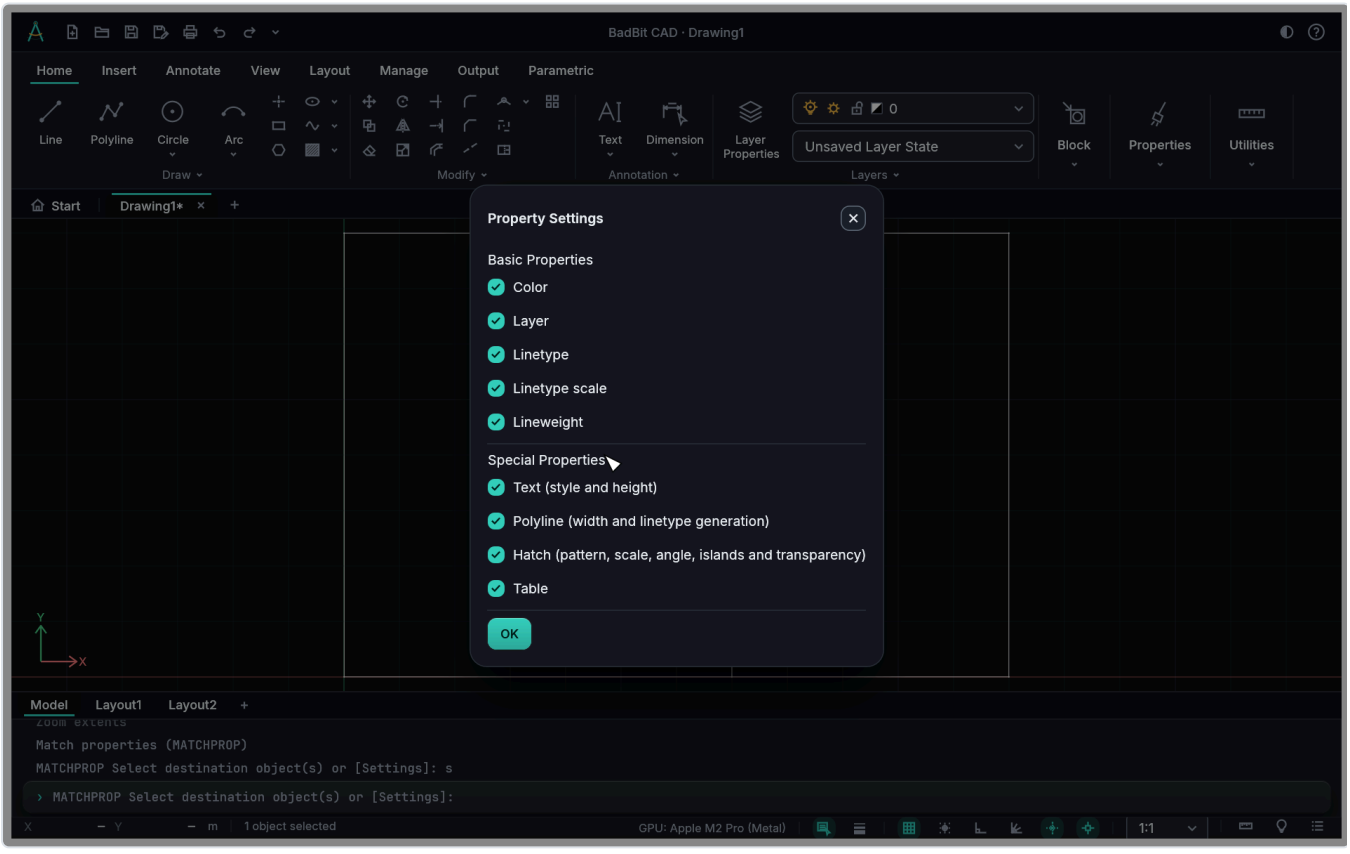
Then click the objects to change, one at a time, until `Enter`; **Undo** (`U`) reverts the last change. On a polyline it changes the first or last segment. On an elliptical arc, the end moves along the ellipse itself (up to almost the full turn; a whole ellipse does not change). On an open spline, shortening cuts the curve and lengthening continues the curve beyond the end, along the same span; the fit points are lost, as in AutoCAD. Angle only applies to arcs.

ALIGN (alias AL)

Aligns the selected objects by pairs of source and destination points. With **one pair** (`Enter` at the second source point), it only moves. With **two pairs**, it moves and rotates to the direction of the destination and asks whether to **scale** by the distance between the points (`Y` / `N`). The third AutoCAD pair is for 3D; here it can be given and changes nothing. It is all a single undo step.

MATCHPROP (aliases MA, PAINTER)

The paintbrush: passes the properties of a source object to the destination objects, picked by click or window, until `Enter` (with an object already selected, it is the source). **Settings** (`S`) opens the dialog that chooses what is passed: color, layer, linetype, linetype scale, lineweight, text (style and height) and polyline (width and linetype generation). Objects on a locked layer are left out, with a warning.



The MATCHPROP Settings.

OVERKILL

Cleans up the selected objects: removes **duplicates** (the same geometry, within the tolerance, with the same properties) and **combines overlaps** (collinear lines that overlap become one; arcs of the same circle too, and a whole circle swallows its arcs). Before running (`Enter`), the options:

Option	What it does
Ignore (`I`)	Properties that do not count in the comparison: `C` color, `LA` layer, `L` linetype, `LTS` linetype scale, `LW` lineweight, separated by commas; `A` all, `N` none.
tOlerance (`O`)	The distance within which two geometries are the same.
combine parTial overlap (`T`)	Combines those that partially overlap (on).
combine Endtoend (`E`)	Also combines those that only touch on the same straight line or the same circle (off).
Plines (`P`)	Optimizes the segments within each polyline (on): zero-length segments are removed; consecutive straight segments in the same direction and consecutive arcs of the same circle become a single segment. Then it asks whether to **ignore the width** of the segments (off: only segments of equal, constant width are combined).
Associativity (`A`)	Maintain associative objects (on). There are no associative objects in BadBit CAD yet; the choice is stored for associative dimensions and hatches.

At the end, the command line says how many duplicates and how many overlapping objects were removed and how many polyline segments were optimized. Ellipses, splines and texts are only removed when they are whole duplicates.

SPLINEDIT (alias SPE)

Edits a spline. The options:

Option	What it does
Close (`C`) or **Open** (`O`)	Closes the spline with tangency or opens a closed one.
Join (`J`)	Joins to the spline the clicked objects that touch its ends.
Fit data (`F`)	On a spline made of fit points: **Add** (`A`, a new point after the chosen one), **Close/Open**, **Delete** (`D`), **Kink** (`K`, a new fit point at the clicked point of the curve, where it can make a sharp corner; the corner remains in later edits), **Move** (`M`), **Purge** (`P`, the spline becomes control vertices only), **Tangents** (`T`, the start one and the end one; `S` returns to the system default) and **toLerance** (`L`).
convert to Polyline (`P`)	Replaces the spline with a polyline, with a **precision** from 0 to 99 (higher, more vertices).
Edit vertex (`E`)	On the control vertices: **Add** (`A`, a vertex at a point on the curve, without changing the shape), **Delete** (`D`), **Elevate order** (`E`, the new order, from degree + 1 up to 26: the curve does not change and gains vertices; only on an open spline), **Move** (`M`) and **Weight** (`W`). Moving the vertices removes the fit data.
Reverse (`R`)	Reverses the direction of the spline.
Undo (`U`)	Undoes the last change made in this SPLINEDIT.

Points are picked with a click (the nearest one is used). Each change is an undo step.

ERASE (alias E)

Erases the selected objects. `Delete` does the same without a command.

Objects on locked layers are ignored in edits, with a warning, as in AutoCAD.

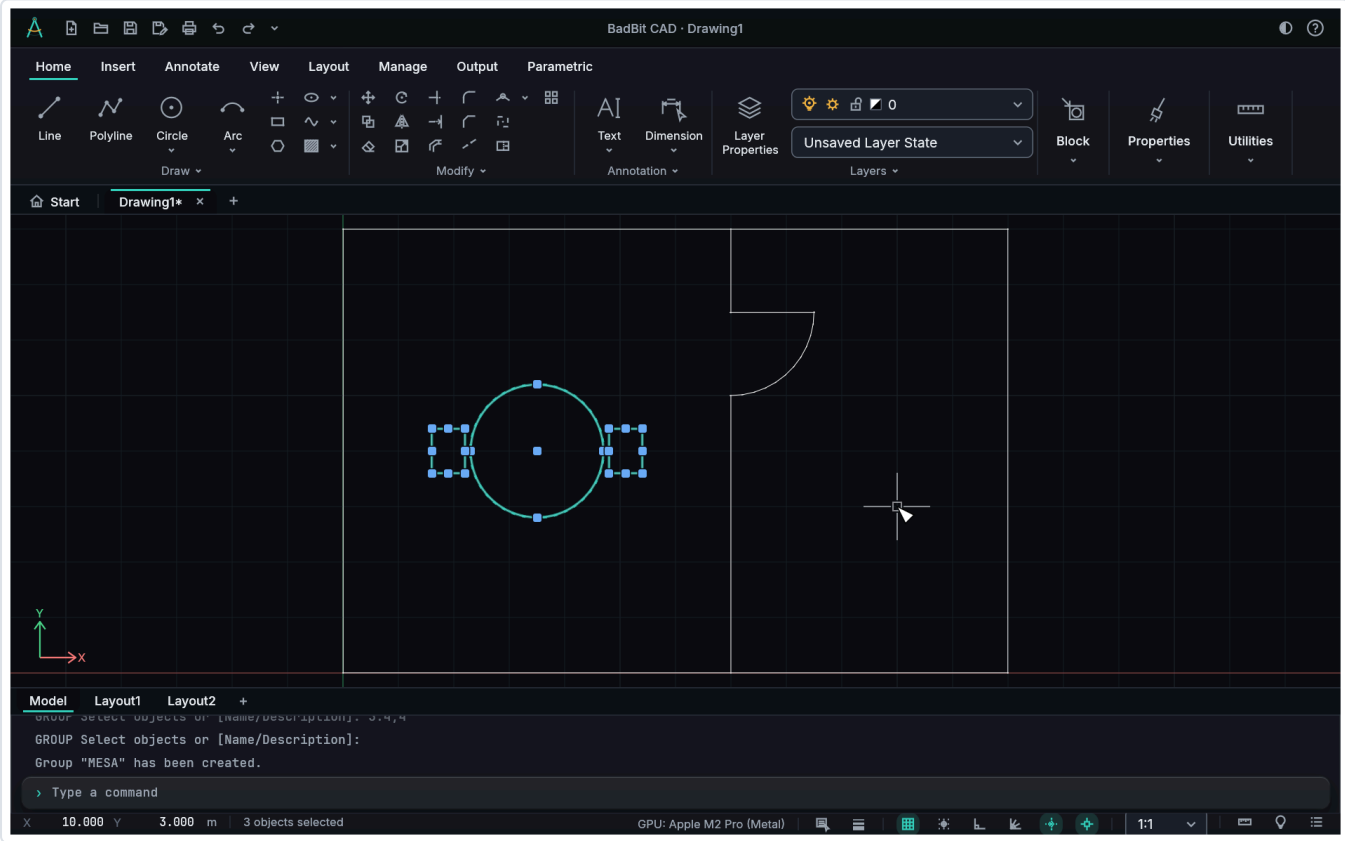
Groups (GROUP, alias G; UNGROUP; GROUPEDIT)

A **group** gathers objects under a name, as in AutoCAD: with group selection on (PICKSTYLE 1, the default), a click, a window or any other way of picking a member picks the whole group, with no command and inside commands (ERASE, MOVE, COPY...). `Ctrl` + `Shift` + `A` turns group selection on and off (the command line says "<Groups on>" or "<Groups off>"); with it off, each object counts on its own. The buttons are on the open **Modify** panel (the small arrow below the title): Group, Ungroup, Group Edit and the group selection checkbox.

- **GROUP** (`G`): "Select objects or [Name/Description]". Pick the objects and `Enter`. **Name** gives the group a name (`?` lists the groups in the drawing); with the name of a group that already exists, it asks whether to replace it (Yes replaces the old group with the new one). **Description** stores a text for the group. Without a name, the group is **unnamed** (`*A1`, `*A2`...). With objects already selected before the command, GROUP creates the unnamed group right away.

- **UNGROUP**: "Select group or [Name]": a click on a member dissolves its group (the objects stay in the drawing); **Name** by name; `ALL` dissolves all of them. With the object in more than one group, it asks "[Accept/Next]": the current group appears selected and Next moves on to the following one. With objects already selected, it dissolves their groups.
- **GROUPEDIT**: the group (click a member or **Name**) and "[Add objects/Remove objects/REName]". In Add and Remove, each picked object counts on its own (group selection does not act); removing all members deletes the group. **REName** shows the current name as the default and also gives a name to an unnamed group.

Erasing a member (by any command) removes it from the group; a group left with no members disappears. **COPY** of a whole group creates a new unnamed group with the copies (the description goes along). Everything comes back with undo, one step per command. Groups go into the `.bbd` file and into **DXF and DWG** (the GROUP of the `ACAD_GROUP` dictionary: name, description, selectable and the order of the members), and groups from an AutoCAD drawing open as groups.



The MESA group (the table and the two chairs) created with Name and Description; a click on the table selects the whole group.

Inquiry

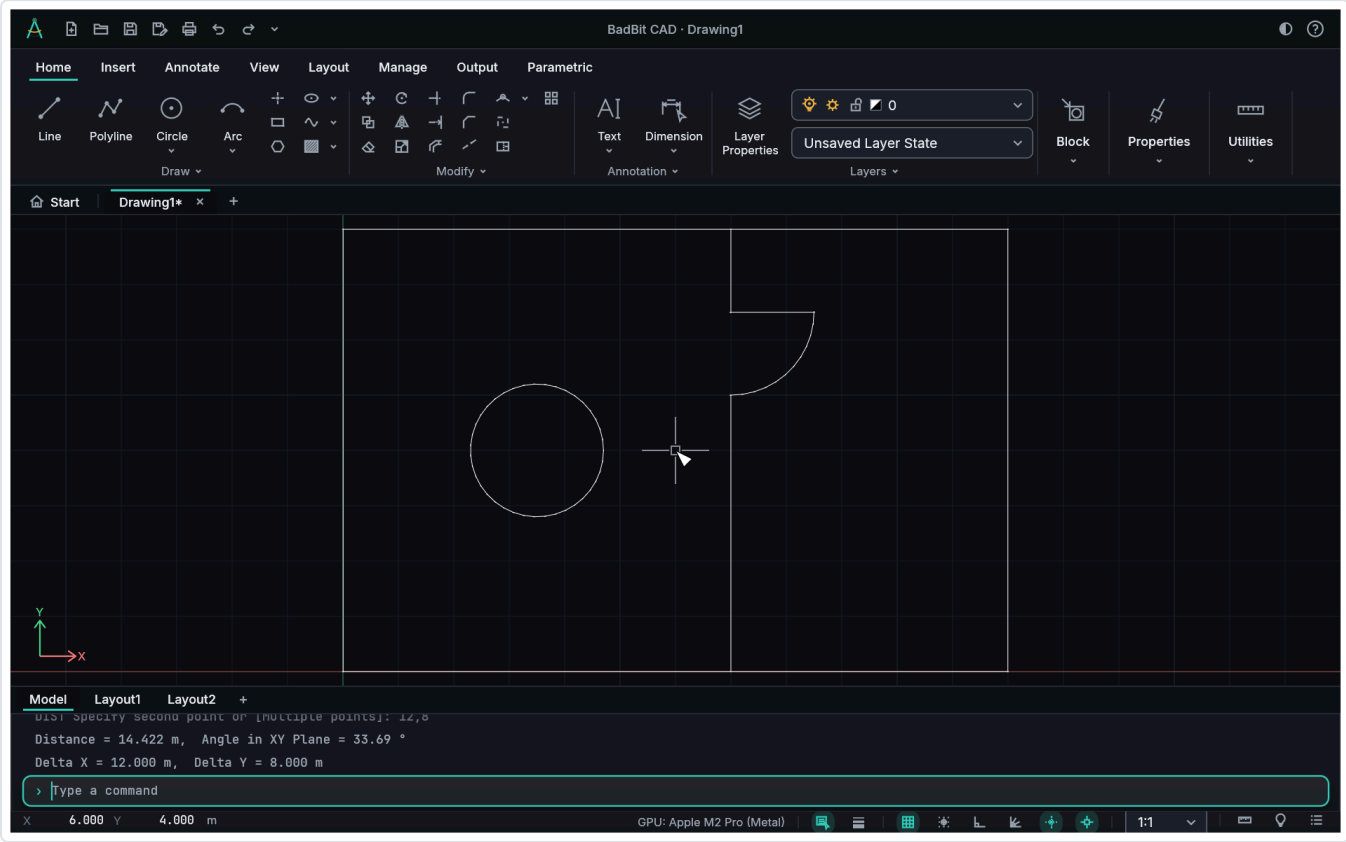
Commands that measure and describe the drawing without changing anything. The results appear on the command line, in the units set in the preferences.

ID

Shows the coordinates of a point (X, Y and Z, which in 2D is always 0). Object snaps apply: `ID` and a click with the Endpoint snap gives the exact coordinates of the end.

DIST (alias DI)

Measures from one point to another: the **distance**, the **angle in the XY plane** and the **deltas** in X and Y. With **Multiple points** (`M`) after the first point, it adds up the segments until `Enter` (or **Total**, `T`), showing the running sum at each point; **Undo** (`U`) removes the last one.



DIST along the diagonal of the floor plan: the distance, the angle and the deltas.

AREA (alias AA)

Measures the **area** and the **perimeter**:

- **By points**: the corners, until `Enter` (with three or more); **Undo** (`U`) removes the last one.

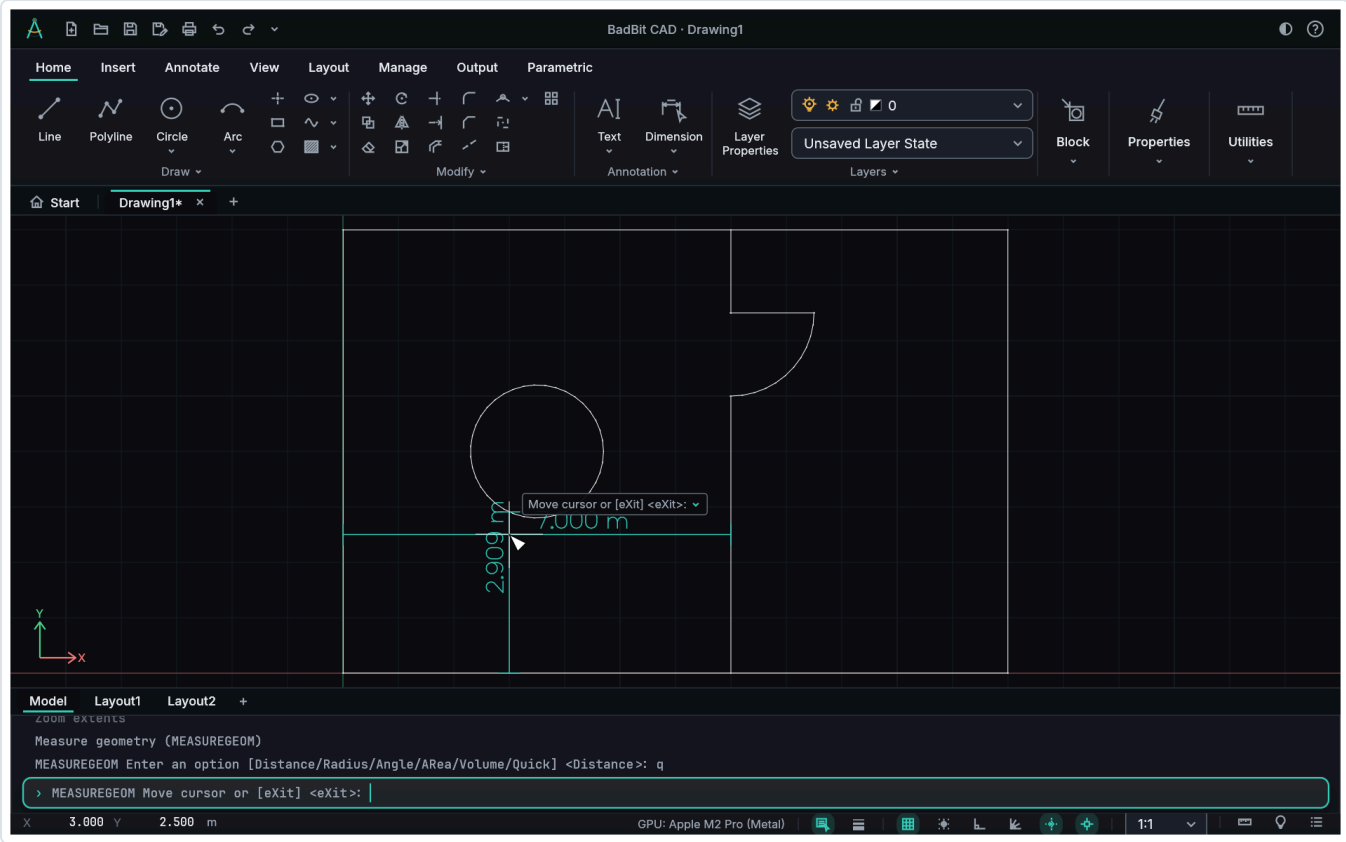
- **Object** (`O`, or `Enter` at the first prompt): a circle, ellipse, polyline, spline, arc or line. On an open object, the area is that of the figure closed by a straight line from one end to the other, and the length is reported instead of the perimeter.
- **Add area** (`A`) and **Subtract area** (`S`): each measured area is added to or subtracted from the **total**, which is shown after each measurement. In Object mode, it keeps asking for objects; **eXit** (`X`) or `Enter` ends with the total.

MEASUREGEOM (alias MEA)

Brings the measurements together and repeats until **eXit** (`X`):

Option	What it measures
Distance (`D`)	Like DIST.
Radius (`R`)	The radius and diameter of an arc, circle or arc segment of a polyline.
Angle (`A`)	Of an arc (the included angle), between two lines (on the side where they were clicked), on a circle (between two points on it) or, with `Enter`, by a vertex and two endpoints.
ARea (`AR`)	Like AREA.
Volume (`V`)	The area of the base (by points or from a closed object, with **Object** `O`) times the **height** requested. **Add volume** (`A`) and **Subtract volume** (`S`) accumulate a total, as in AREA; **eXit** (`X`) ends with it.
Quick (`Q`)	With the cursor resting in one place, the measurements to the nearest objects appear in the drawing: the horizontal and vertical distances between them (or from the cursor to the only one found). A click writes the measurements to the command line and, inside a closed region, its area and perimeter. `Enter` returns to the options.

The Volume is that of a right prism (2D CAD: the base on the plane and the given height).



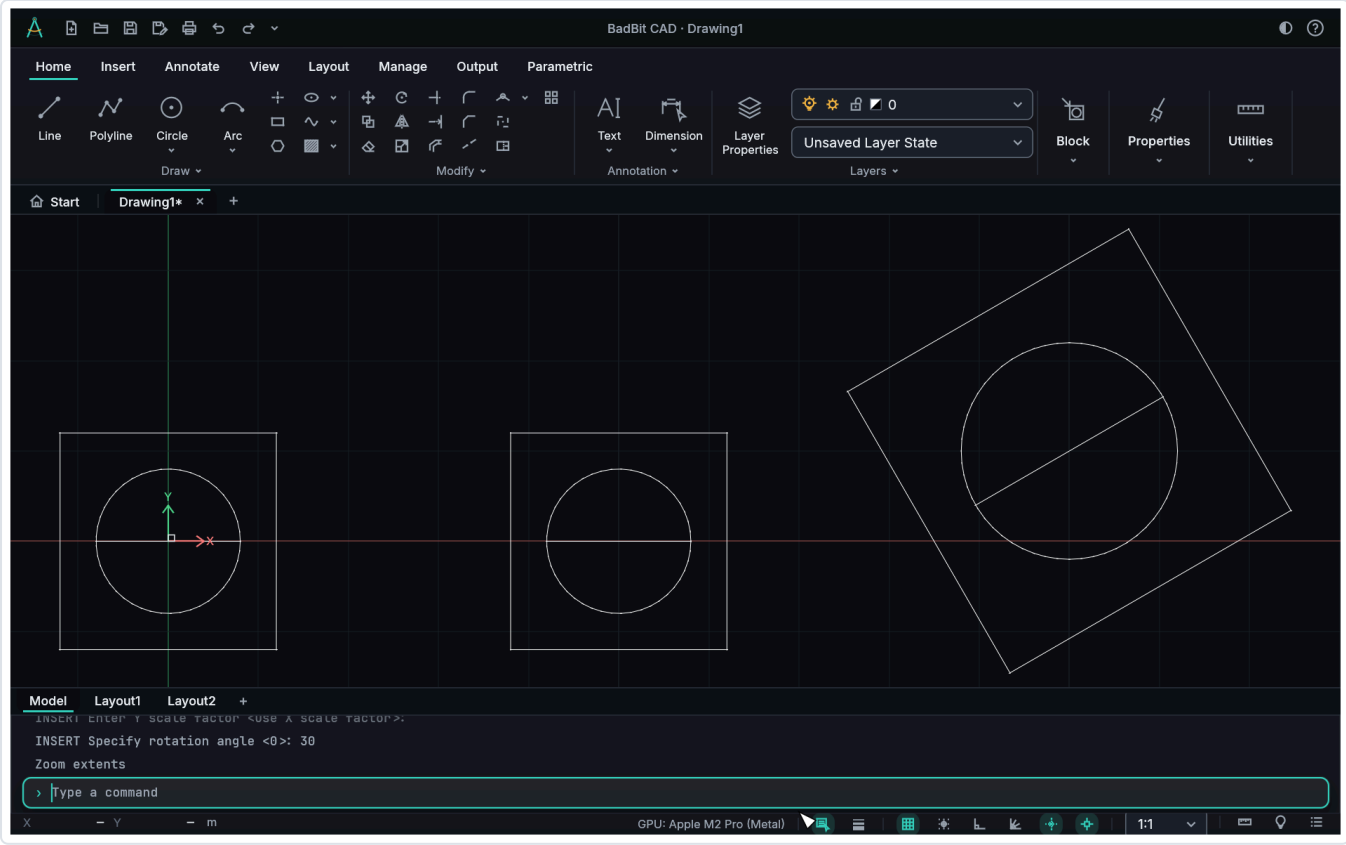
Quick inside the floor plan: the width and height of the room around the cursor.

LIST (aliases LI, LS)

Describes the selected objects: the type (the DXF name: LINE, LWPOLYLINE, CIRCLE…), the layer, the object's own color, linetype and lineweight (or ByLayer), and the geometry: endpoints, length and angle of a line; center, radius, circumference and area of a circle; angles and length of an arc; vertices, bulges, width, area and perimeter of a polyline; axes of an ellipse; order and points of a spline; style, height, rotation and contents of texts.

Blocks, associative arrays and regions

A **block** is a named group of objects, stored in the drawing, that you can insert as many times as you like: doors, tables, symbols. Each **reference** is a single object (selecting, moving, rotating and scaling take all of it), and they all change together when the block is redefined. The buttons are in the **Block** panel of the Home tab (Insert, Create, Edit, Edit Attributes and Define Attributes; expanded, Write Block, Manage Attributes and Synchronize Attributes) and on the **Insert** tab (Block, Block Definition and Reference panels).



The MESA block inserted three times, one of them rotated and larger.

BLOCK (alias B)

Creates a block from the chosen objects:

1. The block **name** (`?` lists the blocks in the drawing). If the name already exists, you are asked whether to **redefine** it (`Y` / `N`).
2. The **base point**: the point of the block that sits on the insertion point.
3. The **objects** (if objects are already selected, they are used).

The objects become a reference of the block in the same place (the *Convert to block* option of the AutoCAD dialog box). When you redefine a block, all its references start showing the new objects. Everything is a single undo step.

The block's items keep their own layer and properties, and follow the AutoCAD rule for color, lineweight and linetype: **ByBlock** takes the reference's value; **ByLayer** follows the item's layer, and items on layer **0** take the reference's layer (its color, lineweight and linetype); an explicit value stays as it is. In a block nested inside another block the rule repeats: the inner reference plays the role of the reference for its own items. Items on other layers disappear when their layer is turned off or frozen. The screen and plotting follow the same rule.

-BLOCK (alias -B)

Like BLOCK, but the objects **are removed from the drawing** without leaving a reference, like AutoCAD's -BLOCK. **OOPS** brings them back.

INSERT (aliases I, -I, -INSERT)

Inserts a block from the drawing:

1. The **name** (`Enter` repeats the last inserted block; `?` lists). A **file** also works: see INSERT from a file.
2. The **insertion point**, with a preview of the block at the cursor. Before the point, **Scale** (`S`) and **Rotate** (`R`) apply to both axes right away and are not asked again later.
3. The **X scale factor** (`Enter`: 1), the **Y scale factor** (`Enter`: the same as X; a negative value mirrors) and the **rotation angle** (`Enter`: 0; the preview rotates with the cursor).

With different X and Y scales, the block's circles and arcs appear as ellipses, as in AutoCAD.

The **Insertion** object snap (INS) snaps to the insertion point. The Properties palette shows the block name and lets you change the point, the scales and the rotation; LIST shows the same. A reference read from a DXF or DWG with a **Z scale**, an **elevation** (the Z of the insertion point) or a **tilted normal** keeps that data: LIST shows the Z scale and the elevation of the base point (and the normal, when it is not the top one), and saving writes it all back to the file, also after moving, rotating, scaling or mirroring the reference. On a tilted reference, changing the X and Y scales or the rotation in the Properties palette applies in the **reference's coordinate system**, as in AutoCAD: the normal (the tilt), the Z scale and the Z stay; the rotation is that of the block's X axis seen in plan. Changing the point moves the reference along the drawing's X and Y, without changing Z. The same applies to a reference that contains another one with a solid: the Z, the Z scale and the normal of the outer one apply to the inner one's solid.

In plan view, a tilted reference looks as it does in AutoCAD: the block's lines, arcs, circles, texts and hatches are the **projection** of the block onto the plane of the reference, foreshortened by the tilt (a circle becomes an ellipse, an arc becomes an elliptical arc, text is distorted). Selection, object snaps (Endpoint, Center, Insertion...) and measurements use this projection; the extents (ZOOM Extents and the saved `EXTMIN` / `EXTMAX`) are AutoCAD's: the box of each object in the block, carried by the reference.

Exploding a reference

EXPLODE breaks the reference into the block's objects, in place, each on its own layer and with ByBlock replaced by the reference's properties. In a reference with different scales, circles and arcs become ellipses and the curved segments of polylines become fine straight segments.

The **solid**, **region** and **surface** of a block do not come out of a reference with different scales (also when only the Z scale is different): the 3D body does not follow the distortion. As in AutoCAD, a reference that contains only these objects stays as it is, with the message **Cannot explode that block.**; in one that has other objects, the others come out and these stay together in a reference of a new **anonymous block** (`*E1`, `*E2`...), in the same place and with the same transformation, with the message **1 was not able to be exploded.** (or **N were not able to be exploded.**). A nested reference with a solid that is rotated inside the non-uniform scale also stays in the anonymous block (rotation and non-uniform scale together do not fit in one reference); without the rotation, or with the non-uniform scale only in Z, it comes out as a reference. With the same scale on all three axes, mirrored or with a tilted normal, the solid comes out on its own, with the reference's dimensions. BURST does the same.

BURST (Express Tools)

BURST (from the old Express Tools, as in AutoCAD) explodes the chosen references like EXPLODE, but **the attributes become texts with their values** (EXPLODE gives back the definitions, with the tag). Choose the references and confirm with `Enter` (or choose them first and type BURST); the other objects in the selection stay as they are.

- Each inner object keeps **its own layer, color and linetype**. What is on **layer 0** goes to the reference's layer, and **ByBlock** color and linetype take the reference's.
- **Invisible** attributes **disappear**; visible constant attributes become text with the definition's value; multiline ones become MTEXT.
- **Nested references** (nested blocks) remain references, in place.
- External references are not exploded. Everything is undone with a single **U**.

BURST is in the expanded **Block** panel of the Home tab.

Associative array

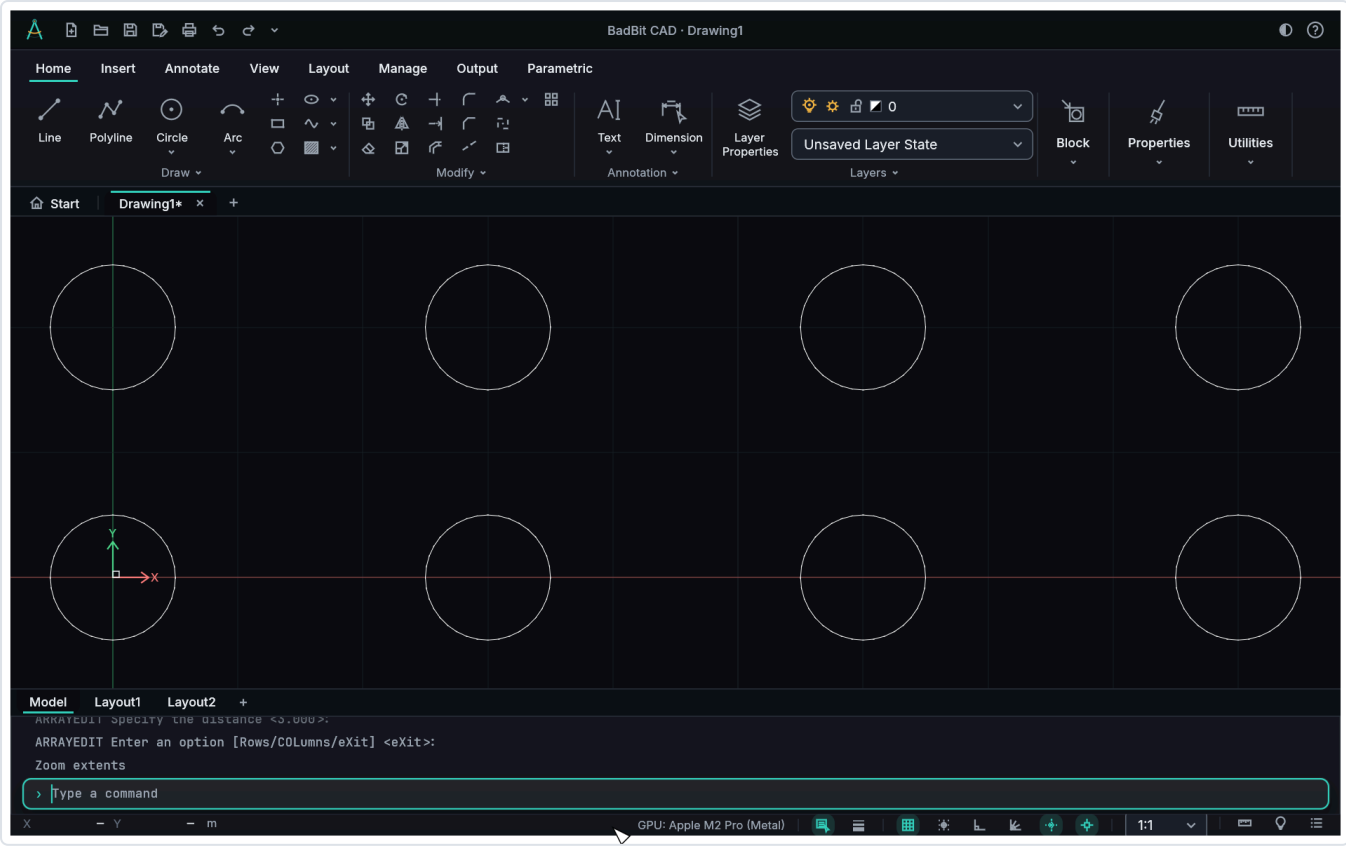
ARRAY creates, by default, an **associative** array, like AutoCAD (`ARRAYASSOCIATIVITY` 1): a single object, with the items computed from the parameters. At the command's first prompt, **ASsociative** (`AS`) switches it to `N`, and the array is once again made of separate copies. The variable applies to the next ones.

ARRAYEDIT

Changes an associative array after it has been created. Click the array and choose:

Type	Options
Rectangular	**Rows** (`R`) and **COLumns** (`COL`): the count and then the distance (`Enter` keeps it).
Polar	**Items** (`I`), **Fill angle** (`F`) and **ROTate items** (`ROT` , `Y` / `N`).
Path	**Items** (`I`) and **Align items** (`A` , `Y` / `N`).

Each change appears immediately; **eXit** (`X`) or `Enter` finishes, and the whole ARRAYEDIT is a single undo step. EXPLODE breaks the array into its copies.



A 2 by 3 associative array that ARRAYEDIT changed to 4 columns.

REGION (alias REG)

Turns closed objects into **regions**: circles, closed polylines, full ellipses, closed splines and the chains of lines, arcs and open polylines that close a boundary. The source objects are removed (with `DELOBJ` 1, the default). The command line says how many regions were created.

A region is displayed by its boundary and counts as an area: **LIST** and **AREA** (Object option) give its area and perimeter, and the Properties palette shows both. **BOUNDARY** also creates regions (advanced options, **Object type** `O` , **Region** `R`): the one of the outer boundary, with the islands as holes. EXPLODE gives back the boundaries as lines and arcs. In regions, ellipses and splines are displayed as fine straight segments, but the area, the perimeter and MASSPROP come from the curve itself, without the chord error (the same applies to the hatch of an ellipse or spline, to hatches and regions made by picking a point inside an area bounded by ellipses or splines, and to HATCH read from DXF and DWG). Also free of rounding error are the boundaries made of lines, arcs and circles whose corners are intersections (a rectangle of four crossing lines, the lens of two circles, a circle cut

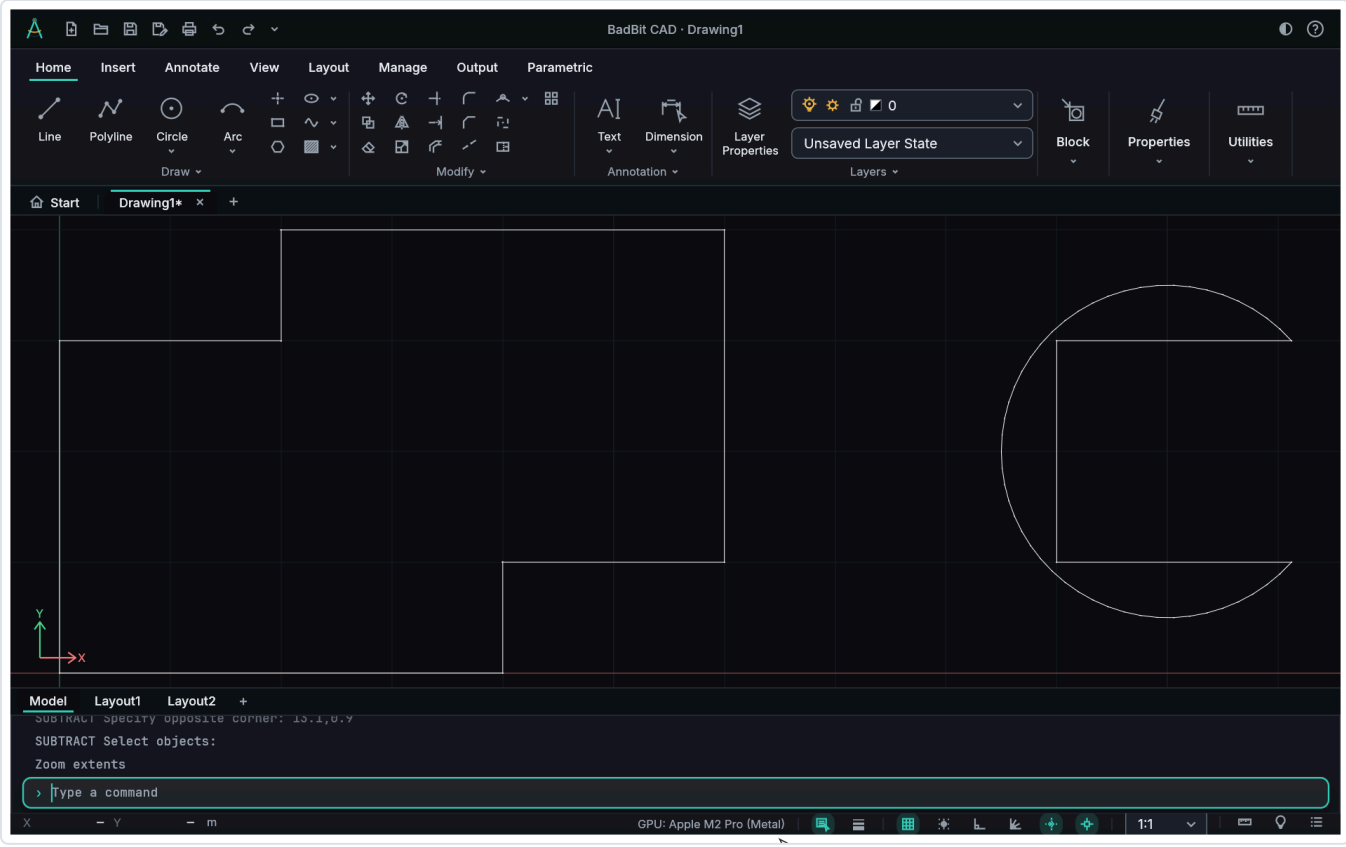
by a line): in UTM coordinates the intersection point stored in a polyline would be rounded to half a billionth of a meter, and the measurement comes from the lines and arcs themselves, which makes a difference in a figure measured in centimeters. Very shallow polyline arcs (bulge of 0.00001 or less) also measure correctly in AREA, LIST, the palette and FIELD. Saved to DXF or DWG, the hatch carries its boundary with the ellipse and spline edges as they are, and AutoCAD measures the same area.

UNION, SUBTRACT and INTERSECT

The operations between regions, as in AutoCAD. The boundaries remain exact: lines are lines and arcs are arcs (the lens of two circles comes out with two arcs), and ellipses and splines are still measured along the curve, cut at the true intersection points. Pieces that end up separate become separate regions, and a region lying entirely inside another in a subtraction becomes a **hole**. Each operation is a single undo step; the chosen regions are removed and the result takes the layer and properties of the first one. When the result would run between two **nearly coincident** curves (closer than the drawing's tolerance along a stretch, such as two equal circles offset by a billionth of their size), the operation is not performed: the boundary there is not defined and the measurement would be wrong. The message says so, and the chosen regions stay as they were.

Command	Alias	What it does
UNION	`UNI`	Joins the chosen regions (two or more): the area of one or the other.
SUBTRACT	`SU`	First the regions to subtract **from** (`Enter`), then the ones that are **removed** (`Enter`).
INTERSECT	`IN`	Keeps only the area common to all of them. With no common area, the regions are removed and nothing is left (AutoCAD's "null region").

Objects that are not regions are left out, with a message. The buttons are in the expanded **Draw** panel, next to Region.



On the left, the UNION of two overlapping rectangles; on the right, the SUBTRACT of a rectangle from inside a circle.

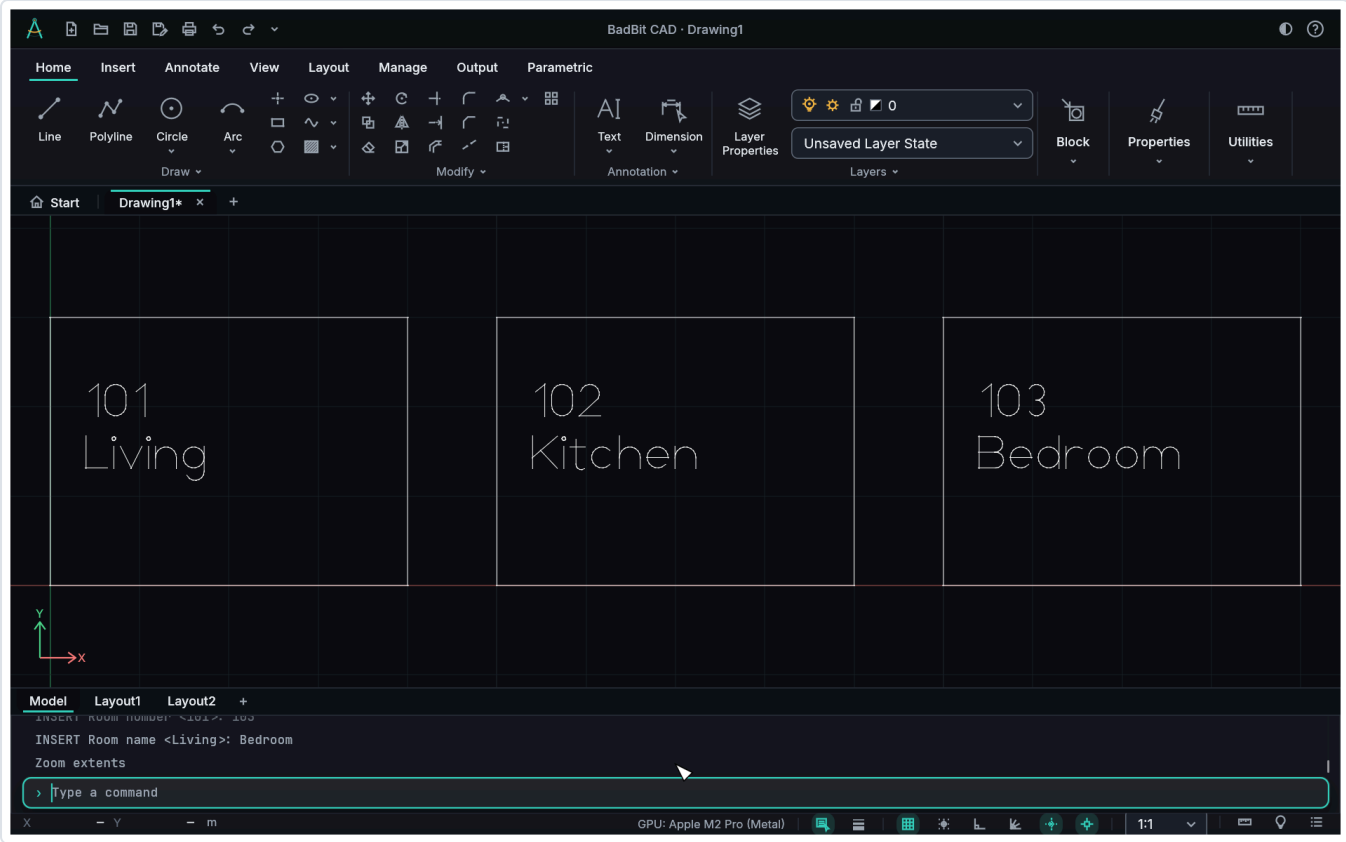
MASSPROP

Shows the **mass properties** of each chosen region, like AutoCAD's MASSPROP in 2D: area, perimeter, bounding box, centroid, moments of inertia about the X and Y axes, product of inertia, radii of gyration and the principal moments about the centroid (with their directions). The calculations use the exact boundaries (arcs are taken as arcs). It is in the expanded **Utilities** panel.

Attributes

An **attribute** is a text that each reference of the block fills in: the room number and name, the part code, the drafter in the title block. The **attribute definition** lives in the block, with the **tag** (no spaces; converted to uppercase), the **prompt** that INSERT asks, the **default value** and the **modes**:

Mode	What it does
Invisible (`I`)	Not displayed with ATTDISP Normal.
Constant (`C`)	The value is the definition's, the same in every reference; it is neither asked nor edited.
Verify (`V`)	INSERT asks for the value twice.
Preset (`P`)	INSERT uses the default without asking.
Lock position (`L`)	The attribute does not move separately from the reference (the default).
Multiple lines (`M`)	The value has several lines; on the command line, `\P` breaks the line.

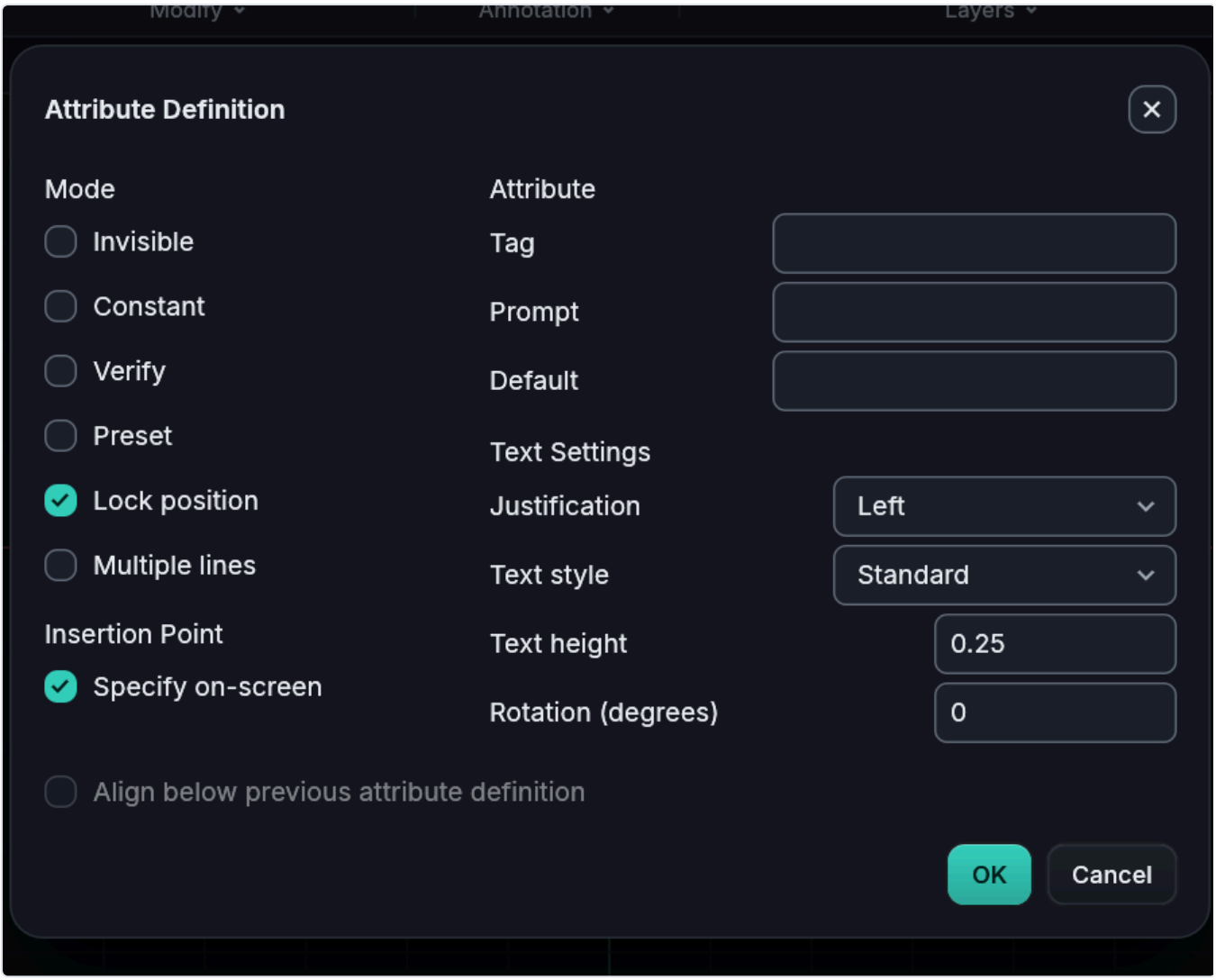


The SALA block inserted three times: each room's number and name are attributes.

ATTDEF (alias ATT) and -ATTDEF (alias -ATT)

ATTDEF opens the **Attribute Definition** dialog box: the modes, the tag, the prompt, the default, the text settings (justification, style, height, rotation and the width for multiple lines) and the insertion point (on screen, after OK, or typed in). **Align below previous attribute definition** places the new one right below the last one, with the same text settings.

-ATTDEF asks everything on the command line, as in AutoCAD: the current modes and the option to change (`Enter` finishes), the tag, the prompt, the default, the **start point** (with **Justify** `J` and **Style** `S`; `Enter` places it below the previous one), the height and the rotation.



The Attribute Definition dialog box.

On its own in the drawing, the definition shows its tag. Choose the definitions along with the objects in **BLOCK**: INSERT then asks for the values, in the order in which the definitions were created (BATTMAN changes the order). A BLOCK that converts the objects leaves the reference with the default values.

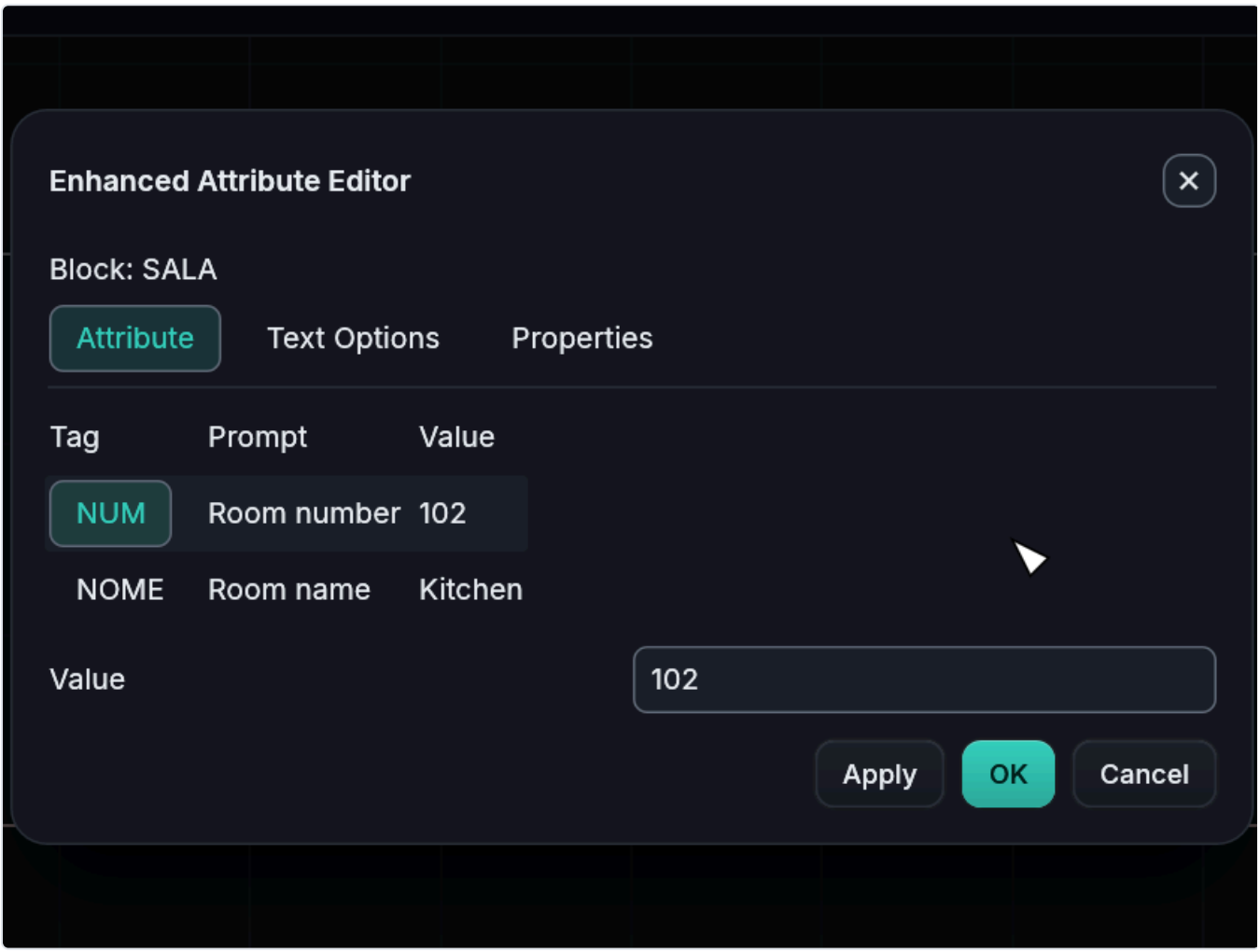
The values in INSERT (ATTREQ and ATTDIA)

After the rotation, INSERT says "Enter attribute values" and asks each definition's prompt, with the default between `< >` (`Enter` accepts it). With **ATTDIA** 1, the values come in a dialog box, as in AutoCAD; with **ATTREQ** 0, references get the defaults without asking.

Editing attributes

- **EATTEDIT** (or **double-clicking** the reference) opens the **Enhanced Attribute Editor**: the list of attributes and the **Attribute** (the value), **Text Options** (style, height, rotation, width factor and oblique angle) and **Properties** (layer and color) tabs, with **Apply**.
- **ATTEDIT** (aliases `ATE`, `DDATTE`) opens the **Edit Attributes** dialog box, with only the prompts and the values.

- **-ATTEDIT** (aliases `-ATE`, `ATTE`) works from the command line: **one at a time** (the block name, tag and value specifications with `*` and `?`, the chosen attributes and, for each one, **Value**, **Position**, **Height**, **Angle**, **Style**, **Layer**, **Color** or **Next**, with the current one marked by an X) or **global** (only those visible on screen or all of them, and the text to replace with the new one in all values). The whole command is one undo step.
- The **Properties** palette of a reference has the **Attributes** section with the editable values; for an attribute definition, the tag, the prompt and the default.



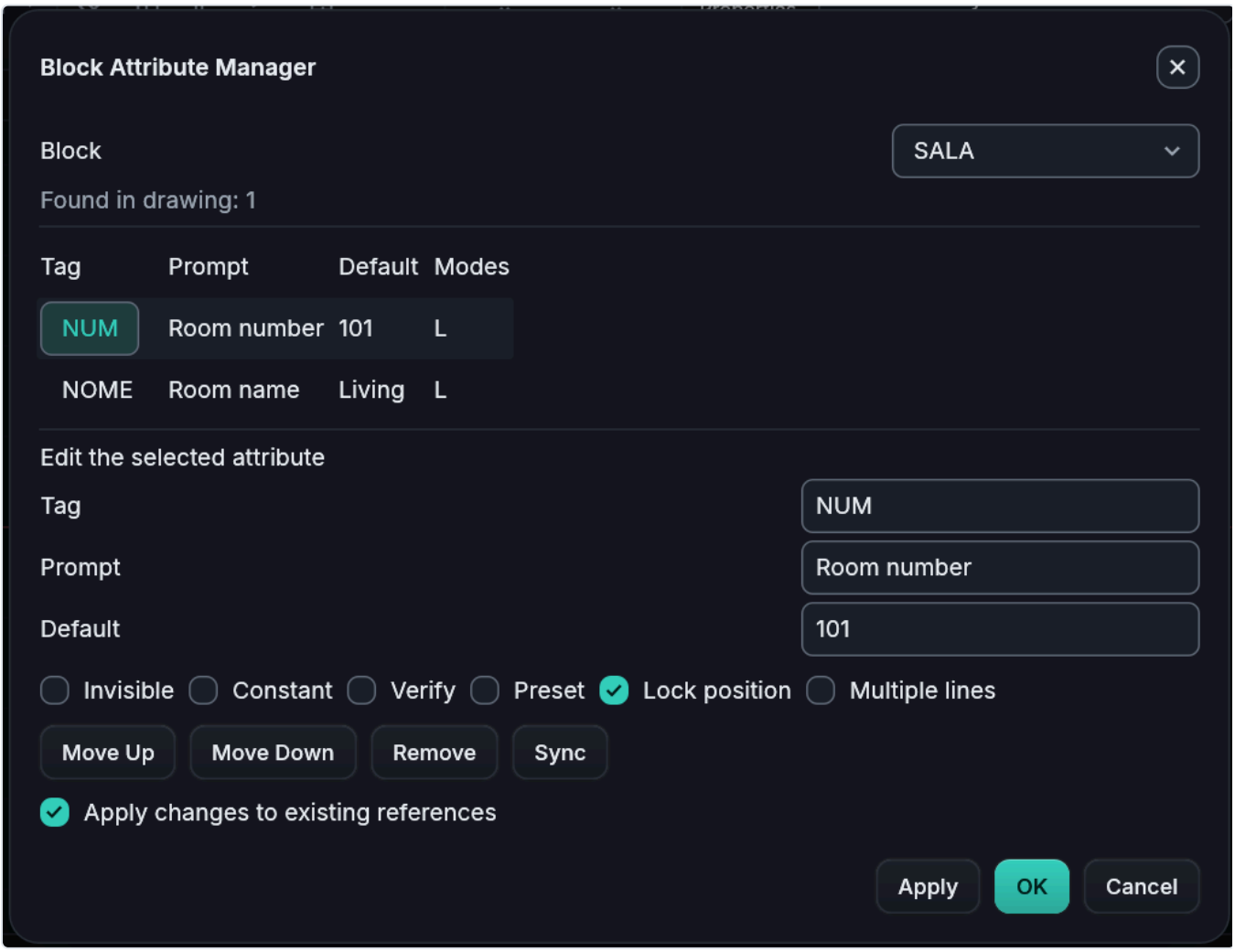
The Enhanced Attribute Editor for room 103.

LIST shows the reference's attributes, **FIND** also searches and replaces in attribute values, and **Quick select** has the Attribute Definition type. **EXPLODE** on a reference gives back the **definitions** (with the tag), and the values are lost, as in AutoCAD.

ATTDISP, ATTSYNC and BATTMAN

- **ATTDISP**: **Normal** (`N`, honors the Invisible mode), **ON** (`ON`, shows all) or **OFF** (`OFF`, hides all). It is the **ATTMODE** variable (0, 1 or 2).
- **ATTSYNC**: brings the current definitions to a block's references (by **Name** `N` or by **Select**ing `S` a reference; `?` lists the blocks with attributes). New tags come in with the default, removed ones disappear, the values stay.

- **BATTMAN**: the **Block Attribute Manager**. Choose the block; the list shows the tag, the prompt, the default and the modes of each definition. Change the fields and the modes, **Move Up/Move Down** (the order of the prompts), **Remove**, **Sync**; with **Apply changes to existing references**, Apply and OK already synchronize.



The Block Attribute Manager with the SALA block.

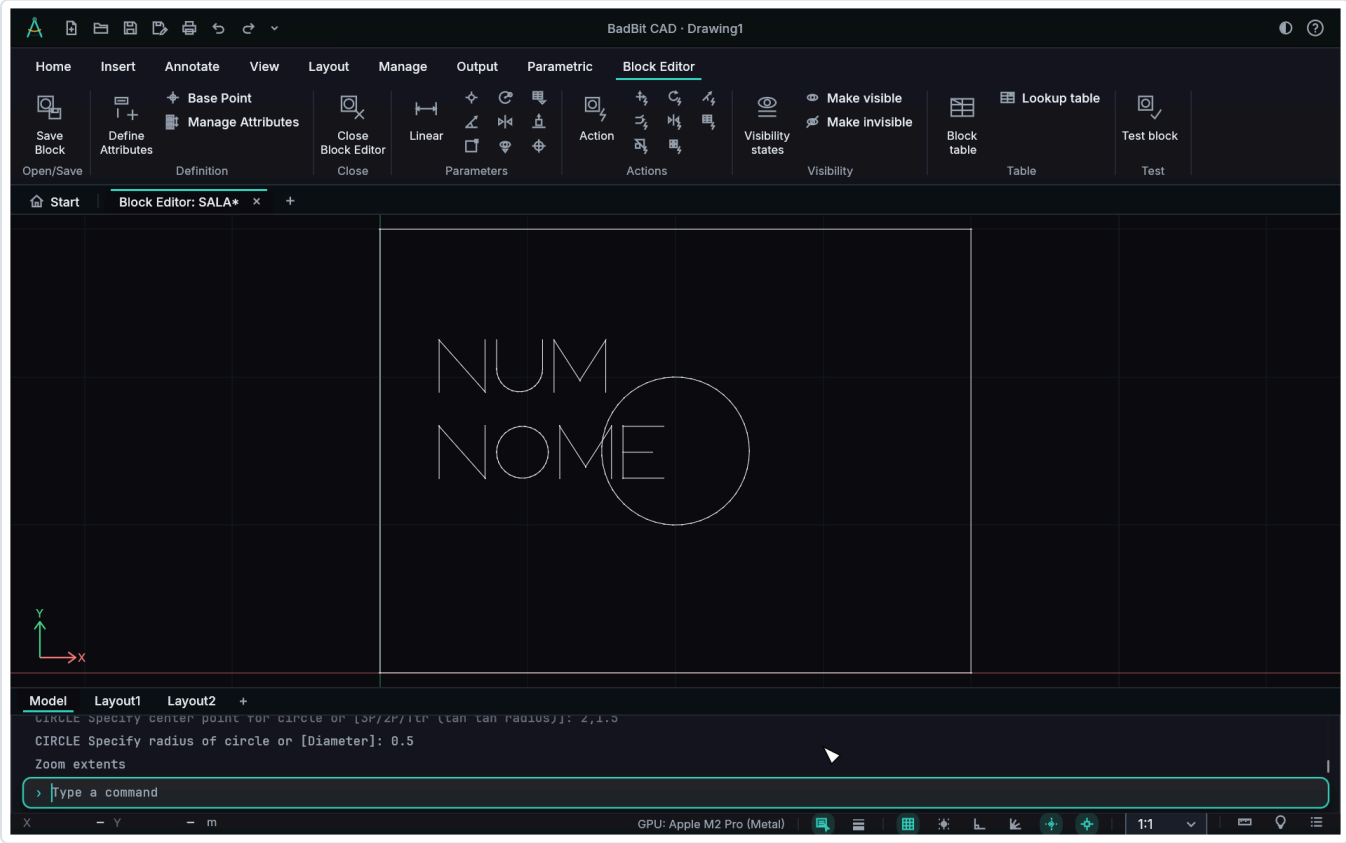
Block Editor (BEDIT, alias BE)

BEDIT opens the block in a view of its own, with only the definition, in the block's coordinates (the base point sits at the block's origin). With a reference selected, it opens that reference's block; otherwise, the **Edit Block Definition** dialog box lists the blocks. **Double-clicking** a reference without attributes also opens the editor.

The **Block Editor** contextual tab appears on the ribbon, with **Save Block** (**BSAVE**, `BS`), **Define Attributes**, **Base Point**, **Manage Attributes** and **Close Block Editor** (**BCLOSE**, `BC`). All drawing and editing commands work inside the editor. **BASE** changes the block's **base point**.

Saving replaces the block's objects and regenerates **all references** at once, including those inside other blocks (and the blocks that contain it), in a single undo step of the drawing. Closing with changes asks whether to save or discard. While the editor is open, opening and saving files and

switching tabs wait for BCLOSE. An external reference does not open in the editor (it is read-only). Changed the attribute definitions? Use **ATTSYNC** to bring them to the references.

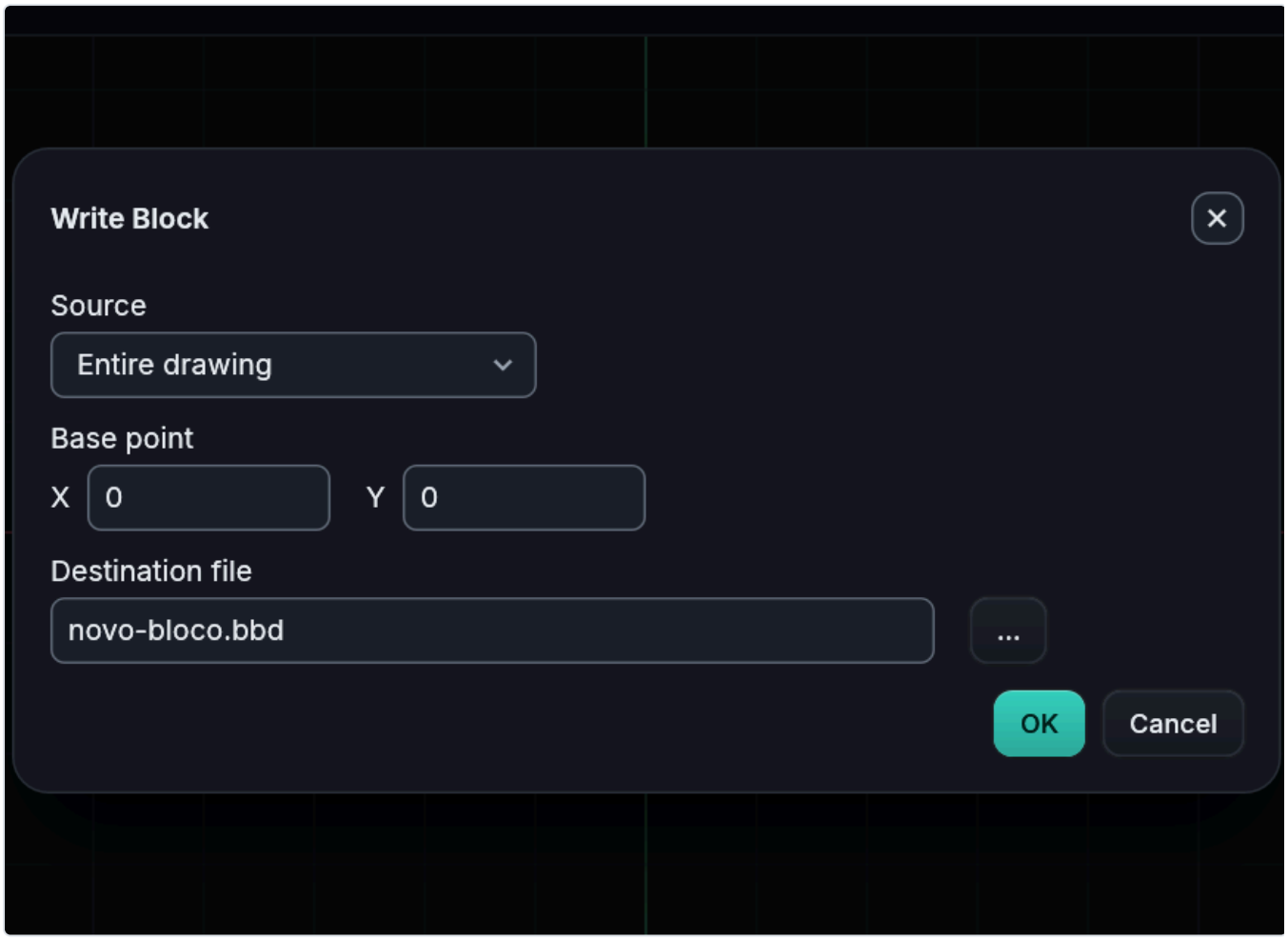


The SALA block in the Block Editor, with the contextual tab.

WBLOCK (alias W) and -WBLOCK (alias -W)

WBLOCK writes a **block** from the drawing, the chosen **objects** or the **entire drawing** to a **new** `.bbd` file, with the **base point** (which becomes the file's insertion base point, **INSBASE**). For objects: **Retain**, **Convert to block** (the objects become a reference of the block with the file's name) or **Delete from drawing**. The layers, linetypes, styles and blocks that the objects use go along.

-WBLOCK asks for the file name and then a block, `=` (the block with the file's name), `*` (the entire drawing) or `Enter` (the base point and the objects, which are removed from the drawing; **OOPS** brings them back).



The Write Block dialog box.

BASE

BASE sets the drawing's insertion base point (**INSBASE**, saved in the file): the point that sits at the cursor when this file is inserted as a block. In the Block Editor, it is the block's base point.

INSERT from a file

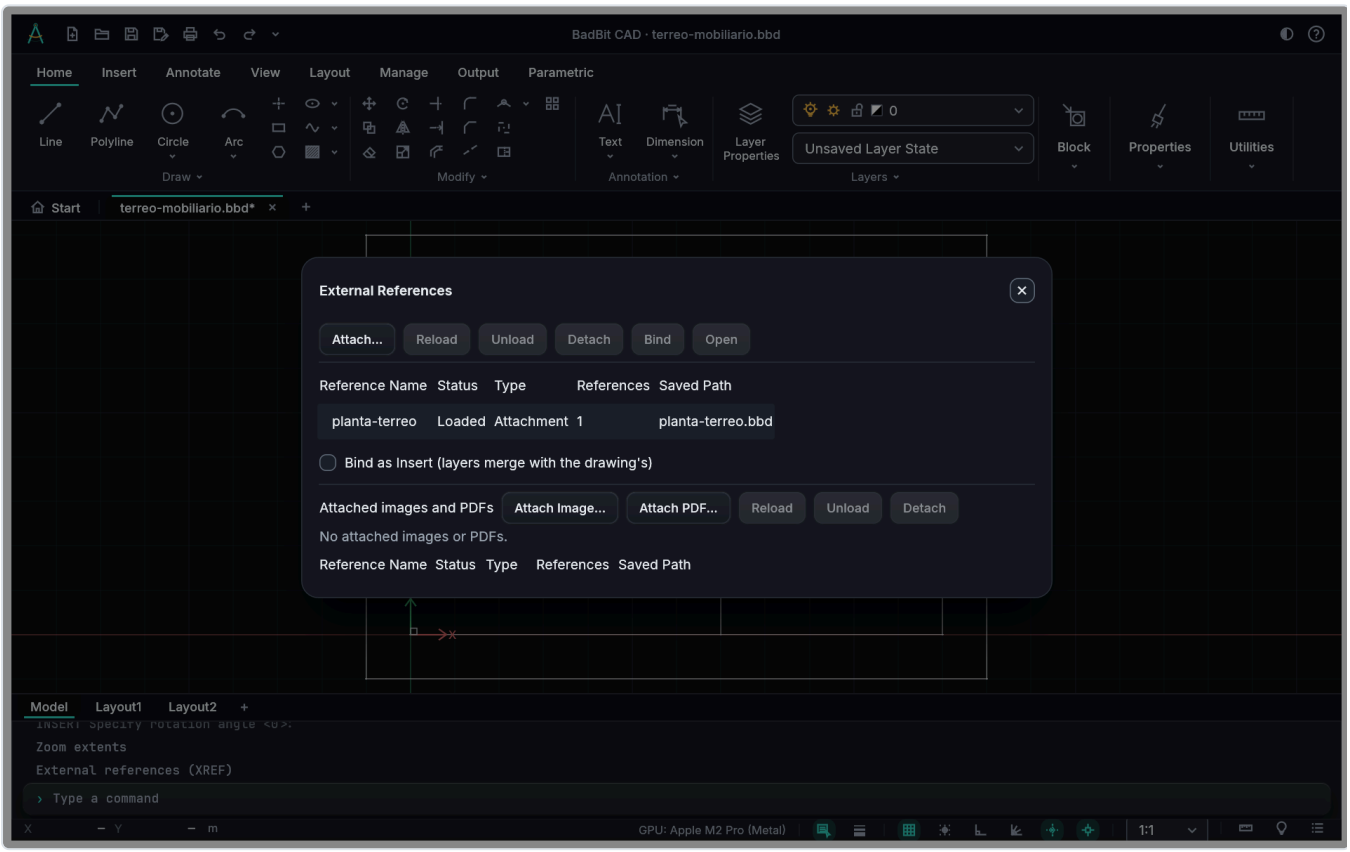
At INSERT's block name, type the **path** of a `.bbd` (`blocos/porta.bbd`): the file becomes the block with its name and INSERT goes on to the insertion point. Missing layers, styles and blocks come in along with it. If the block already exists, the existing one is kept; `name=path` **redefines** it (the references change together), and `~` opens the system dialog box. On the ribbon, the Insert button has **From a file...**.

External references

An **external reference** shows another drawing inside this one, like a **read-only** reference: the file stays separate, and whatever changes in it shows up here. The reference's layers come in with its name as a prefix (`planta-terreo|PAREDE`, as in AutoCAD) and can be turned on, turned off and colored in the layer manager. BLOCK, BEDIT and EXPLODE refuse an external reference; to change its drawing without leaving this one, use **REFEDIT** (below), which saves the reference's file.

- **XATTACH** (`XA`): choose the file; in the **Attach External Reference** dialog box, the **name**, the type (**Attachment**, or **Overlay**: not displayed when this drawing is a reference in another one) and the path type (**full**, **relative** to the drawing's folder, or **no path**, searched for in the drawing's folder). After OK, the insertion point, the scales and the rotation on the command line.
- **XREF** (`XR`, `ER`): the **External References** manager, with each reference's name, status (loaded, unloaded, not found), type, references and saved path; **Attach...**, **Reload**, **Unload** (disappears without detaching), **Detach** (removed along with its references and layers), **Bind** (becomes an ordinary block: the layers become `name$0$layer`, or, with **Bind as Insert**, merge with the drawing's; this is **BINDTYPE**) and **Open** (the file in a tab).
- **-XREF** (`-XR`): the same options from the command line: `?`, **Bind** (`B`), **Detach** (`D`), **Path** (`P`), **Unload** (`U`), **Reload** (`R`), **Overlay** (`O`) and **Attach** (`A`); the names accept commas and `*`.

BadBit CAD checks the reference files every second and **reloads automatically** the one that changed (AutoCAD only notifies); when the drawing is opened, the references come from the files. Reloading is recorded in undo. The reference can be a `.bbd`, a DXF or a DWG.



The ground floor plan attached as a reference and the External References manager.

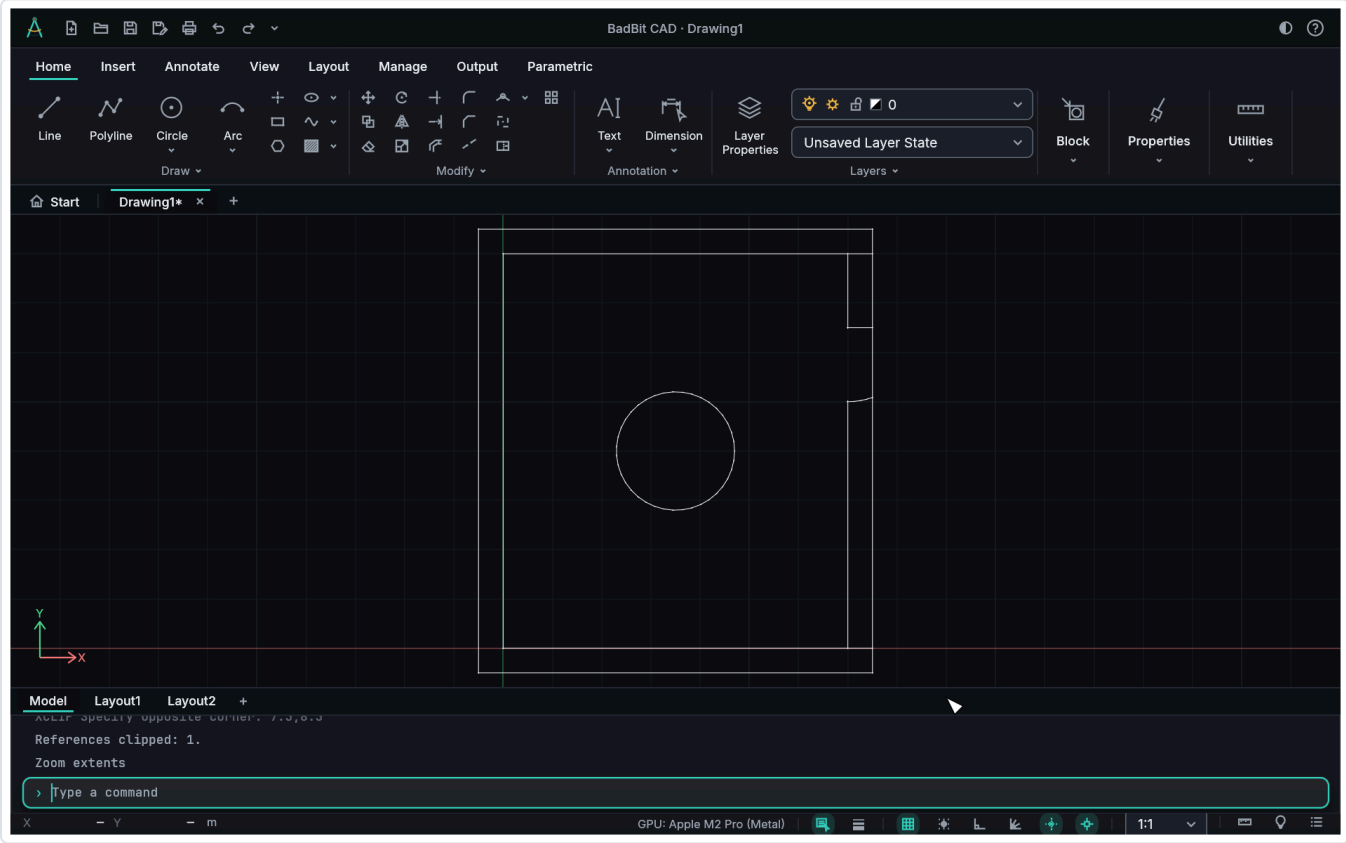
Clipping references (XCLIP, alias XC)

XCLIP clips block references and external references: only what is **inside** a boundary is displayed (or, inverted, what is **outside**). The clip belongs to the reference (each one has its own), moves with it (move, rotate, scale) and applies on screen, in selection and object snaps (what is hidden cannot be selected or snapped to) and in plotting (PDF, SVG and PNG come out clipped). The block's drawing does not change: EXPLODE gives everything back.

Choose the references and answer **Enter clipping option**:

- **New boundary** (`N`, or `Enter`): if there is already a clip, it asks whether to delete the old one (**Yes** `Y` / **No** `N`). Then the boundary: **Rectangular** (`R`, or `Enter`: the two corners), **Polygonal** (`P`: the points, with **Undo** `U`, and `Enter` to close) or **Select polyline** (`S`: a polyline; arcs come in as fine straight segments). **Invert clip** (`I`) switches between hiding what is outside (the outside mode, the default) and what is inside.
- **ON** (`ON`) and **OFF** (`OFF`): turn the clip on and off without losing the boundary.
- **Delete** (`D`): removes the clip.
- **generate Polyline** (`P`): draws the boundary as a polyline, on the current layer.
- **Clipdepth** (`C`): the front and back planes in 3D; does not apply to 2D drawings.

XCLIPFRAME controls whether the boundary is displayed: **0** not displayed and not plotted, **1** displayed and plotted, **2** (the default) displayed only on screen. It is stored in the drawing, as in AutoCAD. The clip goes to the `.bbd`, the DXF and the DWG the way AutoCAD stores it (the spatial filter in the reference's extension dictionary). On the ribbon: **Insert** tab, **Reference** panel, **Clip**.



The floor plan inserted as a block, clipped at the room on the left, with the boundary (XCLIPFRAME 2).

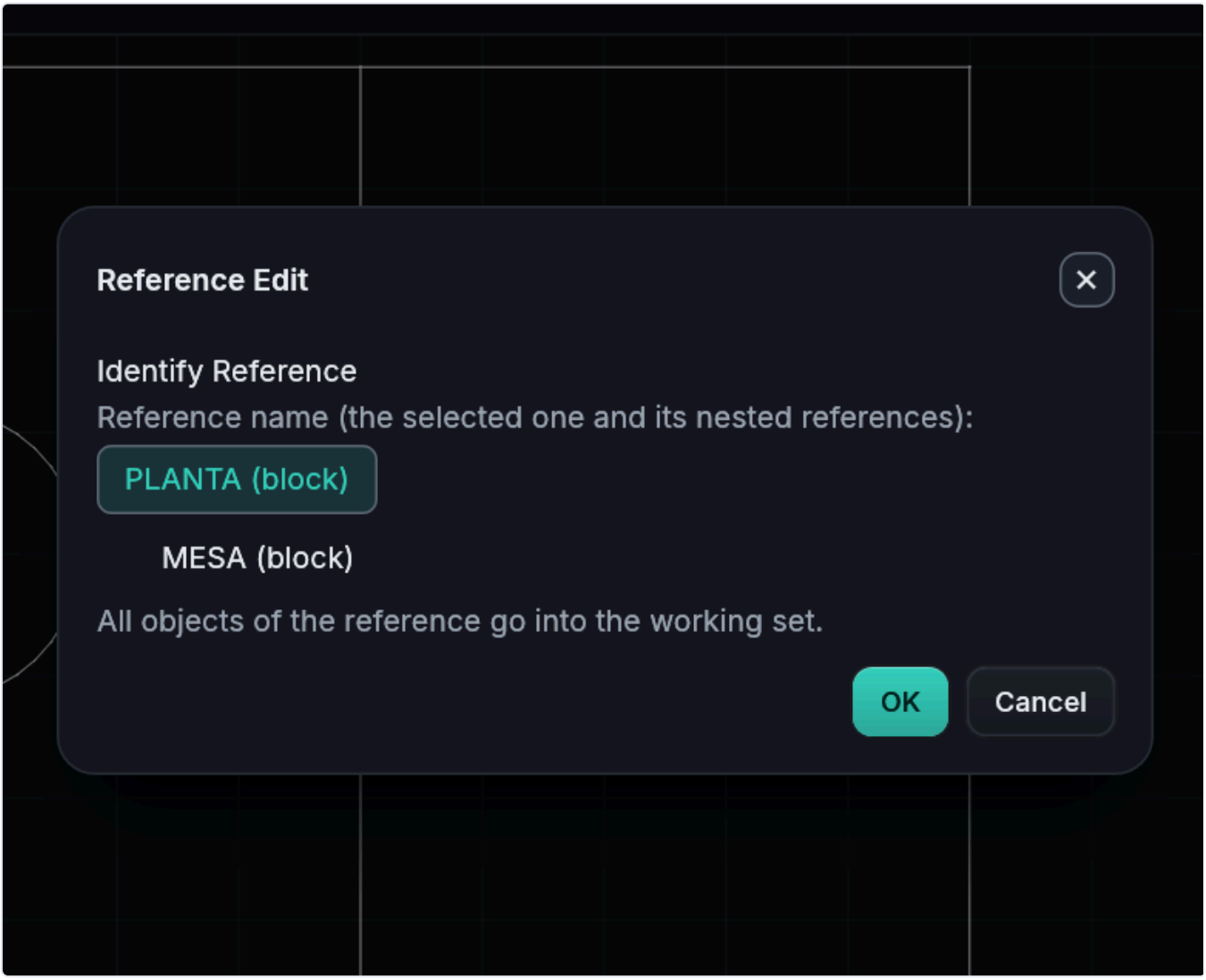
Editing a reference in place (REFEDIT)

REFEDIT edits a block or an external reference **in place**, without leaving the drawing and without opening the Block Editor. Choose the reference (or **double-click** an external reference, or right-click a selected reference, **Edit Reference In-Place**); the **Reference Edit** dialog box shows the tree with the chosen reference and the ones nested in it. Choose which one to edit and click **OK**.

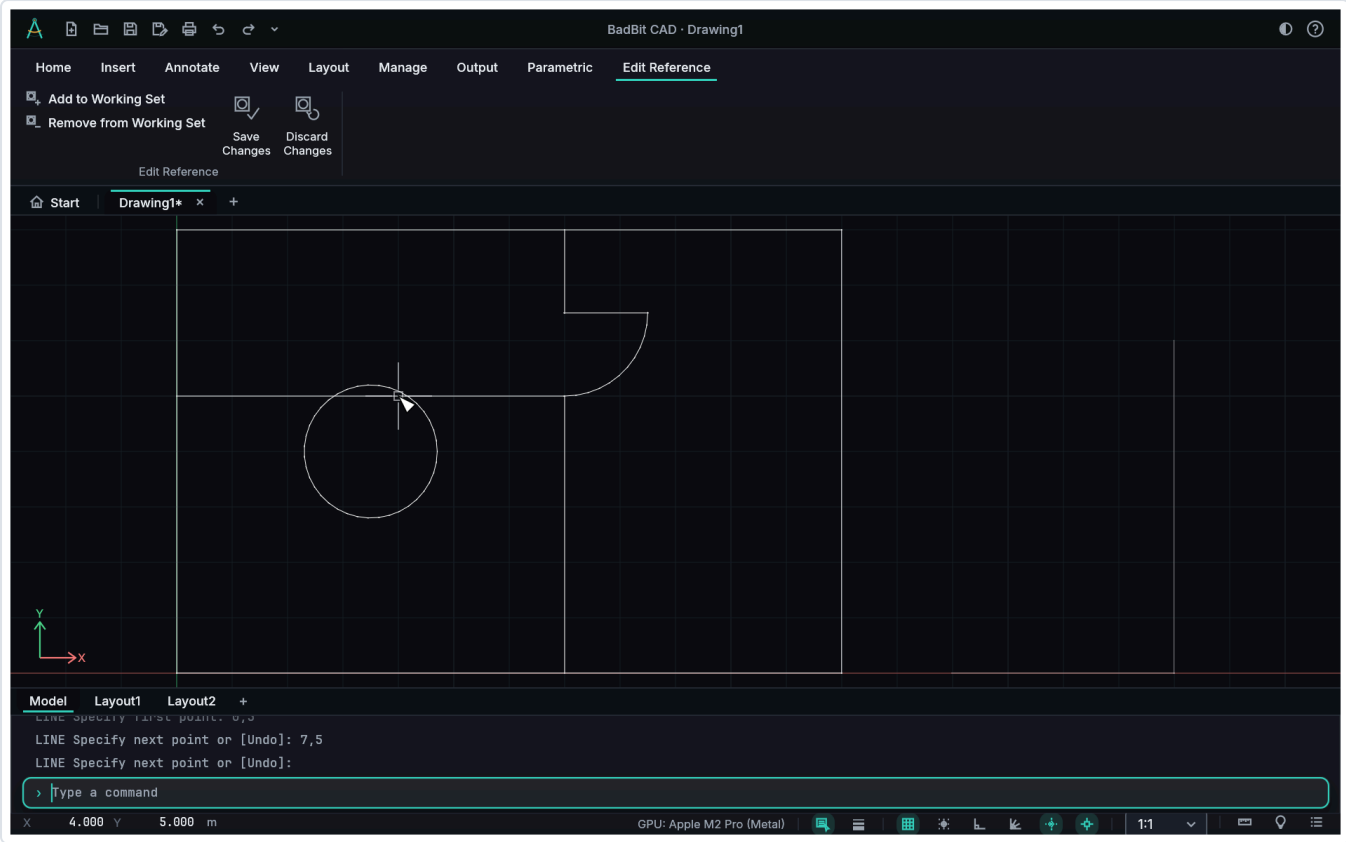
The reference's objects become editable in its place (with the reference's scale and rotation); the rest of the drawing is **faded** (**XFADECTL**, from 0 to 90%, default 50) and cannot be selected. The **Edit Reference** contextual tab appears with:

- **Add to Working Set** and **Remove from Working Set** (**REFSET**, options **Add** `A` and **Remove** `R`): put objects from the drawing into the **working set** (they go to the block when saving and leave the drawing) or take objects out of it (they stay in the drawing). Objects drawn during the editing session join the set automatically.
- **Save Changes** and **Discard Changes** (**REFCLOSE**, options **Save** `S` and **Discard** `D`).

Saving regenerates **all references** of the block (and the blocks that use it), in a single undo step. For an **external reference**, saving writes **its file** in the original format (`.bbd`, DXF or DWG, in the same version) and reloads it; a reference in a **read-only** file or one **open in another tab** cannot be opened for editing (a message appears). The block's attribute definitions stay as they are (use BEDIT or BATTMAN to change them). While the editing session is open, opening and saving files and switching tabs wait for REFCLOSE.



The Reference Edit dialog box: the floor plan and the table nested in it.



The floor plan being edited in place: the rest of the drawing faded, a new partition and the Edit Reference tab.

Purging unused blocks

PURGE (Manage tab, Cleanup panel) now has **Blocks** (`B`): it removes block definitions that no reference uses. Layers used by the blocks' items are not purged.

Dynamic blocks

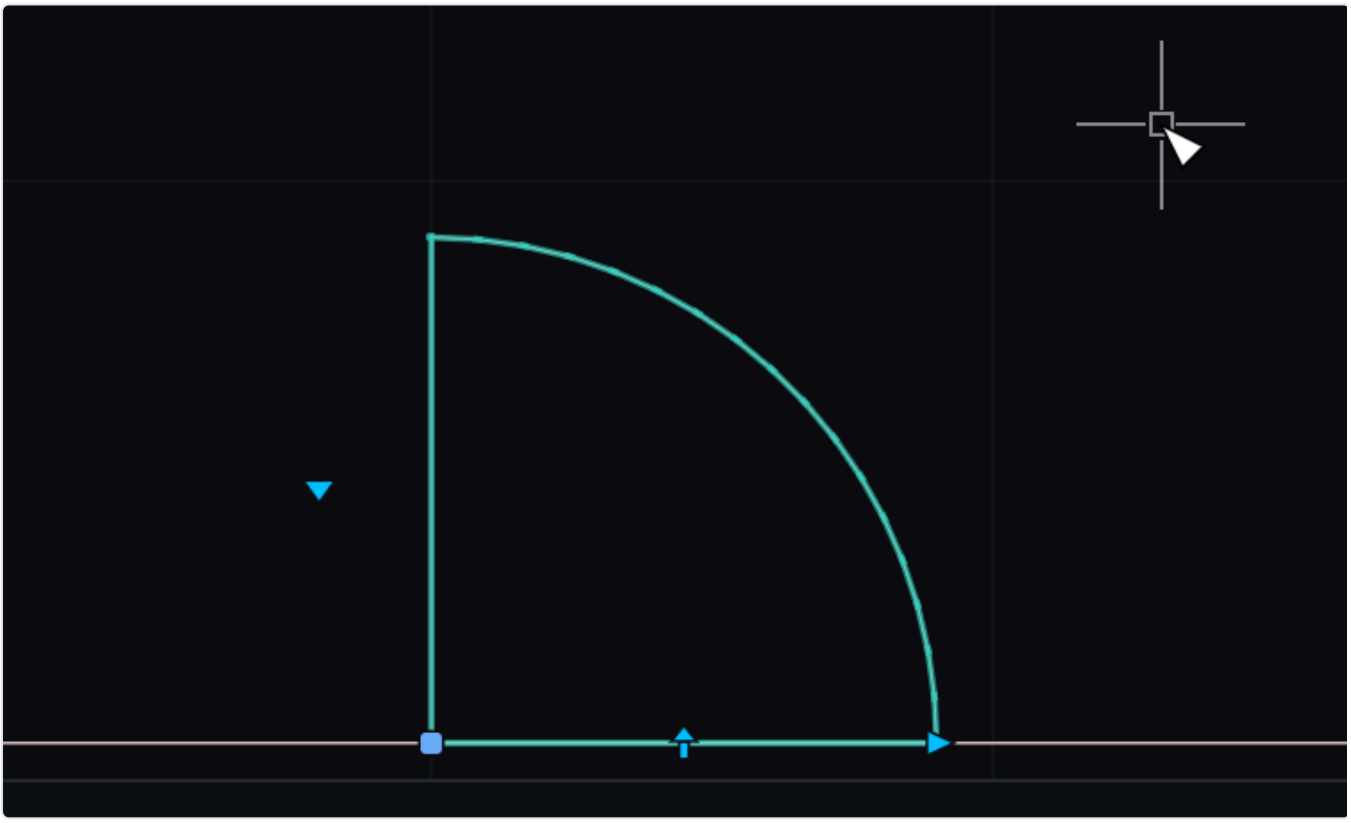
A **dynamic block** carries parameters and actions (defined in the Block Editor or read from an AutoCAD DWG or DXF): the same door, for example, changes width, opens to the other side and appears open or closed, without needing one block for each case. Each reference stores its own values.

Dynamic grips

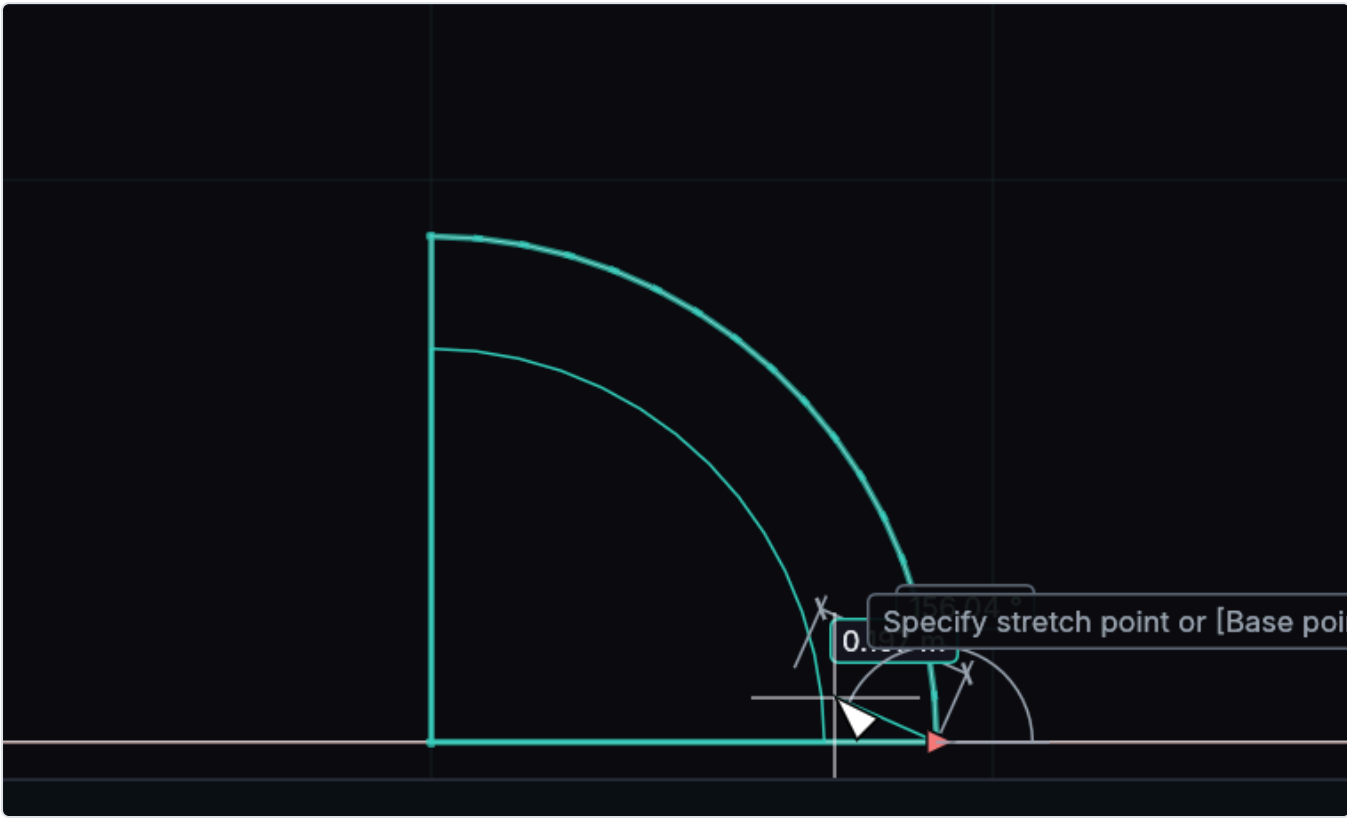
Select the reference: besides the insertion point grip (the blue square), the **dynamic grips** appear, in cyan, with the icon of each parameter type, as in AutoCAD:

- **square**: point, polar and XY;
- **arrow** in the parameter's direction: linear;
- **circle**: rotation;
- **flip arrow**, perpendicular to the reflection line: flip;
- **notched arrow**: alignment;
- **down arrow** (inverted triangle): the visibility, lookup and block properties table list.

They follow **GRIPS** and **GRIPOBJLIMIT** like the other grips. Click a dynamic grip (or press and drag): it turns **red** and the reference follows the cursor, with the actions already applied (the door scales, stretches, rotates...). The value is bound to the parameter's value set: free, in increments or only the values in the list. Object snaps work as with other grips. When you release (or on the second click) the change goes in as a single undo step.



The dynamic door selected: the insertion point grip, the width arrow, the flip arrow and the visibility down arrow.



The width grip hot: the door follows the cursor, bound to the values in the list.

The **flip** grip is not dragged: a click toggles it. The **list** grip opens a menu with the visibility states (or the rows of the lookup or of the table), with the current one checked; choosing one saves it.

A linear parameter with **two grips** also shows the **base** grip, with the base at the start or in the middle: with the base at the start, dragging it moves only the base (the end stays put and the actions tied to the start point move); with the base in the middle, both sides move by half. When the dynamic grip falls on the same spot as the insertion point (the base of the linear, the alignment), the click picks the dynamic one, as in AutoCAD.

The **alignment** grip carries the reference along with the cursor; resting over an object (line, arc, polyline, circle...), the grip falls on its nearest point and the reference **rotates to align**: the parameter's direction becomes **tangent** to the object (or **perpendicular**, depending on the type chosen in BPARAMETER), on the side closest to the current one. Away from any object, it only moves.

An **attribute** hidden in a visibility state keeps its value: when you return to a state that shows it, it appears with the value it had. The hidden value also goes to the DXF and DWG (the attribute is saved as invisible in the reference) and comes back when the file is opened.



The visibility grip menu, with the current state checked.

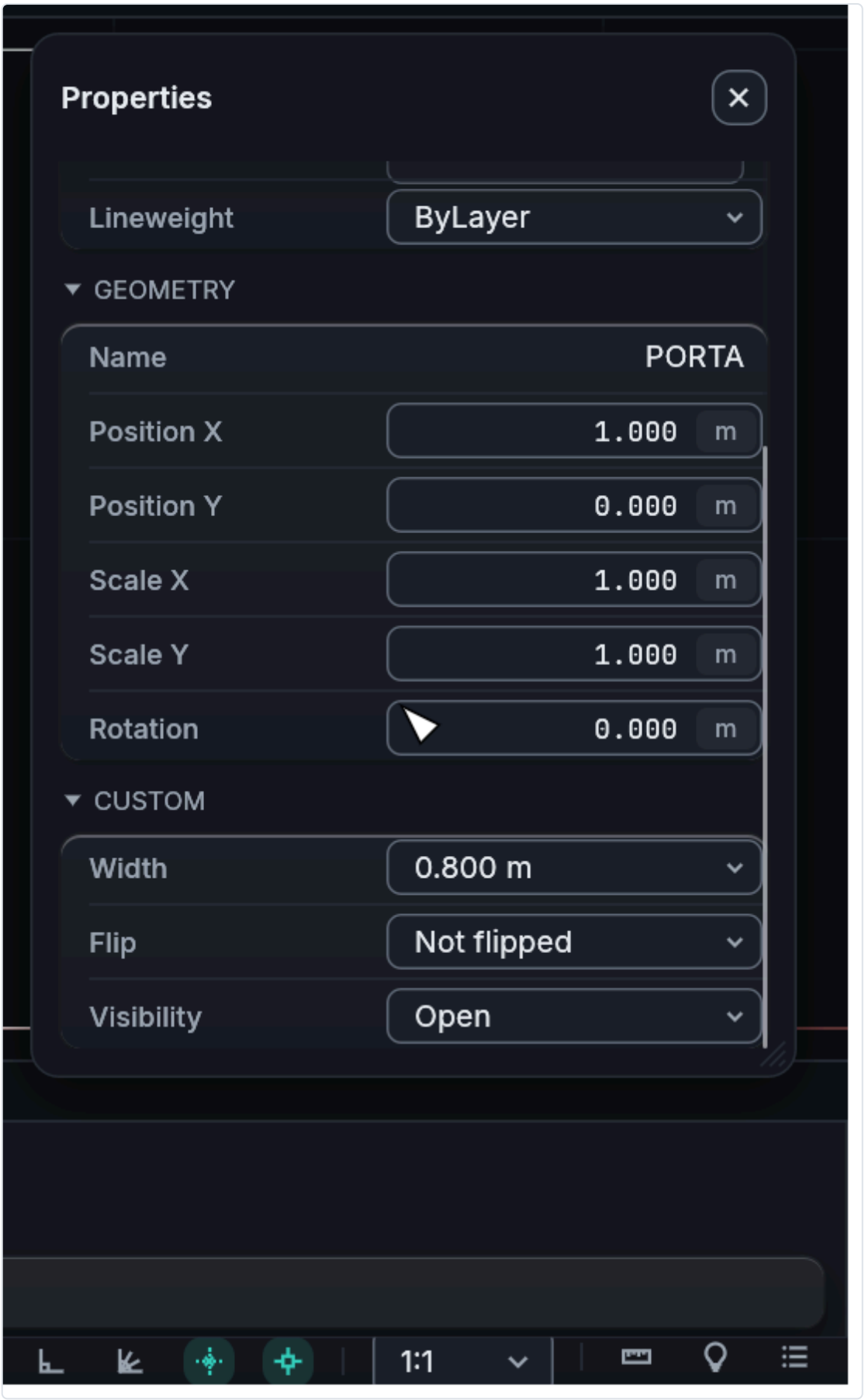
The Custom section

In the **Properties** palette (Ctrl+1), one or more dynamic references of the **same block** get the **Custom** section, with the parameters that the block displays:

- editable distances and angles, in the drawing's units like the other fields; when the parameter only accepts the values of a list, the row becomes a **drop-down list**;

- distances (and the positions of the point parameter) already including the **reference's scale**, as in AutoCAD: a door of 10 inserted at scale 2 shows 20, and typing 30 leaves the door at 15 in the block (the scale in the parameter's direction: for XY, the X and Y scales);
- flip, visibility, lookup and block properties table through the list; a **Read only** lookup shows only the row's text (no list).

With several references, values that differ appear blank ("mixed"), and the chosen value applies to all of them, in one undo step. The section also appears in **Quick Properties**.



The door's Custom section: the width in the list, the flip and the visibility.

Reset Block (RESETBLOCK)

RESETBLOCK (AutoCAD's name in all three languages) returns the chosen dynamic references to the values in the block definition, in one undo step. Select first or answer "Select objects:". Right-clicking a selected dynamic reference also offers **Reset Block**.

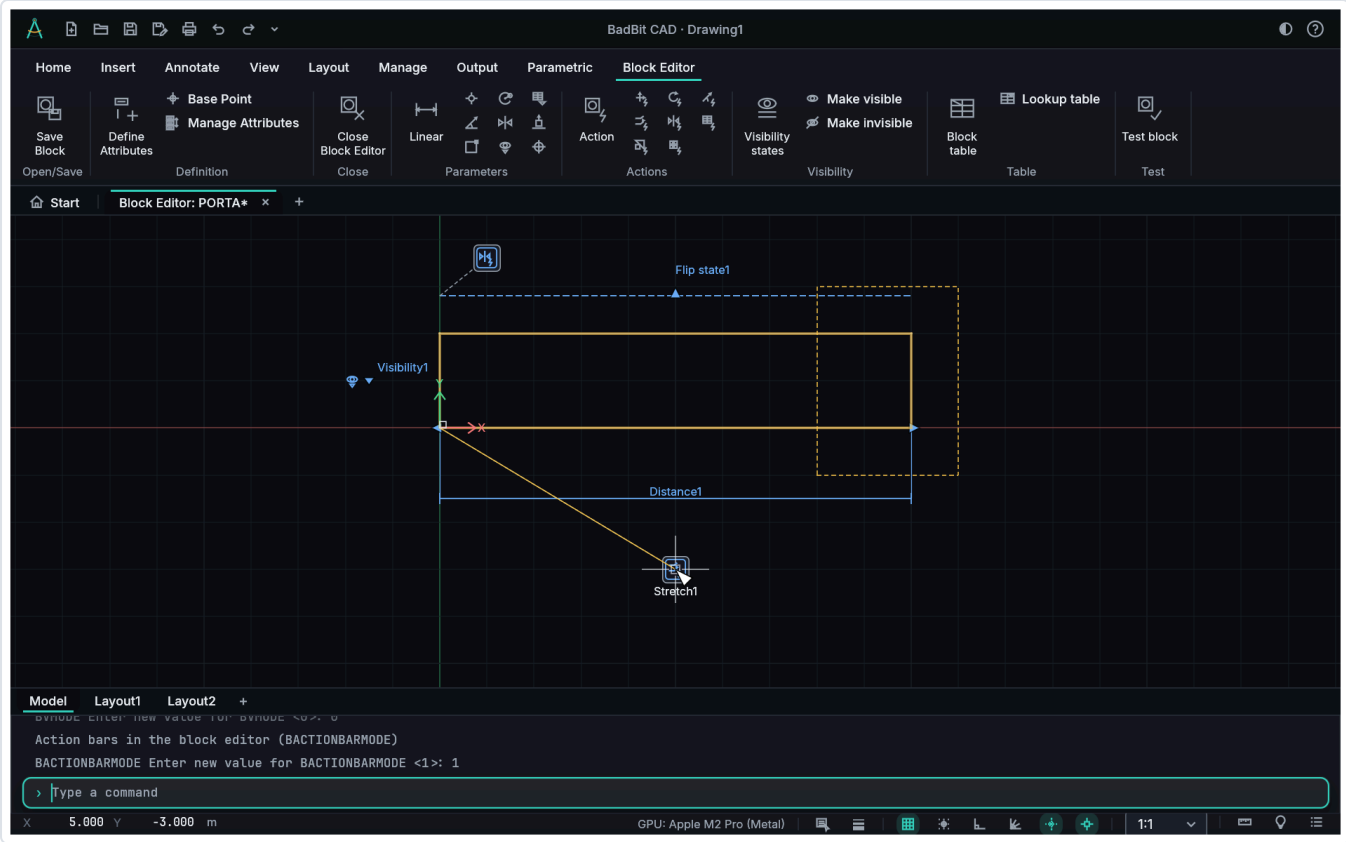
When you insert a dynamic block (INSERT), the preview already shows the block's visibility state, and the command line says which one it is ("Visibility state: Open").

Creating dynamic blocks in the editor

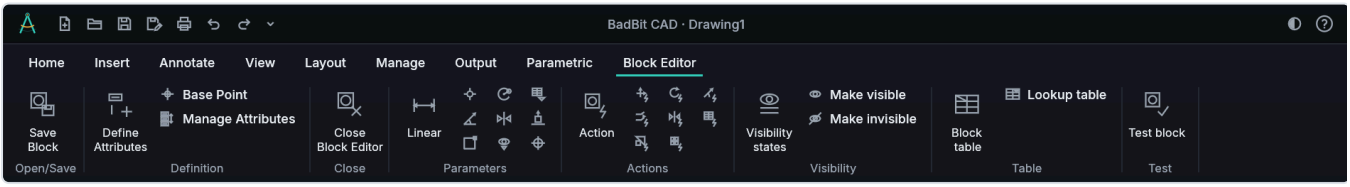
Inside the Block Editor (**BEDIT**), the block gets **parameters** (what changes: a distance, an angle, a state) and **actions** (what each parameter does to the objects: stretch, move, rotate...), as in AutoCAD. At the end, the **Block Editor** tab shows the **Parameters**, **Actions**, **Visibility**, **Table** and **Test** panels. Parameters and actions are displayed over the drawing: the parameter with its label and grips, and the **yellow alert** while it has no action. Actions are displayed in **action bars** (**BACTIONBARMODE** 1, the AutoCAD default): one bar per parameter, at the location of its first action, with the icon of each action and a dashed line to the parameter; with **BACTIONBARMODE** 0, each action has its icon and name at its own location, the old way. With the cursor over an action's icon, its name, the stretch frame and the objects it carries are displayed.

Parameters and actions **are erased on screen**, like objects: click a parameter (or an action's icon) and it is highlighted in yellow (Shift+click removes it from the selection, Esc clears); **Delete** or **ERASE** erases it, in one undo step. Erasing a parameter takes its actions and the references to it (in selection sets and tables). ERASE also accepts clicking a parameter or an action at the "Select objects" prompt.

BSAVE saves the dynamic part along with the objects: the references in the drawing then have dynamic grips and the **Custom** section in the Properties palette, storing the values by parameter name. **Undo** inside the editor also undoes parameters, actions and states.



The door in the editor: the linear Distance1 with the action bar of Stretch1 (the cursor over the icon shows the name, the frame and the objects), the flip and the visibility, with the handle hidden in the Plain state (BVMODE 0).



The Parameters, Actions, Visibility, Table and Test panels at the end of the Block Editor tab.

Parameters (BPARAMETER, alias PARAM)

BPARAMETER asks for the type: **Alignment** (`A`), **Base** (`B`), **pOint** (`O`), **Linear** (`L`), **Polar** (`P`), **XY** (`X`), **Rotation** (`R`), **Flip** (`F`), **Visibility** (`V`) and **looKup** (`K`); the prompt shows the last type as the default (`<Linear>` the first time) and Enter repeats it. The buttons in the **Parameters** panel choose the type already. Then come AutoCAD's prompts:

- the **first point** (the start point, the base point or the parameter location), with the options **Name** (`N`, the name of the parameter object, such as `Linear1`), **Label** (`L`, the name of the property in the palette, such as `Distância1`), **Chain** (`C`, chain the actions), **Description** (`D`), **Base** (`B`, linear only: the base at the **Startpoint** `S` or the **Midpoint** `M`, both sides moving by half), **Palette** (`P`, whether to show it in the Properties palette) and **Value set** (`V`);
- the **endpoint** (for flip, the end of the reflection line; for rotation, the **radius** and the **default angle**, with the **Base angle** `B` option, the reference line that the editor draws dashed and that is stored; for alignment, the direction or **Type** `T`, **Perpendicular** or **Tangent**);
- the **label location**: it is stored; the editor draws the label there and, for linear and polar, moves the dimension line out to it, as in AutoCAD;
- the **number of grips** (0, 1 or 2 for linear and polar; 0, 1, 2 or 4 for XY; 0 or 1 for the others).

With **Chain** on (AutoCAD's **Chain Actions**), a parameter that is in the selection set of an action of another parameter triggers its own actions when it is carried along: a point with a move, carried by the stretch of a linear, takes the move's objects along. Without Chain, the parameter moves with the action but its own actions do nothing. A chain that comes back to a parameter that has already fired stops there.

The **value set** limits what the reference accepts: **None** (`N`), **List** (`L`, values separated by commas, or by semicolons if you use the decimal comma; angles in degrees) or **Increment** (`I`, the step, the minimum and the maximum; Enter leaves it unset). Polar asks for the distance one and the angle one; XY, the horizontal and the vertical. A block has at most one **visibility** parameter and one **base point** parameter.

Actions (BACTION, alias AC, and BACTIONTOOL)

BACTION asks for the **parameter** (click near it, or type its name) and the **action type** it accepts, as in AutoCAD: point accepts **Move** and **Stretch**; linear and XY, **Array** (`A`), **Move** (`M`), **Scale** (`S`) and **sTretch** (`T`); polar also **Polar stretch** (`P`); rotation, **Rotate**; flip, **Flip**; lookup, **Lookup**. **BACTIONTOOL** (the buttons in the **Actions** panel) chooses the type first and then the parameter.

Depending on the type, these come next:

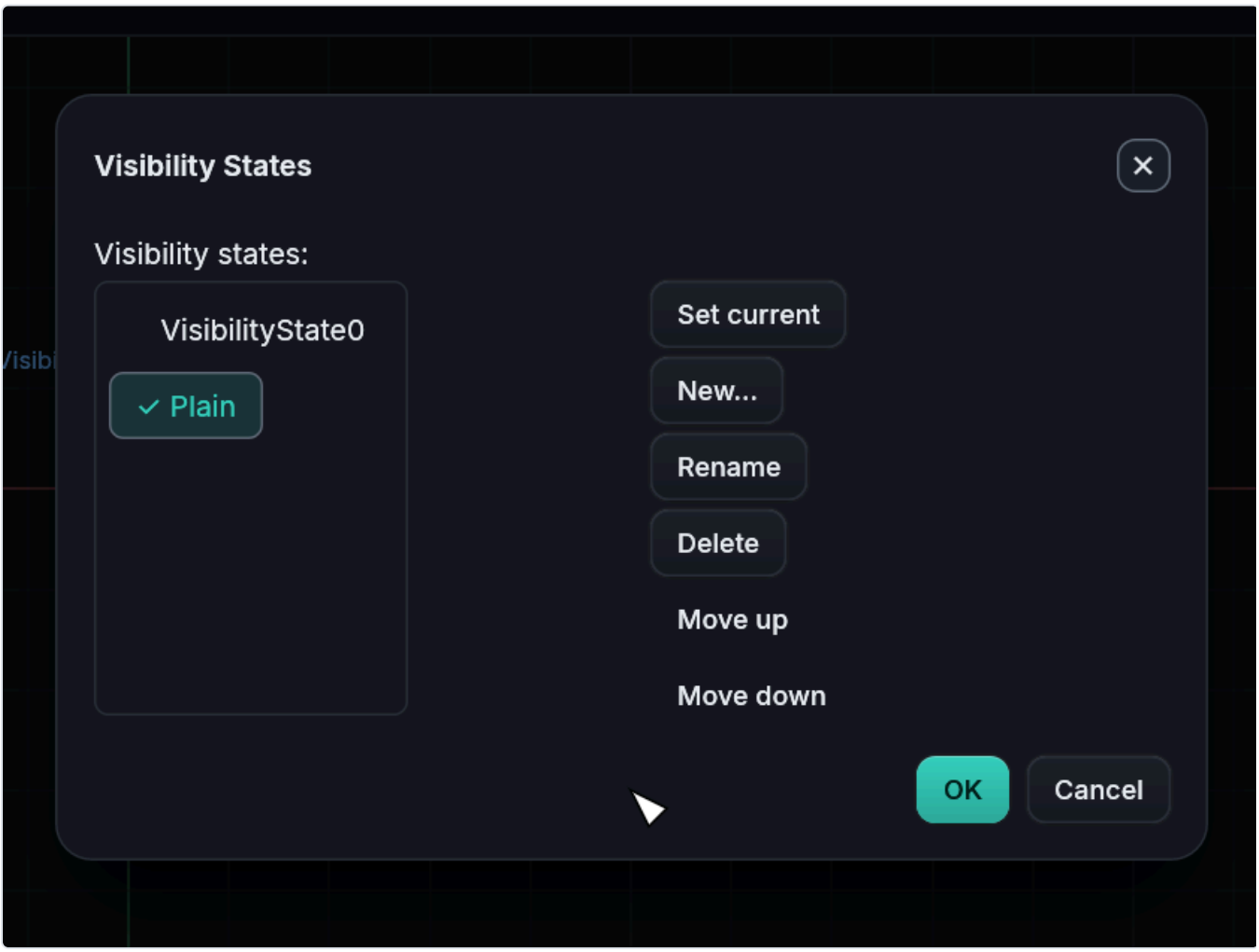
- the **parameter point** that drives the action (**sTart point** `T` or **Second point** `S`, or a click near one of them);
- the **stretch frame** (two corners, or **CPolygon** `CP`): what is entirely inside moves; of what crosses it, only the points inside; other parameters with a point inside the frame go along;
- the action's **objects** (the usual selection); for polar stretch, afterwards, the objects that **only rotate**;
- for array, the **distance between rows** (for XY) and **between columns**: typed or by **two points** (picked or typed as `x,y`); for XY, the two corners of the **unit cell** give both distances at once;
- the **action location** (where the icon goes), with the options **Multiplier** (`M`) and angle **Offset** (`O`) for move and stretch, **Base type** (`B`: **Dependent**, the parameter's base point, or **Independent**, a point of its own) for scale and rotate, and **XY type** (`X`: X, Y or XY) for the actions of an XY parameter.

The **lookup** action opens the **Property Lookup Table** dialog box (**BLOOKUPTABLE** opens it again): the **input properties** (the parameters chosen with **Add properties...**) and the **lookup** property, one row per combination, with the text shown in the reference's list. The last row of the lookup column chooses, as in AutoCAD, **Read only** (the default: the reference shows only the row that matches the inputs, with no list or grip) or **Allow reverse lookup** (the grip and the palette choose the row, and it sets the inputs to its values).

Visibility (BVSTATE, alias BVS; -BVSTATE; BVHIDE; BVSHOW)

With a visibility parameter, **BVSTATE** opens the **Visibility States** dialog box: **Set current**, **New...** (with the options **Hide all existing objects in new state**, **Show all existing objects in new state** or **Leave visibility of existing objects unchanged in new state**), **Rename**, **Delete** and **Move up/Move down**. The state that is current when saving is the one for new references. On the command line, **-BVSTATE** has **New** (`N`), **Set** (`S`) and **Delete** (`D`).

BVHIDE and **BVSHOW** hide and show the chosen objects in the **Current** (`C`) state or in **All** (`A`). **BVMODE** sets how the objects hidden in the current state appear in the editor, as in AutoCAD: **0** (the default) they disappear and cannot be selected; **1** they are **faded**, so they can still be selected (for BVSHOW, for example). Type `BVMODE` and the value, or use SETVAR.



The Visibility States dialog box with the new Plain state, without the handle.

Block properties table (BTABLE)

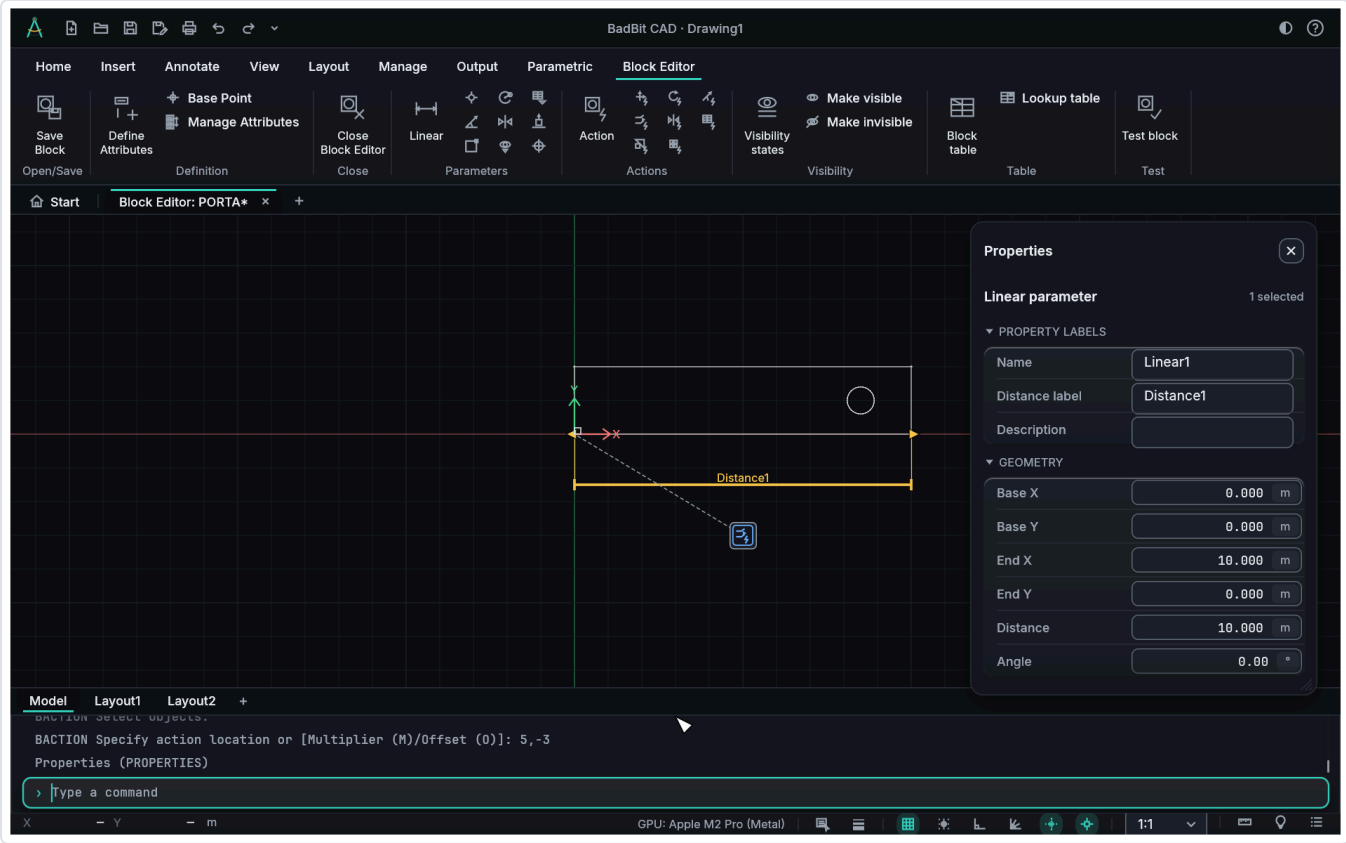
BTABLE asks for the table parameter's **location** and the **number of grips** (0 or 1) and opens the **Block Properties Table** dialog box: the columns are the parameters (**Add properties...**), each row is a combination of values (numbers as AutoCAD displays them, angles in degrees, states by name), and **Block properties must match a row in the table** rejects values outside it. On the reference, the table grip chooses the row.

Parameter and action properties (Ctrl+1)

As in AutoCAD, clicking a parameter (or an action's icon) in the Block Editor and opening the **Properties** palette (`Ctrl`+`1` or **PROPERTIES**) shows its fields, all editable; each change is an undo step of the editor:

- **Property labels**: the **Name** of the parameter object (blank: the type name), the property **label** (the name in the references' Custom section; for polar and XY, both labels), the **Description** and, for flip, the names of the **Not flipped** and **Flipped** states;
- **Geometry**: **Base X/Y** and **End X/Y** (for rotation, the **Center**, the **Radius**, the **Angle** and the **Base angle**; for single-point parameters, the **Position**), the **Distance** and **Angle** of linear and polar (changing the distance moves the end along the parameter; the angle rotates it around the base) and the **horizontal and vertical distances** of XY;
- **Value set**: the **type** (None, Increment or List) and, depending on it, the **increment**, the **minimum** and the **maximum** (blank: no limit) or the **value list** (separated by semicolons); for polar, the distance one and the angle one; for XY, the horizontal and the vertical;
- **Misc**: the linear's **Base location** (startpoint or midpoint), the **Number of grips**, **Show properties**, **Chain actions**, the alignment type and, for the block table, whether the properties must match a row.

For an **action**: the **Action name**, its type and parameter (read-only), the **Distance multiplier** and the **Angle offset** (move, stretch and polar stretch), the **XY type** (for the actions of an XY parameter), the **Base type** (Dependent or Independent, with its own point) for scale and rotate, and the **column and row offsets** of the array. With several chosen, only the fields they all have appear; differing values are shown as mixed and the typed value applies to all.

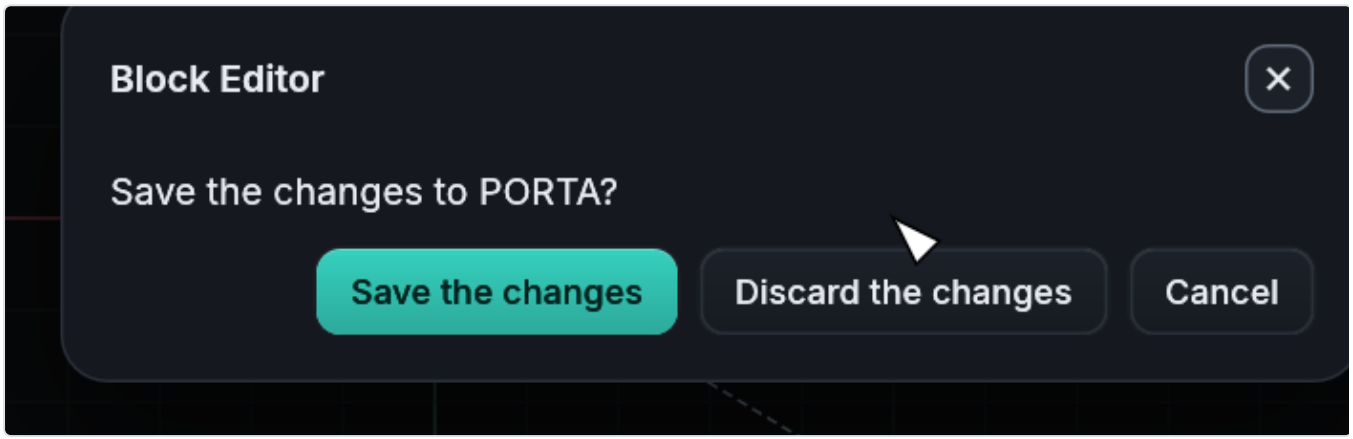


The linear parameter Distância1 chosen in the editor: the Properties palette with the labels, the geometry, the value set and the Misc fields.

Testing the block (BTESTBLOCK)

BTESTBLOCK (**Test** panel) opens the **test block window**, as in AutoCAD: it takes the editor's place (the drawing tab then reads **Test block: NAME**) with a drawing containing only one reference of the block as it is in the editor, and the ribbon gets the **Test Block Window** tab. Everything that works in a drawing works there: select the reference to drag the **dynamic grips** (the list ones open the list of states or rows), change the values in the **Custom** section of the Properties palette, use the commands (MOVE, COPY, INSERT...). BSAVE waits for the test to close.

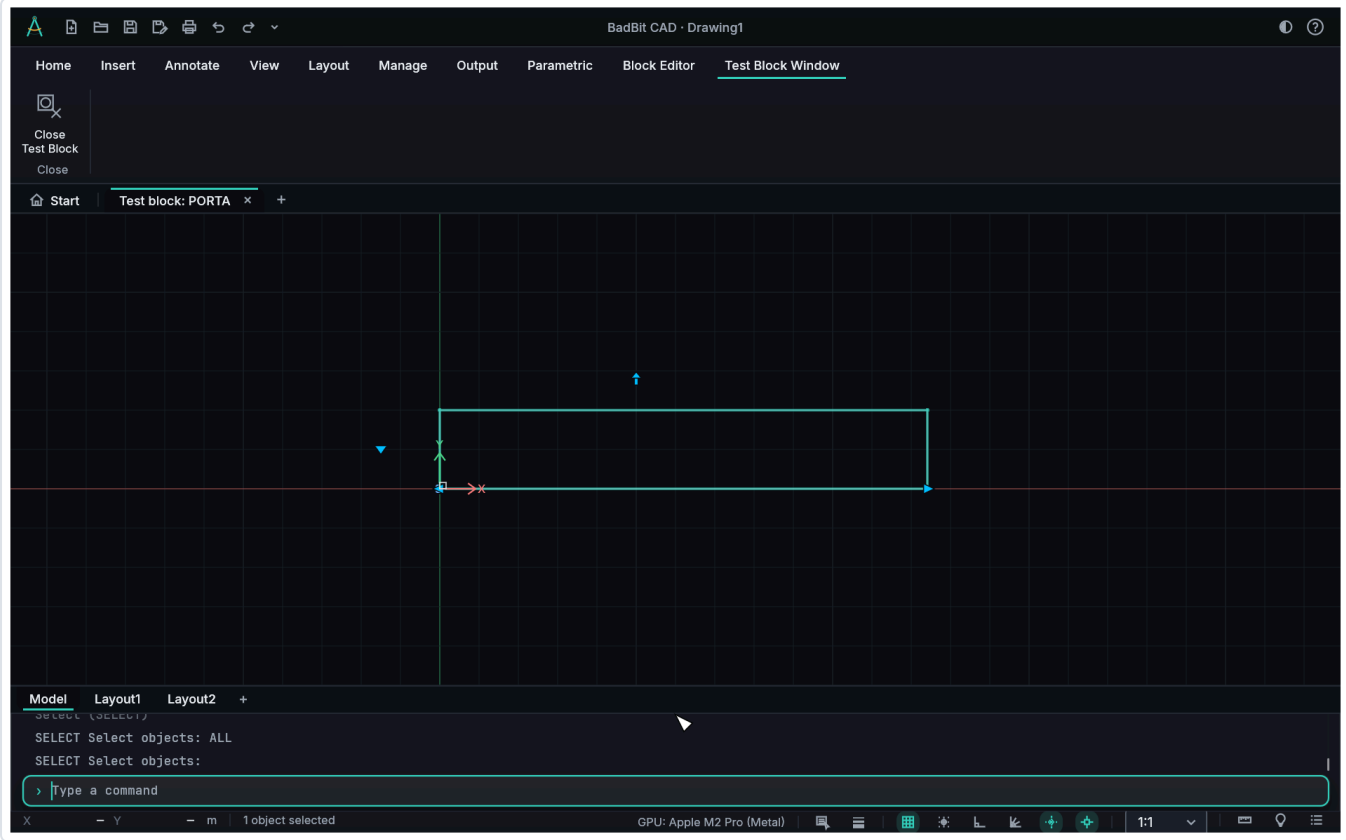
- **Close Test Block** (the contextual tab button) or the drawing tab's **×**: goes back to the editor as it was, without asking anything, as in AutoCAD (what was done in the test window does not go to the block).
- **BCLOSE** typed in the test window: closes the window and goes on to close the Block Editor; if the block was changed in the editor and not yet saved, the prompt **Save the changes to NAME?** appears, with **Save the changes** (saves the block, and the references in the drawing change together), **Discard the changes** and **Cancel** (stays in the editor). Without changes, it goes straight back to the drawing.



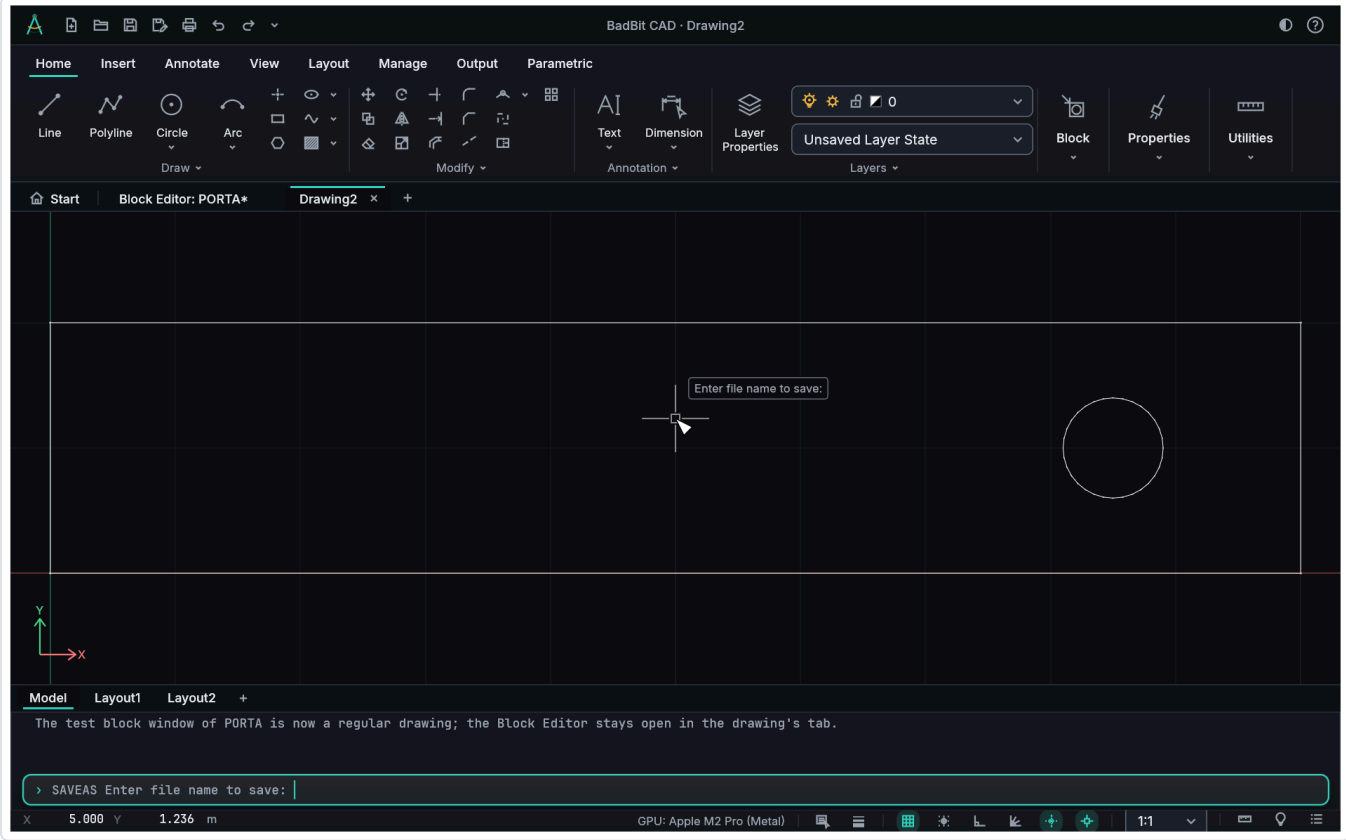
BCLOSE in the test window, with the door changed in the editor: the Save the changes prompt.

- **BTESTBLOCK** again in the test window: closes the window and opens another with the block as it is in the editor (what was changed in the test is lost), as in AutoCAD.
- Closing the program with the test window open closes the test first and saves the block, like closing with the Block Editor open.
- **SAVE**, **QSAVE** (`Ctrl` + `S`) or **SAVEAS** in the test window: as in AutoCAD, it **stops being a test window and becomes an ordinary drawing**, in a new untitled tab (with everything done in the test), and saving asks for the file name. The Block Editor stays open in the drawing's tab, with what was not saved: click the tab (or `Ctrl` + `Tab`, or BEDIT in it) and it comes back as it was. (AutoCAD's help mentions SAVE and QSAVE; SAVEAS doing the same is assumed.)

Switching tabs with the Block Editor open also leaves it **parked in its tab** (the tab reads **Block Editor: NAME**): the other tabs work normally and, back in that tab, the editor comes back. Only one Block Editor is parked at a time; closing the drawing's tab also closes the editor parked in it, without saving the block.



The test block window: the door stretched by the linear grip and the Test Block Window tab.



After SAVE in the test window: the test became the drawing in the new tab, and the Block Editor waits in the drawing's tab.

Parametric constraints inside the block (**BCPARAMETER**, **BCONSTRUCTION**) are not yet supported in the editor; blocks that bring them from AutoCAD keep them as they came.

Exporting and importing attributes: ATTOUT and ATTIN

ATTOUT writes the attribute values of the chosen blocks to a tab-delimited text file (the one from the old Express Tools), for editing in a spreadsheet: the first line has **HANDLE**, **BLOCKNAME** and the tags of all the blocks; each following line is a reference, with its handle (starting with `'`, so that the spreadsheet leaves it alone) and `<>` where the block does not have that tag. **ATTIN** reads the edited file and replaces the values by each reference's handle, in a single undo step (references on locked layers are left out, with a message; lines whose handle is not in the drawing are counted). Reading and writing the file run in the background. Both are in the **Block** panel of the Insert tab (in the expanded panel).

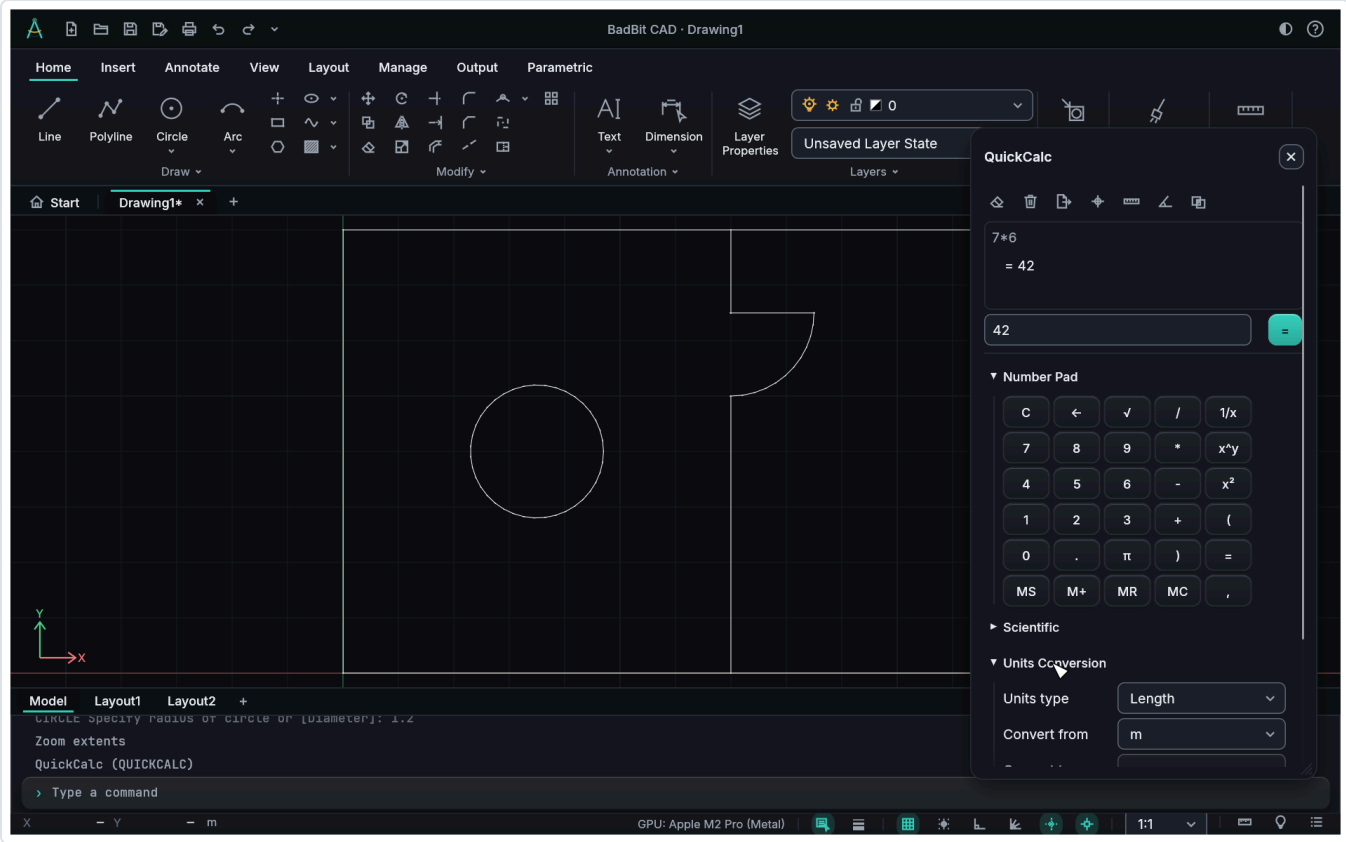
Calculator, DesignCenter and tool palettes

Three AutoCAD palettes that sit beside the drawing without blocking anything: **QuickCalc** (with **CAL** on the command line), **DesignCenter** and the **Tool Palettes**. All three open from the **Palettes** panel of the **View** tab, with the AutoCAD shortcuts and with the commands. What belongs to the user (the calculator variables, the DesignCenter favorites and the tool palettes) is stored in the **profile** and goes along when it is exported.

QuickCalc (QUICKCALC, alias QC, Ctrl+8)

QUICKCALC opens the **QuickCalc** palette (**Ctrl+8** opens and closes it; **QCCLOSE** closes it). From top to bottom:

- the **toolbar**: **Clear** (the field), **Clear History**, **Paste Value to Command Line** (the last result goes to the command line, ready for Enter) and the **get from screen** buttons: **Get Coordinates** (one point), **Distance** (two points), **Angle** (of the line through two points) and **Intersection** (of the lines through four points). The points are requested on the command line, with object snaps on;
- the **history**: each calculation and its result; clicking a result puts it in the field;
- the calculation **field**: **Enter** or **=** calculates, and the result stays in the field for the next calculation;
- **Number Pad**: the digits, the four operations, **x^y**, **x²**, **√**, **1/x**, **π**, the parentheses and the memory (**MS** stores the result, **M+** adds to it, **MR** recalls it, **MC** clears it);
- **Scientific**: **sin**, **cos**, **tang**, **asin**, **acos**, **atan** (in degrees, as in AutoCAD), **log**, **10^x**, **ln**, **e^x**, **r2d**, **d2r**, **abs**, **rnd**, **trunc**, **x²**, **x³**, **√** and **3√**: the button wraps what is in the field (`sin(30)`), or the last result;
- **Units Conversion**: the **type** (length, area, volume or angular), **Convert from**, **Convert to** and the **value**; the **Converted value** appears immediately and **Return to the input area** takes it to the field;
- **Variables**: the **constants** (a value) and the **functions** (a calculation evaluated at each use, which can get points from the screen). AutoCAD's sample ones come included: **Phi** (the golden ratio) and the functions **dee**, **ille**, **mee**, **nee**, **rad**, **vee** and **vee1**. **New**, **Edit** and **Delete** change the list (in the profile); **Return to input area** (or a double-click) puts the name in the field. A constant stores the value calculated when it is saved.



*QuickCalc after 7*6 on the number pad, with units conversion expanded.*

Inside a command that asks for a number or a point, **'QC** (or **'QUICKCALC**) opens the calculator transparently: the **Apply** button calculates and sends the value to the command (a point as `x,y`), and the palette closes.

CAL

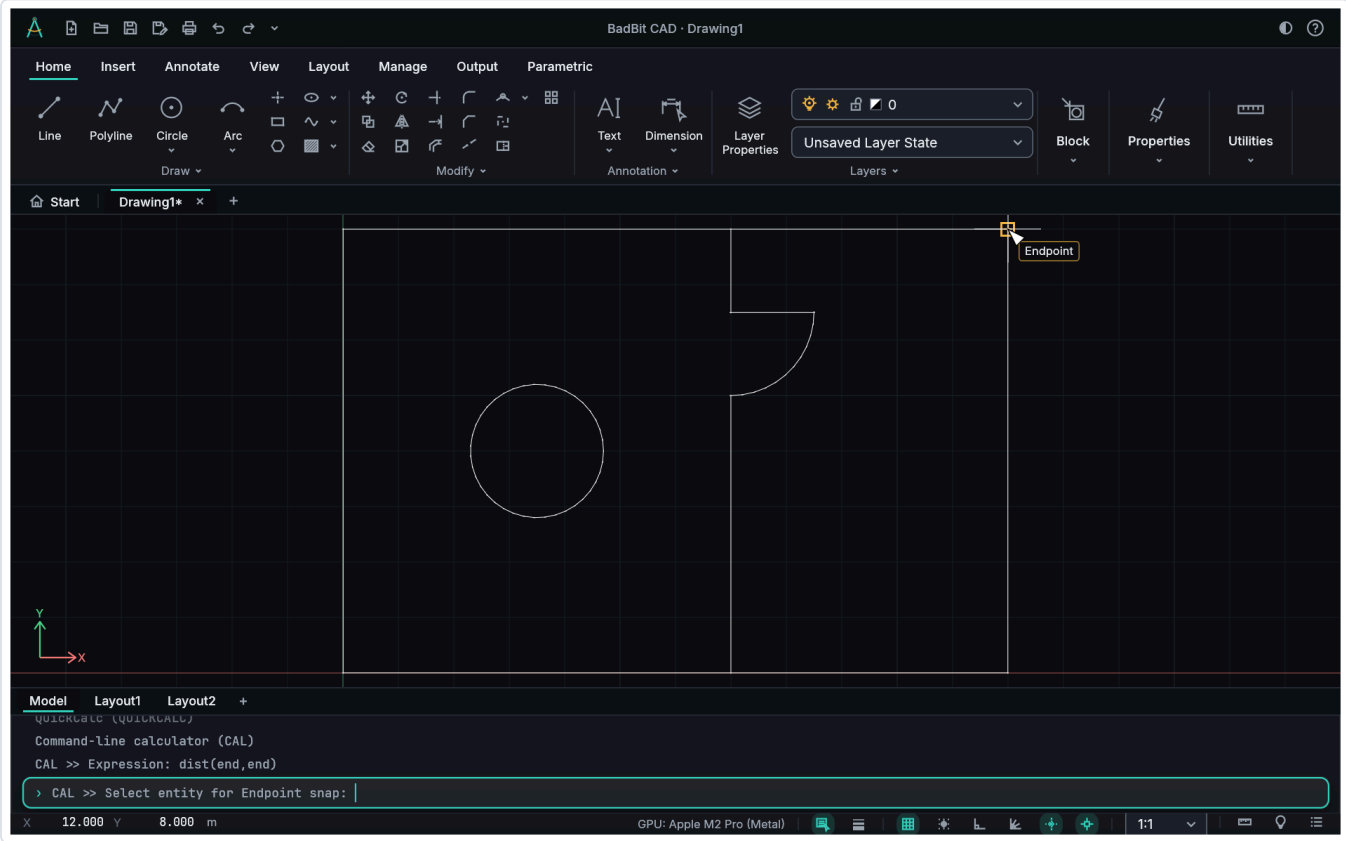
CAL asks for the expression on the command line (`>> Expression:`; spaces are part of it) and shows the result. Inside a command, **'CAL** is transparent: the result goes to the command that asked for the number or the point (`CIRCLE`, center, `'CAL`, `2*3` makes a circle of radius 6).

The expressions are those of AutoCAD's CAL:

- **numbers** with a decimal point (`2.5`, `1e-3`) and with an **attached unit**: lengths (`30cm`, `2m`, `5'6"`, `12"`), which become drawing units, and angles (`30d`, `30d15'20"`, `1.2r`, `50g`), which become degrees;
- **points and vectors** in brackets: `[x,y,z]` (what is missing is 0: `[,5]`), polar `[d<a]`, cylindrical `[d<a,z]`, spherical `[d<a<b]`, relative to the last point `[@1,2]` and the last point itself `@`;
- **operators** in CAL's order: `^` (power, right to left: `2^3^2` = 512), `*` and `/` (`*` between two vectors is the dot product), `&` (cross product), `+` and `-`, with parentheses; the minus sign comes after the power (`-2^2` = −4);
- **functions**: `sin`, `cos`, `tang`, `asin`, `acos`, `atan` (degrees), `ln`, `log`, `exp`, `exp10`, `sqr`, `sqrt`, `abs` (for a vector, its length), `round`, `trunc`, `r2d`, `d2r`, `pi` and `cvunit(value,from,to)` (`cvunit(1,inch,cm)` = 2.54);

- **geometry**: `dist(p1,p2)`, `dpl(p,p1,p2)` (point to line), `dpp(p,p1,p2,p3)` (point to plane), `ang(v)`, `ang(p1,p2)`, `ang(vertex,p1,p2)`, `pld(p1,p2,d)` (the point at `d` from `p1` on the line), `plt(p1,p2,t)` (by parameter), `rot(p,origin,angle)` and `rot(p,axis1,axis2,angle)`, `vec(p1,p2)`, `vec1(p1,p2)` (unit), `nor(v)`, `nor(p1,p2)`, `nor(p1,p2,p3)`, `ill(p1,p2,p3,p4)` (intersection of two lines), `ilp(p1,p2,p3,p4,p5)` (line and plane), `xyof`, `xzof`, `yzof`, `xof`, `yof`, `zof`, `rxof`, `ryof`, `rzof`, `w2u` and `u2w`;
- **what you get from the screen**: the object snaps `end`, `mid`, `cen`, `int`, `nea`, `nod`, `qua`, `per`, `tan` and `ins` (each asks for the point with that snap only, and rejects a loose point), `cur` (any point) and `rad` (the radius of a circle, arc or polyline arc segment); the shortcuts `dee` (distance between two endpoints), `ille`, `mee` (the midpoint), `nee`, `vee` and `vee1`; and the QuickCalc variables by name.

The points you get can be typed (`10,5`). Esc stops the calculation (the transparent one gives the prompt back to the command).



CAL with dist(end,end) asking for the first endpoint, with the snap marker.

DesignCenter (ADCENTER, aliases ADC, DC, DCENTER; Ctrl+2)

ADCENTER opens **DesignCenter** (**Ctrl+2** opens and closes it; **ADCCLOSE** closes it), with three tabs:

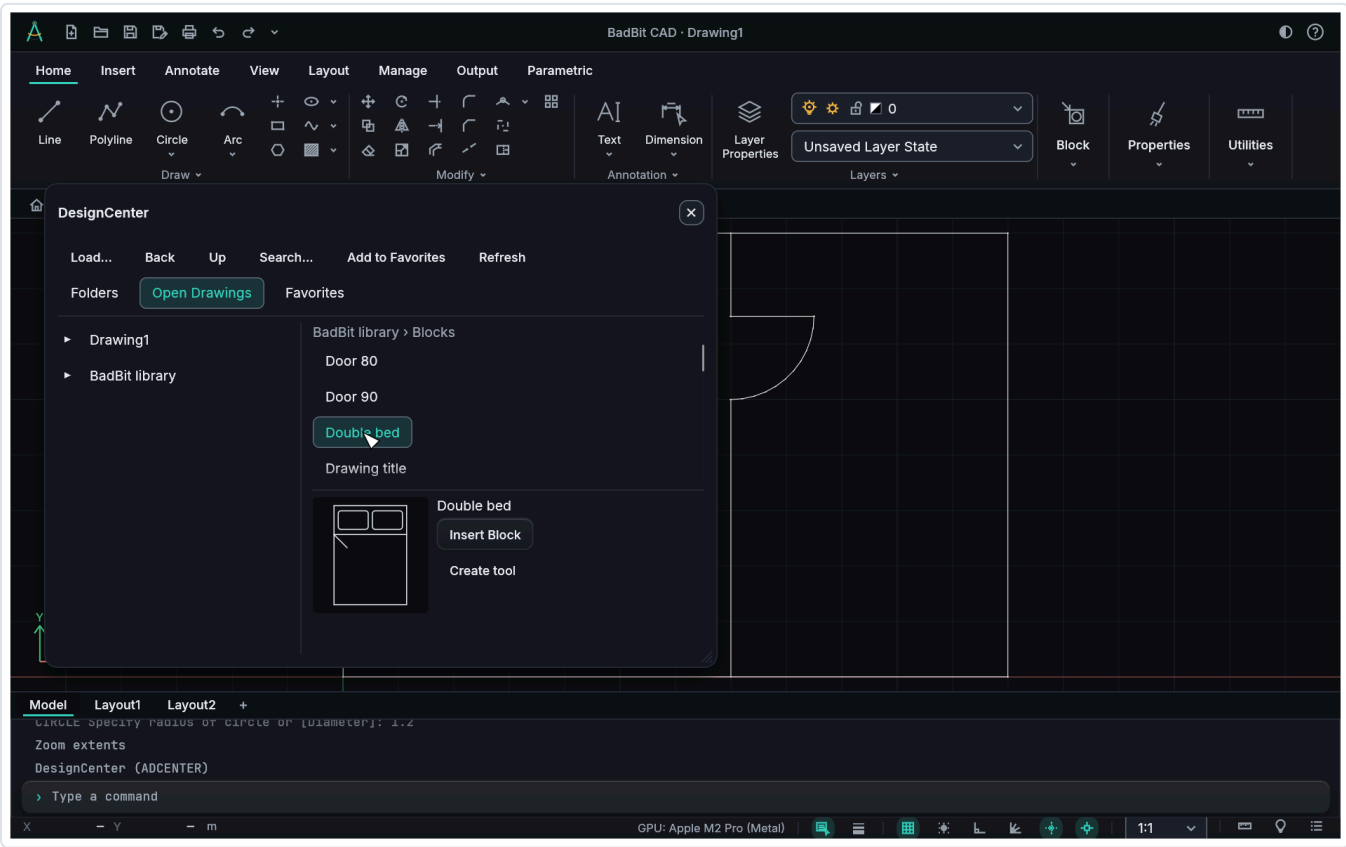
- **Folders**: the folder tree (the drawing's folder, the home folder and the root of the disk) with the **DWG**, **DXF** and **.bbd** drawings;
- **Open Drawings**: the drawings in the tabs and the **BadBit CAD library** (the blocks that come with the program);
- **Favorites**: the folders and drawings stored with **Add to Favorites** (the **×** removes them).

In the tree, each drawing opens into the content types: **Blocks**, **Layers**, **Text styles**, **Dimension styles**, **Table styles**, **Multileader styles**, **Linetypes**, **Layouts** and **Xrefs**. On the right is what the chosen place contains (double-clicking or **Explore** goes in), and below, the block **preview** and the description. The toolbar has **Load...** (a drawing through the system dialog box), **Back**, **Up**, **Search...**, **Add to Favorites** and **Refresh**. Drawings on disk are read in the background (the last eight are kept and are read again if the file changes).

To use an item: **double-click**, the button (**Insert Block**, **Add to drawing**, **Attach**) or **drag it into the drawing**:

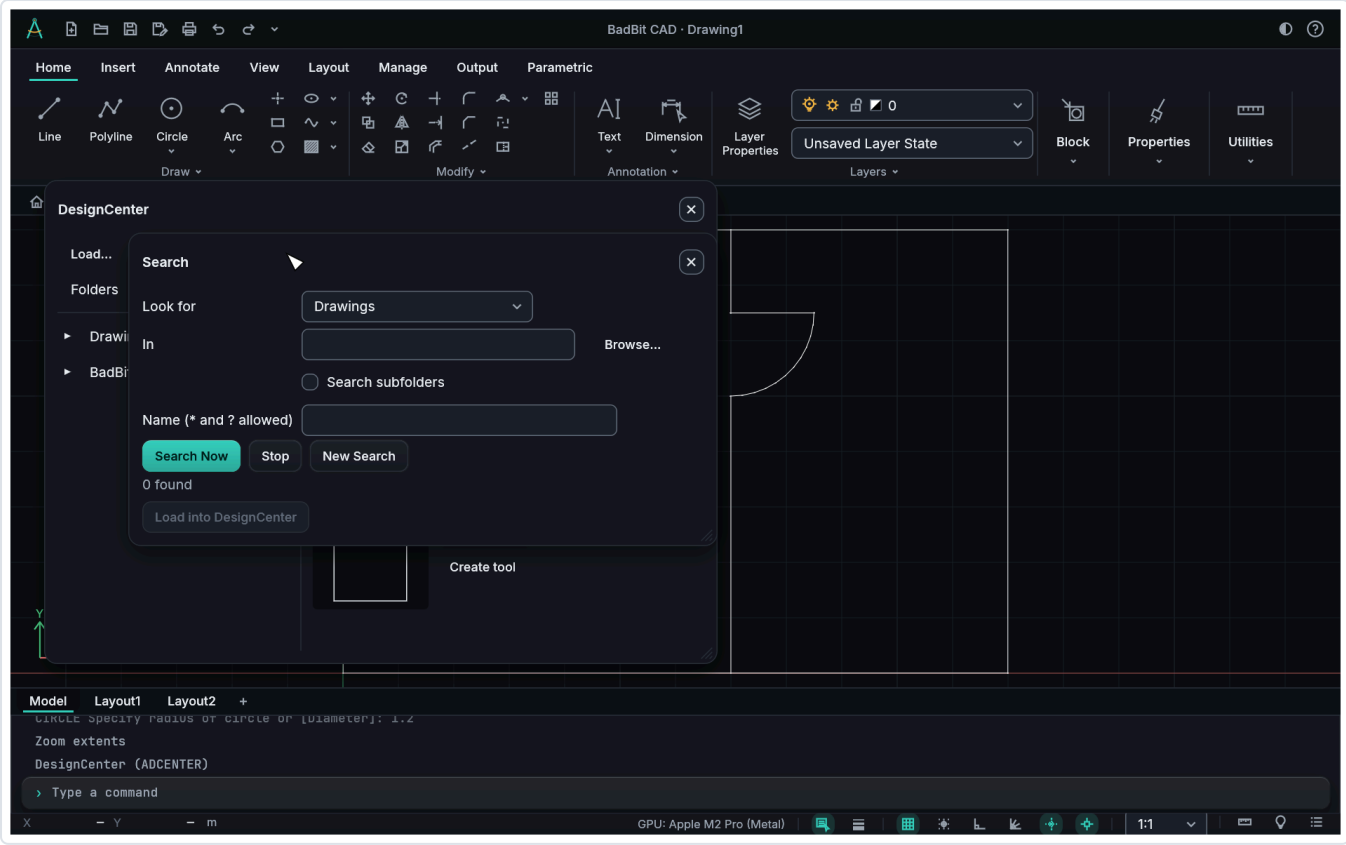
- the **block** comes into the drawing (with the layers, linetypes, styles and blocks it needs) and **INSERT** asks for the point, with the scale of the source drawing's unit; when dragged, the block follows the cursor over the drawing and comes in where it is dropped (`Esc` during the drag cancels);
- **layers**, **styles** and **linetypes** come in (the dimension style brings its text style; the layer brings its linetype);
- the **layout** comes in with what is in it (with another name if it already exists: `Planta (2)`);
- the **external reference** is attached (the path relative to the source drawing's folder).

Whatever already exists in the drawing with the same name **stays as it is**, and the command line says so (as AutoCAD does). Everything goes in as a single undo step.



The blocks of the BadBit CAD library in DesignCenter, with the preview of the double bed.

Search... opens the search: **Look for** (drawings, by file name, or a content type), **In** (the folder), **Search subfolders** and the **Name** (`*` and `?` are allowed; without them, the name contains the text). **Search Now** runs in the background (with **Stop**); **Load into DesignCenter** (or a double-click) takes you to the result.



The DesignCenter Search dialog box.

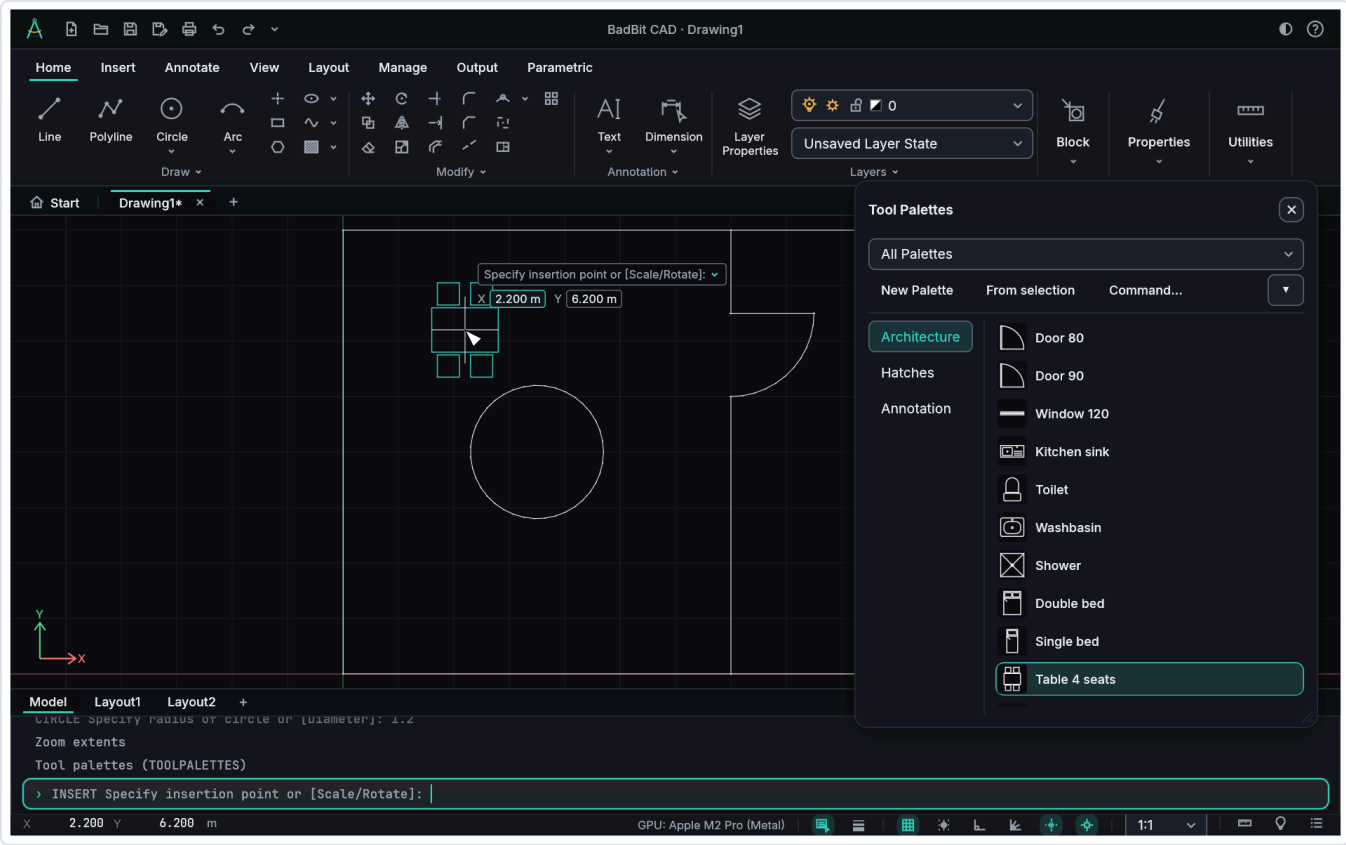
Tool palettes (TOOLPALETTES, alias TP; Ctrl+3)

TOOLPALETTES opens the **Tool Palettes** (**Ctrl+3** opens and closes them; **TOOLPALETTESCLOSE** closes them): the tabs (palettes) on the left and the tools of the open tab. Three palettes come included, with generic blocks made for BadBit CAD:

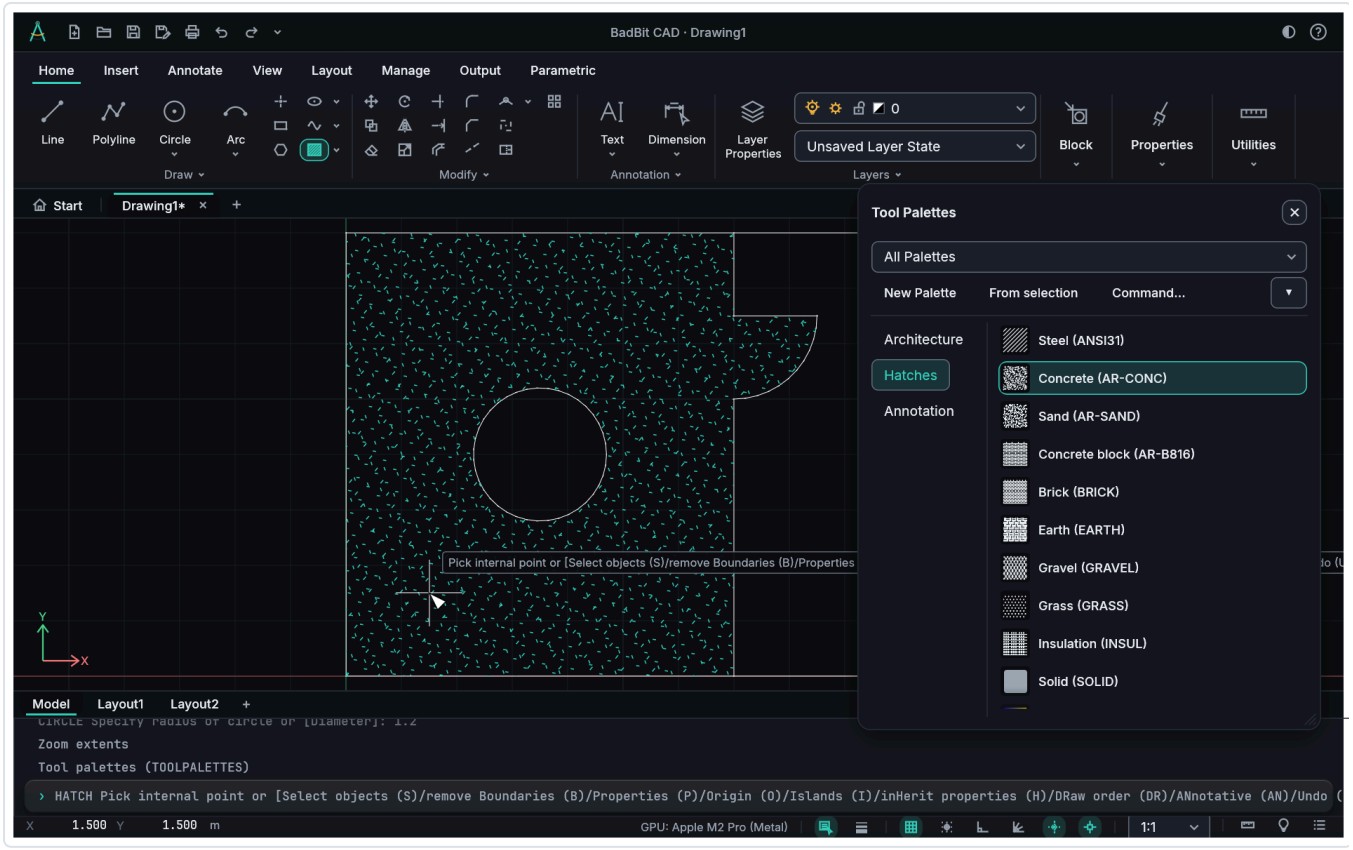
- **Architecture**: 80 and 90 doors (base point at the hinge), a 120 window in a 15 cm wall, kitchen sink, toilet, washbasin, shower, double and single beds, table with four chairs, three-seat sofa, refrigerator, four-burner stove and tree;
- **Hatches**: steel (ANSI31), concrete, sand, concrete block, brick, earth, gravel, grass, insulation, solid and three gradients;
- **Annotation**: north arrow, plan and section levels, section mark, drawing title, revision mark and the text, dimension, multileader, revision cloud and table commands.

Moving the mouse over a tool highlights its row (the icon and the name: the whole row can be clicked or dragged). **Clicking** a tool uses it, and it stays **highlighted** in the palette while its command runs (when it finishes or `Esc` cancels, the highlight goes away): a block asks for the insertion point (with the tool's scale and rotation), hatch and gradient tools start **HATCH** with their pattern, scale and angle (the Hatch Creation panel stays open for changes), and a command tool runs its command. With a hatch, each click inside a closed area adds it: it stays drawn even with the mouse outside it (or over the palette) and the prompt continues for more areas; **Enter**, **Space** or a right-click finish and create the hatch.

Dragging into the drawing uses the tool at the drop point. As in AutoCAD, as soon as the mouse enters the drawing the command starts and shows the result before you drop: the **block** follows the cursor (with the tool's scale, rotation and layer) and the **hatch** or the **gradient** appears in the closed area under the cursor, changing area as the mouse moves. Dropping inserts the block there or fills the area and finishes (that is why the Hatch Creation panel does not open when dragging); `Esc` during the drag, or dropping outside the drawing (back over the palette, for example), cancels without leaving anything. A command tool runs only when dropped. Blocks dragged from **DesignCenter** and from the **Blocks** palette also follow the cursor. In a drawing with many visible objects, the area search is built in the background: the drawing keeps responding and the hatch preview appears when it is ready.

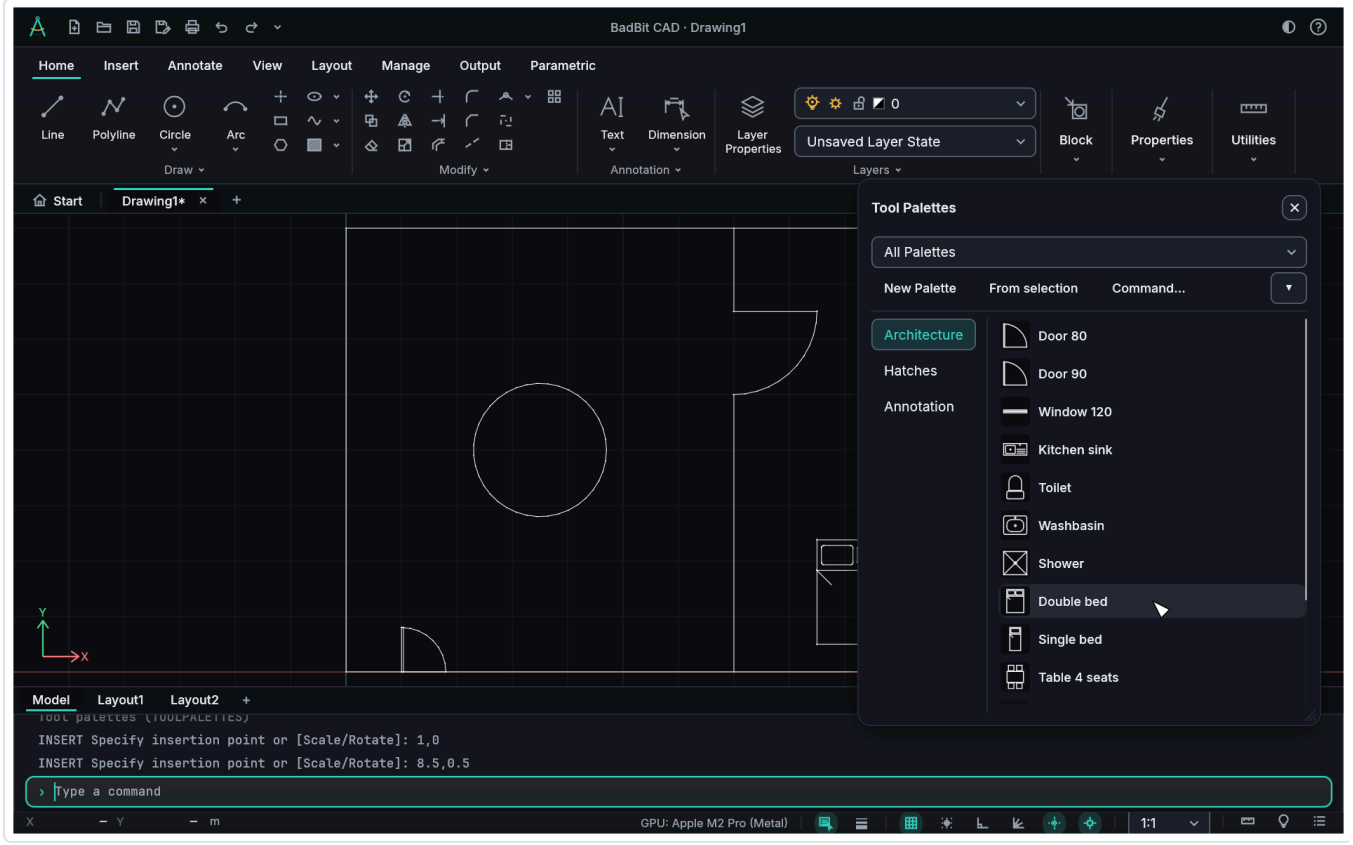


The Table 4 seats block dragged over the floor plan, not yet dropped: the block follows the cursor and the tool stays highlighted.

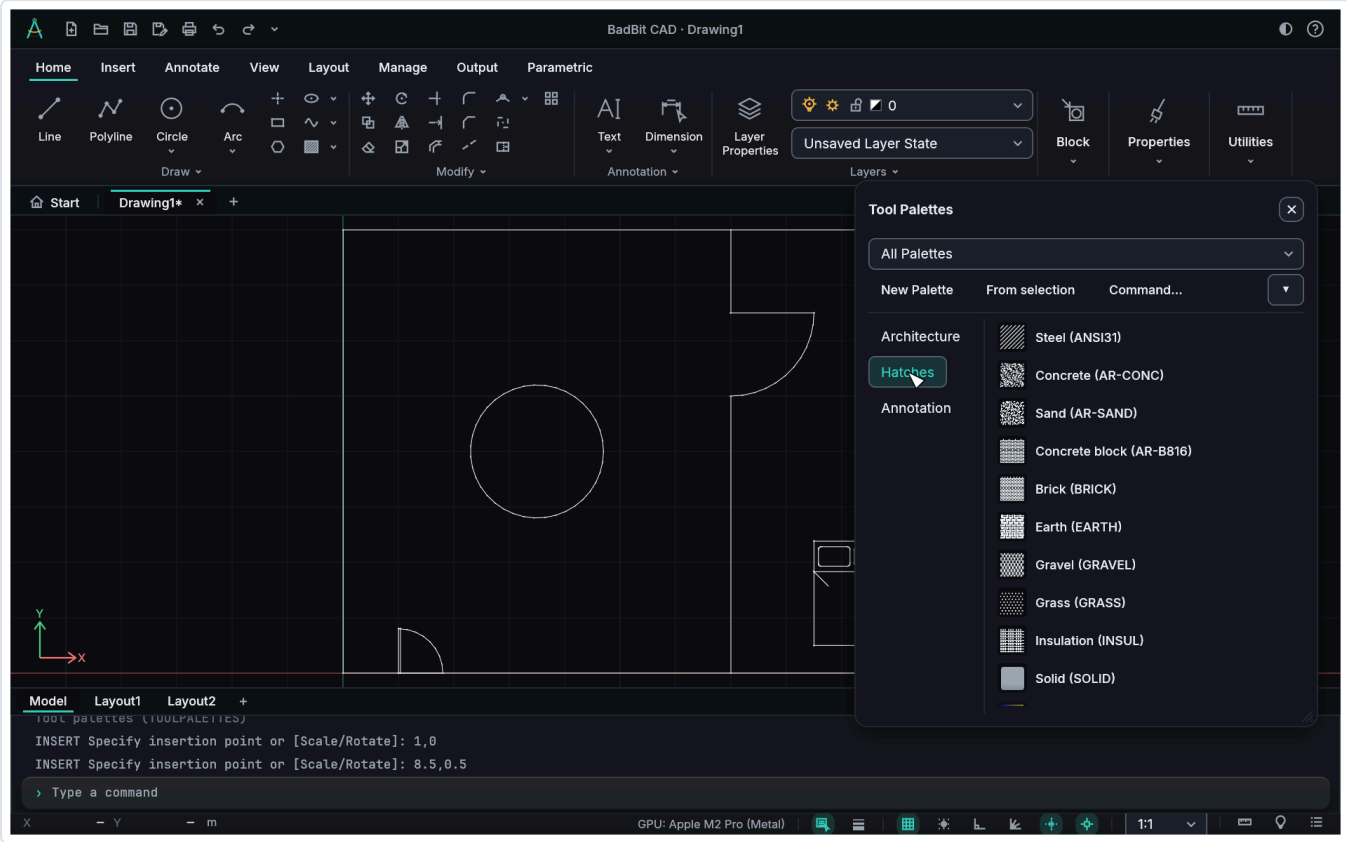


The Concrete tool dragged over the room, not yet dropped: the preview fills the closed area under the cursor (the circle is left out, as an island).

The tool's **layer**, **color** and **linetype** apply only while its command lasts (a layer that does not exist is created); afterwards the previous ones come back.

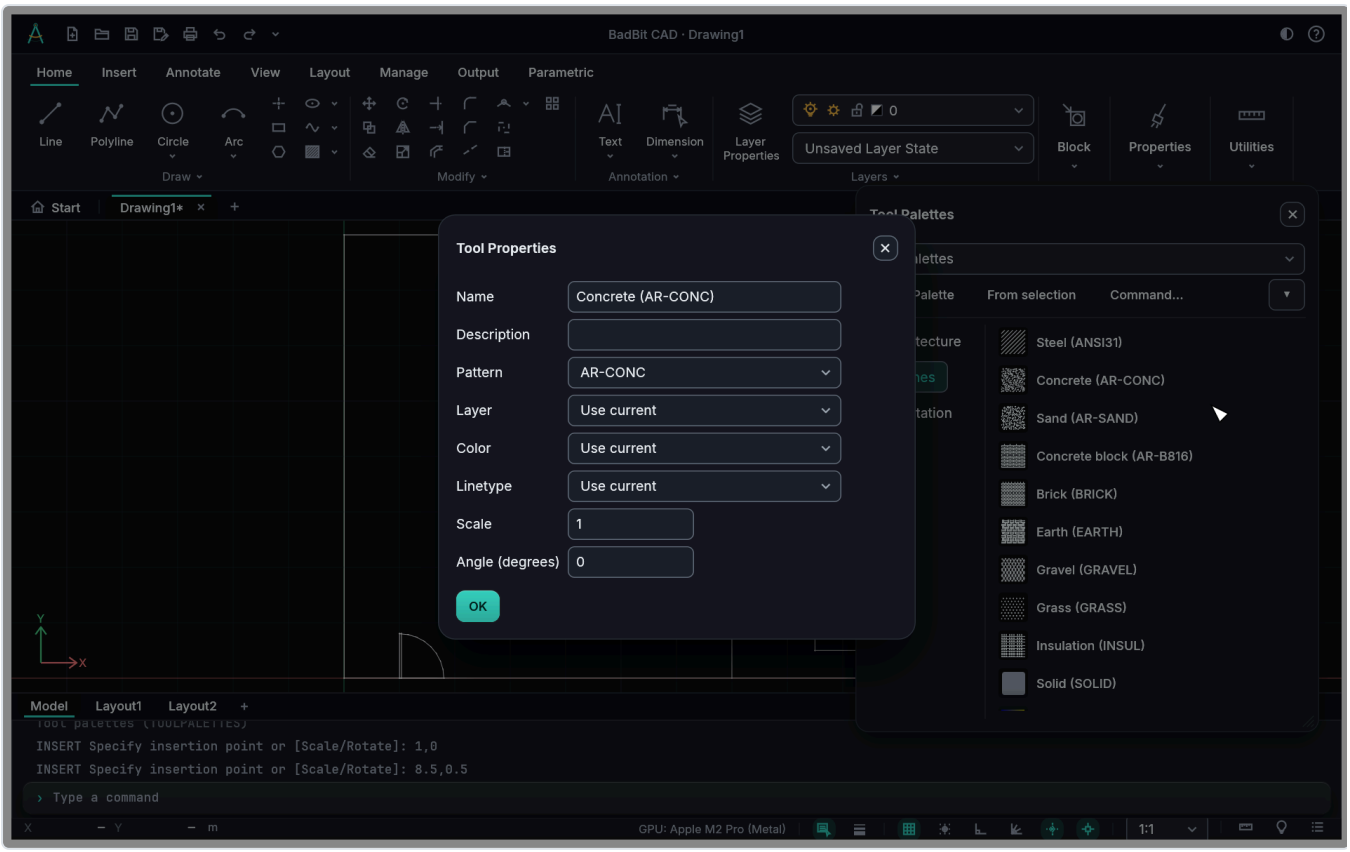


The Architecture palette, with the door and the double bed inserted in the floor plan.



The Hatches palette, with the thumbnail of each pattern.

Right-clicking a tool opens **Properties...** (name, description, the block and its source, the pattern or gradient, the command, the layer, the color, the linetype, the scale and the rotation or angle), **Move Up/Move Down** and **Delete**; tools that came with BadBit CAD also have **Restore default**, which returns them to their factory properties. `Esc` closes this menu (and the **More** menu) without canceling the command that is running, as in AutoCAD.



The properties of the Concrete tool.

To create tools:

- **drag a block from DesignCenter** onto the palette (or **Create tool** in DesignCenter);
- **drag chosen objects from the drawing** (press on one of them and drag it to the palette), or the **From selection** button: a block reference becomes a block tool, a hatch becomes a hatch or gradient tool, and other objects become the tool of the command that draws them (line, circle, polyline, text, dimension...), with the object's layer, color and linetype;
- **Command...**: the command and the answers, separated by spaces (`;` is Enter, as in AutoCAD macros).

New Palette creates a tab; the **More** menu, on the right of the same row (when the window is narrow, the buttons that do not fit go into it), holds **Rename**, **Delete Palette**, **Move Up/Move Down**, **Import...**, **Export...**, **Groups...** and **Restore default palettes**, which, after confirmation, recreates Architecture, Hatches and Annotation as they came (at the top, with their factory names) without touching the palettes you created or the groups. **Groups** gather palettes (the list at the top of the window chooses the group on display, or **All Palettes**). **Export** saves all palettes and groups to a **.bbd-palettes** file (JSON, described in `docs/formatos/paletas-de-ferramentas.md`); **Import** adds those in a file to the current ones (those with repeated names come in with another name). AutoCAD's `.xtp` file is not read: its tools point to AutoCAD files and resources.

Blocks palette (BLOCKSPALETTE)

BLOCKSPALETTE opens the AutoCAD 2026 **Blocks** palette (**BLOCKSPALETTECLOSE** closes it). It is also in the **Palettes** panel of the View tab and in the expanded **Block** panel of the Home and Insert tabs. **INSERT** still asks on the command line, like **-INSERT** (in AutoCAD 2026 INSERT opens the

palette; here it was kept as it was so as not to change what people already use).

At the top, the name **filter** (part of the name, or with `*` and `?` as in AutoCAD; the small arrow remembers the filters used), the **List style** (Large icons, Small icons or List), **Insert DWG as block...** (an entire drawing becomes a block, with the file's name) and the **Palette position**: floating or **docked** on the left or right of the drawing area. The tabs:

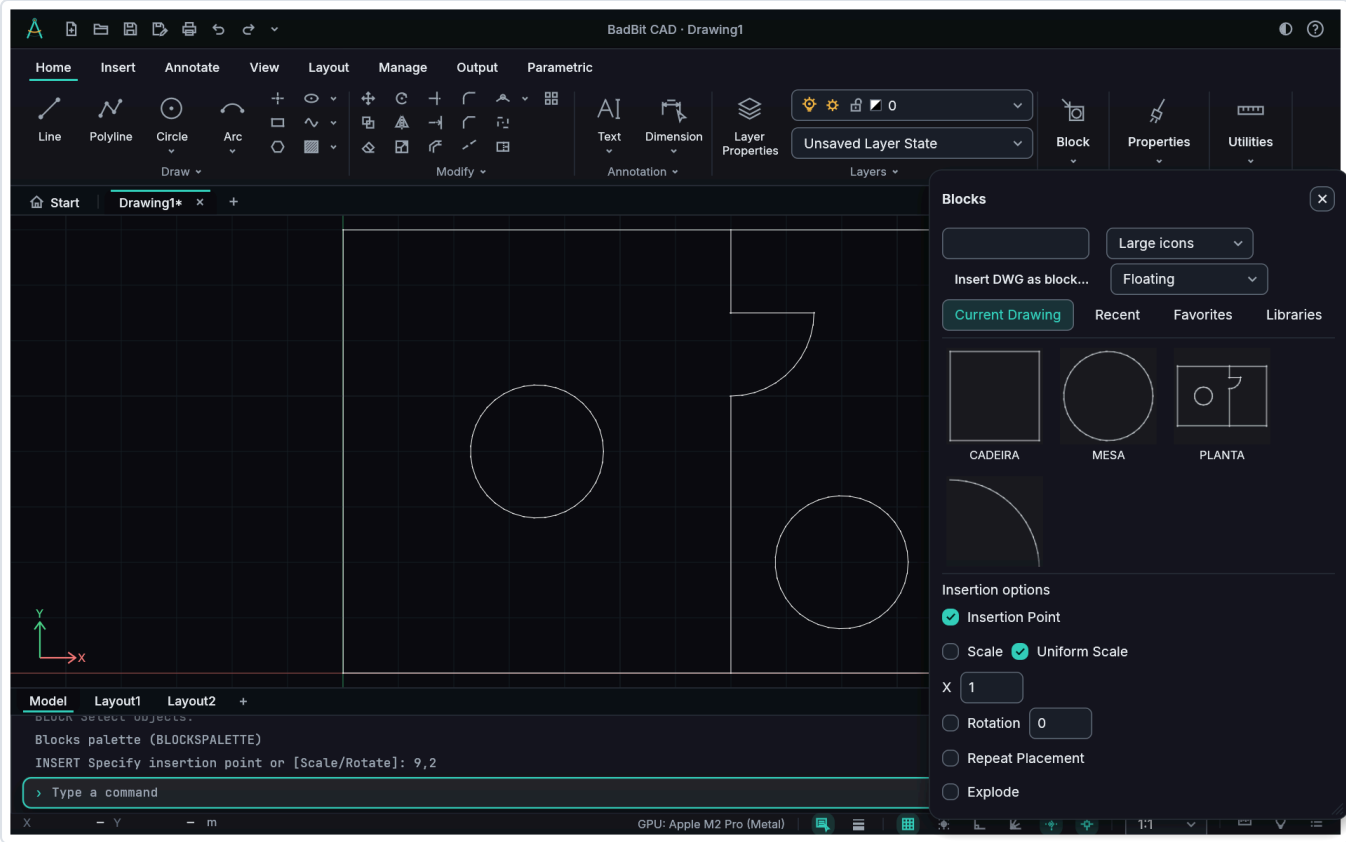
- **Current Drawing**: the blocks in the drawing, with thumbnails (dynamic ones have a lightning bolt in the corner).
- **Recent**: the blocks inserted or created recently (from the palette or the command line), from any drawing, up to 50; the list is stored in the profile and comes back next time. Right-clicking has **Remove from Recent**.
- **Favorites**: the drawings in a **local folder** (the default one sits next to the profiles folder; **Favorites folder...** changes it). Right-clicking a block in any tab has **Copy to Favorites**, which saves the block there as a drawing; **Remove from Favorites** deletes that copy. Nothing goes to the cloud.
- **Libraries**: the blocks of a drawing (**Browse block libraries...**) or the drawings in a folder (**Library folder...**; double-click a subfolder to go in and **Up** to go back). The libraries used stay in the list, in the profile.

The **thumbnails** are drawn by the CPU off the interface thread (the drawing keeps responding while they arrive), like those on the start screen; the thumbnail of a drawing on disk is the preview the file carries.

Clicking a block inserts it with the **insertion options** at the bottom: what is **checked is asked on screen**; what is unchecked uses the field's value.

- **Insertion Point**: unchecked, the block comes in at the last point used by the palette.
- **Scale** (with **Uniform Scale** or different X and Y; negative mirrors) and **Rotation** (degrees, from the UCS X axis).
- **Repeat Placement**: INSERT starts again after each block placed, until `Esc`.
- **Explode**: the block comes in exploded (only with uniform scale), in a single undo step.

Dragging a block into the drawing inserts it at the drop point (while dragging, the block follows the cursor; `Esc` cancels), with scale 1 and rotation 0 (the options do not apply to dragging, as in AutoCAD). A block from another drawing comes into the current drawing right away (if the drawing already has a block with the same name, the one already here is used), with the scale of the file's units, as in INSERT. The tabs, the style, the position, the options, the Favorites folder, the libraries and Recent are stored in the **profile**.



The Blocks palette on the Current Drawing tab, with the thumbnails and the table inserted from the palette.

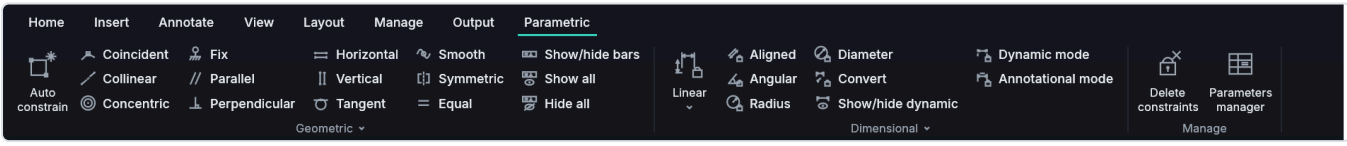
Auto-placement and synchronization with the Autodesk cloud do not exist in BadBit CAD.

Parametric drawing

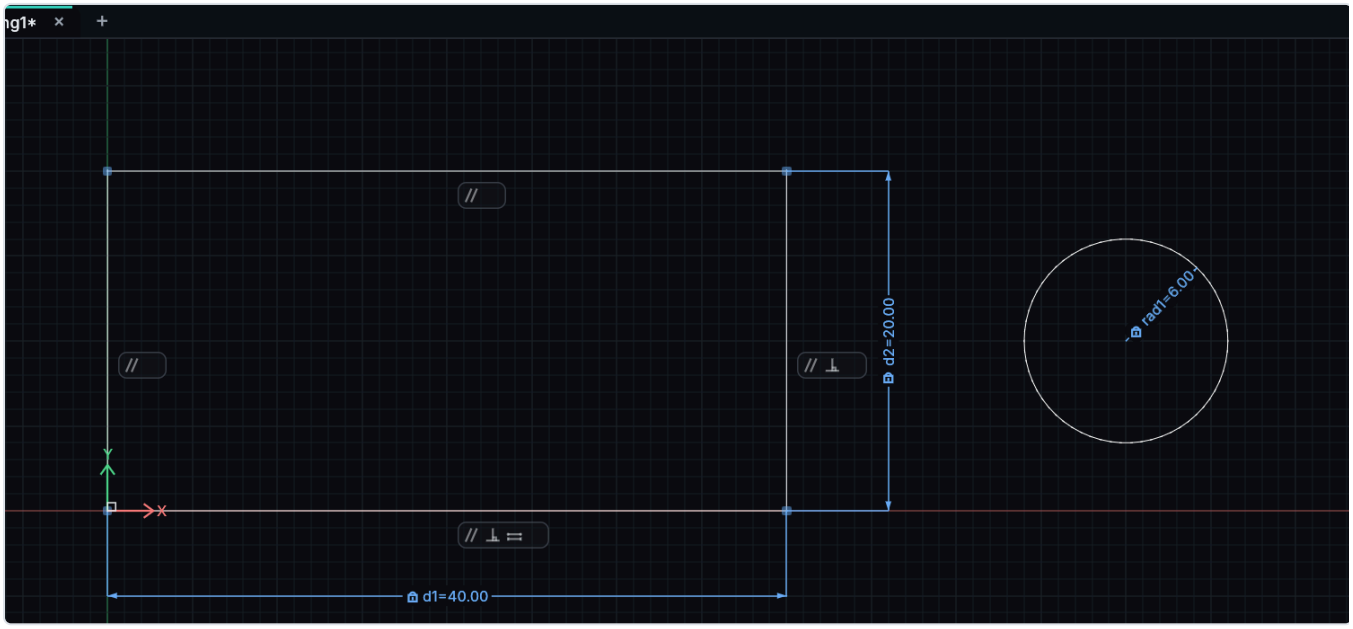
Parametric drawing ties relationships and measurements to the geometry, as in AutoCAD: two lines stay perpendicular, a side stays at 40, the door width is half the wall. Whoever changes a constrained object sees the connected objects adjust along with it. There are two kinds of constraint:

- **geometric** constraints hold relationships (horizontal, parallel, tangent, coincident…) and appear as icons in the **constraint bars** near each object;
- **dimensional** constraints hold measurements (distance, angle, radius, diameter) with a **name** and an **expression** (`d1 = 40`, `d2 = d1/2`) and appear as dimensions with a lock.

The buttons are on the **Parametric** tab of the ribbon, in the **Geometric**, **Dimensional** and **Manage** panels.



The Parametric tab: the geometric constraints, the dimensional ones and the manager.



A rectangle and a circle after AUTOCONSTRAIN, with the dynamic dimensions d1, d2 and rad1. The small squares at the corners are the coincident constraints.

Geometric constraints: GEOMCONSTRAINT (alias GCON)

GEOMCONSTRAINT asks for the type and then the objects; each type also has its own command (the buttons in the Geometric panel). The type prompt shows the last type used between `<` and `>` (`<Coincident>` the first time): `Enter` repeats that type. The **second** object adjusts to the **first**, which stays where it is. A constraint that would over-constrain the geometry or that conflicts with existing ones is not applied, and the command line says why.

Type (option)	Command	What it holds	Options
Horizontal (`H`)	GCHORIZONTAL	A line (or two points) horizontal	**2Points** (`2P`)
Vertical (`V`)	GCVERTICAL	A line (or two points) vertical	**2Points** (`2P`)
Perpendicular (`P`)	GCPERPENDICULAR	Two lines at 90° (the second one rotates)	—
Parallel (`PA`)	GCPARALLEL	Two parallel lines (the second one rotates)	—
Tangent (`T`)	GCTANGENT	Two tangent curves (line, arc, circle)	—
Smooth (`SM`)	GCSMOOTH	The end of a spline with another curve, with no corner and no jump in curvature	—
Coincident (`C`)	GCCOINCIDENT	Two points in the same place, or a point on an object	**Object** (`O`), **Autoconstrain** (`A`)
Concentric (`CON`)	GCCONCENTRIC	Circles, arcs and ellipses with the same center	—
Collinear (`COL`)	GCCOLLINEAR	Lines on the same straight line	**Multiple** (`M`)
Symmetric (`S`)	GCSYMMETRIC	Two objects or two points symmetric about a line	**2Points** (`2P`)
Equal (`E`)	GCEQUAL	The same length (lines) or the same radius (circles and arcs)	**Multiple** (`M`)
Fix (`F`)	GCFIX	A point or an object held where it is (for a line or arc, its infinite line or circle; the endpoints slide along it)	**Object** (`O`)

Points are chosen by clicking near an endpoint, a midpoint, a center or a vertex (the constraint point closest to the cursor). Polyline segments count as lines and arcs, and the **midpoint of each segment** is also a constraint point. On an elliptical arc, the two **arc endpoints** are constraint points that follow the ellipse when it rotates or changes size. A point object (POINT) counts as a point.

Ellipse axes and texts. As in AutoCAD, where the constraint asks for a line (Horizontal, Vertical, Parallel, Perpendicular, Collinear and the axis of Symmetric), clicking an ellipse picks the ellipse **axis** closest to the click (the major or the minor one, whole, from end to end): the ellipse then moves and rotates with the line that the axis is tied to. A single-line **text** (TEXT) counts as its **baseline** (click on the letters) and as its **text point** (the justification point; for Aligned and Fit, the second point). The text does not change size: when the line its baseline is tied to rotates, it rotates around its own point; when the object its point is tied to moves, it goes along without rotating.

AUTOCONSTRAIN and CONSTRAINTINFER

AUTOCONSTRAIN applies to the chosen objects the constraints that the geometry already nearly satisfies: corners that touch become coincident, nearly horizontal lines become horizontal, and so on. As in AutoCAD, **ellipse axes** and **text baselines** count as lines (a major axis on the same straight line

as a line becomes collinear; a text aligned with a line, parallel to it), constraints inside each **polyline** (between its segments) come before those between objects, and **Equal** is turned off (AutoCAD's AUTOCONSTRAIN does not tie lines or circles of the same size; turn it on in the tab if you want). Like AutoCAD's, it accepts a parallel between two lines that are already held only by the two axes of the same ellipse (which are perpendicular to each other). The types, the priority order and the tolerances are on the **AutoConstrain** tab of Constraint Settings (the command's **Settings** option, `S`, opens that tab).

AUTOCONSTRAIN handles selections of **thousands of objects**: objects that touch are found through a spatial index, and lines with the same direction (parallel, collinear) or the same length (equal) are tied in sequence, without comparing every one with every other one. Each group of connected objects is checked separately. Between groups that do not touch (two separate rectangles, for example), parallel, collinear and equal constraints are only added while the resulting group stays small (up to 32 objects), so that the drawing does not turn into a single system that is slow to solve.

With **CONSTRAINTINFER** on (1), commands that create objects (LINE, PLINE, CIRCLE, ARC, RECTANG, copies…) get the constraints the drawing suggests, in the same undo step: coincident where the click came from a point snap and between segments drawn in sequence, perpendicular and tangent from the snaps of those names, **parallel** when the point came from the **Parallel** snap (parallel to the acquired line), horizontal and vertical on straight segments drawn with Ortho or with **polar tracking** (F10) at 0° or 90°.

Dimensional constraints: DIMCONSTRAINT (alias DCON)

DIMCONSTRAINT asks for the type (or an associative dimension to convert); each type also has its own command:

Type (option)	Command	Measures
Linear (`LI`)	DCLINEAR	The horizontal or vertical distance between two points, depending on where the dimension is placed
Horizontal (`H`)	DCHORIZONTAL	The horizontal distance
Vertical (`V`)	DCVERTICAL	The vertical distance
Aligned (`A` , the default)	DCALIGNED	The distance between two points, a point and a line (**Point & line**, `P`) or two lines (**2Lines**, `2L` : the second becomes parallel to the first)
Angular (`AN`)	DCANGULAR	The angle between two lines, the sweep of an arc or the angle by **3Point** (`3P` , the `Enter` default: the vertex and the two points)
Radius (`R`)	DCRADIUS	The radius of a circle or arc
Diameter (`D`)	DCDIAMETER	The diameter of a circle or arc
Convert (`C`)	DCCONVERT	Turns associative dimensions into constraints
Form (`F`)	DCFORM	Chooses the form of the next constraints

After the points (or **Object**, `O`) and the dimension line location, the command line shows the dimension text: `Enter` accepts the current measurement, or type `name=expression` (`largura=40`) or just the expression.

Aligned: Point & line and 2Lines. At the first point prompt of DCALIGNED (or of DIMCONSTRAINT's Aligned), `P` asks for a point and then the line (or, with `Enter` or `L`, the line and then the point): the dimension measures the distance from the point to the line's infinite line. `2L` asks for two lines: the second becomes parallel to the first, at the dimension's distance.

Angular by 3 points. In DCANGULAR (and in DIMCONSTRAINT's Angular), `Enter` (or `3P`) at the first line prompt asks for the **angle vertex** and the **two points**; the dimension measures the angle at the vertex, from the first to the second point (the smaller of the two sides), with the extension lines from the vertex to the dimension arc.

Names. Each dimensional constraint is a named **parameter**: `d1`, `d2`... for distances, `ang1` for angles, `rad1` for radii and `dia1` for diameters. The name is not case-sensitive and must be unique among the parameters (letters, digits and `_`, starting with a letter).

Expressions. The expression can reference other parameters and use `+ - * / ^`, parentheses, the constants `pi` and `e` and the functions `sin`, `cos`, `tan`, `asin`, `acos`, `atan`, `sqrt`, `abs`, `round`, `trunc`, `floor`, `ceil`, `ln`, `log`, `exp`, `exp10`, `pow`, `min`, `max`, `sign`, `d2r` and `r2d`. The decimal separator is the period; the comma separates arguments. Angles are in **degrees** (trigonometric functions too); a number can carry `deg`, `d` or `°` (degrees), `rad` or `r` (radians) and `grad` or `g` (grads): `30d` is 30°. A cycle (`d1 = d2`, `d2 = d1`) or an unknown name is rejected.

Units in lengths. Lengths are in the drawing's unit, and a number can carry a unit: `mm`, `cm`, `dm`, `m`, `km`, `in`, `ft`, `yd` and `mi`. The unit is converted to the drawing's: in a BadBit CAD drawing (in meters), `250mm` is 0.25; in a DXF or DWG in millimeters (its unit, INSUNITS), `2.5m` is 2500. In a **unitless** drawing, a length with a unit is rejected (there is nothing to convert to).

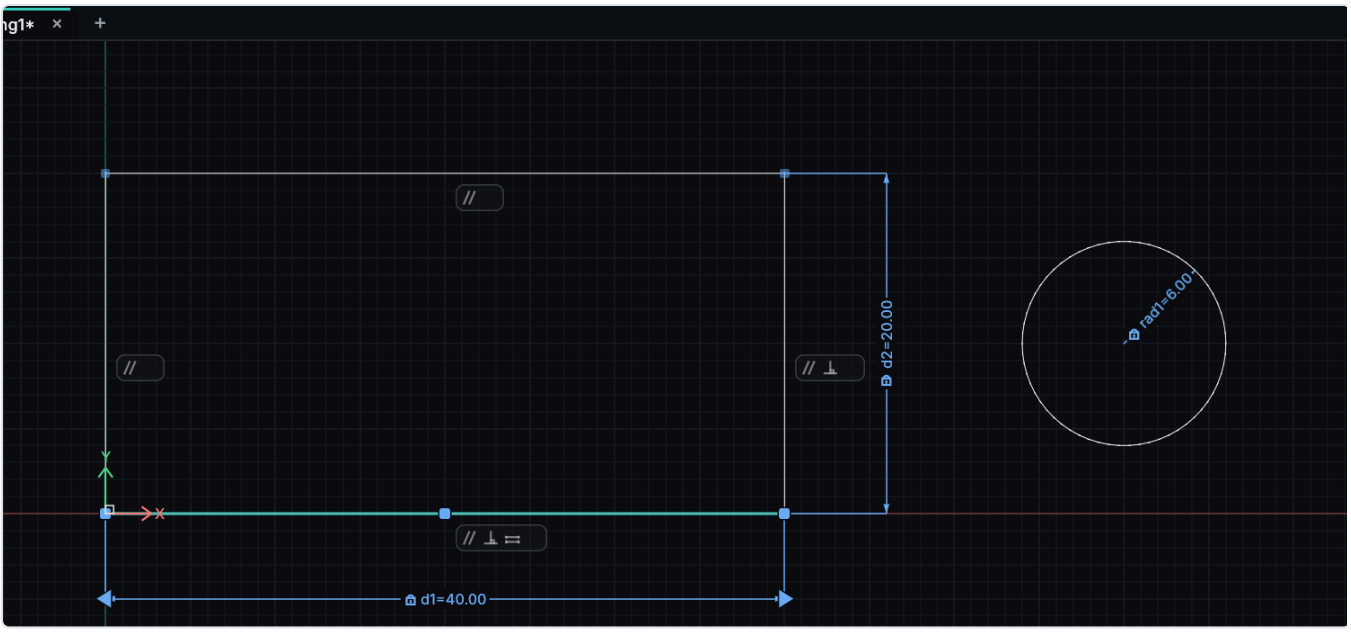
Forms. A **dynamic** constraint (the default, CCONSTRAINTFORM 0) appears only on screen, at a fixed screen size, and does not plot; the value in its text uses the language's decimal separator (`d1=40.00` in English, `d1=40,00` in Portuguese), like the Value column of the Parameters Manager. An **annotational** one (1) is a real dimension, with the current dimension style, and plots. The **Dynamic mode** and **Annotational mode** buttons in the Dimensional panel switch the form of the next ones.

Reference dimensions. When the constraint would over-constrain the geometry, AutoCAD and BadBit CAD offer to create it as a **reference dimension** (`CR`): it measures and follows the geometry, but does not drive it. The text comes in parentheses and has no lock.

DCDISPLAY shows (`S`) or hides (`H`) the dynamic constraints of the chosen objects; DYNCONSTRAINTDISPLAY hides all of them at once.

Value grips

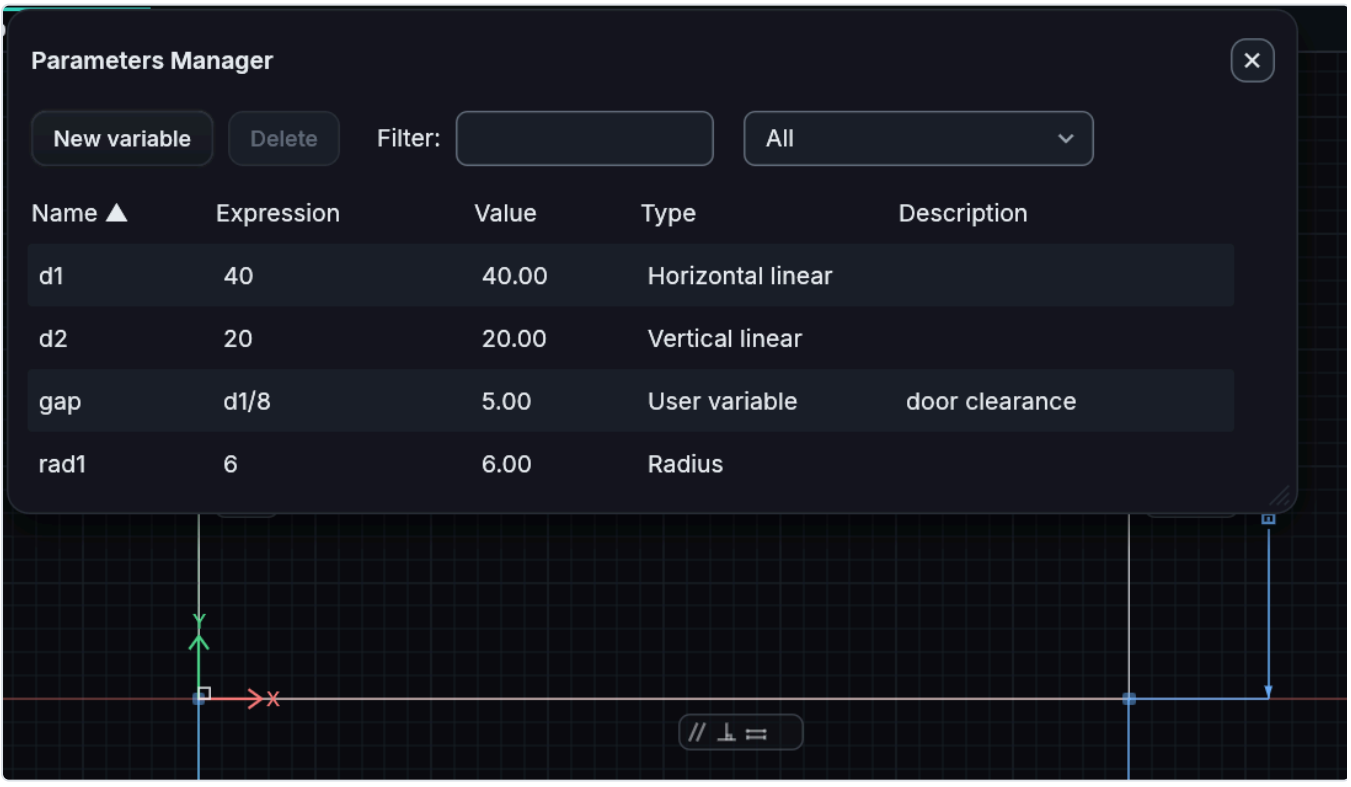
With a constrained object selected, the ends of the constraint dimension get **triangular grips**. Dragging one of them changes the constraint's value (the geometry goes along); typing a number at that moment uses that number.



The bottom line selected: the triangles at the ends of dimension d1 change the measurement.

Parameters Manager: PARAMETERS (aliases PAR, PARAM)

PARAMETERS opens the palette with all the drawing's parameters: the dimensional constraints and the **user variables**. The palette does not block the drawing.



The dimensions d1, d2 and rad1 and the variable gap, with the expression, value, type and description.

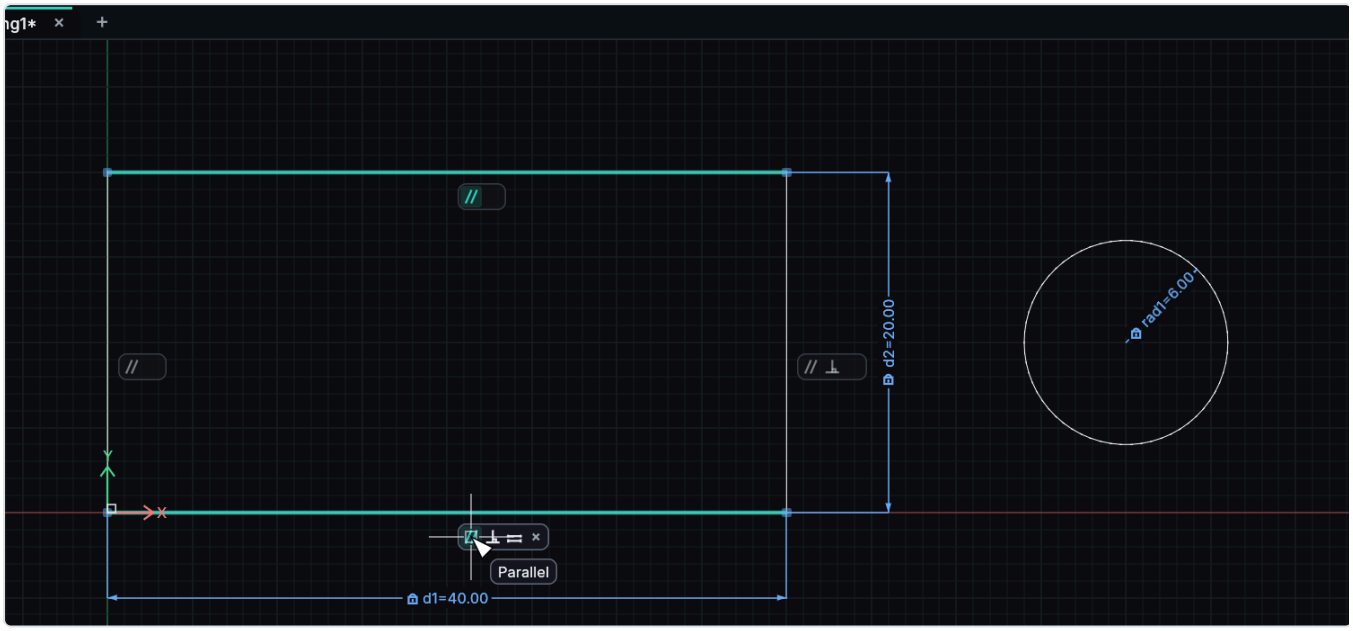
- **In-cell editing.** Click the name, the expression or (for variables) the description to edit; `Enter` confirms, `Esc` cancels. Renaming updates the expressions that reference the parameter. Changing a constraint's expression changes the geometry immediately. An error (invalid or

duplicate name, invalid expression, cycle, geometry with no solution) appears in the palette and nothing changes.
- **New variable** creates `user1 = 1` with the name already in edit mode.
- **Delete** erases the selected parameter. A parameter referenced in another one's expression is not removed (the palette says which).
- **Filter** by text (in the name, the expression or the description) and by group: **All**, **Dimensional constraints**, **User variables** and **Used in expressions**.
- **Sort**: click a column header; another click reverses the order.

Each change is an undo step. **-PARAMETERS** does the same from the command line: **New** (`N`), **Edit** (`E`), **Rename** (`R`), **Delete** (`D`) and list (`?`, accepts `*` and `?` in the name).

Constraint bars: CONSTRAINTBAR (alias CBAR)

Each object with geometric constraints shows, near it, a small semi-transparent bar with the icon of each constraint; coincident constraints appear as a **small square** at the point. Hovering over an icon highlights the objects tied by that constraint (and its icon in the other bars) and shows its name.

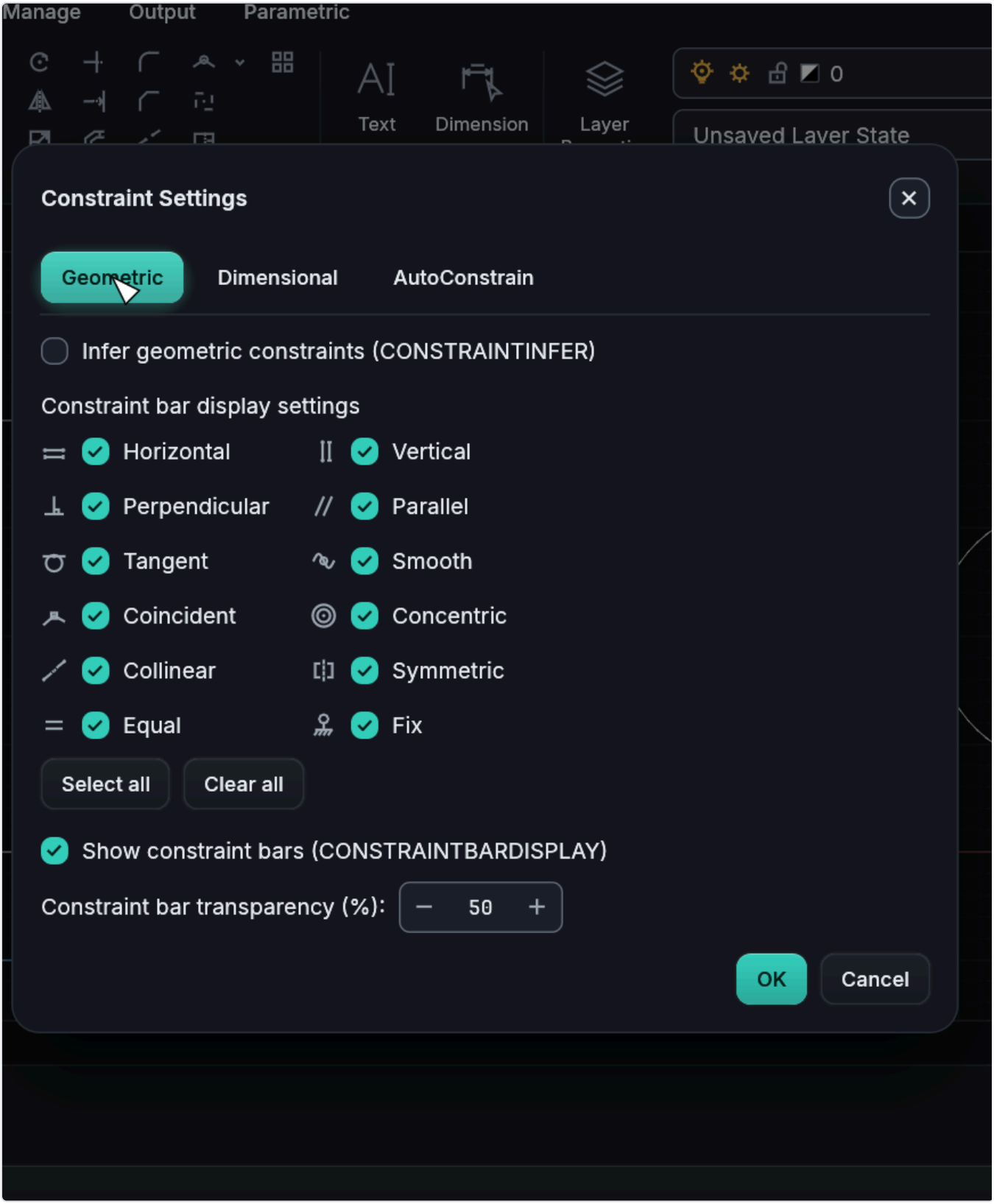


The pointer over an icon: the constraint's objects highlighted and its name.

- **Dragging the bar** (click it and drag) puts it somewhere else; it stays at that distance from the object, also when zooming and when the object moves.
- The **x** that appears at the end of the bar when the mouse is over it hides it.
- **Right-clicking** an icon opens the menu: **Delete this constraint**, **Hide this bar**, **Hide all bars** and **Constraint bar settings...**.
- **CONSTRAINTBAR** shows (`S`), hides (`H`) or resets (`R`: shows the bar and returns a dragged bar to its automatic position, near the object) the bars of the chosen objects; with no objects, **Showall** (`S`, the default) and **Hideall** (`H`). The **Show all** and **Hide all** buttons in the Geometric panel do the same.
- **DELCONSTRAINT** (alias DELCON) removes all constraints from the chosen objects. Expressions that referenced a removed dimensional constraint take its value.

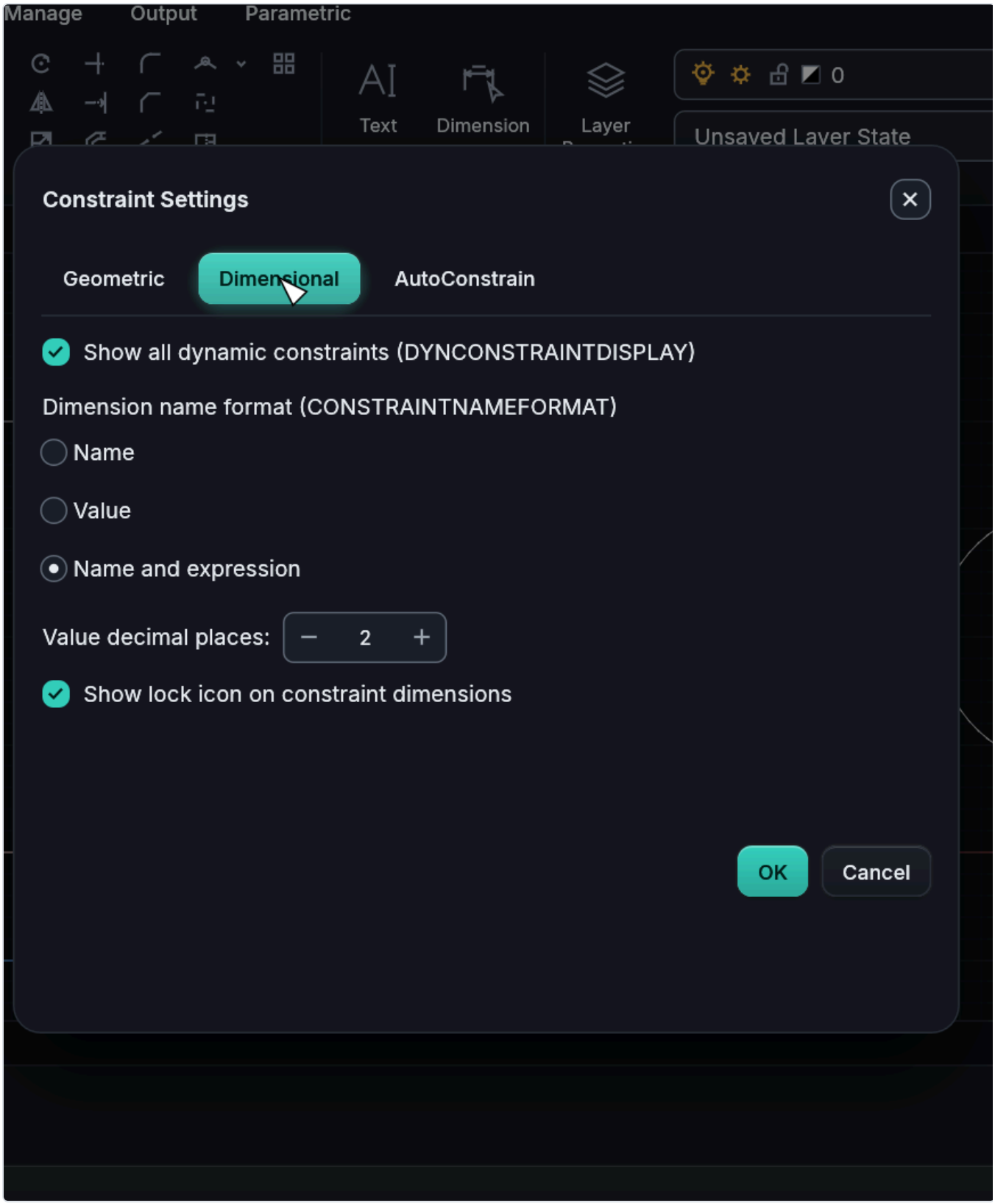
Constraint Settings: CONSTRAINTSETTINGS (alias CSETTINGS)

The dialog box has AutoCAD's three tabs; **OK** saves, **Cancel** discards.

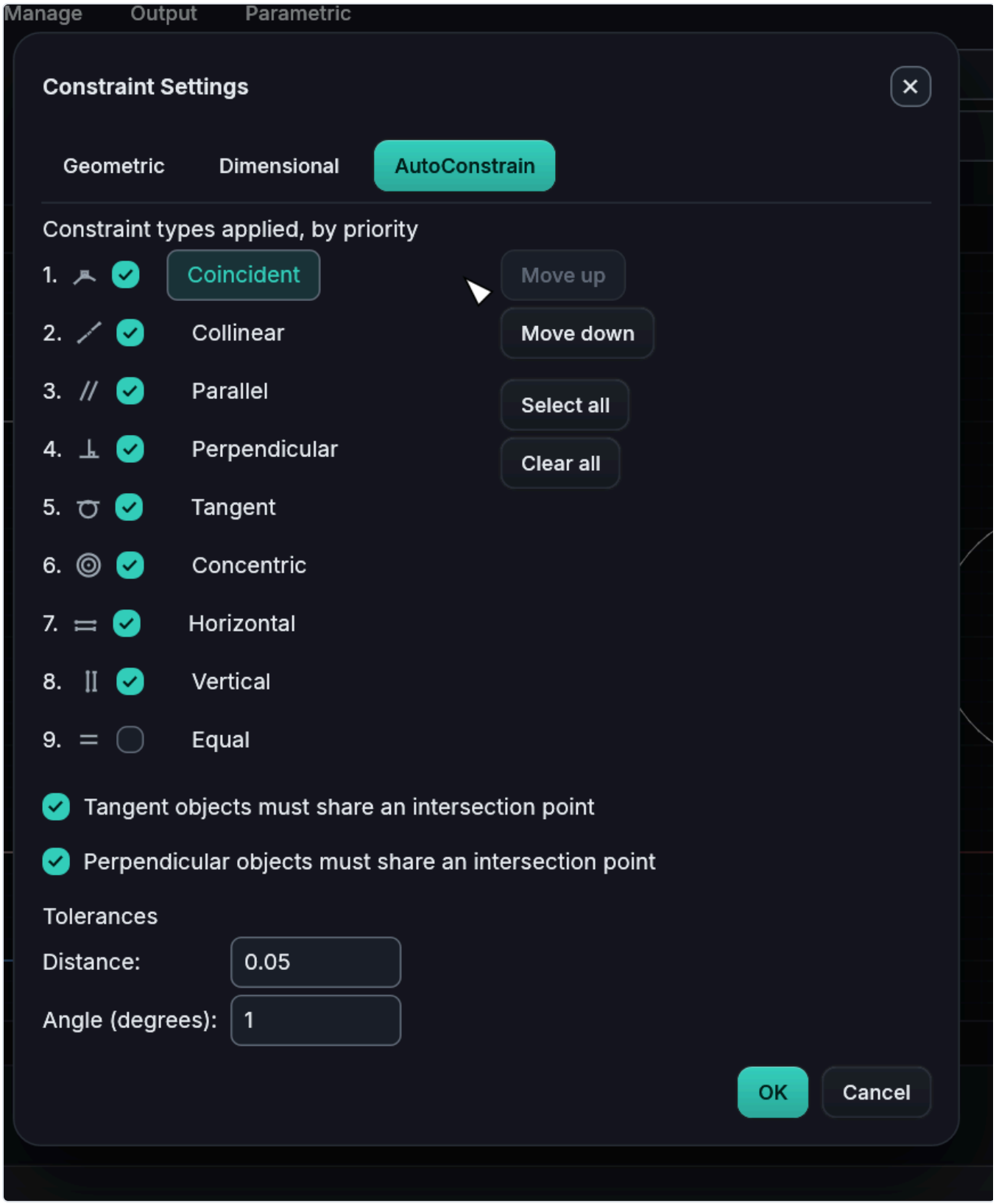


Geometric tab: infer constraints, the types shown in the bars and the transparency.

- **Geometric**: infer constraints (CONSTRAINTINFER); the **types shown in the bars** (CONSTRAINTBARMODE), with **Select all** and **Clear all**; show the bars (CONSTRAINTBARDISPLAY); the bars' **transparency** (0 to 90%).
- **Dimensional**: show all dynamic constraints (DYNCONSTRAINTDISPLAY); the dimensions' **name format** (CONSTRAINTNAMEFORMAT): **Name** (`d1`), **Value** (`40.00`) or **Name and expression** (`d1=40.00`, the default); the value's **decimal places**; and the **lock** on constraint dimensions (this choice goes to the user profile).
- **AutoConstrain**: the types AUTOCONSTRAIN applies, in **priority order** (select one and use **Move up** and **Move down**; the check box turns the type on or off); **Tangent objects must share an intersection point**; **Perpendicular objects must share an intersection point**; and the distance (in the drawing's unit) and angle (in degrees) **tolerances**.



Dimensional tab: the name format and the lock.

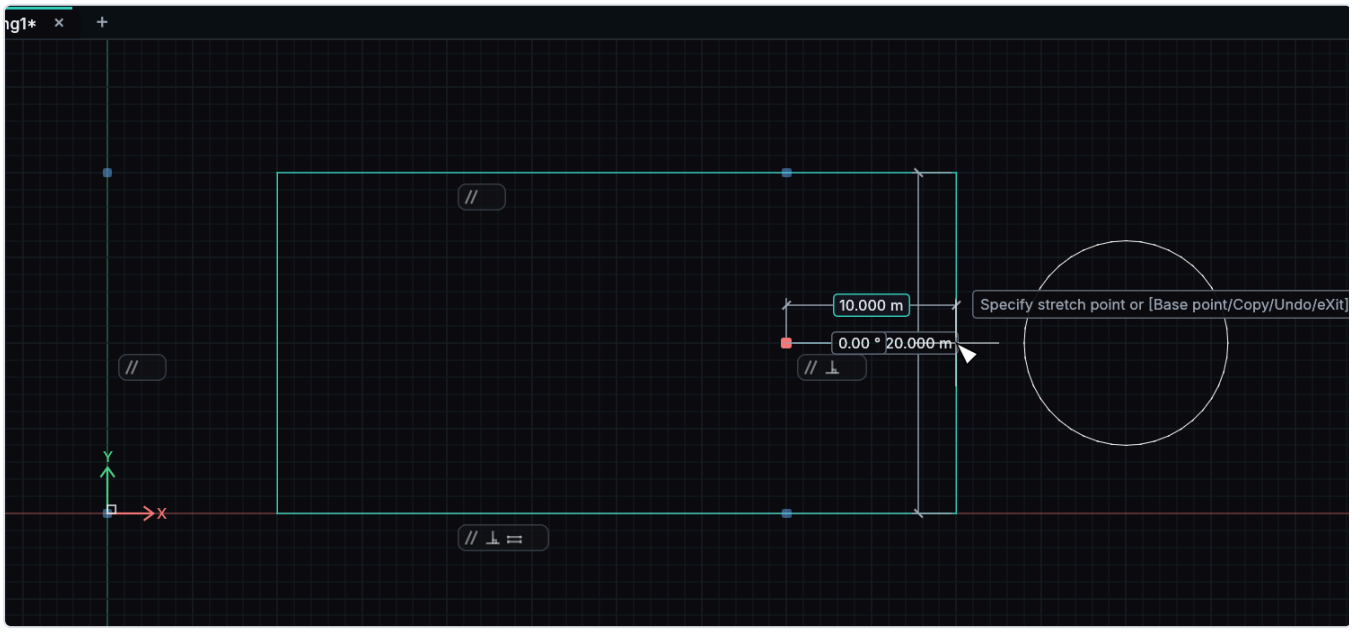


AutoConstrain tab: the order of the types and the tolerances.

Editing with constraints

Moving, stretching or rotating a constrained object (with grips, MOVE, STRETCH and the other commands) takes along the objects tied to it by constraints. During the drag, the **live preview** shows the connected objects already adjusted and, as in AutoCAD, **their original geometry disappears** from

the screen until the button is released (nothing is left dashed on top). Large drawings are solved off screen, without freezing the interface.



The right side dragged by its midpoint grip: the top and bottom lines stretch along, because they are coincident and horizontal; the originals leave the screen during the drag.

How the geometry moves, as in AutoCAD: of the line the user moved, **the endpoint they dragged** stays put (the one that moved the most; in a tie, the start), and the line keeps the length and direction they gave it as far as the constraints allow. A rotated horizontal line returns to horizontal around the endpoint that moved, with the same length; stretched by one endpoint, that endpoint stays where it was placed and the other follows; with a length dimension, the stretched endpoint stays and the line keeps its direction. Connected lines that were not touched **move whole**, without stretching or rotating unless they need to (a rectangle with one side moved moves as a whole); when they need to rotate, they rotate around their start. When the **corner** of a rectangle made of horizontal and vertical lines is stretched (both lines of the corner at once), the left side keeps its length and the other lines, from left to right, keep theirs: with the top right corner dragged, the left side stays where it was (it only goes up) and the top and bottom sides stretch, as in AutoCAD. When the corner of a polygon held by parallel and perpendicular constraints (no horizontal or vertical) is stretched, one of the two lines of the corner stays **whole**, where the user placed it: the first one that entered the constraint network (that of the first constraint created), as in AutoCAD; the others adjust to it. In the rectangle, of the two lines tied to the endpoints of the one that stays whole, the **left** one stays rigid when the two are separated horizontally; otherwise, the free endpoint of the line that was not touched goes to the foot of the perpendicular and the opposite side rotates around where it was, as AutoCAD does in most cases. A spline held by smooth only to a line or to the straight end segment of a polyline goes along, without deforming, when that segment is moved. When that segment is rotated, or the other spline of a smooth between two splines is moved, the spline that has only that smooth settles as in AutoCAD: it moves whole and only the junction vertex and one inner vertex change (the second or third counting from the junction). When the value of a constraint dimension changes, its **first point** stays: a line's dimension changes its end; a dimension between two lines moves the second one. With a Fix point on the edited line, it straightens first and then goes to the fixed point with the other endpoint as close as possible to where it was. **Fix on a whole object** holds, for a line (and a straight polyline segment), its **infinite line**, and for an arc (and an arc segment), its **circle**: the endpoints slide along it, as in

AutoCAD. The fixed side of a rectangle **stretches** when the top side goes up (the bottom one stays); a fixed line stretched in its own direction grows, and when pulled off its infinite line it stays as it was; the endpoint of a fixed arc slides along its circle. For a polyline stretched by a vertex, when it is not possible to leave all of them where the user placed them, the dragged vertex stays and the others slide.

When the requested edit has no solution with the constraints (pulling the endpoint of a fixed line, for example), the edit does not happen and the command line says so. With **CONSTRAINTRELAX** on (1), the constraints that prevent the edit are relaxed (removed) and the edit happens.

Undo and redo leave the constraints satisfied: if the restored state has an unsatisfied constraint (a drawing that came in that way from outside, for example), it is solved again in the same step. Changes made by **scripts** go through the constraints like commands do.

Constraints in DXF and DWG

The associative network of an AutoCAD file (geometric and dimensional constraints, user parameters, actions of associative surfaces) **comes back whole** when saving to DXF or DWG, in the same version and in any other (DWG 2000 to 2018, DXF 2010 and 2018), in the format AutoCAD writes in each version: AutoCAD keeps the constraints when it opens the file. AutoCAD's constraints **become BadBit CAD constraints**: they appear in the constraint bars and in the Parameters Manager, with AutoCAD's names and expressions, and are edited like those made here. Saving without changing any of them, the network comes back exactly as it came; changing any, it is rebuilt in AutoCAD's format. Constraints on the whole axis of an ellipse and on a text (like those AutoCAD's AUTOCONSTRAIN creates) also become BadBit CAD constraints. A constraint that BadBit CAD does not have leaves its group as it came, without editing, with a message when reading. The reserved layer for the dimensions of dynamic constraints keeps AutoCAD's name, `*ADSK_CONSTRAINTS` (the `*` marks a system layer: the user cannot create a layer with it). An **associative loft surface** (LOFT, with the cross sections, guides and path) goes to all versions of DWG and DXF the way AutoCAD writes it: from 2013 on its parameters are in the action; from 2000 to 2010, in the action's body, flagged for AutoCAD to re-evaluate on opening. What does not fit in the chosen version (the actions of other associative surfaces before 2013) or depends on an object deleted in BadBit CAD is left out of the file entirely, together with what depends on it, with a message in the save report.

Constraints and variables created in BadBit CAD **go to DXF and DWG** (DWG 2000 to 2018, DXF 2010 and 2018) in the format AutoCAD itself writes: when the file is opened in AutoCAD, the constraints work (moving, rotating or changing a parameter takes the connected objects along) and appear in the Parameters Manager with the same name and expression. All geometric types go, including smooth (between two splines and between a spline and a line, an arc or the end of a polyline) and those on splines (by control vertices, point on curve and fix; also splines made by fit points and the closed spline AutoCAD makes), on the end of the ellipse's major axis, on the whole ellipse axes, on texts and on paper space objects of layouts, plus the dimensional constraints (dynamic, annotational and reference) and the user variables, including expressions that use other parameters. The dimensions of dynamic constraints go on the reserved layer `*ADSK_CONSTRAINTS`, as AutoCAD does.

These stay only in the `.bbd` file, with a message when saving: those on closed periodic splines (AutoCAD renumbers their vertices on opening and does not reopen correctly even those it creates itself), those on whole splines other than point on curve or fix, those that tie objects in different spaces, and all of them in DXF R12, which has nowhere to store them. AutoCAD's constraints on splines (also smooth with another spline, with a line and with an arc, splines by fit points, closed and periodic ones), on the end of the ellipse's major axis and in paper space also become BadBit CAD constraints when reading.

Parametric drawing variables

Variable	What it controls	Values	Default
CCONSTRAINTFORM	Form of new dimensional constraints: 0 dynamic, 1 annotational	0 or 1	0
CONSTRAINTBARDISPLAY	Show constraint bars	0 or 1	1
CONSTRAINTBARMODE	Types shown in the bars (sum: horizontal 1, vertical 2, perpendicular 4, parallel 8, tangent 16, smooth 32, coincident 64, concentric 128, collinear 256, symmetric 512, equal 1024, fix 2048)	0 to 4095	4095
CONSTRAINTINFER	Infer constraints while drawing and editing	0 or 1	0
CONSTRAINTNAMEFORMAT	Text of constraint dimensions: 0 name, 1 value, 2 name and expression	0 to 2	2
CONSTRAINTRELAX	Relax the constraints that prevent an edit	0 or 1	0
DYNCONSTRAINTDISPLAY	Show dynamic constraints	0 or 1	1

All of them can be typed by name on the command line or set with SETVAR.

Undo and redo

Command	Shortcut	Effect
U	`Ctrl` + `Z`	Undoes the whole last command (a LINE with several segments goes away at once).
UNDO		Asks how many steps to undo, or an option (below).
REDO	`Ctrl` + `Y` or `Ctrl` + `Shift` + `Z`	Redoes what U undid, right after it.

The command line confirms what was undone: "Undone: LINE". A new command discards whatever there was to redo, as in AutoCAD. Inside LINE and PLINE, the **Undo** option (or `Ctrl` + `Z` while the command is still active) removes only the last segment; after the command ends, `Ctrl` + `Z` undoes the whole command.

UNDO has AutoCAD's options:

Option	What it does
A number	Undoes that many steps (`Enter` : one).
Mark (`M`)	Places a mark in the history.
Back (`B`)	Undoes everything back to the last mark (with no mark, the whole history).
BEgin (`BE`) and **End** (`E`)	Whatever is done between the two becomes a single undo step.

Undo keeps the history in memory with a ceiling of 512 MiB (about two million ordinary commands). The oldest steps beyond the ceiling go to a file in the recovery folder (see *Files*), and U still reaches them, all the way back to the start of the drawing; closing the drawing deletes that file. The ceiling also applies to **REDO**: after many U's, the most distant redo steps go back to the file, and REDO reaches them one by one.

The history **survives a crash**: after **Recover** (see *Recovery after a crash*, in *Files*), U goes back through the same history as before the crash, including the steps that were in the file and those from before the last save, and REDO redoes what had been undone. A crash right after saving, with no changes to recover, does not lose the history either: when you open the file again, it opens as it was saved, without asking, and U goes back through the history from before the crash. If the file was changed externally in the meantime (saved by another window or another computer), the history no longer applies to it and is discarded.

Without a recovery folder (the Block Editor, REFEDIT), the long undo file goes in the system's temporary folder; in the browser, the steps stay in memory, compressed. The Block Editor and the REFEDIT working drawing have their own undo, which does not mix with the drawing's.

OOPS

Brings back what the last **ERASE** removed, even after other commands (which are not undone, unlike U). It works once for each ERASE; the objects come back as they were (on a layer that was deleted afterwards, they come back on the current one).

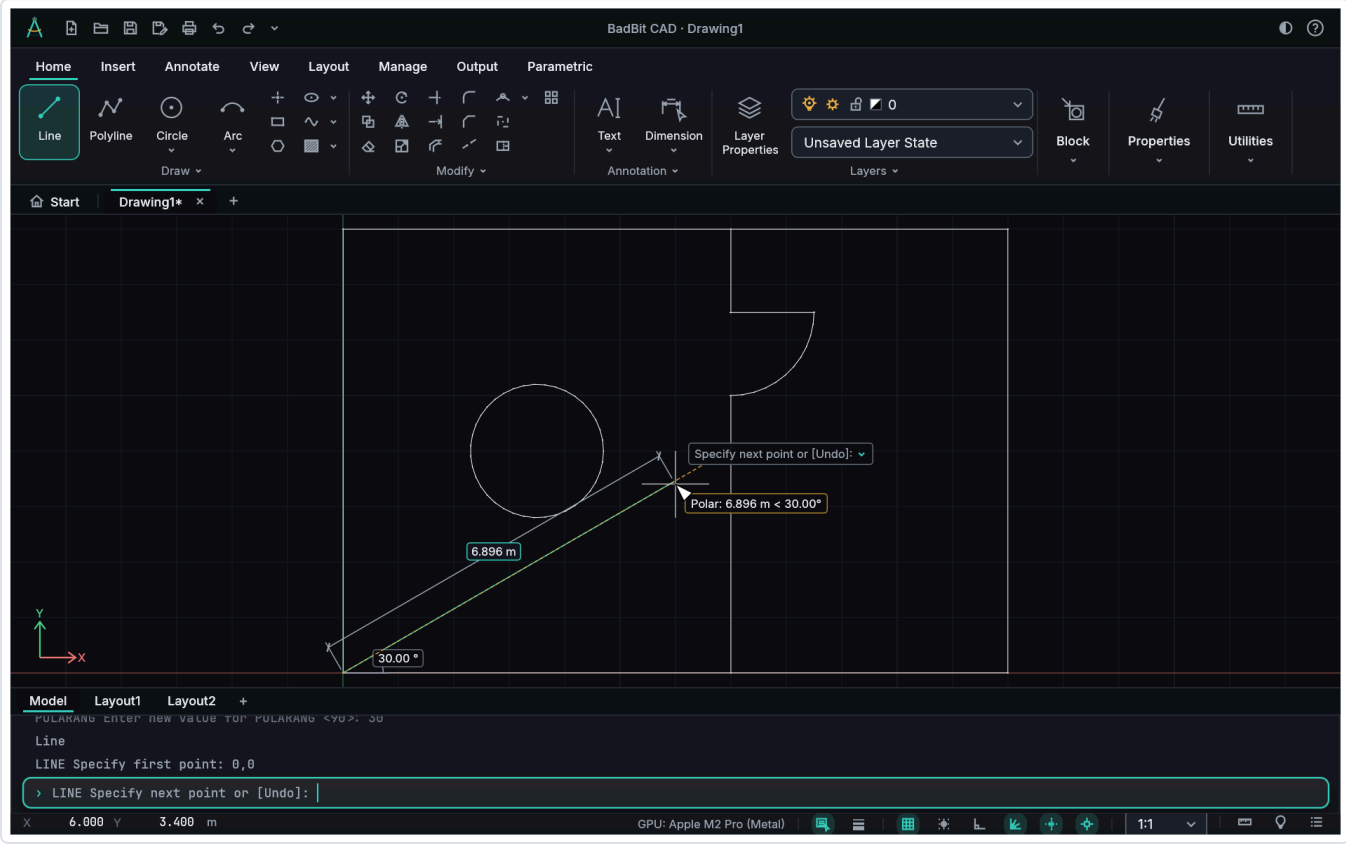
Precision: Ortho, Polar, object snaps and tracking

Ortho (F8)

Constrains the point to horizontal or vertical from the last point. Ortho and Polar cannot be on at the same time.

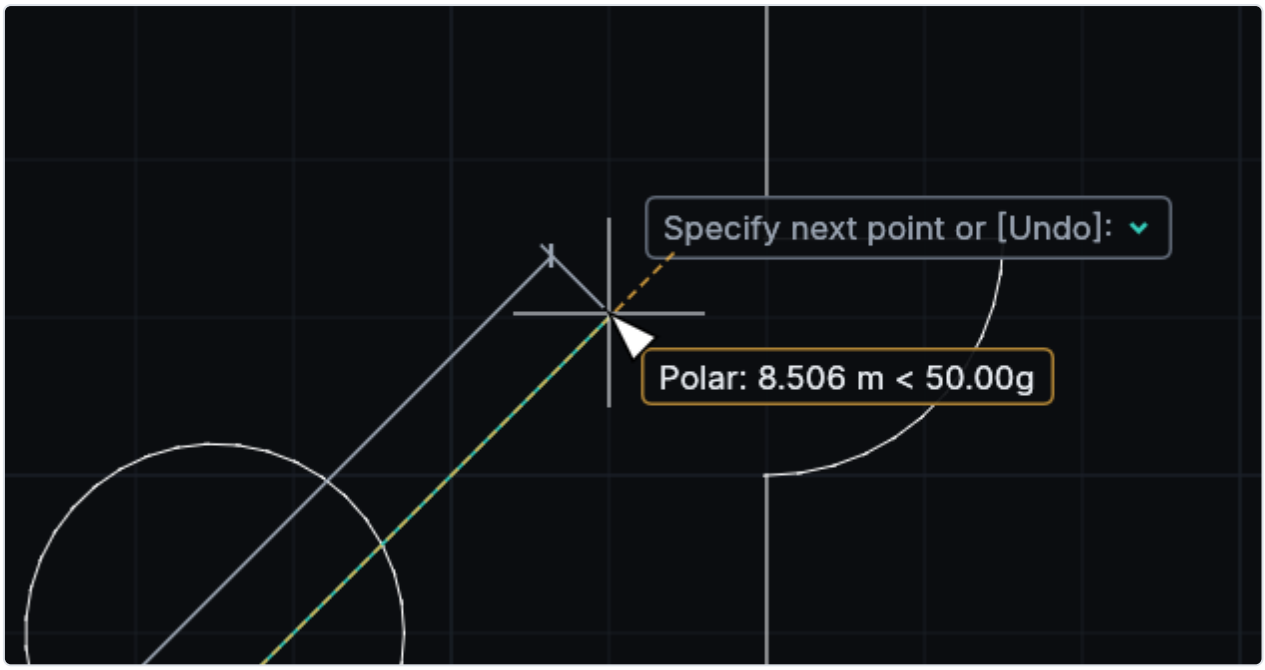
Polar (F10)

With Polar on, the cursor snaps to angles that are multiples of the **POLARANG** increment (default 90°) from the last point. The dotted alignment path and the tooltip with the distance and angle appear. Typing a distance follows the polar ray.



Polar with POLARANG 30: the alignment path and the tooltip "Polar: distance < angle".

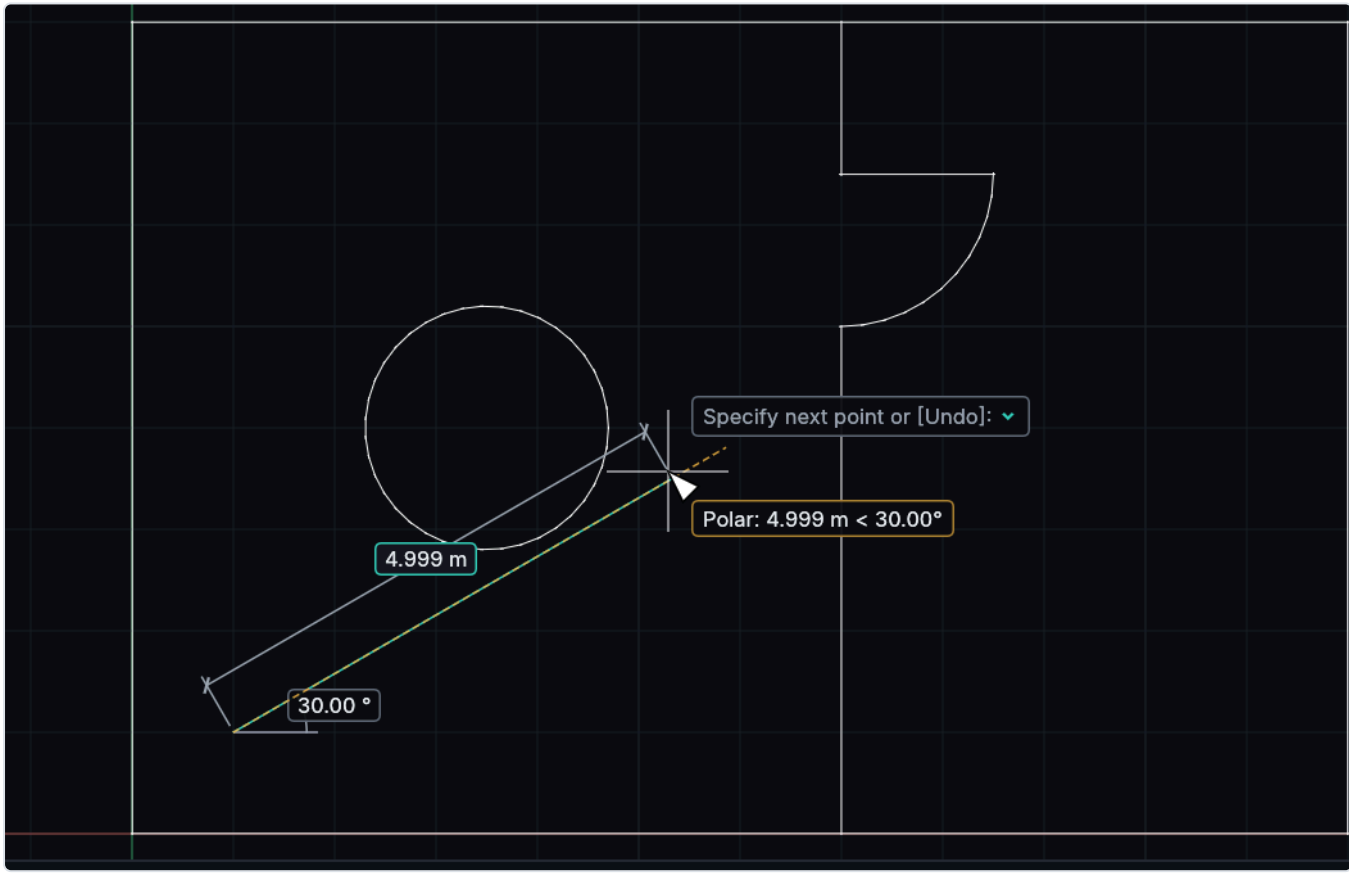
The polar, tracking and extension tooltip follows the drawing's formats, like AutoCAD's: the **distance** in the drawing's unit and decimal places (`LUNITS`, `LUPREC`; in a drawing in mm, `1500 mm`) and the **angle** measured from the zero direction (`ANGBASE`) in the `ANGDIR` direction, in the `AUNITS` format with `AUPREC` decimal places: decimal degrees (`45°`), degrees/minutes/seconds (`45°00'`), grads (`50g`), radians (`0.79r`) or, in surveyor's units, the bearing (`N 45° E`). In polar relative to the last segment, the relative angle comes out in the same format (the bearing, which applies only to directions, becomes degrees/minutes/seconds).



Polar at 45° in a drawing with angles in grads: "Polar: … < 50g".

Additional angles and relative measurement

Besides the multiples of the increment, polar follows up to ten **additional angles** (**POLARADDANG**, separated by semicolons: `30;135`), turned on by bit 4 of **POLARMODE**. As in AutoCAD, an additional angle applies only to itself, without multiples: with POLARANG 90 and the additional angle 30, the cursor at 31° snaps to 30°, but at 60° it snaps to nothing.

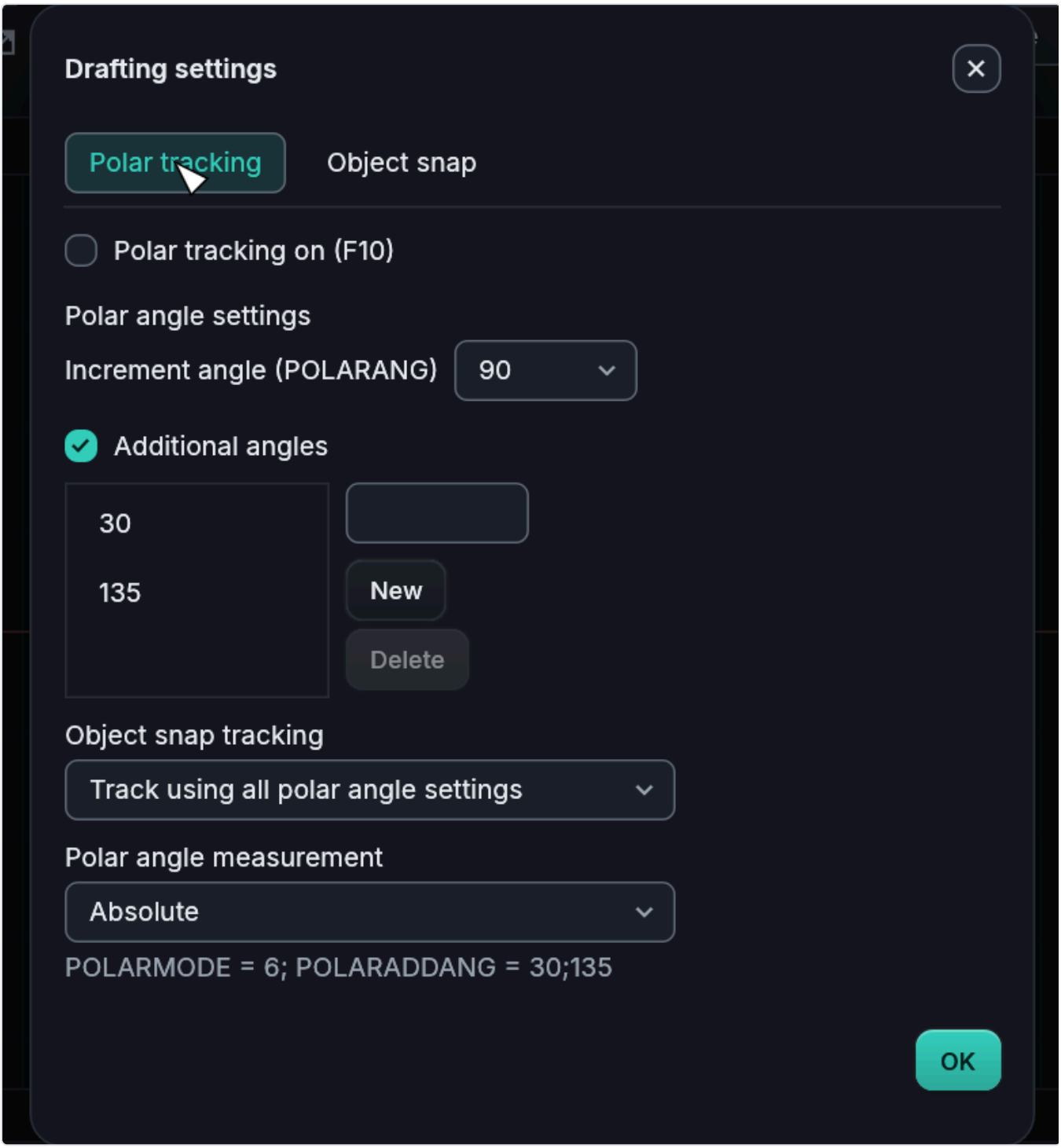


POLARANG 90 and the additional angle 30: the cursor near 30° snaps to the additional angle.

Measurement can be **absolute** (from the X axis, the default) or **relative to the last segment** (bit 1 of POLARMODE): angles are counted from the direction of the last segment of LINE or PLINE (for an arc, the tangent at its end), and the tooltip then reads "Relative polar". With no segment yet in the command, measurement goes back to absolute.

Drafting settings (DSETTINGS)

The **Drafting settings** dialog box (`DS`, `DSETTINGS`, `SE`; or right-click the polar icon on the status bar) has two tabs. The **Polar tracking** tab turns polar on, chooses the increment from the list (90, 45, 30, 22.5, 18, 15, 10, 5 or 60; another value is typed in POLARANG), maintains the **additional angles** (type the angle beside it and click **New**; select one in the list and click **Delete**), chooses **object snap tracking** (orthogonal only or using all polar angle settings) and the **measurement** (absolute or relative to the last segment). Below, the resulting POLARMODE and POLARADDANG values are shown. The **Object snap** tab is the OSNAP one.



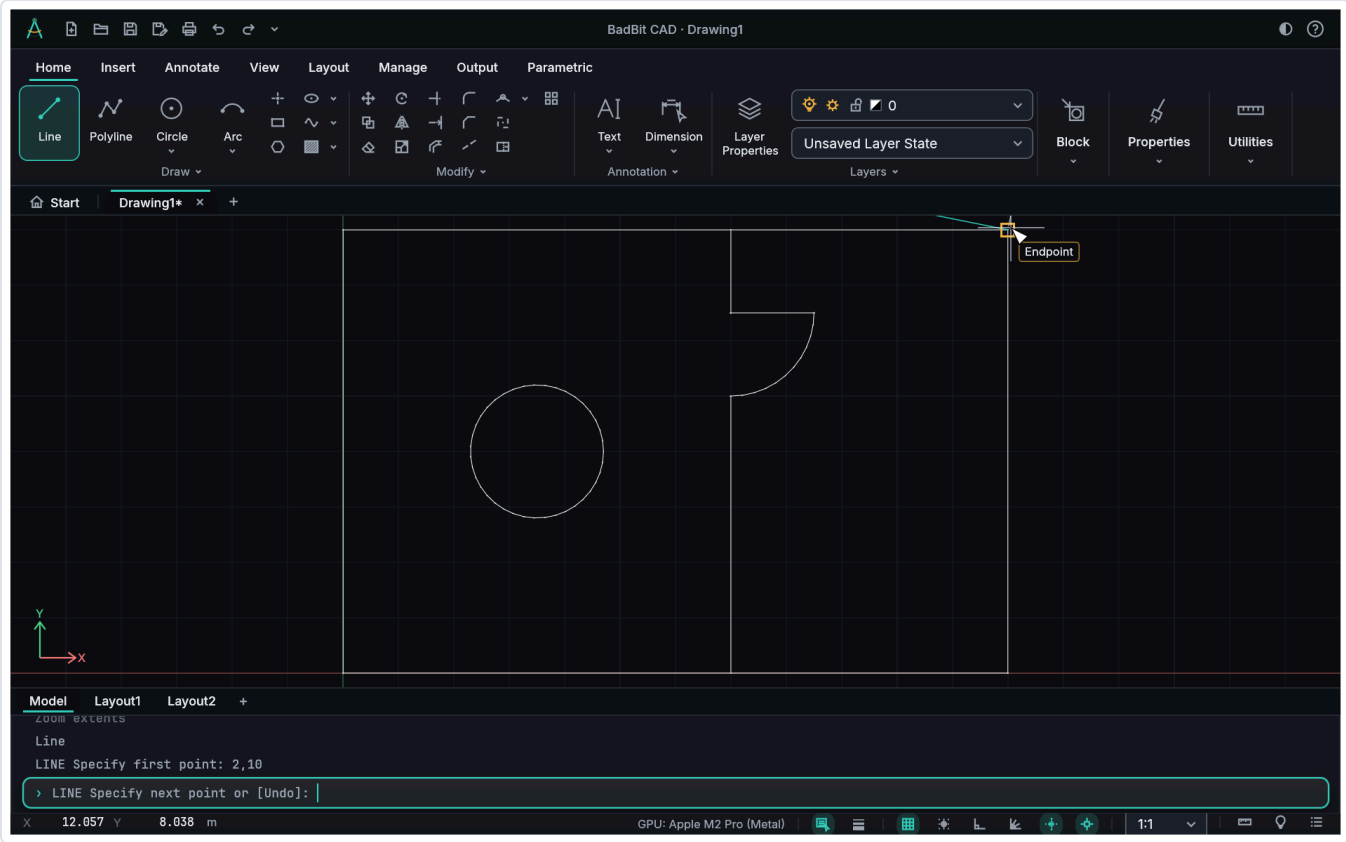
The Polar tracking tab, with the additional angles 30 and 135 and tracking using all polar angle settings.

POLARMODE adds up AutoCAD's bits: **1** measurement relative to the last segment, **2** tracking using all polar angle settings, **4** use the additional angles, **8** acquire points only with `Shift`. As in AutoCAD, POLARANG, POLARMODE and POLARADDANG are stored in the user profile (they apply to all drawings), and not in the drawing.

Object snaps (F3)

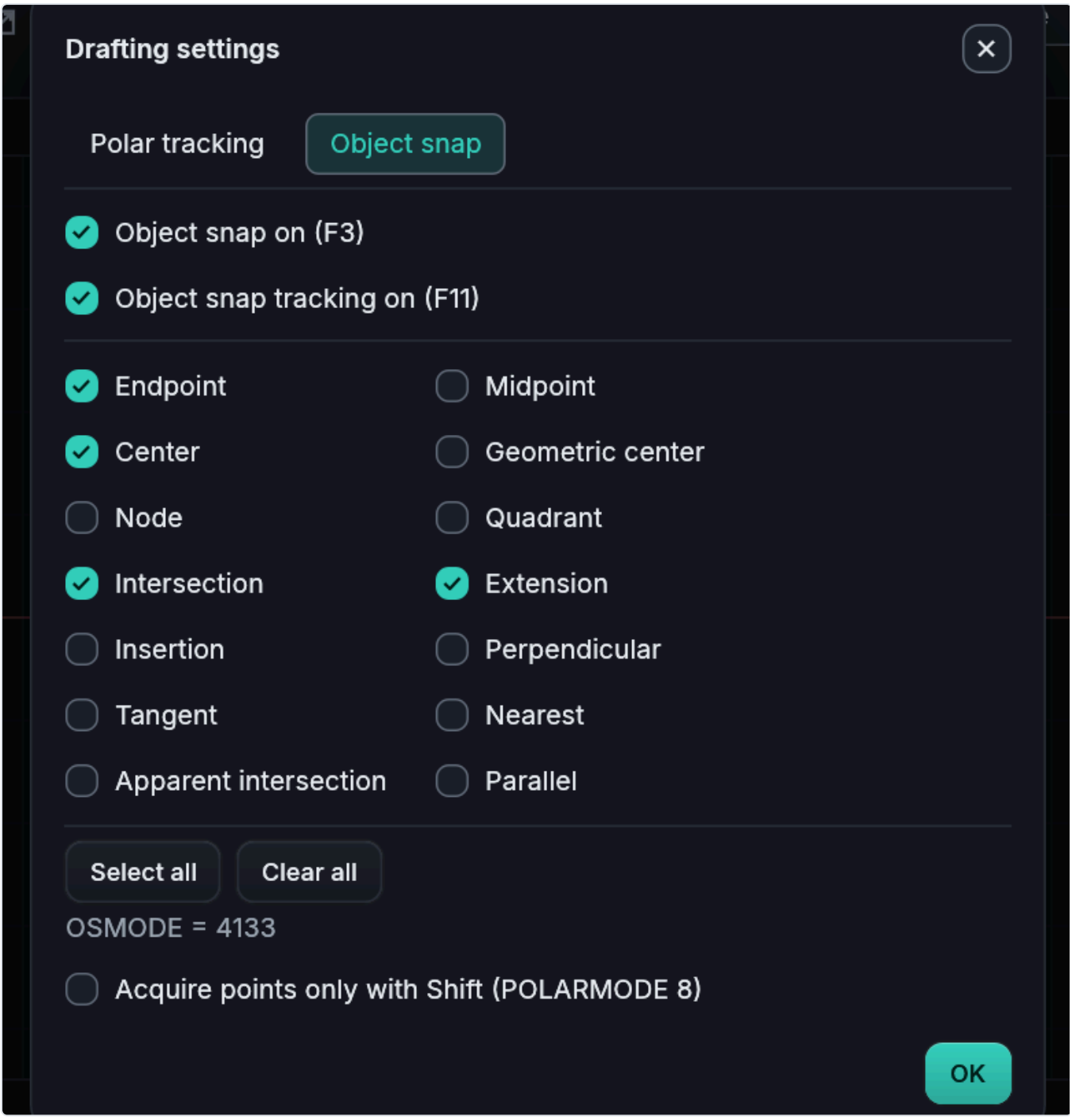
With the cursor near an object, the point snaps to one of its notable points. The marker (in yellow) and the name appear. Object snap takes precedence over Ortho, as in AutoCAD.

Mode	Marker	Snaps to
Endpoint (END)	Square	Ends of lines, arcs, elliptical arcs, open splines and segments; the start of a ray.
Midpoint (MID)	Triangle	Middle of lines, segments, arcs and elliptical arcs; the root of a construction line.
Center (CEN)	Circle	Center of circles, arcs and ellipses (with the cursor over the curve).
Geometric center (GCEN)	Hexagon	Center of closed polylines.
Node (NODE)	Circle with X	Point objects.
Quadrant (QUA)	Diamond	0°, 90°, 180° and 270° of circles and arcs; the ends of the ellipse axes.
Intersection (INT)	X	Crossings between objects.
Apparent intersection (APP)	Square with X	Crossing of the extensions of acquired lines.
Extension (EXT)	Three dots	Extension of an acquired line or arc.
Perpendicular (PER)	Right angle	The foot of the perpendicular from the last point.
Tangent (TAN)	Circle with tangent	Tangency from the last point.
Parallel (PAR)	Two bars	The parallel to an acquired line.
Nearest (NEA)	Hourglass	The point on the object closest to the cursor (only if there is no other).
Insertion (INS)	—	Blocks and texts (when they exist).



Endpoint snap with its name beside it.

The default is AutoCAD's (OSMODE 4133): endpoint, center, intersection and extension. The modes are turned on and off on the **Object snap** tab of Drafting settings (the **OSNAP** command, alias `OS`, opens it directly; or right-click the object snap icon on the status bar) or with the **OSMODE** variable, with the same numbers as AutoCAD (1 endpoint, 2 midpoint, 4 center, 8 node, 16 quadrant, 32 intersection, 64 insertion, 128 perpendicular, 256 tangent, 512 nearest, 1024 geometric center, 2048 apparent intersection, 4096 extension, 8192 parallel).



The Object snap tab of Drafting settings.

Snap for a single point

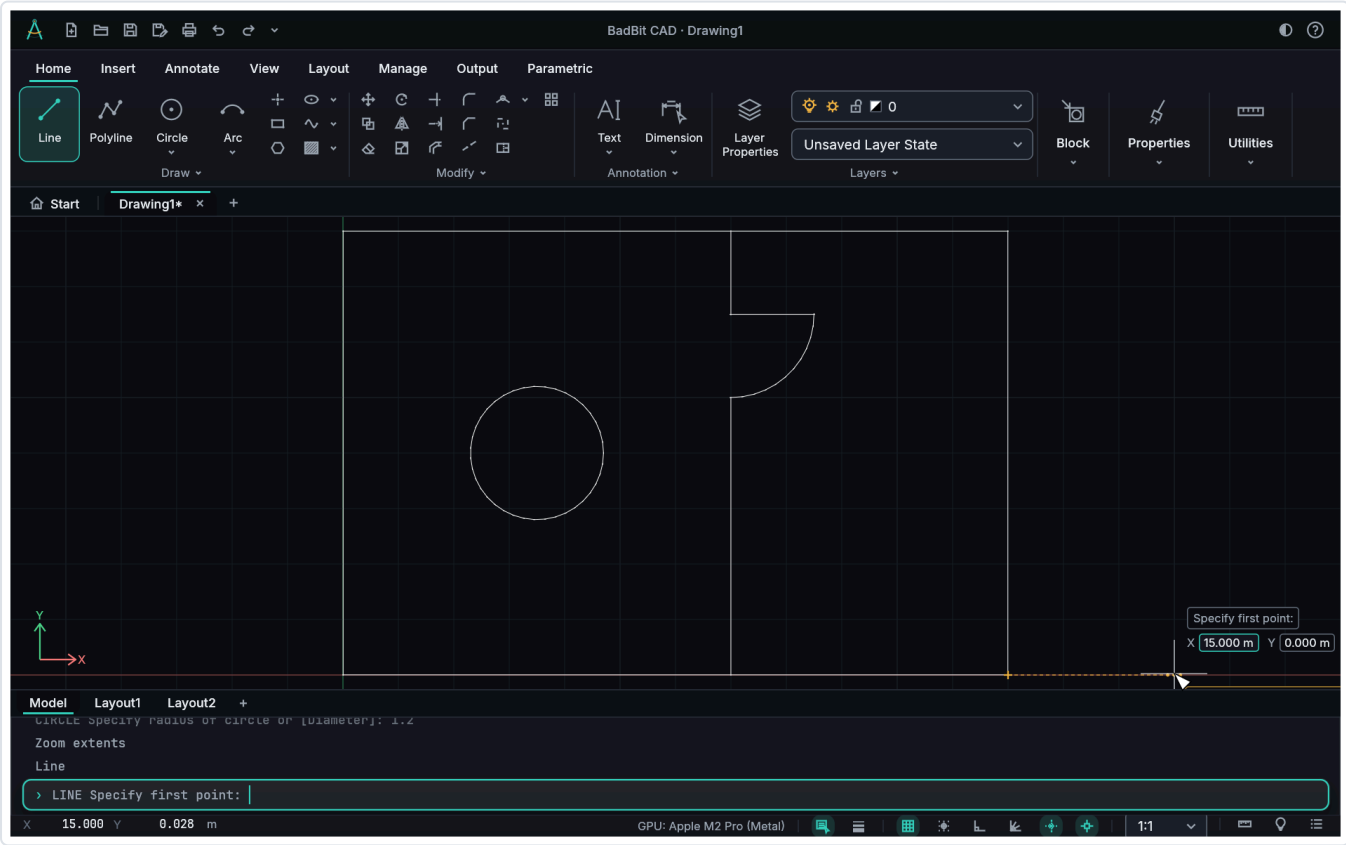
During a command, typing the mode name applies only to the next point: `END`, `MID`, `CEN`, `GCE`, `NOD`, `QUA`, `INT`, `APP`, `EXT`, `PER`, `TAN`, `PAR`, `NEA`, or `NON` (none). With or without an underscore.

Acquisition, extension and tracking (F11)

Extension, parallel and tracking start from **acquired** points and lines: pause the cursor for a moment (half a second) over an endpoint (or another snap point) or over a line, until a small yellow cross appears. Pausing over it again releases it. With **Acquire points only with Shift** (on the Object snap tab;

bit 8 of POLARMODE), pausing the cursor acquires nothing: press `Shift` with the cursor over the point or the line (again, to release). Then:

- **Extension:** the cursor runs along the extension of the acquired line (or arc), with the dotted path.
- **Parallel:** the cursor runs along the parallel to the acquired line, from the last point.
- **Tracking (F11):** horizontal and vertical lines pass through the acquired points; where two of them cross, the cursor snaps. With **Track using all polar angle settings** (bit 2 of POLARMODE), the lines from the acquired points also follow the polar angles (the multiples of POLARANG and the additional angles, always absolute). With Polar on, the polar ray also crosses these lines. With **Ortho** on, the point stays constrained to the horizontal or vertical of the last point, and tracking marks where that line crosses the line of the acquired point: this is how you close a rectangle, acquiring the first corner and coming down to its plumb line.



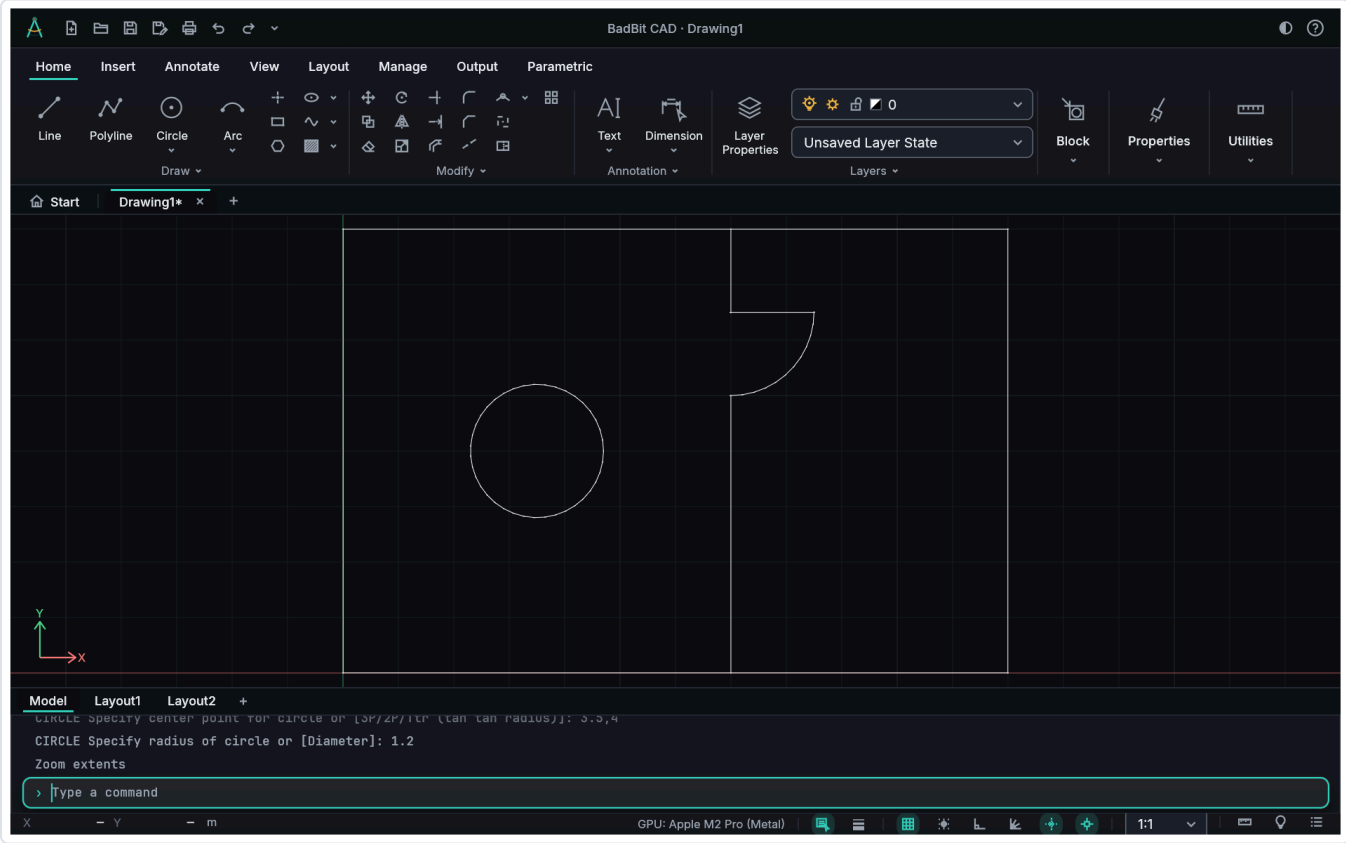
Extension: the acquired endpoint (small cross) and the path with the distance.

Acquired points are released when the point is used or the command ends.

Grid and units

Grid (F7)

The grid is on when the program opens, as in AutoCAD, and `F7` turns it off and on. It sits beneath the drawing, with a heavier line every five lines and the X and Y axes tinted. The grid is adaptive, as in AutoCAD: when you zoom out, the spacing grows five times so that it never turns into a blur; when you zoom in and the lines get too far apart, lines five times closer appear between them (from 1 m they go to 0.2 m, then 0.04 m). The GRIDUNIT lines stay in the grid at any zoom. Grid snap (F9) does not change with zoom: it follows SNAPUNIT.



The grid turned on.

The spacing is the **GRIDUNIT** variable, in the project's unit. The default follows the unit as long as it has not been chosen: 1 m in a project in meters, 10 cm in centimeters, 10 mm in millimeters. Once chosen by hand, the value stays even when the unit changes.

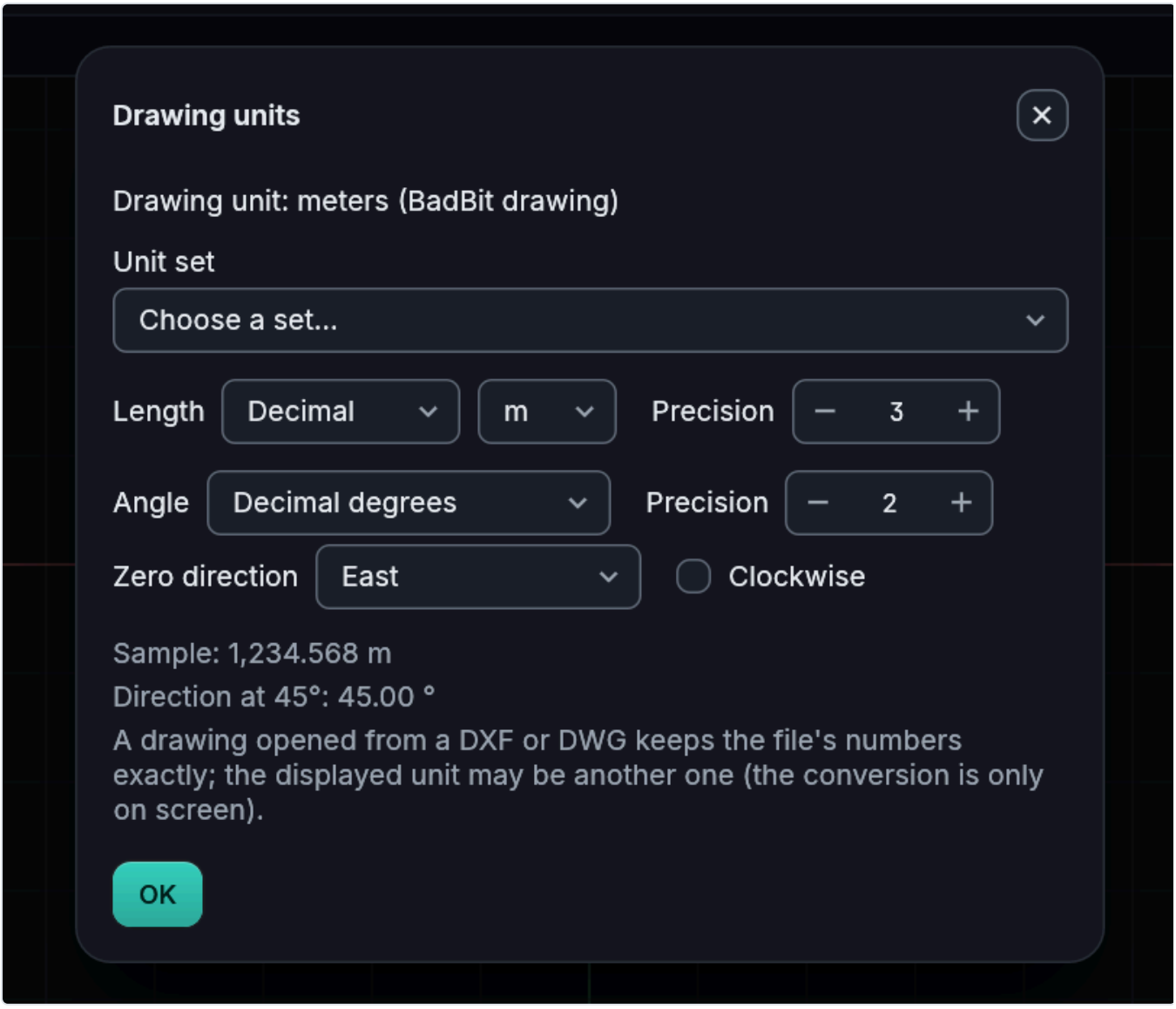
Grid snap (F9)

With grid snap on (`F9`, or the icon on the status bar), the cursor crosshair moves from grid point to grid point while the command asks for a point: clicks fall exactly on the grid. The step is the **SNAPUNIT** variable; as long as it has not been chosen, it is the same as the grid (GRIDUNIT). Near an object, object snap still applies, as in AutoCAD. For selecting objects, the cursor is free.



With F9, the line goes from point (1, 1) to the grid point (5, 2), even with the mouse slightly off it.

Drawing units (UNITS, alias UN)



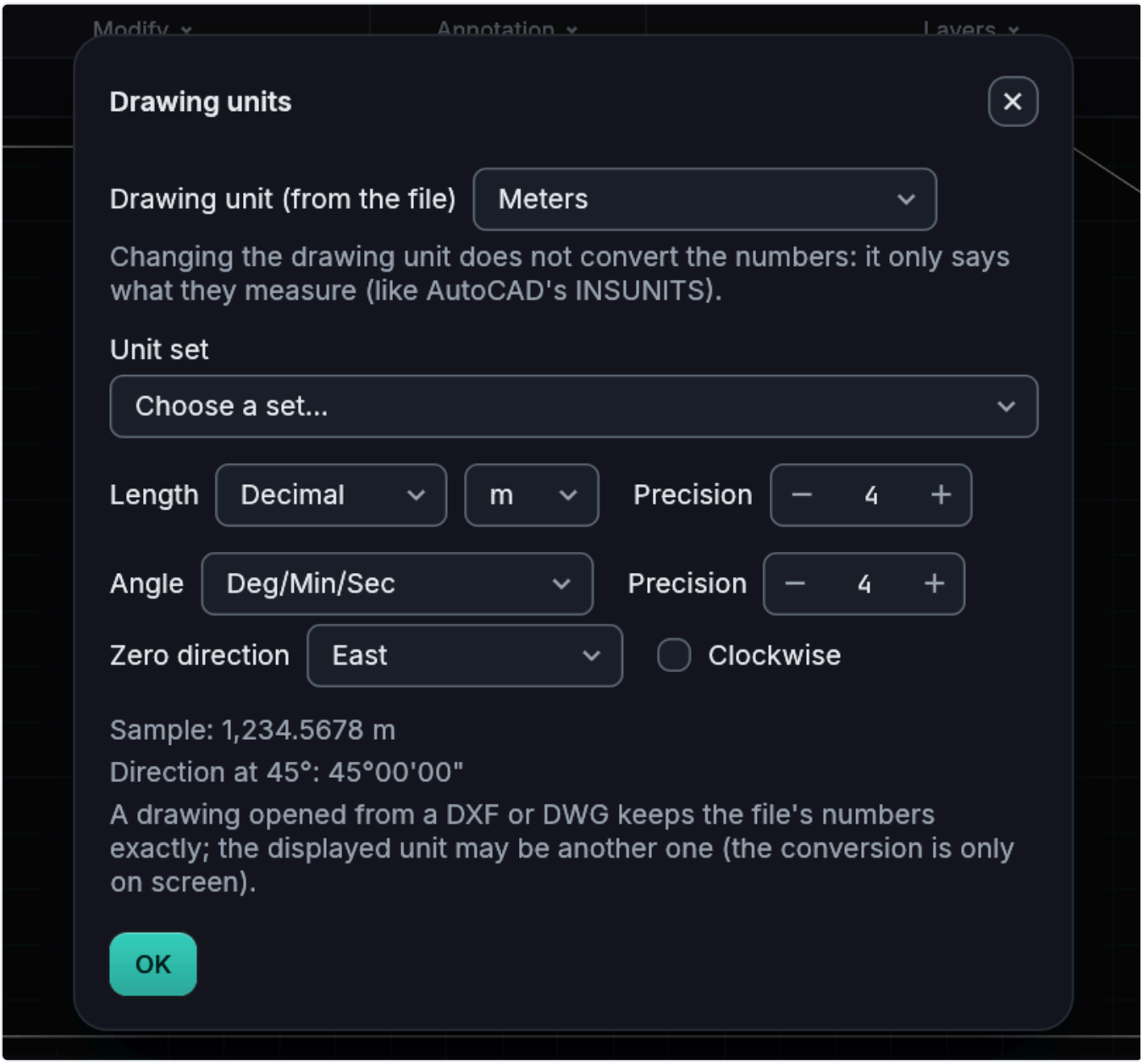
The units dialog box of a drawing made in BadBit CAD.

The dialog box chooses a ready-made set (SI, Brazil – structures, Brazil – architecture and MEP, Detailing in millimeters, US imperial), the length unit and the decimal places, and, like AutoCAD's:

- **Length**: the type (scientific `1.5500E+01`, decimal `15.50`, engineering `1'-3.50"`, architectural `1'-3 1/2"`, fractional `15 1/2`; in engineering and architectural the number is read in inches) and the precision (the decimal places, or the fraction from 1/2 to 1/256).
- **Angle**: the type (decimal degrees, degrees/minutes/seconds `45°30'15"`, grads `50g`, radians `0.7854r`, surveyor's units `N 45°30'15" E`) and the precision (in degrees/minutes/seconds: 0 degrees only, 1–2 minutes, 3–4 seconds, 5 to 8 seconds with decimal places).
- **Zero direction** (east, north, west or south, `ANGBASE`) and **Clockwise** (`ANGDIR`): they apply to angles typed in polar coordinates (`@10<45`) and to displayed directions. With zero to the north and clockwise, angles are azimuths. Typed angles accept the marks of any type: `45d30'15"`, `45°30'`, `50g`, `0.78r` and the bearings `N45d30'E`.

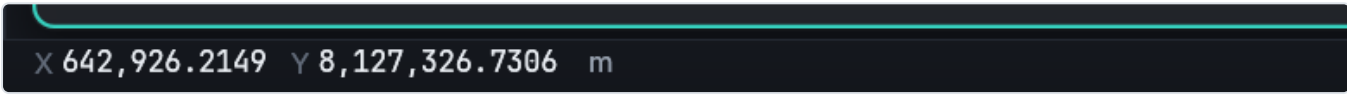
In a drawing made in BadBit CAD, coordinates are stored internally in meters: the unit changes what you type (without a unit, `250` in a project in centimeters is 2.5 m), what you see in the coordinates and the default grid.

The drawing unit that came from a DXF or DWG



A UTM survey opened from a DXF in meters, with angles in degrees/minutes/seconds.

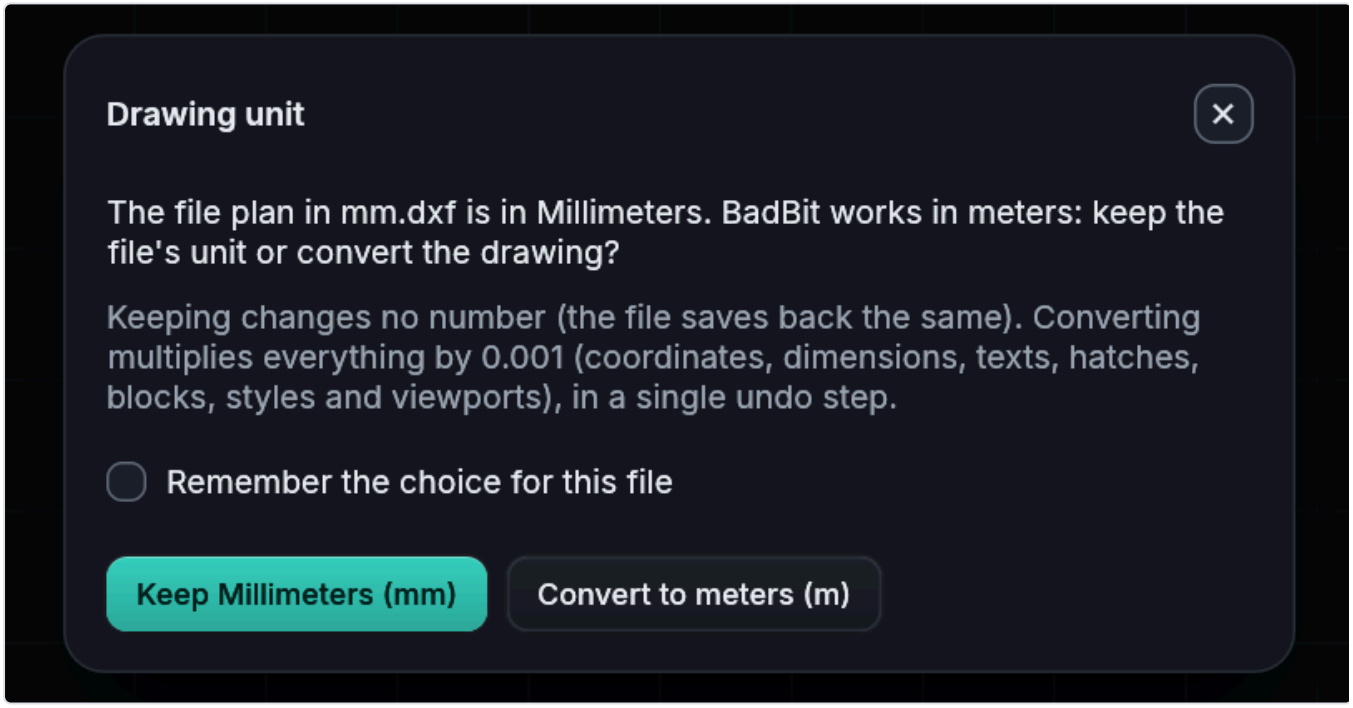
A drawing opened from a DXF or DWG keeps **exactly the file's numbers** (not even the last digit changes) and uses its unit and formats (`INSUNITS`, `LUNITS`, `LUPREC`, `AUNITS`, `AUPREC`, `ANGBASE`, `ANGDIR`): the dialog box shows **Drawing unit (from the file)**. Changing that unit does not convert the numbers, it only says what they measure (like AutoCAD's `INSUNITS`); the displayed unit can be a different one (the conversion is only on screen). Each tab has its own unit: a drawing in millimeters and another in meters opened side by side each show their own.



The status bar with the drawing's "m" next to the UTM coordinates.

The status bar shows the drawing's unit next to the coordinates ("unitless" in a drawing without a unit).

Keeping or converting the file's unit



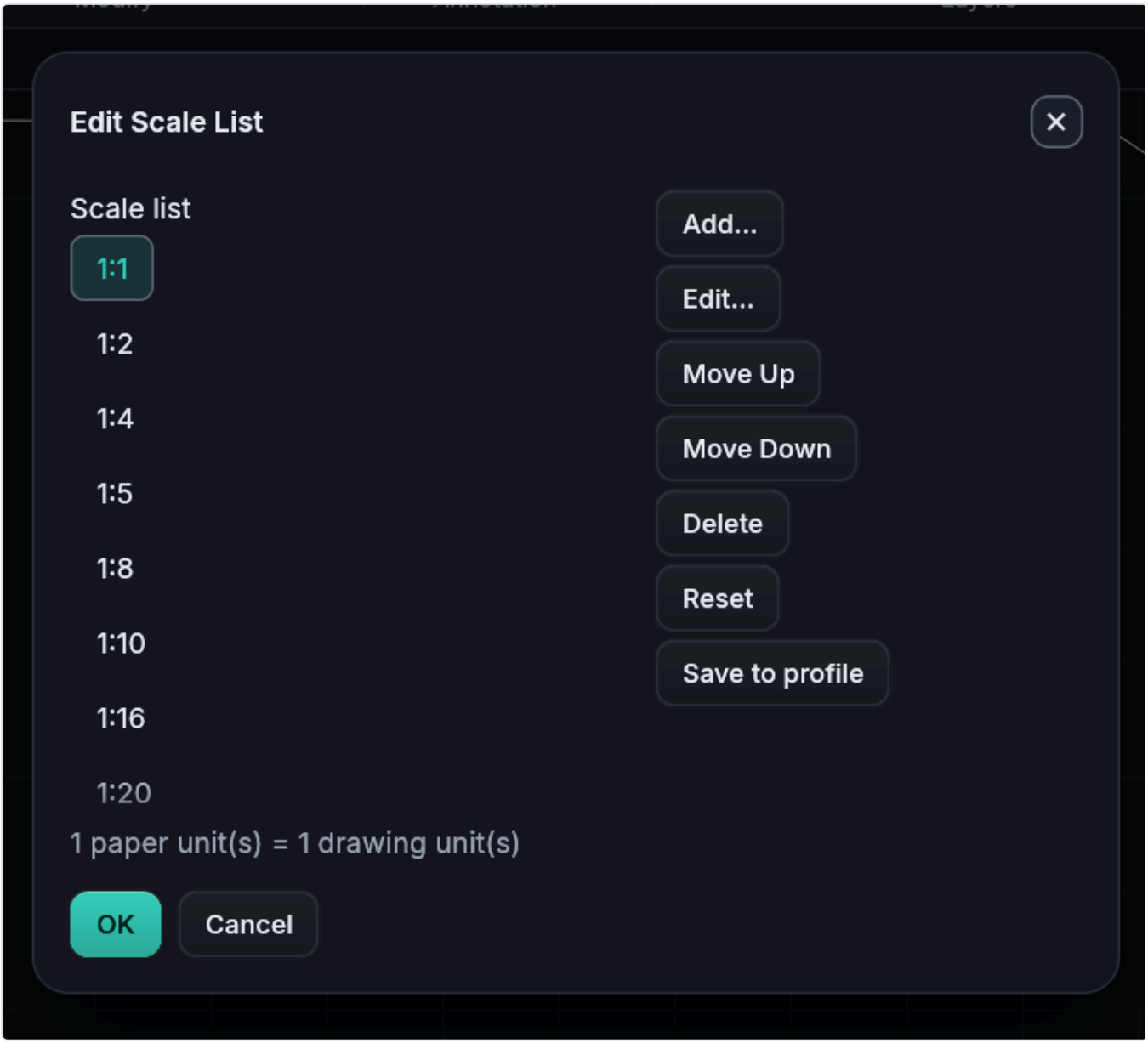
A DXF in millimeters: Keep (the default) or Convert to meters.

When you open a DXF or DWG in a unit other than the meter (millimeters, centimeters, inches, feet...), the **Drawing unit** dialog box asks:

- **Keep <unit>** (the main button, the default; also `Esc` and closing the window): the numbers stay as in the file and the drawing works in its unit. Saving back to DXF or DWG writes the same numbers.
- **Convert to meters (m)**: the whole drawing is multiplied by the factor (0.001 from mm to m): model coordinates, blocks and references, dimensions (with the style sizes and overrides), texts, hatches (with the pattern scale), multileaders, tables, text, dimension, multileader and table styles, linetypes and the model side of viewports (the paper stays the same size). External references stay in their own files, with the factor in the reference's scale. It is **a single undo step**: **U** (or `Ctrl` + `Z`) brings back the file's numbers and unit together. Converting a large drawing is done in parallel.
- **Remember the choice for this file** stores the answer in the profile (next time the file opens directly, as stated on the command line).

With **FILEDIA 0** there is no prompt: the drawing stays in the file's unit and the command line says so.

Scale list: SCALELISTEDIT and -SCALELISTEDIT



The Edit Scale List dialog box with the default metric list.

SCALELISTEDIT opens the drawing's **Edit Scale List** dialog box, like AutoCAD's: **Add...** and **Edit...** (the name, the paper units and the drawing units; `1:50` is 1 paper unit to 50 drawing units), **Move Up**, **Move Down**, **Delete** (the last one stays), **Reset** (back to the default list) and **Save to profile** (the list becomes the default for new drawings and for Reset). **OK** saves the list in the drawing; **Cancel** undoes what was changed in the dialog box.

-SCALELISTEDIT does the same from the command line, with AutoCAD's options: `[? (?)/Add (A)/Delete (D)/Reset (R)/Exit (E)] <Add>`. **?** lists the scales; **A** asks for the name and the ratio (`1:50`, `1=50`, `2:1` or the imperial ones `1/4" = 1'-0"`); **D** asks for the name; **R** asks `[Yes (Y)/No (N)] <No>`.

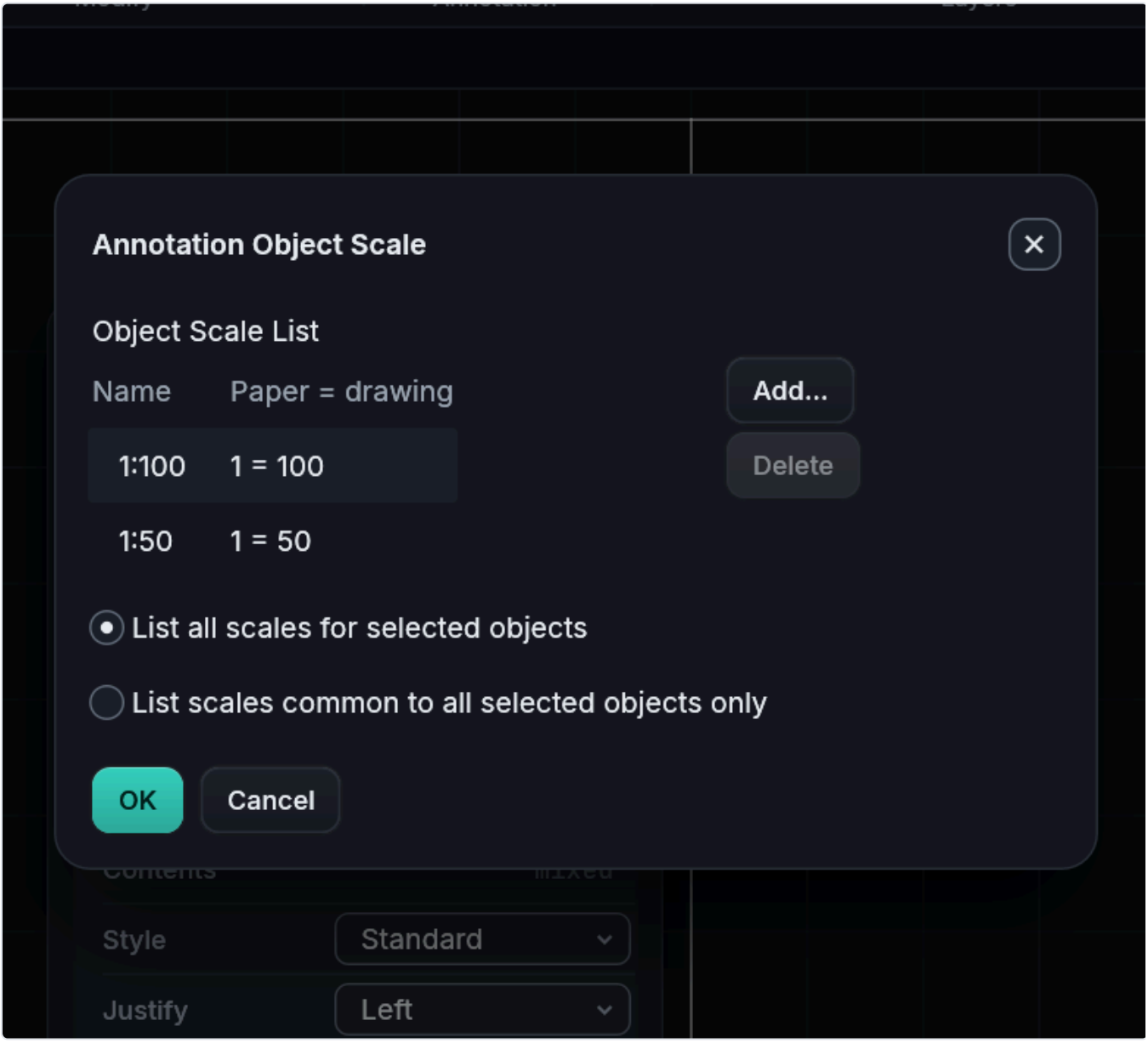
The list belongs to the drawing (it goes in the `.bbd`). A new drawing starts with the profile's list or, without it, with AutoCAD's **metric** list (1:1 to 1:1000, 2:1 to 100:1, with 1:25, 1:75, 1:200, 1:250 and 1:500 from Brazilian floor plans) or the **imperial** one (1/128" = 1'-0" to 1-1/2" = 1'-0") depending on whether

the drawing is metric or imperial (`MEASUREMENT`). It appears in the **annotation scale** list and in the **viewport scale** list of the status bar and in the **Plot** dialog box.

Changing the list (the dialog box's **OK**, or each Add, Delete and Reset of -SCALELISTEDIT) is an **undo step** with the command's name and leaves the drawing **modified** (the asterisk on the tab and the save prompt on closing): **U** brings back the previous list, **REDO** puts it back again.

Scales are **true ratios**, in any unit: 1:100 is one hundredth of the real size. In a drawing in mm, a viewport at 1:100 shows 100,000 mm of the model per meter of paper; in one in meters, 100 m. **ZOOM nXP** inside the viewport (`1/100XP`) sets the viewport to 1:100 in the drawing's unit, and a new **MVIEW** viewport in an empty drawing starts at 1:100.

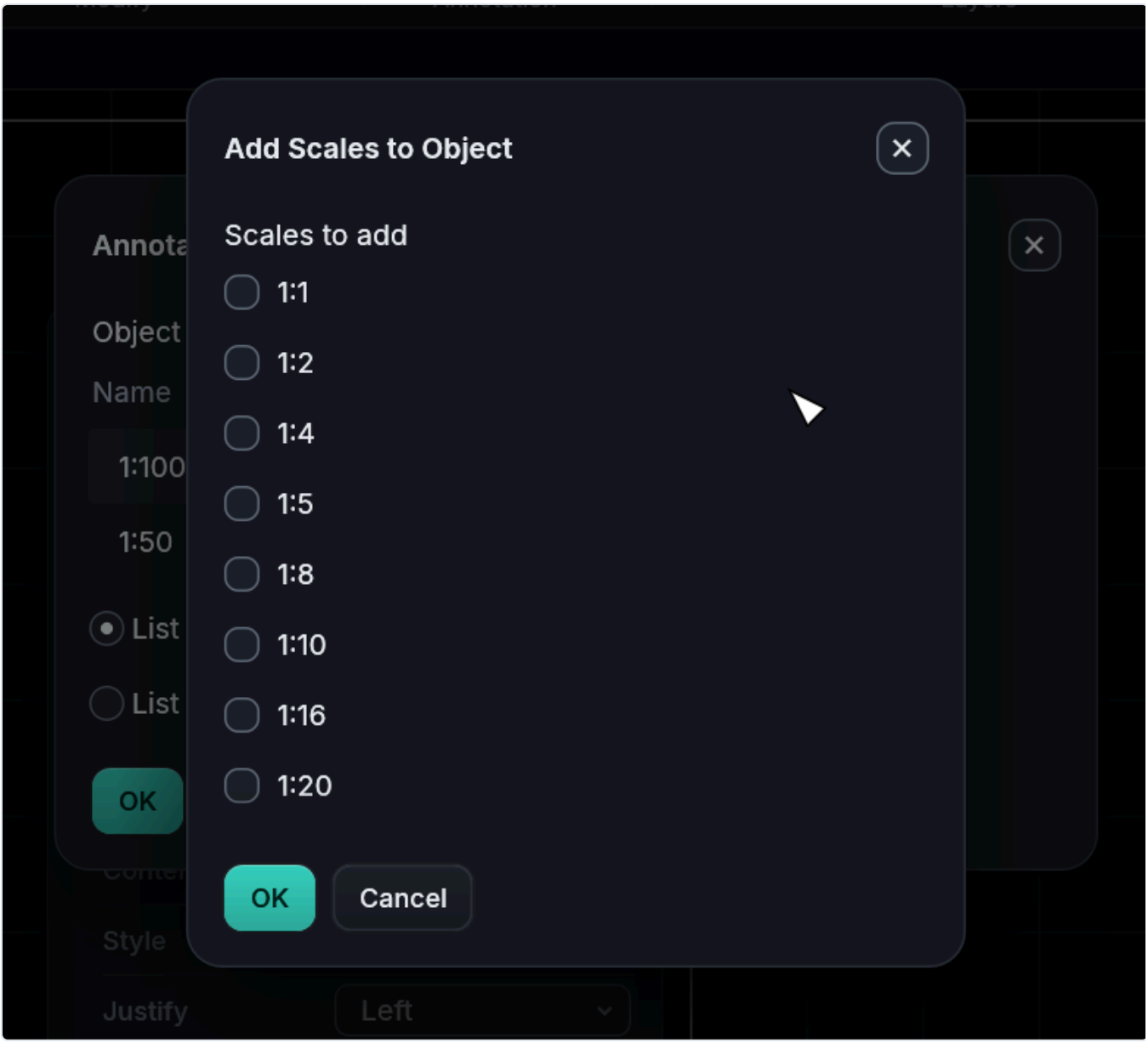
Scales of annotative objects: OBJECTSCALE and -OBJECTSCALE



OBJECTSCALE with two annotative texts chosen: 1:100 for one and 1:50 for the other.

OBJECTSCALE adds and removes **annotation scales** on annotative objects (texts, multiline texts, dimensions, multileaders and tolerances), like AutoCAD's. With no selection, it asks `Select objects:` (those that are not annotative are left out; the command line says how many annotative ones were included) and opens the **Annotation Object Scale** dialog box:

- the **object's scale list**, with the name and the paper and drawing units (`1 = 100`);
- **List all scales for selected objects** or **List scales common to all selected objects only**;
- **Add...** opens the **Add Scales to Object** dialog box with the drawing's scale list (the one from SCALELISTEDIT): check one or more and click **OK**;
- **Delete** removes the selected scale (the object always keeps at least one).

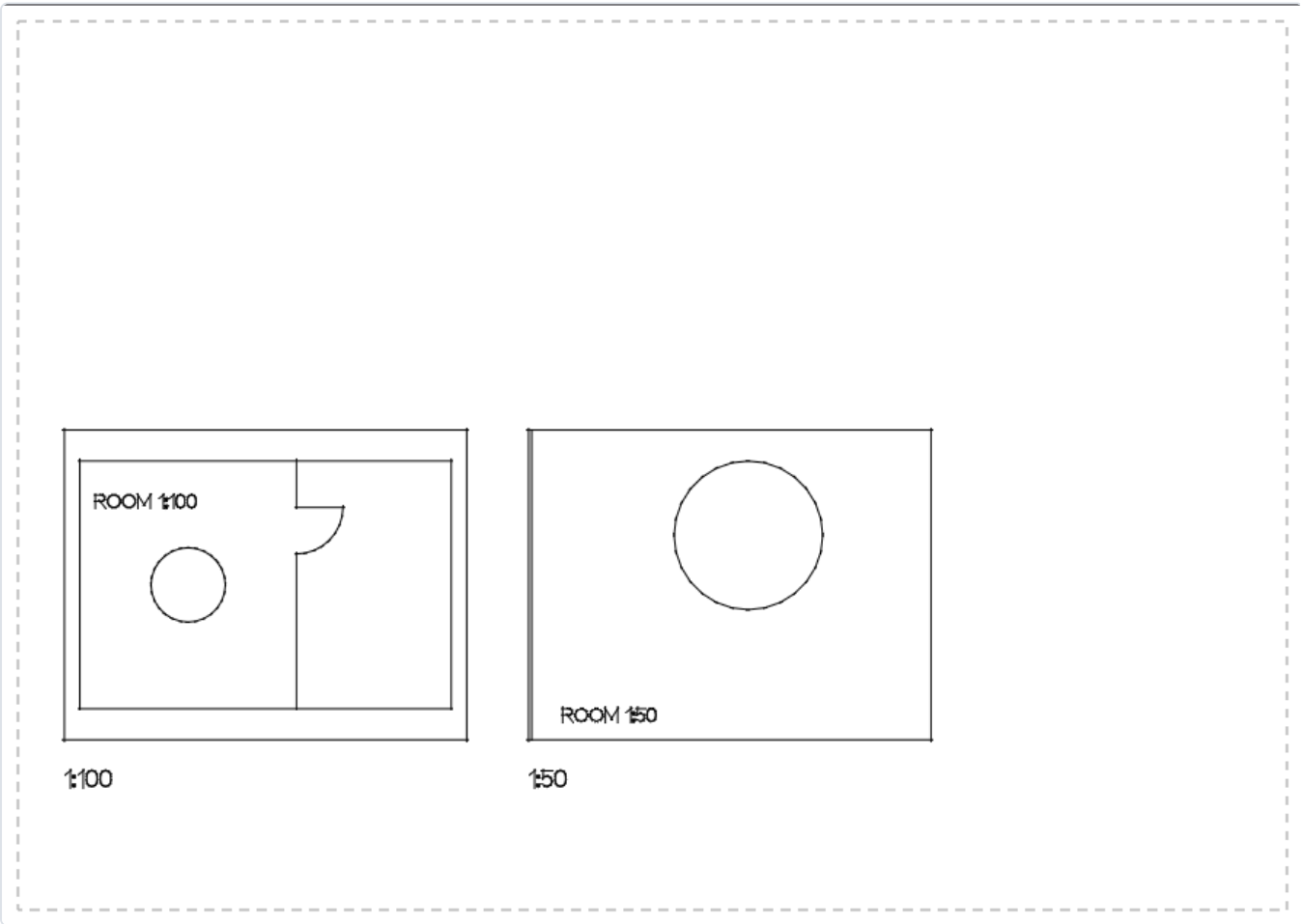


The Add Scales to Object dialog box, on top: the drawing's scales, each with a check box.

OK applies everything in **a single undo step**; **Cancel** changes nothing. An object that gains the current scale (CANNOSCALE) starts appearing at it; one that loses the scale at which it was appearing starts appearing at the current one, if it supports it, or at the first one left.

-OBJECTSCALE does the same from the command line, with AutoCAD's options: `Enter an option [Add (A)/Delete (D)/? (?)] <Add>:`. **A** asks `Enter named scale to add or [? (?)] <1:100>:` (the name in the list or a ratio such as `1:30`); each scale typed is added immediately and the prompt comes back, until `Enter` (`Enter` right away adds the current scale). **D** does the same to remove, and **?** lists the objects' scales and those common to all.

ANNOALLVISIBLE controls whether annotative objects that **do not support** the current scale are displayed: **1** (the default) shows all of them (each at the scale it is at); **0** shows only those that support CANNOSCALE, and the others also cannot be selected or snapped to. Changing the annotation scale with 0 changes which ones are displayed. In a **layout**, each viewport shows annotative objects at **its own scale** (the one on the status bar inside the viewport), as AutoCAD does — and the plot comes out the same. The variable belongs to the drawing, as in AutoCAD: it goes in the `.bbd` and in DXF and DWG (2007 onward), is recorded in undo and leaves the drawing modified. A positive **ANNOAUTOSCALE** still adds the new scale to all annotative objects when CANNOSCALE changes.



ANNOALLVISIBLE 0: the viewport on the left, at 1:100, shows only "SALA 1:100"; the one on the right, at 1:50, only "SALA 1:50".

The dynamic input angle

The angle that dynamic input (F12) shows next to the cursor follows the drawing: it is measured from the zero direction (`ANGBASE`), in the `ANGDIR` direction, on the shorter side, in the `AUNITS` format with `AUPREC` decimal places (decimal degrees, `45°30'00"`, `50g`, `0.7854r`; in surveyor's units, the bearing `N 30°00' W`). An angle typed in the angle field (after `Tab`) is read like the one in `@10<45`, in the same format and from the same base.

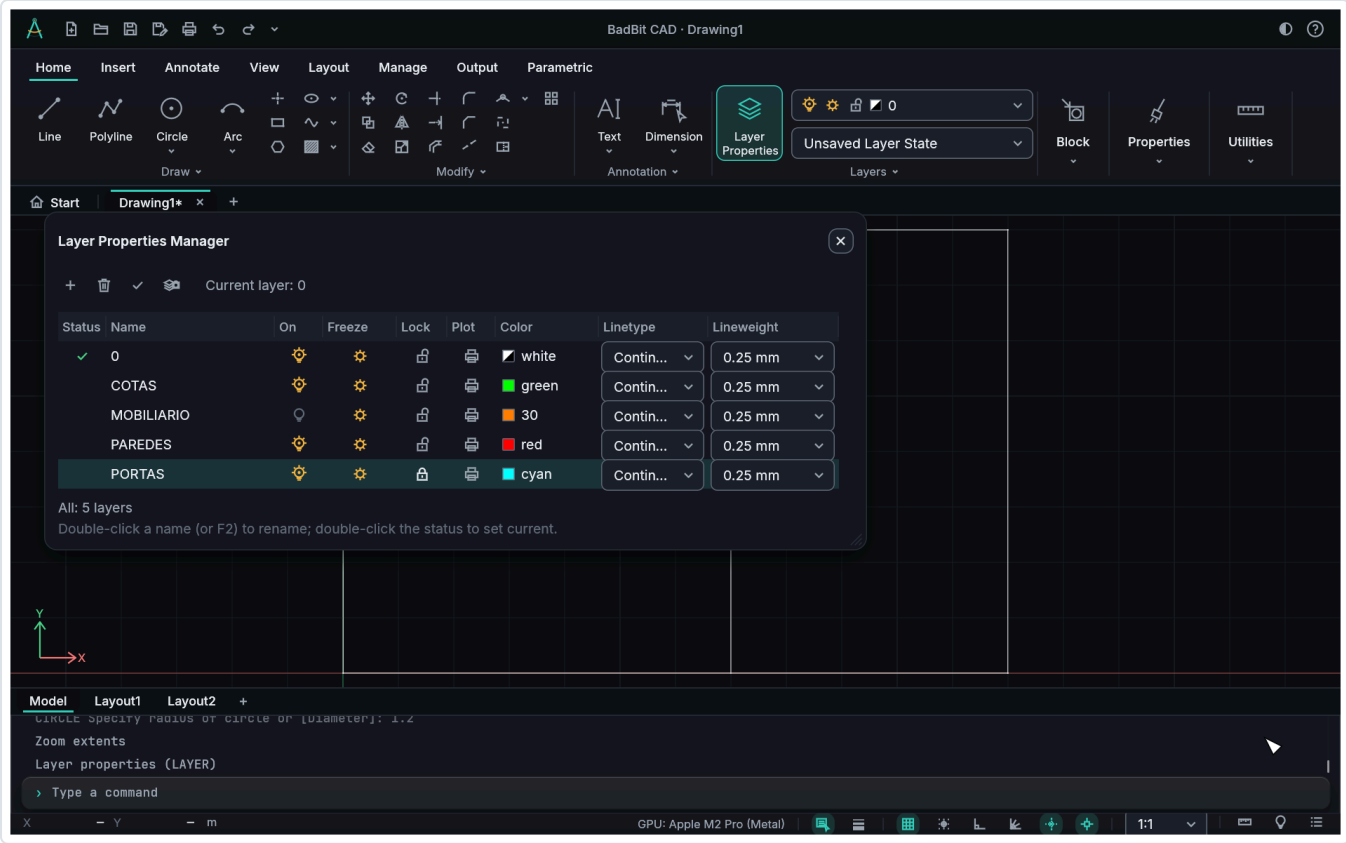
An **external reference** in another unit comes in scaled by the factor between the units, and its file's unit is **stored in the drawing**: if the reference's file changes to another unit (from mm to cm, for example), reloading the reference, even after saving, closing and reopening the drawing, multiplies the scale of its references by the ratio between the new factor and the old one (0.001 → 0.01). Converting the drawing to another unit does not disturb the calculation.

The **INSUNITSDEFSOURCE** and **INSUNITSDEFTARGET** variables (stored in the profile, as in AutoCAD's registry; the `INSUNITS` code, 6 = meters by default) set the unit of an incoming unitless file (INSERT, XREF or opened with FILEDIA 0) and that of the unitless drawing that receives it.

Layers

Layers organize the drawing as in AutoCAD. Each object belongs to a layer. By default, the object's color, linetype and lineweight come from the layer (ByLayer). Every drawing starts with layer **0**, which cannot be renamed or deleted. New objects go on the **current layer**.

Layer Properties Manager (LAYER, LA)



The `LA` command (or the **Layer Properties** button in the Layers panel of the ribbon) opens the manager. It is a palette: it stays open while you draw, can be dragged by its background and resized by its edges.

Button	Shortcut	What it does
+ New Layer	`Alt` + `N`	Creates "Layer1", "Layer2"... with the properties of the selected layer, with the name ready to be typed.
Trash can: Delete Layer	`Alt` + `D`	Deletes the selected layer.
✓ Set Current	`Alt` + `C`	The selected layer starts receiving new objects.

In the table:

- **Status**: the green check mark indicates the current layer. Double-click the status to make a layer current.

- **Name**: double-click (or select the row and press `F2`) to rename. `Enter` confirms and `Esc` cancels.
- **On** (light bulb): when off, the layer disappears from the screen and its objects cannot be selected or snapped to.
- **Freeze** (sun/snowflake): when frozen, the layer disappears like an off layer and is also left out of view calculations: zoom extents (and the Extents plot area) does not count its objects; those on a layer that is merely off do count, as in AutoCAD.

Blocks follow the AutoCAD rule: a block object on layer **0** takes the reference's layer (color, linetype, lineweight and also visibility), so it disappears when the reference's layer is turned off or frozen, and does not disappear when "0" is frozen; a block object on its own layer disappears when that layer is turned off or frozen, even with the reference's layer on. Freezing the reference's layer hides the whole block; turning it off hides only what is on "0". Loose objects on layer "0" disappear when it is frozen. What is not displayed also cannot be selected, snapped to or plotted. - **Lock** (padlock): objects are displayed and can be snapped to, but cannot be moved, copied or erased. The command says how many were left out. - **Plot** (printer): sets whether the layer is plotted. - **Color**: opens the *Select Color* dialog box. - **Linetype**: the list of linetypes loaded in the drawing (see *Linetypes*, below). The last item, **Load...**, opens the dialog box for loading linetypes (from the BadBit CAD library or from a `.lin` file), like AutoCAD's Load button; the first linetype checked is applied to the layer right away. - **Lineweight**: list with AutoCAD's standard values, from 0.00 to 2.11 mm.

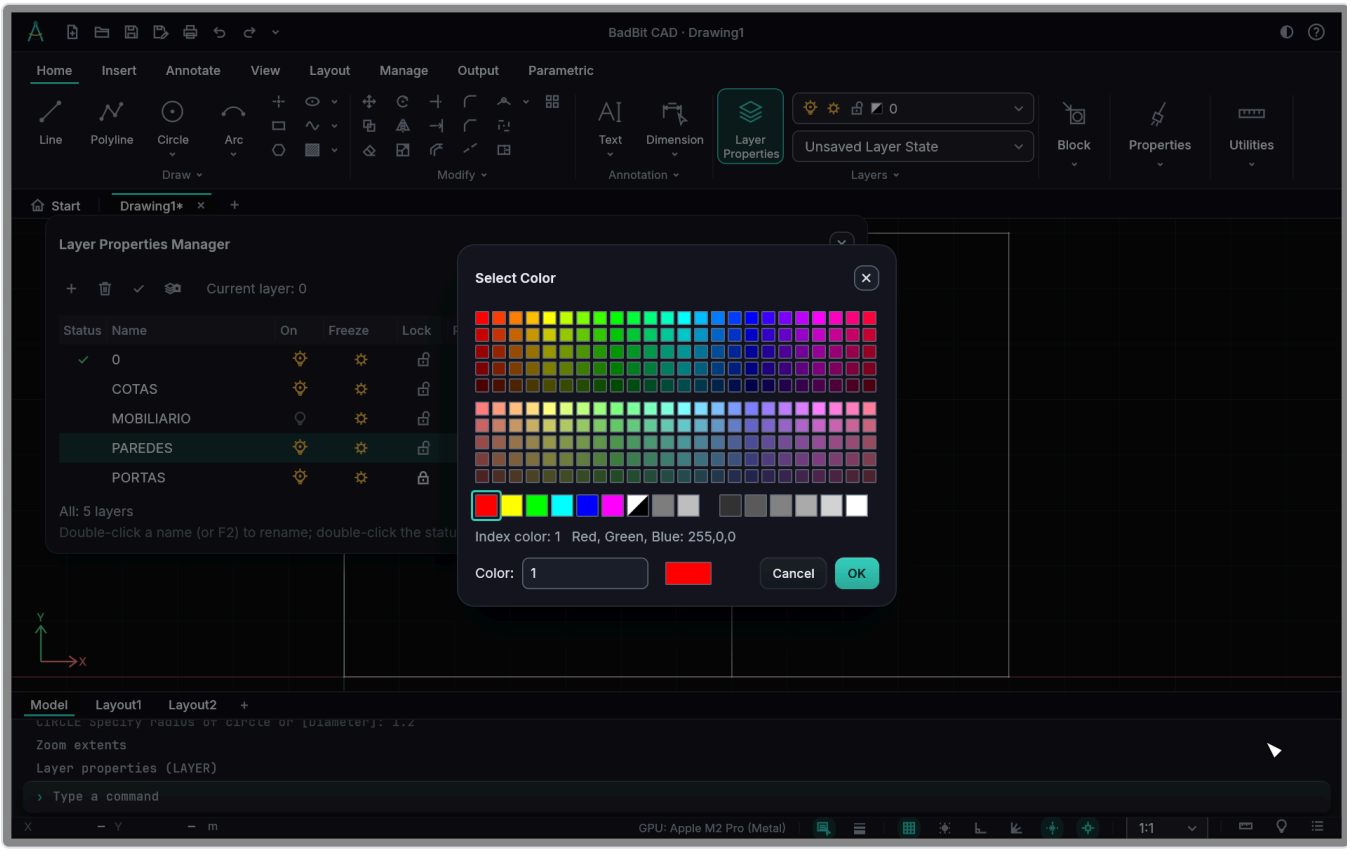
The footer shows the total number of layers and, when the request cannot be carried out, the reason.

Rules (the same as AutoCAD's)

Request	Result
Rename or delete layer 0	Not possible.
Delete the current layer	Not possible: make another one current first.
Delete a layer with objects	Not possible: move or erase the objects first.
Freeze the current layer	Not possible. Turning it off is possible.
Make a frozen layer current	Not possible.
Duplicate name (case-insensitive) or empty name	Not accepted.
Name with `< > / \ " : ; ? * | = '` or a backquote	Not accepted.

Every layer change can be undone with `U` (`Ctrl`+`Z`). If undo deletes or freezes the current layer, the current layer goes back to 0.

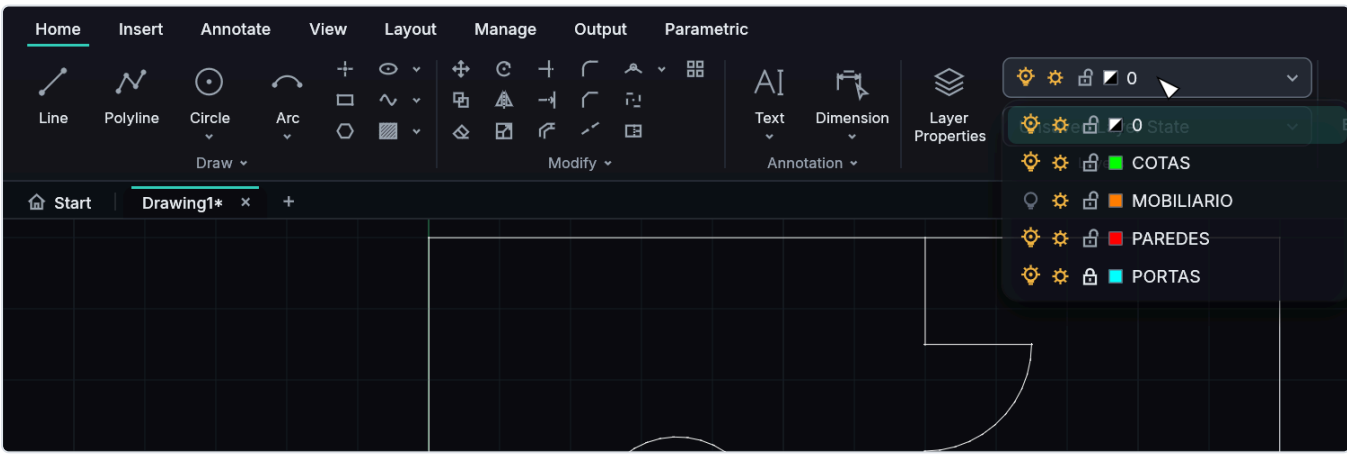
Select Color



This is AutoCAD's index color tab (ACI). At the top are the 240 palette colors: each column is a hue, with the full variations at the top and the soft ones at the bottom. Below them are the nine standard colors (1 red, 2 yellow, 3 green, 4 cyan, 5 blue, 6 magenta, 7 white/black, 8 and 9 grays) and the six grays from 250 to 255. Resting the mouse over a color shows its index and RGB value.

To choose, click the color and **OK**, double-click the color, or type the number in the **Color** field. Color 7 appears white on a dark background and black on a light background, as in AutoCAD.

Ribbon layer list



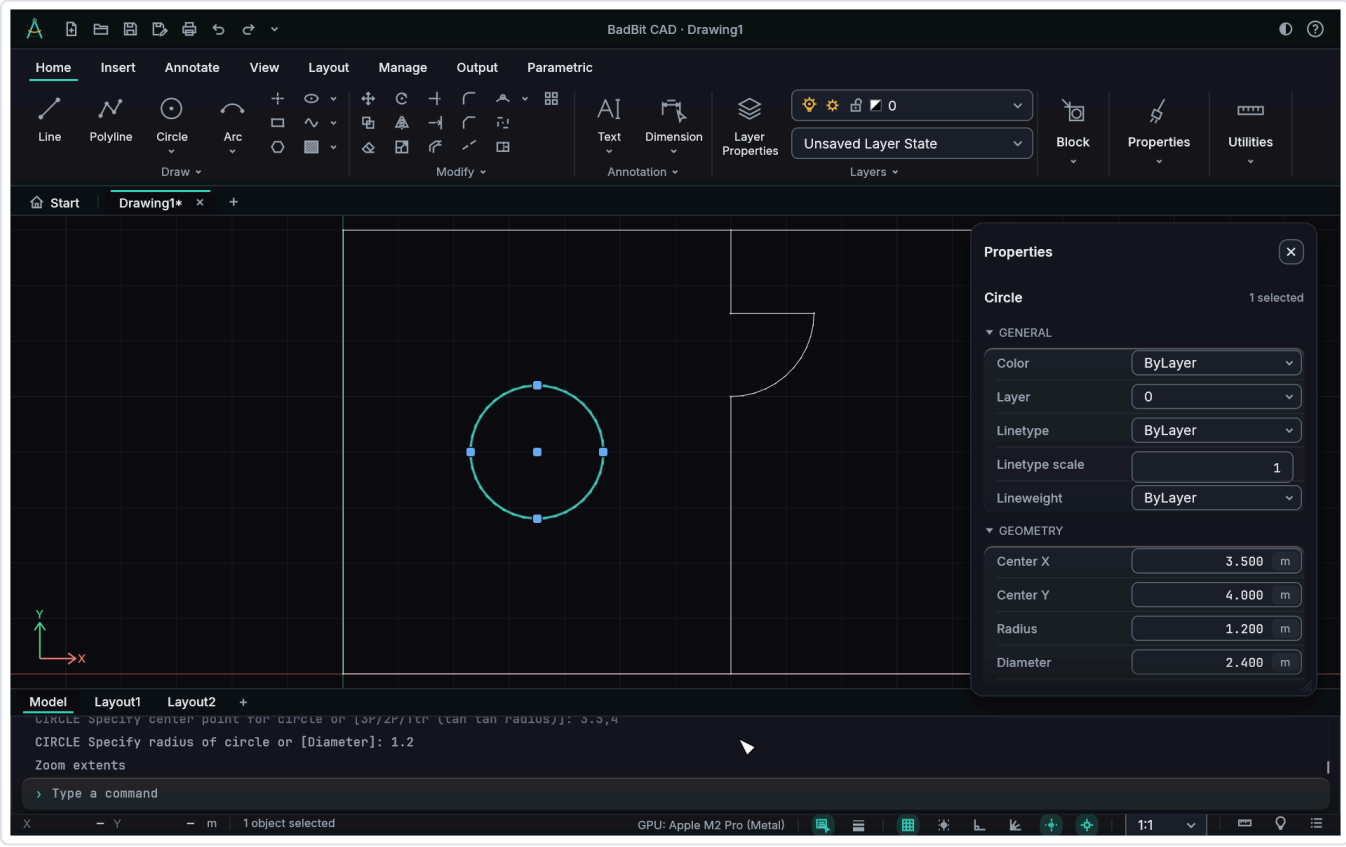
In the **Layers** panel of the Home tab, the list shows the current layer. With objects selected, it shows their layer, or stays blank if they are on different layers. When you open the list:

- Clicking the **name**, with nothing selected, makes the layer current.
- Clicking the **name** with objects selected moves the objects to that layer. The command line reports how many changed. Objects on a locked layer are left out, with a message.
- The light bulb, sun and padlock icons toggle the states right in the list, without opening the manager.

`Esc` or a click outside closes the list.

Object properties

Properties palette (PROPERTIES, Ctrl+1)



`Ctrl` + `1` opens and closes the palette. The `PR` command (or `PROPS`, `CH`, `MO`) and the **Properties** button on the ribbon also open the palette. Like the layer manager, it stays open while you draw and follows the selection. At the top are the object type and how many are selected.

General, for any object:

Property	Options
Color	ByLayer, the seven named colors, the explicit color in use and **Select Color...**, which opens the index color dialog box.
Layer	The drawing's layers. Moves the objects to the chosen layer.
Linetype	ByLayer, Continuous or one of the loaded linetypes; **Load...** loads others and gives the first one to the objects.
Linetype scale	The object's own scale (multiplies the global one, LTSCALE).
Lineweight	ByLayer or one of the standard values (0.00 to 2.11 mm).

The lineweight can also be changed in the **Lineweight** list of the Properties panel on the ribbon: with objects selected, it shows theirs ("mixed" if they differ) and changes theirs; with no selection, it shows and changes that of new objects (CELWEIGHT), as in AutoCAD.

With nothing selected, the palette shows what applies to new objects. In that case, changing **Layer** changes the current layer, and **Linetype** and **Linetype scale** change those of new objects (CELTYPE and CELTSCALE).

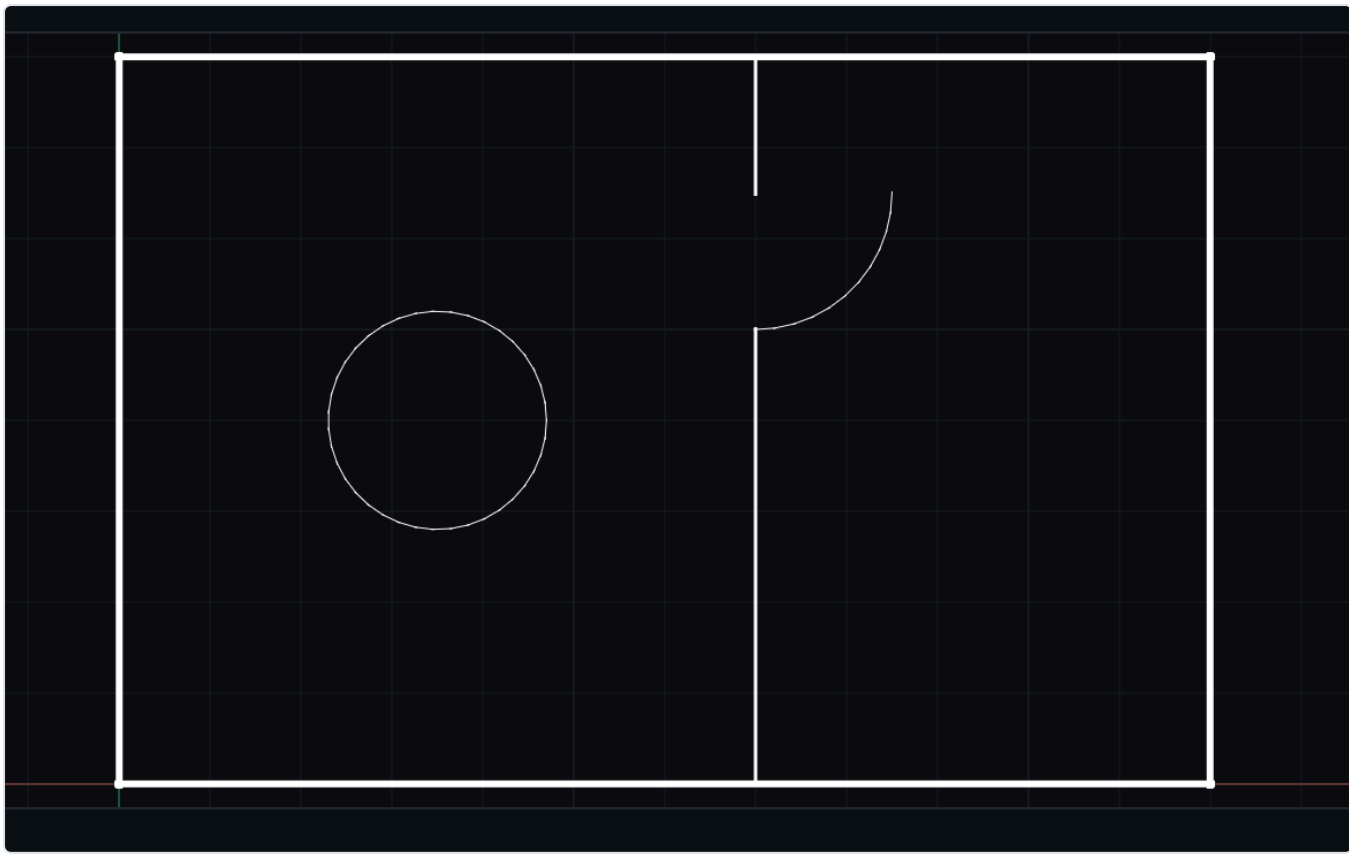
Geometry, depending on the type:

Object	Editable fields	Read-only
Line	Start X/Y, End X/Y	Delta X/Y, Length, Angle
Circle	Center X/Y, Radius, Diameter, Circumference, Area	—
Arc	Center X/Y, Radius, Start angle, End angle	Total angle, Arc length
Polyline	Vertex (chooses which), Vertex X/Y, Closed, Linetype generation	Length, Area (if closed)
Point	Position X/Y	—

The values follow the project's units and accept another unit when typed (`250 cm`, `45°`). The value is applied when you press `Enter` or leave the field, and `Esc` cancels. As in AutoCAD, changing a circle's diameter, circumference or area changes its radius. Changing an arc's start angle keeps the end angle, and vice versa.

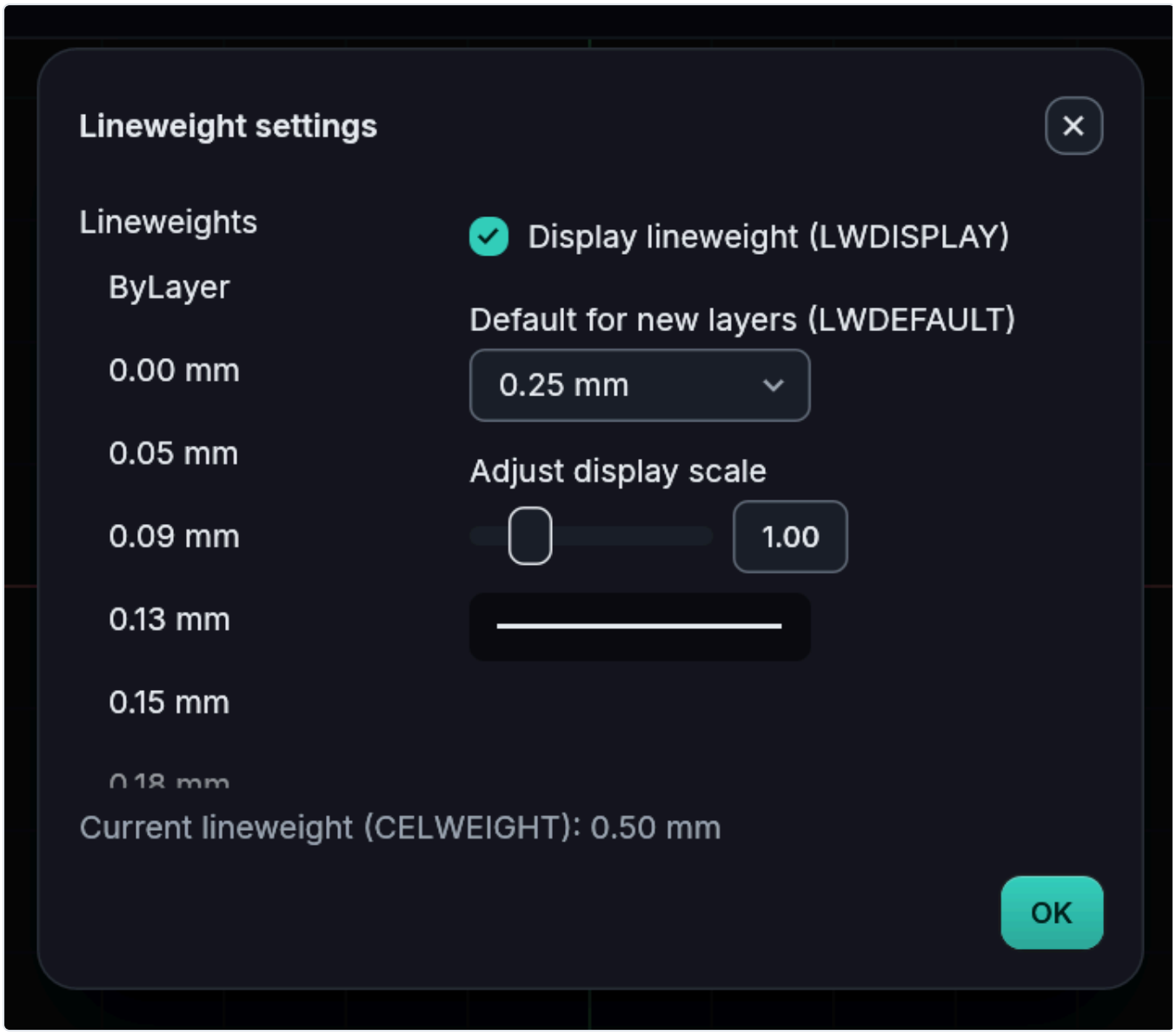
Lineweight on screen (LWDISPLAY, LWEIGHT)

As in AutoCAD, lineweights are not displayed on screen until they are turned on: the **Show/hide lineweight** button on the status bar (or the **LWDISPLAY** variable) turns them on and off. When on, each line is drawn with its lineweight: its own, or, for ByLayer, the layer's (a new layer has 0.25 mm). In model space the lineweight is measured in screen pixels and does not change with zoom: 0.25 mm (the default) or less is drawn thin, one pixel wide, and above that the width grows in whole pixels (0.50 mm is twice as wide as 0.25 mm). Lineweight 0.00 is always drawn thin, through the same fast path as always, even with millions of lines.



LWDISPLAY on: 1.00 mm walls, 0.50 mm partitions and the rest ByLayer (0.25 mm, thin).

The **LWEIGHT** command (alias `LW`, or `LINEWEIGHT`; or right-click the button on the status bar) opens **Lineweight settings**: the list chooses the lineweight of new objects (**CELWEIGHT**: ByLayer or a value), **Display lineweight** is LWDISPLAY, **Default for new layers** is **LWDEFAULT** (the lineweight the layer manager gives a new layer when no other layer is selected to copy from) and **Adjust display scale** makes all lineweights on screen heavier or lighter (from 0.25 to 4 times), with a sample beside it.



The LWEIGHT dialog box.

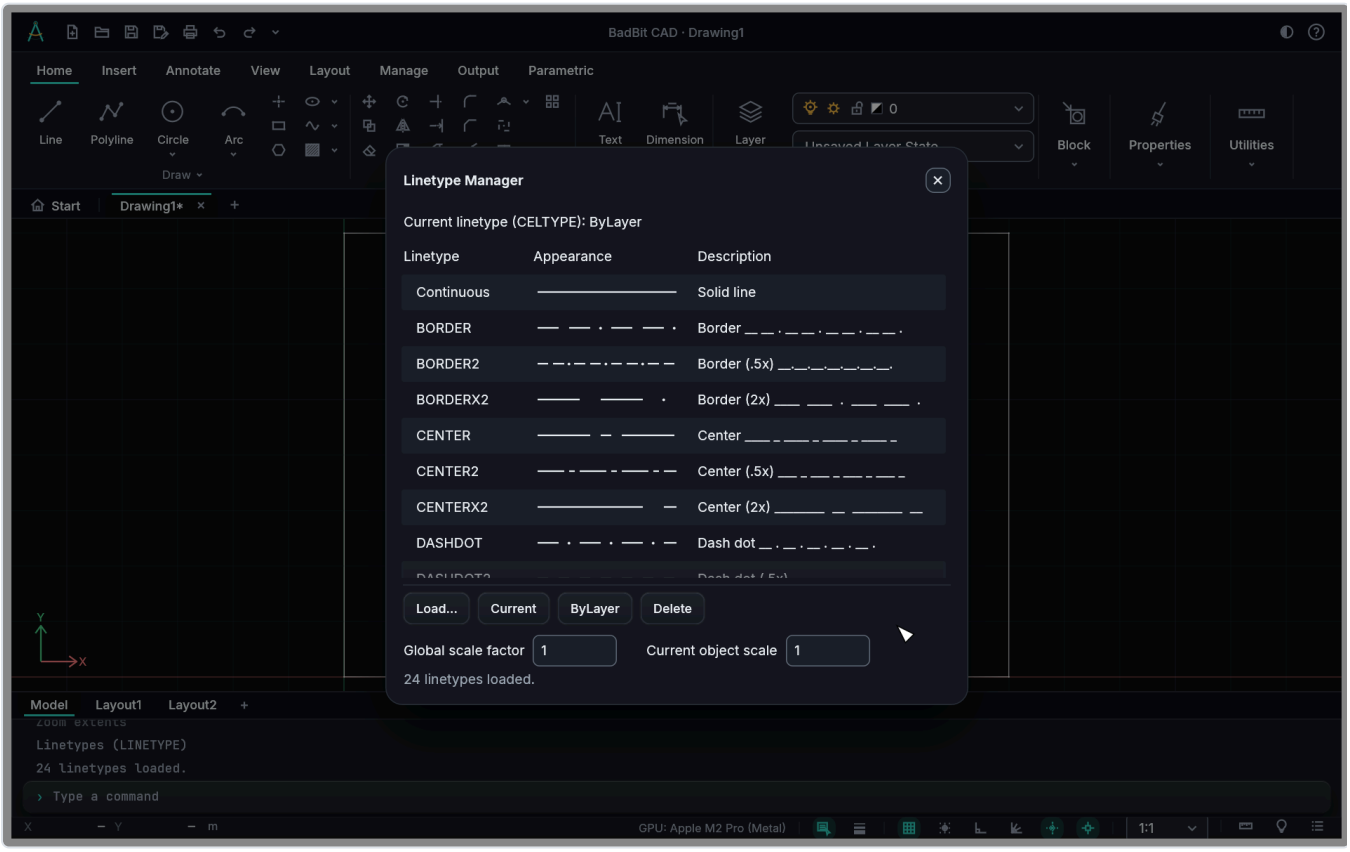
The display scale and LWDEFAULT are stored in the user profile (as in AutoCAD's registry); LWDISPLAY and CELWEIGHT belong to the open drawing. Difference from AutoCAD: the "Default" value in AutoCAD's lineweight list (which follows LWDEFAULT) does not exist in BadBit CAD yet; a new layer receives the LWDEFAULT value at the moment it is created.

Linetypes (LINETYPE, alias LT)

As in AutoCAD, a new drawing has only the continuous linetype (Continuous); the others are **loaded** before use. The LINETYPE command (or `LT`) opens the manager, with each loaded linetype, its sample and its description:

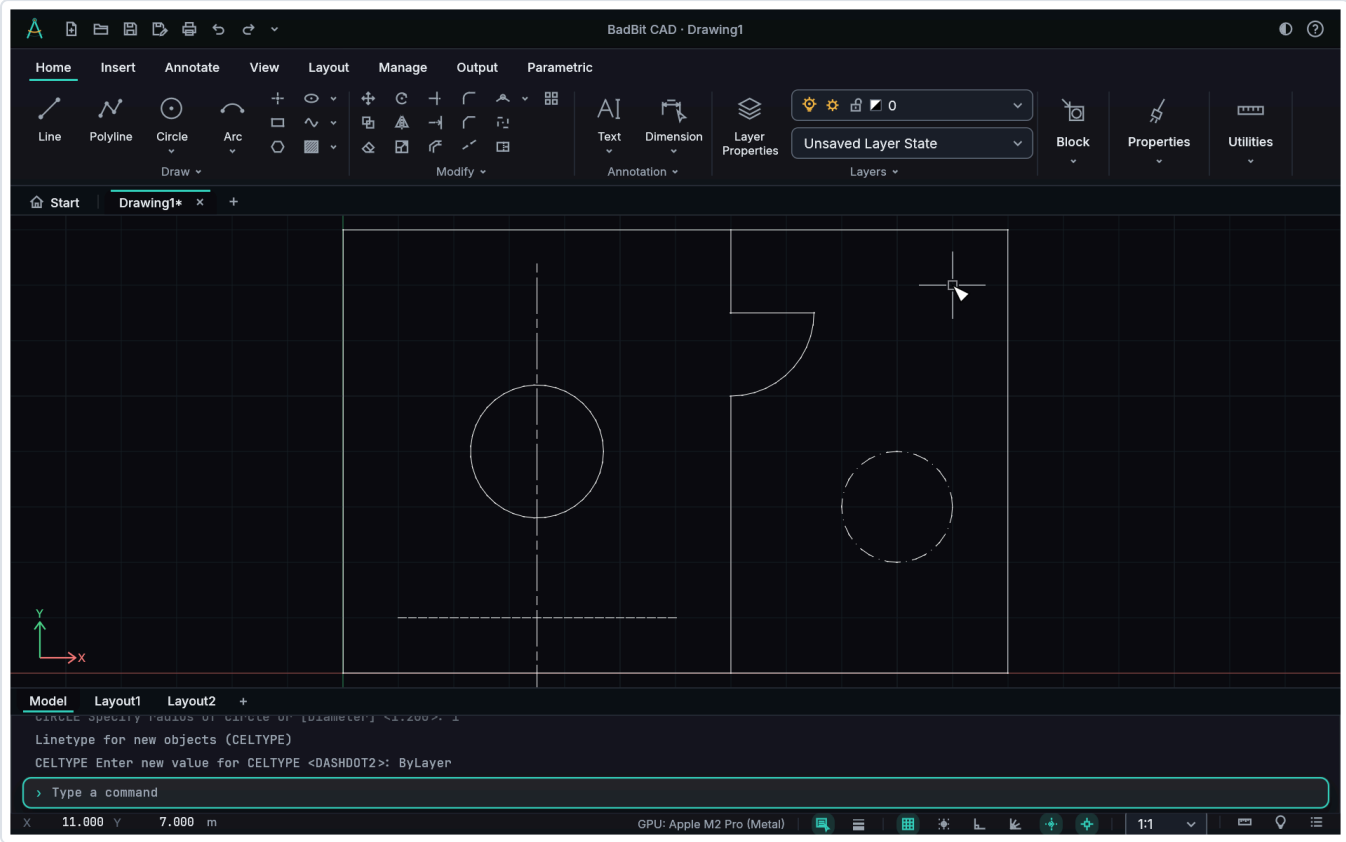
- **Load...**: opens the BadBit CAD library list (dashed, hidden, center, phantom, dot, dash-dot, border, divide, each also at half and double size) to check the ones to bring in. **.lin file...** loads the simple linetypes from an AutoCAD `.lin` file (those with text or shapes in them are left out).
- **Current**: the selected linetype becomes the one for new objects (CELTYPE). **ByLayer** returns to the layer's.

- **Delete**: removes the linetype from the drawing, if no object, no layer and CELTYPE use it.
- **Global scale factor** (LTSCALE) and **Current object scale** (CELTSCALE): the dash size is the pattern's times the global scale times the object's scale.



The manager with loaded linetypes and their samples.

Each object's linetype is chosen in the Properties palette (or stays ByLayer, and the layer's applies, chosen in the layer manager). Dashes appear on screen with and without the GPU, identically. The library patterns are in meters, designed for floor plans; the global scale adjusts them to the drawing's size. On a segment too short for one repetition of the pattern, or with dashes too small to see, the line is drawn continuous, as in AutoCAD.



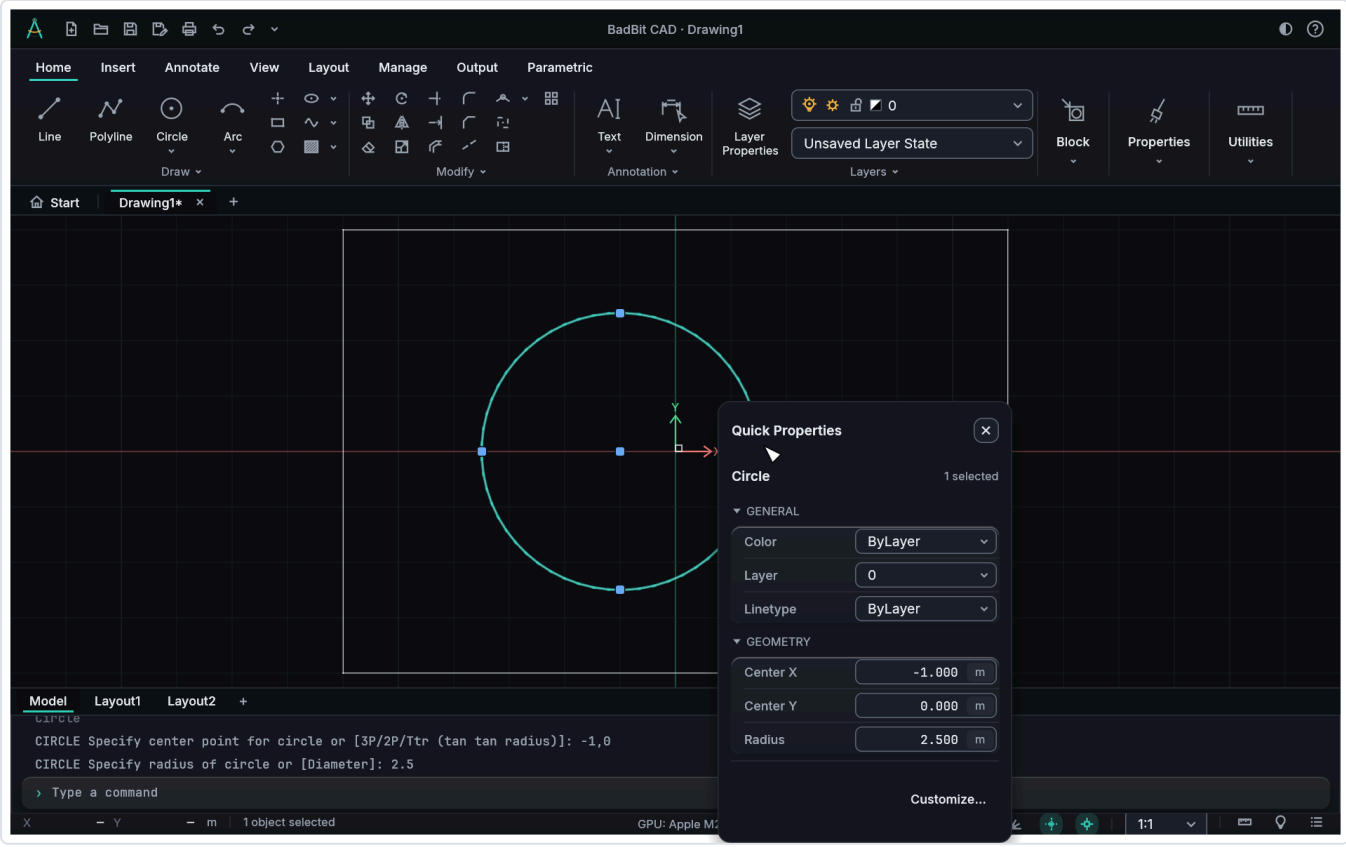
Walls dashed by layer and a center line axis.

On polylines, the pattern restarts at each vertex; with **Linetype generation** on (the PLINEGEN variable for new ones, PEDIT's `L` option or the Properties palette), it runs along the whole polyline, without breaking at the corners.

Quick Properties (QP, Ctrl+Shift+P)

With Quick Properties on, selecting objects (with no active command) opens a small window next to the cursor with the essentials of the selection, as in AutoCAD: color, layer and linetype, and the measurements for the type (the length of a line; the center and radius of a circle and an arc; the radii of an ellipse; the base point of a construction line...). The values are edited right there, as in the full palette, and each change is an undo step.

The small window can be dragged and closed (it comes back at the next selection). It does not appear with the Properties palette open or during a command. It is turned on and off with the **Quick Properties** button on the status bar or with `Ctrl` + `Shift` + `P`; it is on by default.



A circle selected: the small window with color, layer, linetype, center and radius.

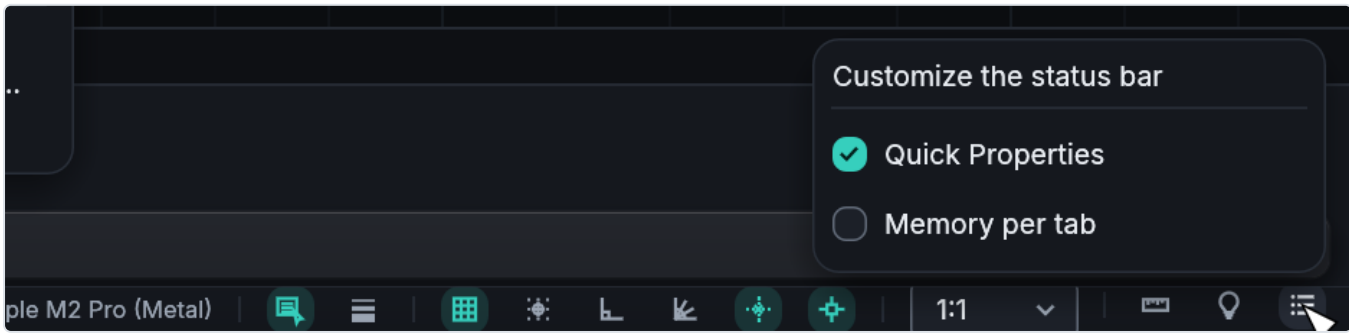
Where the small window appears (QPLOCATION)

The **QPLOCATION** variable chooses, as in AutoCAD: **0** (default) the small window moves next to the cursor at each new selection; **1** it is **floating**: it appears next to the cursor only the first time and then stays where you left it. **QUICKPROPERTIES** shows the Quick Properties of the selection right away, even with QPMODE 0.

Customizing the rows

The **Customize...** button, at the bottom of the small window, opens the **Customize User Interface (CUI): Quick Properties** dialog box, where you choose the rows shown for each object type (the **Object types** list and, below it, the checked **Properties**), like the Quick Properties part of AutoCAD's CUI. It starts on the type of the selection; what you check applies immediately and is stored in the profile.

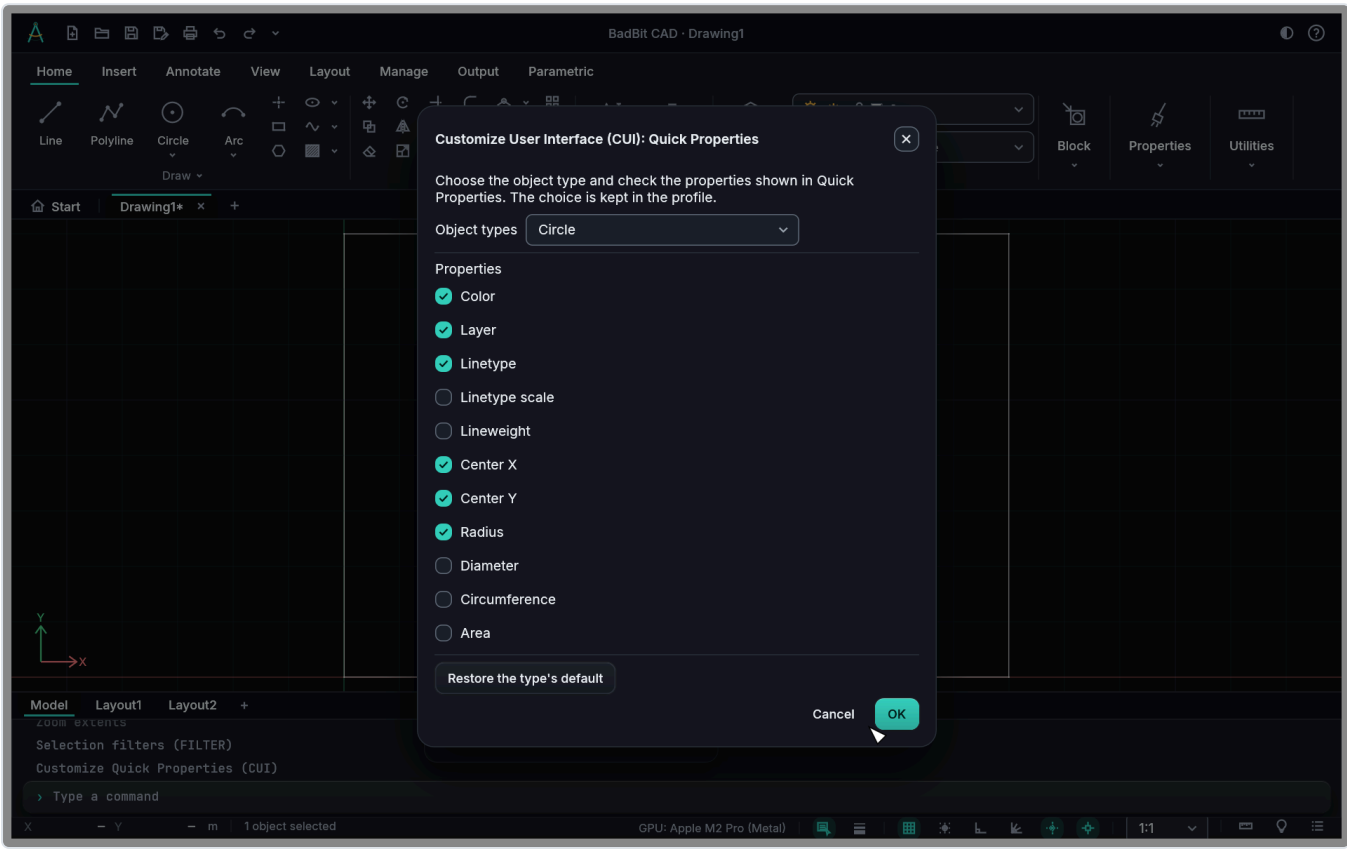
The **Quick Properties** button is removed from and returned to the status bar through the **Customize the status bar** menu (the last button, on the right), where it is checked by default.



The menu for customizing the status bar: Quick Properties and Memory per tab.

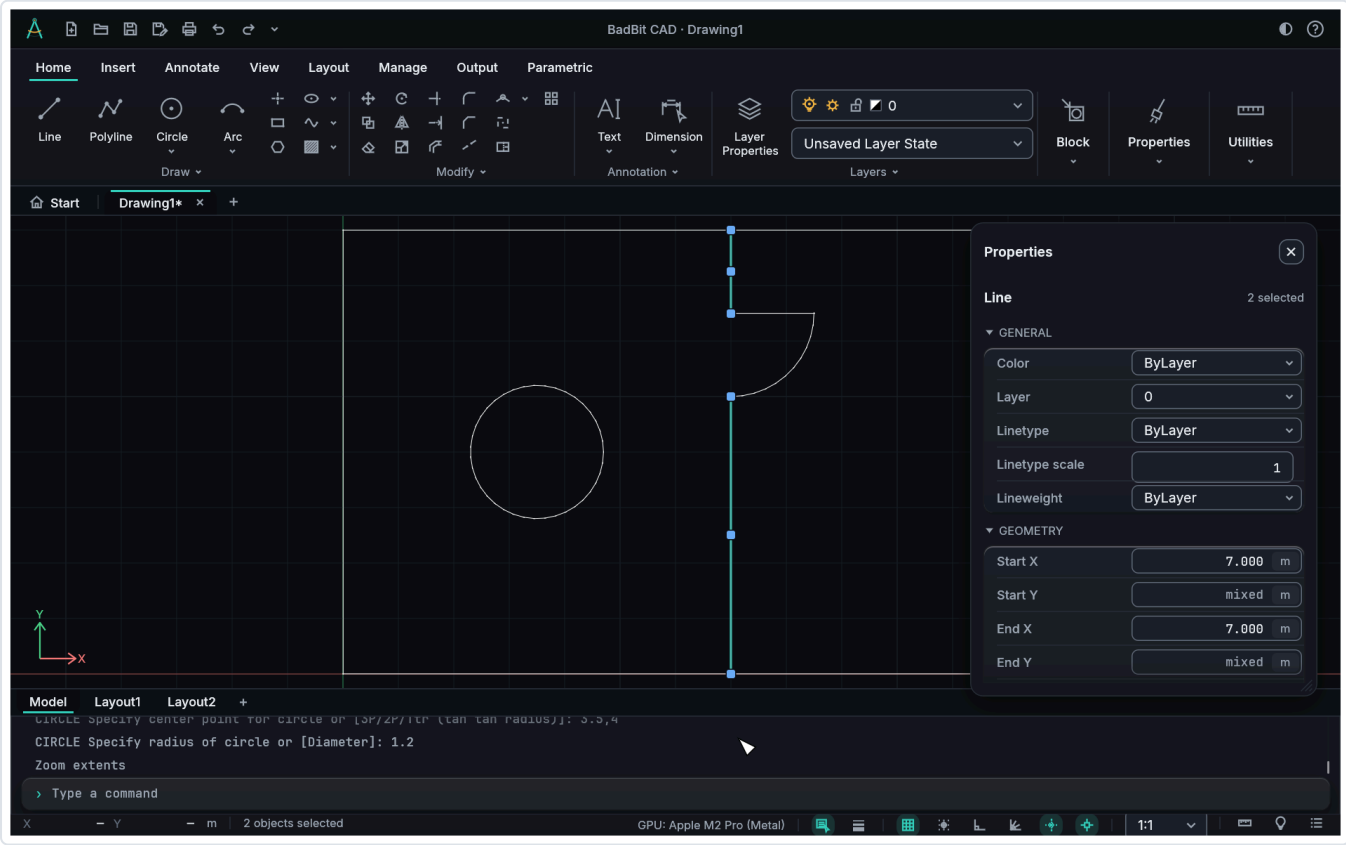
Customizing Quick Properties (CUI)

CUI opens the **Quick Properties** part of AutoCAD-style customization: choose the **object type** and check the **properties** shown in the small window (the list is that of the full palette for an object of that type). **Restore the type's default** returns to AutoCAD's choice. Customized types appear with `*`; the choice is stored in the **profile** (it goes along when the profile is exported and imported).



CUI: the circle's properties in Quick Properties (the default: color, layer, linetype, center and radius).

Multiple objects



With several objects of the same type, fields with different values show **mixed**. A value typed there applies to all of them, in a single undo step. With different types (for example, lines and circles), the top shows **All** and only the General group appears.

Objects on a locked layer are left out of the change, with a message. Meaningless values, such as a zero radius, are not accepted: the reason appears at the bottom of the palette and nothing changes. Each change in the palette is undone with `U` (`Ctrl`+`Z`).

Layer tools

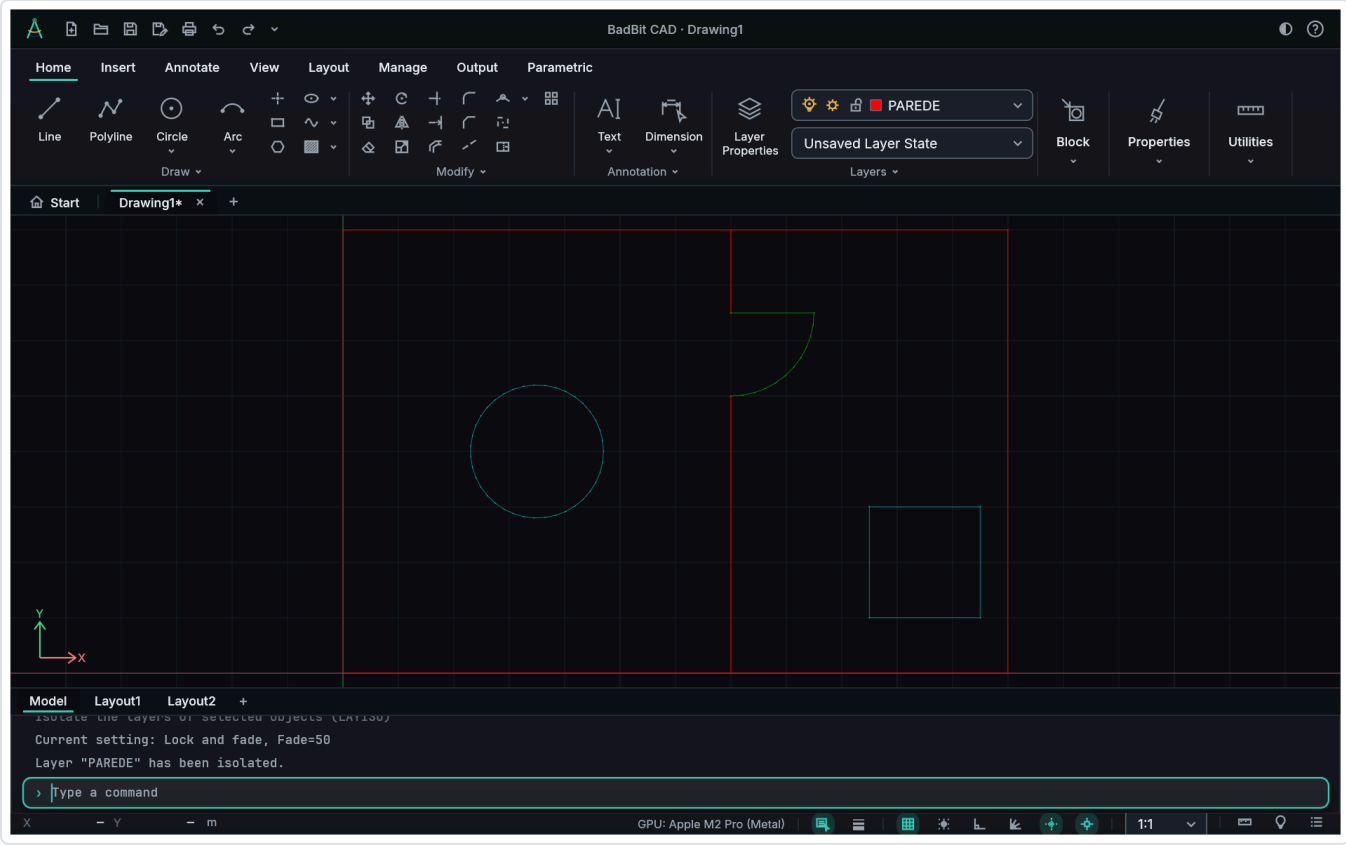
AutoCAD's layer tools (in the **Layers** panel of the ribbon, Home tab, expanded with the small arrow below the title, or typed). The prompts, the options (with AutoCAD's English letters in parentheses) and the refusals are AutoCAD's; each one is a single undo step (`U`).

Command	What it does
LAYISO	Isolates the layers of the chosen objects (before or after the command). The others are **locked and faded** (the default) or **turned off**, through the **Settings** option (`S`): Off (`O`), with **Vpfreeze** (`V` , freezes only in the layout viewport) or Off (`O`) in the viewport; Lock and fade (`L`), with the fade value (0 to 90). The current layer becomes one of the isolated ones.
LAYUNISO	Restores what the last LAYISO changed, except what you changed afterwards (as in AutoCAD: a layer you unlocked by hand stays unlocked). Works after saving and reopening.
LAYOFF	Click an object: its layer is turned off; repeats until `Enter` . **Undo** (`U`) restores the last one; **Settings** (`S`): **Viewports** (`V` : Vpfreeze or Off inside a layout viewport) and **Block selection** (`B` : **Block**, the reference's layer; **Entity**, the layer of the object inside the block; **None**). On the current layer, it asks first.
LAYON	Turns all layers on.
LAYFRZ	Like LAYOFF, freezing. The current layer is not frozen.
LAYTHW	Thaws all layers.
LAYLCK and **LAYULK**	Lock and unlock the layer of the clicked object.
LAYCUR	Moves the chosen objects to the current layer.
LAYMCH	Moves the chosen objects to the layer of a clicked object, or by **Name** (`N`).
COPYTOLAYER	Copies the chosen objects to the layer of a clicked object (or by **Name**, `N` , the default); then asks for the base point and the second point (or the **Displacement**, `D`) to move the copies; **eXit** (`X`) leaves them in place.
LAYMRG	Merges layers into one: click objects on the layers to merge (or **Type-it**, `T` , the name; `?` lists), `Enter` , and the target layer. After the warning (**Yes**, `Y`), the objects in all spaces, block items and attributes move to the target layer and the others are removed from the drawing.
LAYDEL	Deletes layers **with their objects** (also inside blocks), after the warning. **Type-it** (`T`) the name, **Undo** (`U`) the last one chosen.

AutoCAD's refusals: layer **0** and **Defpoints** cannot be deleted or merged into another; the current layer cannot be deleted (in LAYMRG it can: the target becomes current); the layers of an external reference (`file|layer`) cannot either.

Faded locked layers (LAYLOCKFADECTL)

Objects on locked layers appear faded, as in AutoCAD: the **LAYLOCKFADECTL** variable goes from 0 to 90 (% of fading; the default is 50). A negative value turns fading off while keeping the value (LAYISO with Lock and fade turns it on again). It is stored in the profile.

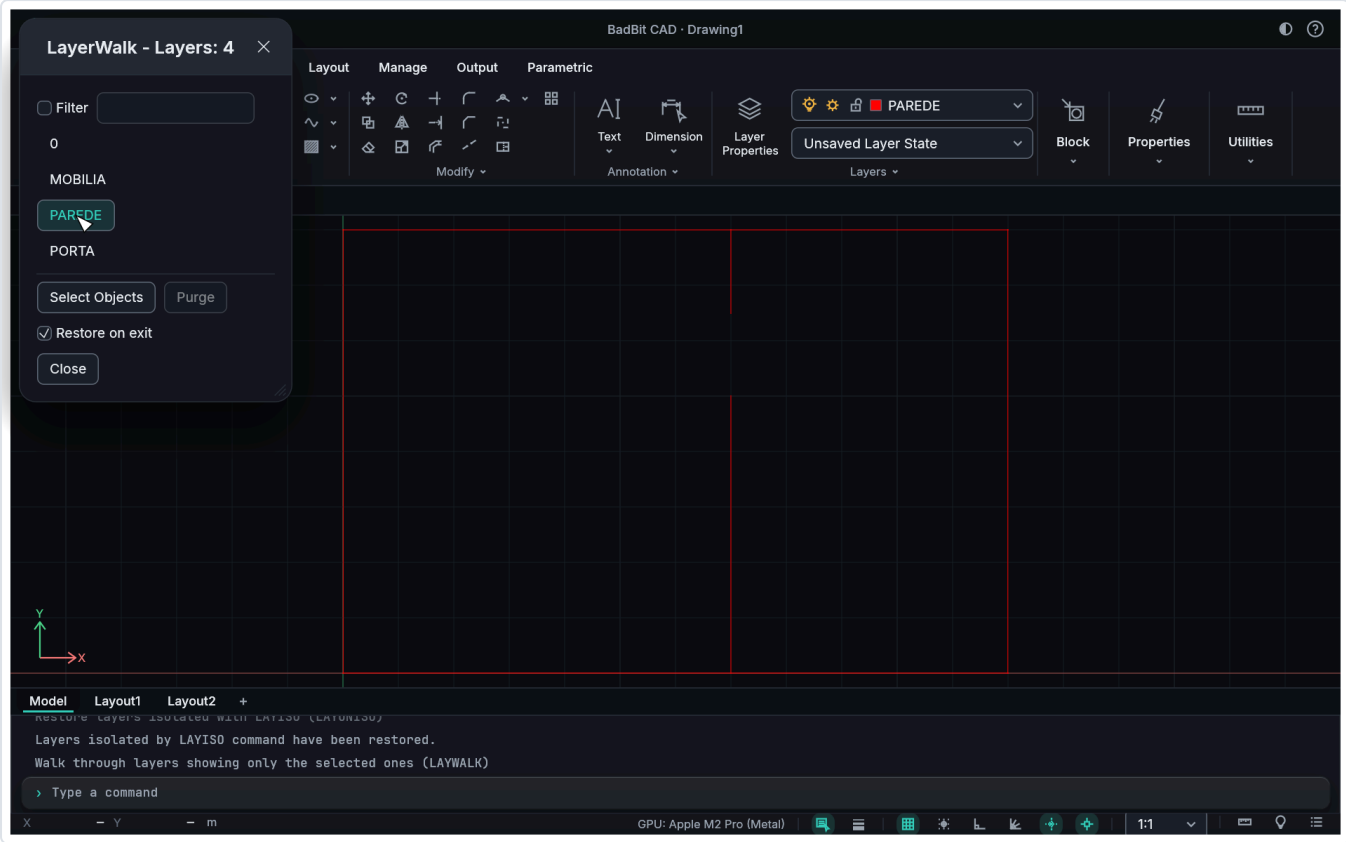


LAYISO on the wall: the other layers locked and faded (LAYLOCKFADECTL 50).

Walking through layers (LAYWALK)

The **LayerWalk** dialog box lists the drawing's layers; only the chosen ones are displayed in the drawing (the drawing does not change while the dialog box is open). A click chooses one; `Ctrl`+click adds or removes; `Shift`+click chooses a range. A double-click **holds** the layer (a `*` in front): it is always displayed. **Filter** shows only the names that match the pattern (`*` and `?`; several separated by commas). **Select Objects** hides the dialog box and chooses the layers of the objects you select. **Purge** deletes the chosen ones that have no objects at all.

With **Restore on exit** checked (the default), closing the dialog box leaves the drawing as it was; unchecked, the layers that were displayed stay on and the others off (one undo step).

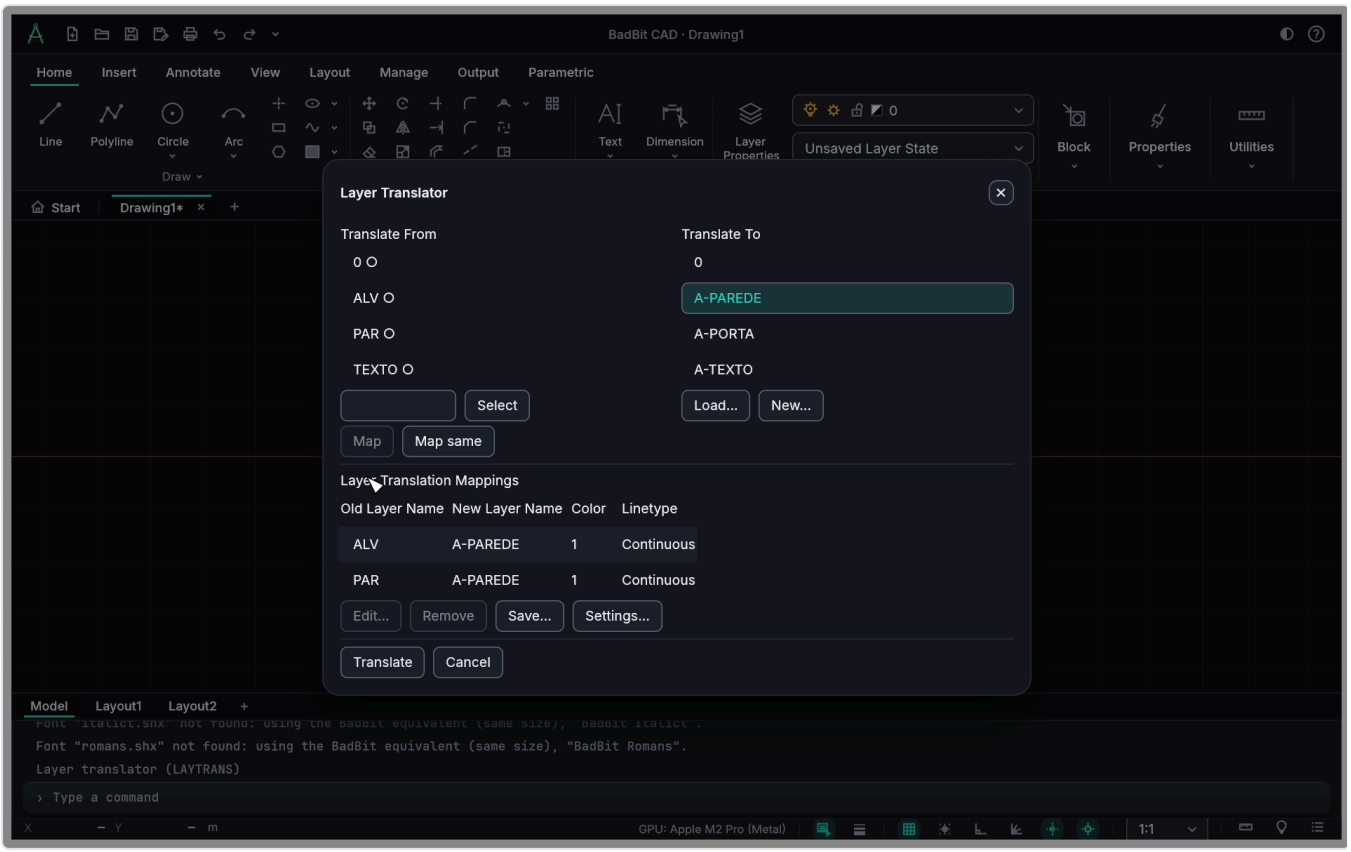


LayerWalk with the PAREDE layer chosen: only it is displayed.

Layer Translator (LAYTRANS)

Replaces the drawing's layers with those of an office standard. On the left, **Translate From**: the drawing's layers (those with no objects marked ○); **Select** chooses by name pattern. On the right, **Translate To**: **Load...** reads the layers of a DWG, DWT, DWS, DXF or `.bbd` (in the background) and **New...** creates one (name, color and linetype). **Map** links the chosen ones on the left to the chosen one on the right; **Map same**, those with the same name. In the **Layer Translation Mappings** list, **Edit...** changes the target layer's properties, **Remove** removes the mapping and **Save...** writes a `.bbd` with the target layers and the mappings (Load on that file brings the mappings back).

Settings...: force object color and linetype to ByLayer, translate objects inside blocks, write the transaction log (`drawing.log`, next to the drawing) and show the contents of the layers chosen on the left. **Translate** applies everything in one undo step: the target layers are created (or keep the file's properties), the objects move to them and the source layers are removed from the drawing.



The Layer Translator: PAR and ALV mapped to A-PAREDE, loaded from a standard.

Layer states (LAYERSTATE, LAS)

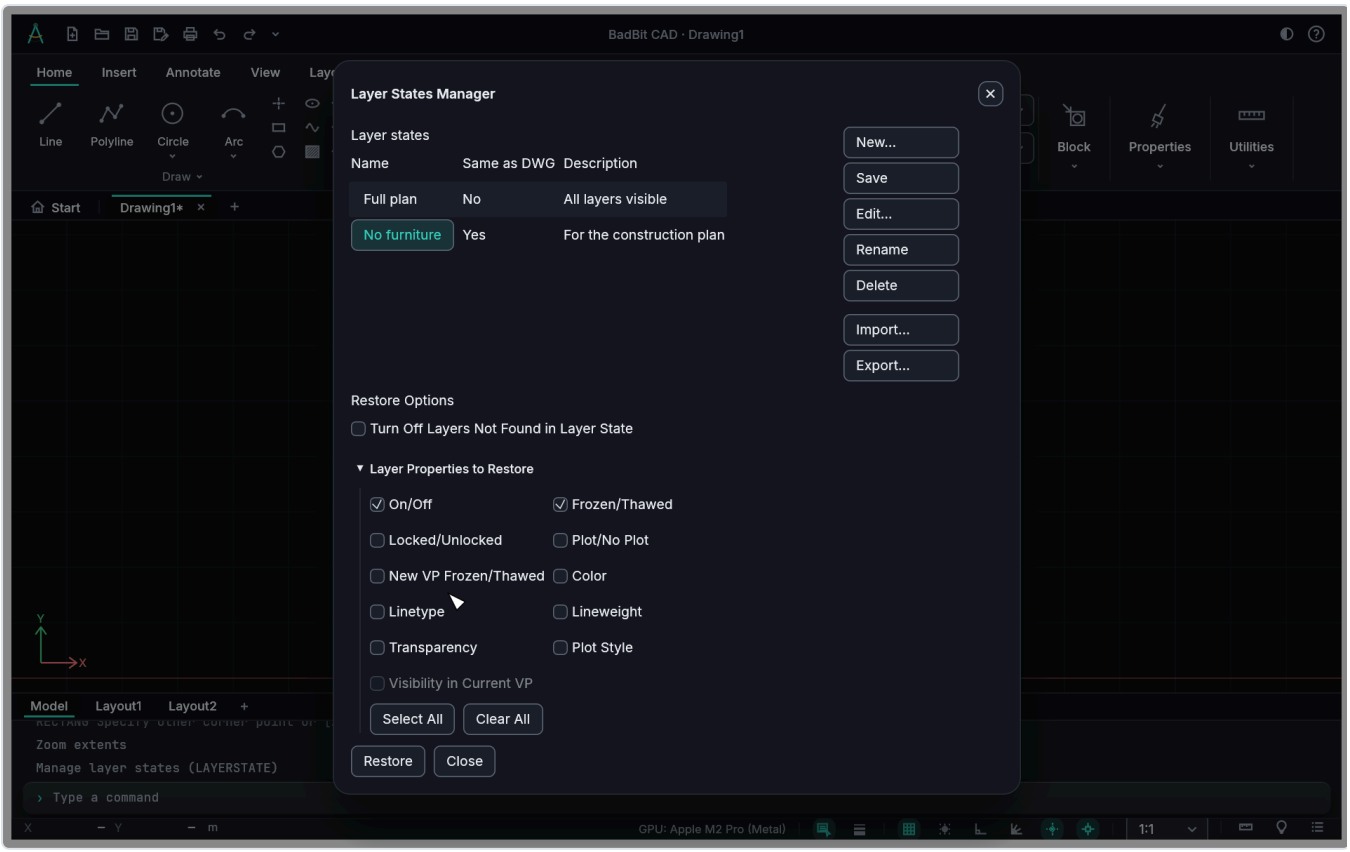
A **layer state** is a snapshot of the settings of all layers (on, frozen, locked, plot, color, linetype, lineweight, transparency, plot style, frozen in new viewport) and of the current layer, saved with a name in the drawing, so you can go back to it later, as in AutoCAD. The states go in the `.bbd` and in DWG/DXF (where AutoCAD stores them: a state saved in BadBit CAD appears in AutoCAD's Layer States Manager, and vice versa).

LAYERSTATE (alias **LAS**, or the **Layer States** button in the layer manager and in the expanded Layers panel) opens the **Layer States Manager**: the list of states (name, whether the layers currently match the state, and the description) and the buttons:

Button	What it does
New...	Saves the current layers as a new state (name and description; if the name already exists, it asks before replacing). It is the same as the **LAYERSTATESAVE** command.
Save	Replaces what is stored in the chosen state with the current layers, with the properties to restore checked below.
Edit...	Opens the state: the stored layers, with on, frozen, locked, plot, color, linetype and lineweight for each; removes layers from the state or adds the missing drawing layers.
Rename and **Delete**	Rename (the name becomes editable in the list; `Enter` confirms) and delete the state.
Import...	Brings in states from a `.las` or from another drawing (DWG, DWS, DWT, DXF or `.bbd`, read in the background): choose the states in the list; those with the same name are only replaced with **Replace states with the same name**. The state's layers that the drawing does not have are created (with the Continuous linetype if the drawing does not have theirs).
Export...	Writes the chosen state to a `.las`.
Restore	Applies the chosen state (also by double-clicking in the list) and makes current the layer that was current when it was saved.

Restore Options: **Turn Off Layers Not Found in Layer State** turns off the layers created after the state (so that the drawing is as it was). In **Layer Properties to Restore** (click the title to expand) you check what comes back: only what is checked changes (**Select All** and **Clear All** help). **Visibility in Current VP** applies only with an active layout viewport: the layers that are off or frozen in the state are frozen only in that viewport, and the others stay on and thawed. Transparency, plot style and new viewport freeze are stored in the state and go to and come back from the file, but BadBit CAD layers do not have these properties yet: when checked, they change nothing.

In the **Layers** panel of the ribbon, below the layer list, the state selector shows the state that matches the current layers (or **Unsaved Layer State**); choosing a state restores it immediately; **New Layer State...** and **Manage Layer States...** open the dialog boxes. Each change to the states and each Restore is a single undo step (`U`).



The Layer States Manager with two states; "No furniture" restores only on and frozen.

The `.las` is AutoCAD's layer states file. Autodesk does not publish the format: BadBit CAD writes the same codes AutoCAD uses in DWG, with the names of the layers and linetypes (the specification is in `docs/formato/`, section 12.10).

Layer of the current object (LAYMCUR)

LAYMCUR makes the layer of the clicked object current (it asks "Select object whose layer will become current:"). With an object already chosen before the command, its layer is used, without asking. A frozen layer cannot be current.

Previous layer (LAYERP, LAYERPMODE)

LAYERP undoes the last change to layer settings, one at a time: color, linetype, lineweight, on, frozen, locked, plot, the current layer — made in the manager, in the ribbon list, with the layer tools or by restoring a state. Objects do not change (for them, use `U`). As in AutoCAD, a renamed layer goes back to its previous properties but keeps the new name, and a deleted or created layer neither comes back nor goes away. LAYERP itself is undone with `U`.

LAYERPMODE turns tracking on and off (it asks **ON** (`ON`) or **OFF** (`OFF`)). When off, LAYERP says that it is not tracking changes; turning it off forgets what was stored. Tracking belongs to the session and starts on for each drawing opened.

Color, ByLayer and names

Color of new objects (COLOR, COL)

COLOR (aliases **COL**, **COLOUR** and **DDCOLOR**) opens the **Select Color** dialog box, the same as in the text editor, with AutoCAD's three tabs: **Index Color** (the grid of the 255 colors and the **ByLayer** and **ByBlock** buttons), **True Color** (RGB or HSL, with the "R,G,B" field) and **Color Books** (BadBit CAD's own book, BADBIT TONS; no third-party books). The chosen color applies to the objects you create from now on (the **CECOLOR** variable); existing ones do not change (for them, use the Properties palette).

-COLOR (**-COL**, **-COLOUR**) does the same from the command line: "Enter default object color [Truecolor/COlorbook] <BYLAYER>:". You can type `BYLAYER`, `BYBLOCK`, an index from 1 to 255, the name of one of the first seven colors (`red`, `yellow`, `green`, `cyan`, `blue`, `magenta`, `white`; also in Portuguese, such as `azul`) or `r,g,b`. `T` asks "Red, Green, Blue:"; `CO` asks for the book name (BADBIT TONS) and the color name. `Enter` keeps the current color. The current color goes in DXF and DWG (`$CECOLOR`, with the true color in group 420), as in AutoCAD.

Changing to ByLayer (SETBYLAYER)

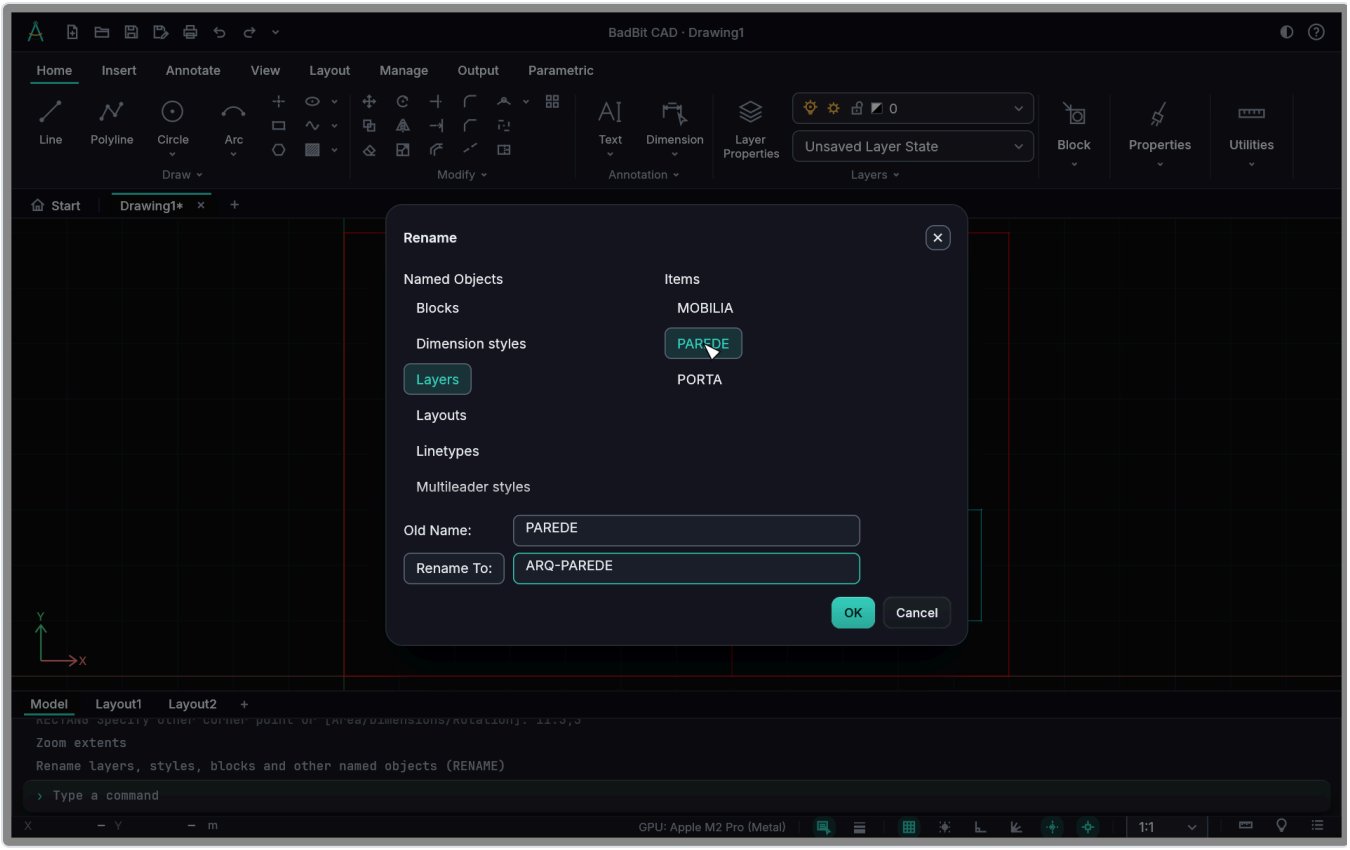
SETBYLAYER shows the active properties ("Current active settings: Color Linetype Lineweight...") and asks "Select objects or [Settings]:". `S` opens the **SetByLayer Settings** dialog box, with the Color, Linetype, Lineweight, Material, Plot Style and Transparency check boxes (the **SETBYLAYERMODE** variable, in the profile). With the objects chosen (or chosen before the command), AutoCAD's two prompts follow:

- **Change ByBlock to ByLayer? [Yes/No] <Yes>:** with **No**, properties that are ByBlock stay as they are.
- **Include blocks? [Yes/No] <Yes>:** with **Yes**, the objects inside the definitions of the chosen blocks (and of the blocks nested in them) also change to ByLayer, and all references of those blocks follow; the attributes of the chosen references also change.

BadBit CAD changes the color, linetype and lineweight; material, plot style and transparency are kept in SETBYLAYERMODE for when objects have these properties. Objects on a locked layer are left out (with a message). Everything is a single undo step (`U`).

Renaming named objects (RENAME, REN)

RENAME (alias **REN**) opens the **Rename** dialog box: on the left, the **Named Objects** (Blocks, Dimension styles, Layers, Layouts, Linetypes, Multileader styles, Table styles, Text styles, UCSs, Views and Viewport configurations); on the right, the **Items** of the chosen type. Click an item: its name goes to **Old Name** and to the field next to **Rename To:**. Change the name and click **Rename To:** (or **OK**, which renames and closes).



The Rename dialog box with the PAREDE layer chosen, becoming ARQ-PAREDE.

Everything that refers to the object by name changes along with it: the block's references (in the drawing and inside other blocks), block items and attributes on the layer, layers and objects with the linetype (and the current linetype), texts, dimensions, multileaders, tables and styles that use the style. **Old Name** accepts the wildcards `*` and `?` and the new name repeats them in place of what matched: `A-*` to `ARQ-*` renames A-PAREDE and A-PORTA to ARQ-PAREDE and ARQ-PORTA, in one step. These do not change (and do not appear in the list), as in AutoCAD: layer 0 and Defpoints, the Continuous, ByLayer and ByBlock linetypes, the Standard styles, anonymous blocks and names dependent on an external reference (`file|name`). The new name cannot be empty, be longer than 255 characters or contain `< > / \ " : ; ? * | , = '` or the backquote; a duplicate name (case-insensitive) is rejected with a message. Each rename is a single undo step.

-RENAME (**-REN**) asks on the command line: "Enter object type to rename [Block/Dimstyle/LAyer/LType/multileadeRstyle/textStyle/Tablestyle/Ucs/VIew/VPort]:", then the old name and the new one (with the same wildcards). Material and plot style do not exist in BadBit CAD yet (-RENAME says so). Renaming a layer in the manager (F2 or double-clicking the name) uses the same renamer, including block items.

Files

Drawings are saved in the `.bbd` format: a single file, robust against crashes and readable by other programs (the specification is public). The commands and shortcuts are AutoCAD's. OPEN and SAVEAS also work with AutoCAD's **DWG** and **DXF** (see the *DXF and DWG files* chapter).

Commands

Command	Shortcut	What it does
NEW	`Ctrl` + `N`	Starts a new drawing in a new tab (the open drawing stays in its tab).
CLOSE	—	Closes the drawing's tab, asking first if there are unsaved changes.
CLOSEALL	—	Closes all tabs, asking about each drawing with unsaved changes (see below).
OPEN	`Ctrl` + `O`	Opens a `.bbd` file in a new tab (or goes to its tab, if it is already open).
QSAVE	`Ctrl` + `S`	Saves to the drawing's file. A drawing that does not have a name yet asks for one, like Save As.
SAVEAS	`Ctrl` + `Shift` + `S`	Saves with another name, in another folder or in another format (`.bbd`, DXF or DWG, choosing the version). The extension is added if missing.

New, Open, Save and Save As are also on the Quick Access Toolbar, at the top left. The file name appears in the title and on the drawing's tab; a `*` on the tab indicates unsaved changes. Several drawings stay open at the same time, each in its own tab (see *File tabs*, in the interface chapter).

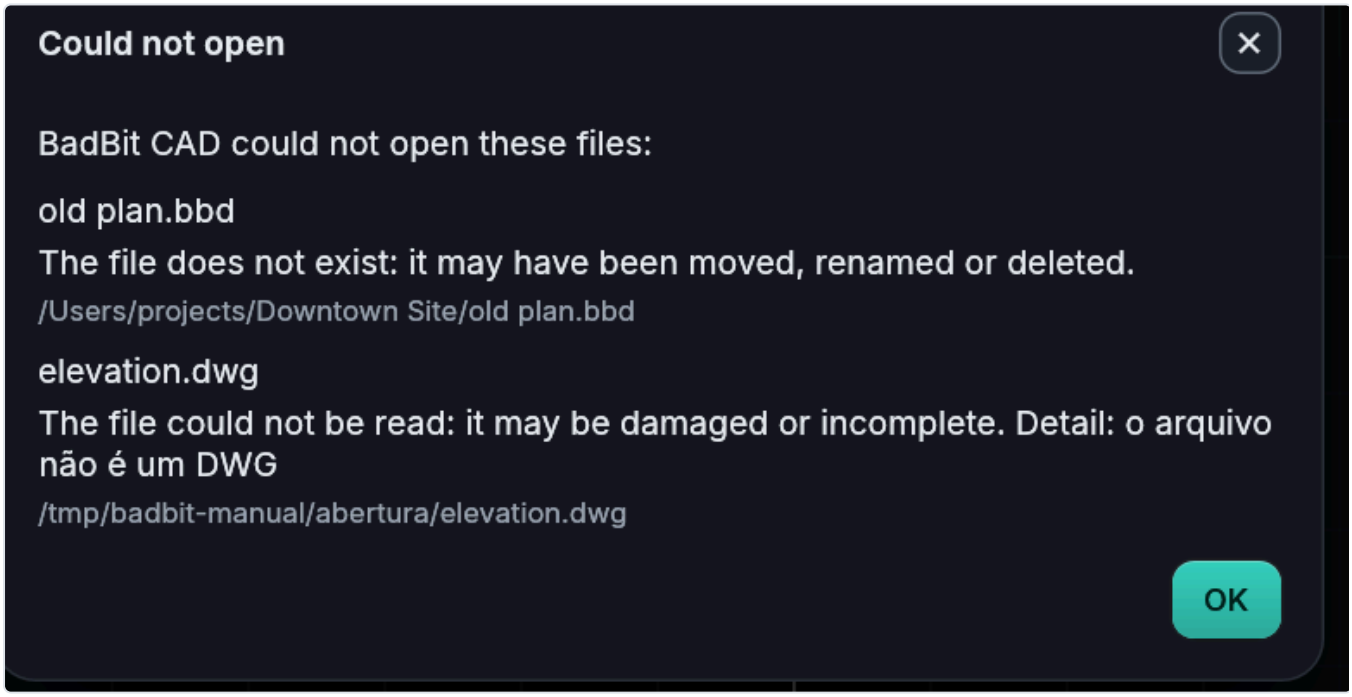
A drawing saved in a layout (AutoCAD's `TILEMODE` 0) opens in that layout; one saved in Model opens in Model. This applies to `.bbd`, DXF and DWG.

Opening from Finder or File Explorer (double-click)

Double-clicking a `.bbd` file opens the drawing in BadBit CAD, like OPEN. A DWG, DXF or DWT opens the same way through **Open with ▸ BadBit CAD** on the right-click menu (BadBit CAD is added as an option, without replacing the default program for those files).

- **With the program closed**, it opens directly with the drawing, in place of the Start tab.
- **With the program open**, the drawing goes to a **new tab** in the same window (it does not open another copy of the program), and the window comes to the front.
- **Several at once** (selected together in Finder, or dropped on the BadBit CAD icon in the Dock): each one opens in its own tab, one after the other, and the last one stays in view. Each file is read in the background, as with OPEN: the window keeps responding while large drawings load.
- **File already open**: BadBit CAD just goes to its tab, without opening it again.
- Opening prompts come one at a time: a password-protected DWG, a unitless DXF or a drawing with changes recovered from a crash asks before the next file opens.

If a file does not open, the others open normally and the **Could not open** dialog box says which one and why: the file does not exist (it was moved, renamed or deleted), you do not have permission to read it, it is not a type BadBit CAD opens, or it could not be read (damaged or incomplete, with the error detail). The same message stays on the command line. **OK** closes the dialog box and the program carries on.



On Windows and Linux the file path reaches the program through the command line (`badbit-shell planta.bbd`, which also accepts several files): BadBit CAD opens each one in a tab, with the same messages. With the program already open, the same applies as on the Mac: the drawing goes to a new tab in the open window, which comes to the front (restored from the taskbar, if it was minimized). File names with accented characters written by older systems (outside UTF-8, common on Linux disks and network folders) open normally.

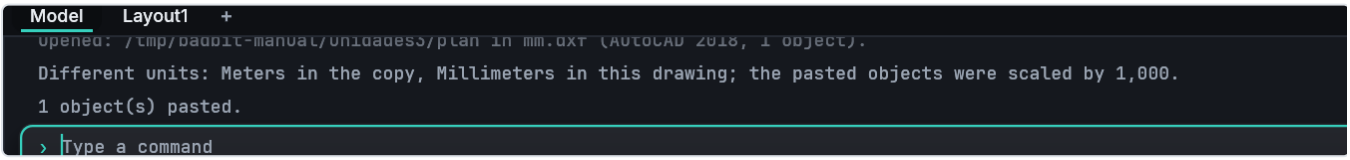
On Windows and Linux, starting BadBit CAD from the Start menu or a shortcut, with no file, while the program is already open, opens **another session** of the program, as in AutoCAD. Files opened afterwards by double-clicking go to the first session; when it closes, the other one starts receiving them.

Copying and pasting between drawings

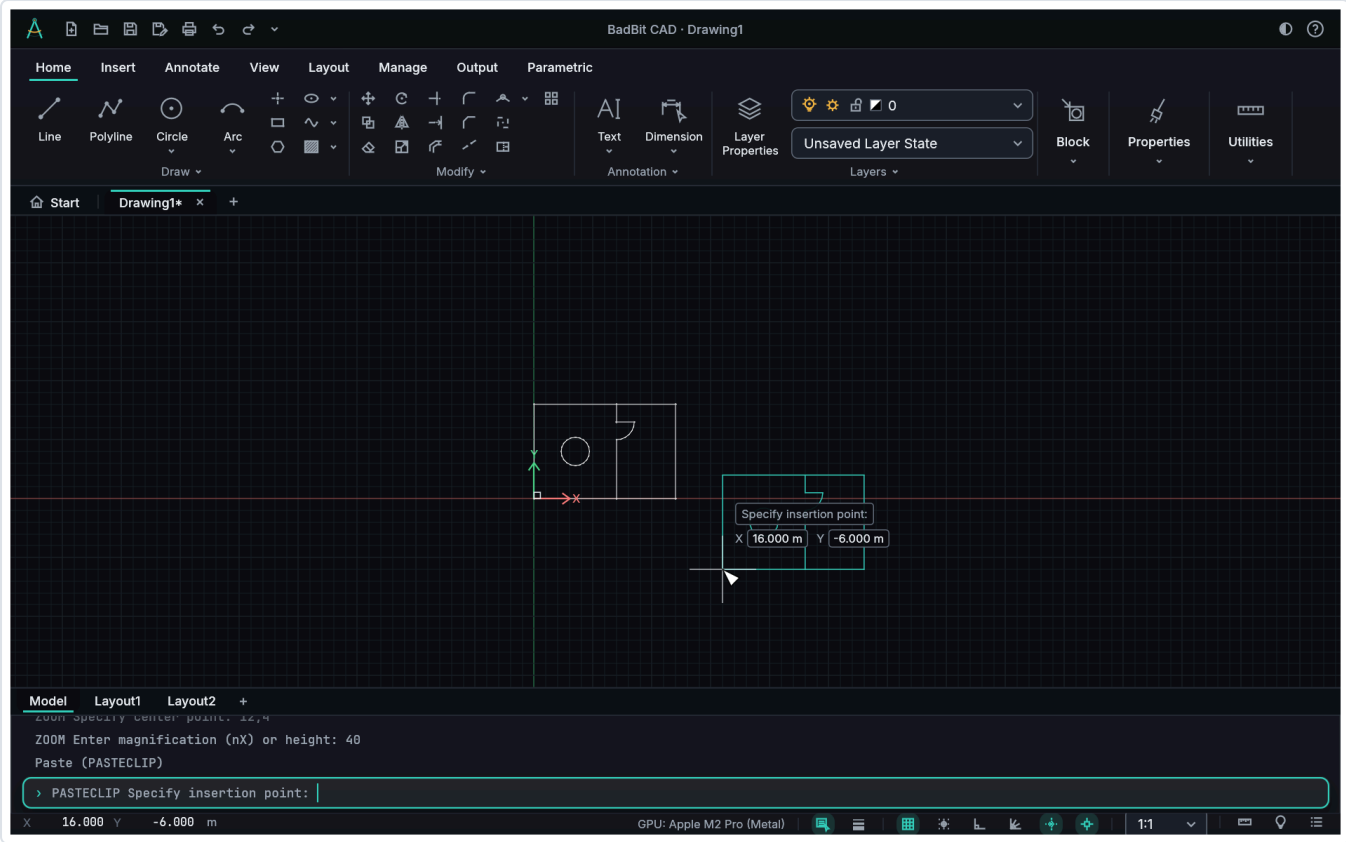
Command	Shortcut	What it does
COPYCLIP	`Ctrl` + `C`	Copies the selected objects to the clipboard, with the base point at their lower left corner.
COPYBASE	`Ctrl` + `Shift` + `C`	Copies the selected objects, asking for the base point.
PASTECLIP	`Ctrl` + `V`	Pastes the objects, asking for the insertion point (the base point lands on it).
PASTEBLOCK	`Ctrl` + `Shift` + `V`	Pastes the objects as an anonymous block (`A$C…`), asking for the insertion point.
PASTEORIG	—	Pastes the objects at the same coordinates as in the drawing they came from.
CUTCLIP	`Ctrl` + `X`	Cuts: copies the selected objects like COPYCLIP and erases them from the drawing.
CUTBASE	`Ctrl` + `Shift` + `X`	Cuts, asking for the base point, like COPYBASE.

With no selection, COPYCLIP (and `Ctrl` + `C`) asks `Select objects:` as AutoCAD does, and COPYBASE asks for the base point and then the objects; `Enter` with nothing chosen cancels. CUTCLIP and CUTBASE ask the same way; the copy goes to the clipboard in the same format as COPYCLIP (PASTECLIP pastes it back, in this or another tab) and the erasing is **a single undo step** (`Undone: CUTCLIP`). Objects on locked layers go into the copy and stay in the drawing, with a message saying how many were on a locked layer, as with ERASE.

On the ribbon, with a wide window (the ribbon at 1,400 points or more; narrower, the panel is dropped so the layer list fits), the **Clipboard** panel, at the end of the Home tab, has **Paste** (with Paste as Block and Paste to Original Coordinates in its list), **Cut** and **Copy Clip** (with Copy with Base Point in its list), as in AutoCAD. The same commands are in the Clipboard submenu of the right-click menu with a selection. The shortcuts work with the drawing in focus and the command line empty; in a text being edited or a table cell, `Ctrl` + `C` and `Ctrl` + `V` belong to the text.



A polygon copied from a survey in meters and pasted into a floor plan in mm: the factor of 1000 stated on the command line.



PASTECLIP with the copied floor plan: the objects follow the cursor until the click.

In PASTECLIP and PASTEBLOCK, the copied objects appear **following the cursor**, with the base point on it (already sized for this drawing's unit), until the click or the typed point; `Esc` cancels. With many objects (more than 2,000), the preview shows the first ones and the outline of all of them.

The insertion point (and COPYBASE's base point) is asked for as in any command: **object snaps** (`F3`; clicking near the end of a line pastes exactly on it), a snap typed for that point only (`MID`, `END`, `NON`...), the **grid** (`F9`) and **tracking** from acquired points (`F11`, with polar `F10`) all work the same.

Pasting brings along the **layers, linetypes, styles and blocks** that the objects use and that are missing from the drawing (those that already exist stay as they are in the drawing, as in AutoCAD), in a **single undo step**. Between drawings with **different units**, the objects are converted by the factor between the units, as with INSERT (from m to mm, 1000; from mm to m, 0.001), with the paste's own message on the command line: `Different units: Meters in the copy, Millimeters in this drawing; the pasted objects were scaled by 1,000.`

The system clipboard also receives the copied objects, as DXF text (with the unit and the base point): pasting into a BadBit CAD open in another window (another process) reads that text. Text that is not a drawing is not pasted into the drawing (the command line says so).

With **FILEDIA** at 1 (the default), opening and saving show the system's file dialog box. With FILEDIA at 0, the name is typed on the command line (spaces in the name are allowed; `Enter` confirms), useful in scripts and for pasting paths.

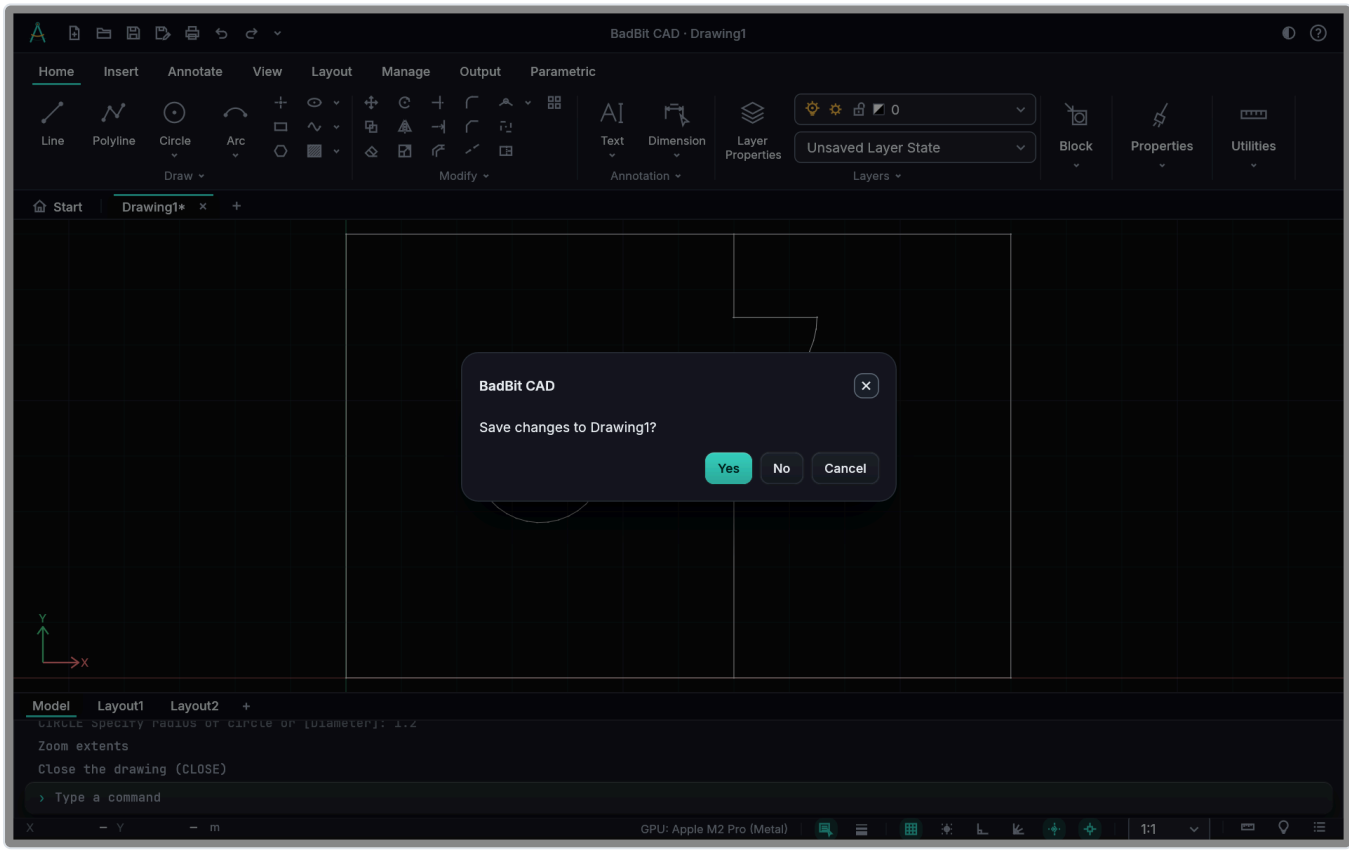
Purging the drawing (PURGE, aliases PU, -PU)

Removes from the drawing what is not in use, like AutoCAD's -PURGE on the command line. Choose the type:

Option	What it purges
Blocks (`B`)	Block definitions that no reference uses.
LAyers (`LA`)	Layers with no objects or block items (never "0", the current layer or Defpoints).
LTypes (`LT`)	Linetypes that no layer, object or the current linetype uses (never Continuous).
All (`A`)	Blocks, layers and linetypes.
Zero-length geometry (`Z`)	Lines, arcs, circles and polylines with no length.
Empty text objects (`E`)	Texts and multiline texts containing only spaces.

For blocks, layers and linetypes, give the **names** (`*` and `?` are allowed, several separated by commas; `Enter` is `*`) and whether you want to **verify each one** (`Y` : asks by name, with `Enter` keeping it). Objects on locked layers stay. Everything is a single undo step.

Save changes?



Closing the tab (CLOSE or the `×`) and closing the window with unsaved changes ask before discarding them (NEW and OPEN go to a new tab and lose nothing):

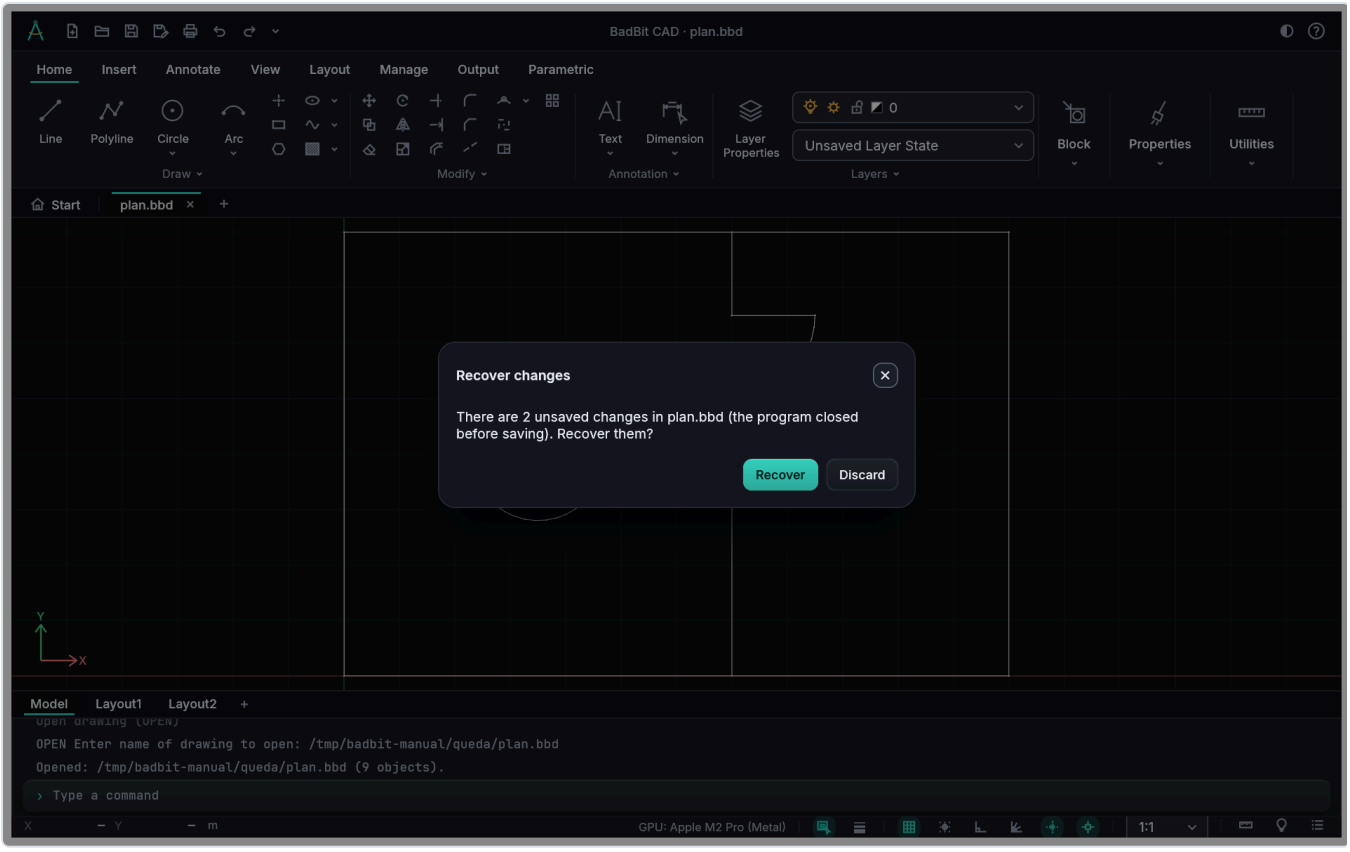
- **Yes** saves and continues. An untitled drawing asks for the name first.
- **No** discards the changes and continues.
- **Cancel** returns to the drawing, without doing anything.

CLOSEALL closes all tabs, as in AutoCAD: drawings without changes close directly; each modified drawing comes to the screen and asks, one at a time. **Yes** and **No** move on to the next one; **Cancel** stops everything (the drawings that have not closed yet stay open). When all are closed, an empty drawing remains, as after the last CLOSE.

Saving is fast, even in large drawings

Saving to the same file writes only the objects that changed since the last save. In a drawing with 200,000 lines, saving one change takes a few milliseconds. Saving and opening run in the background: the window keeps responding, and the result appears on the command line ("Saved to ...", "Opened: ..."). Each save is a complete operation: either everything goes into the file, or nothing in it changes.

Recovery after a crash



While the drawing is open, each command is recorded in a **journal** in the recovery folder (see *Automatic save*, below). Only the `.bbd` file sits in your drawing's folder. Saving empties the journal, and closing without saving deletes it.

If the program or the computer crashes, the journal stays on disk. When you open the file again (with OPEN or from the *Drawing recovery* dialog box), BadBit CAD reports how many unsaved changes it found:

- **Recover** reapplies the changes on top of the file. The drawing is as it was before the crash, still unsaved, and **so is undo**: U goes back through the same history as before the crash (including what was in the long undo file and what came before the last save), and REDO redoes what had been undone. A journal from an earlier version of BadBit CAD recovers the changes, but without the history: each recovered change becomes an undo step.
- **Discard** deletes the journal (and the long undo file of the session that crashed) and opens the file as it was last saved.

If the crash came right after saving, with no changes to recover, there is no prompt: the file opens as it was saved and U goes back through the history from before the crash (the journal records which saved version of the file it belongs to). If the file was changed externally after the crash, that history is discarded.

A new, still untitled drawing also has a journal: it stays in the recovery folder (below) until the drawing is saved with a name.

Automatic save (SAVETIME)

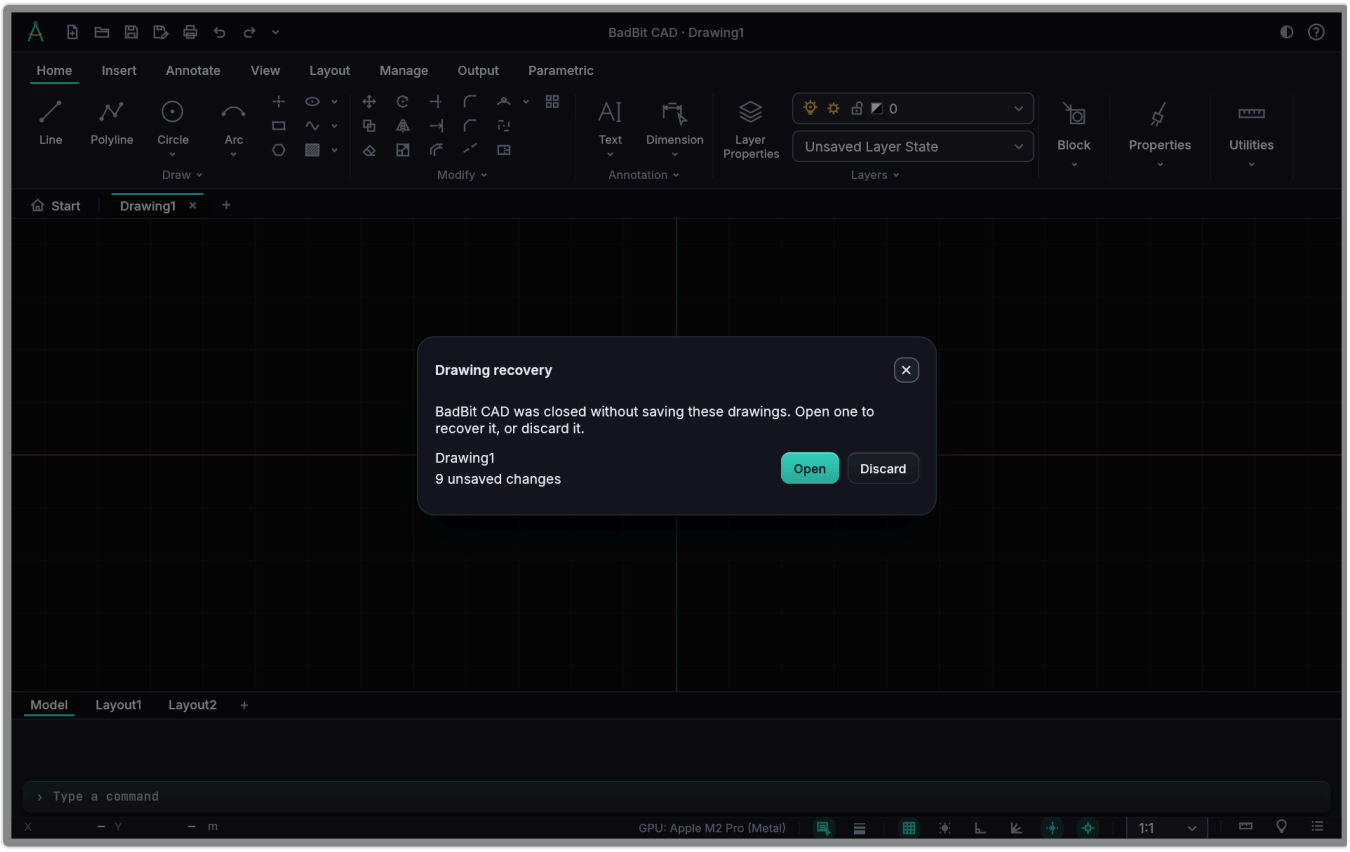
As in AutoCAD, BadBit CAD periodically writes an **automatic copy** of the open drawing. The interval is the **SAVETIME** variable, in minutes: the default is 10, and 0 turns it off. The copy goes to a separate recovery folder and **never touches your file**: it only changes when you save. Writing runs in the background, and the command line says "Automatic save to ...".

The recovery folder is in the user data folder:

System	Folder
macOS	`~/Library/Application Support/BadBit/recuperacao`
Windows	`%LOCALAPPDATA%\BadBit\recuperacao`
Linux	`~/.local/share/badbit/recuperacao`

Closing the drawing normally (with or without saving) deletes its copy and its journal. The journal applies from the first command, even in an untitled drawing: this way not even Drawing1 is lost in a crash. In a drawing from a newer version of BadBit CAD, the journal also keeps what the copies carried that this version does not understand, and recovery gives that back too.

Drawing recovery



If the program crashes, the copies stay in the folder. The next time it starts, the **Drawing recovery** dialog box lists each drawing with how many unsaved changes it has (a named drawing with no changes to recover does not appear: its undo history comes back when the file is opened):

- **Open**, for a named drawing, opens your file and offers the changes in the journal (see above). If the file has disappeared or does not open, the automatic copy is opened instead.
- **Open**, for an untitled drawing, brings the automatic copy plus the commands in the journal. The drawing stays untitled and unsaved: use **Save As**.
- **Discard** deletes the copy.
- **Close** keeps the list for the next start.

Several BadBit CAD windows open at the same time do not get mixed up: the copies of an open window are locked and do not appear as crash leftovers in the others.

Files from newer versions

A file saved by a newer version of BadBit CAD opens read-only, with a message. To save, use **Save As** with another name (the same name is refused, so as not to damage the original).

What this version does not understand is not lost. Objects of a new type (from another module or a newer version) are kept without being displayed, and the command line says how many there are ("3 objects from a newer version of BadBit were kept without being displayed."). New data on a known

object (a new property of a line, for example) is also kept, even if you edit the object. All of this goes back to the file unchanged when you save, including with **Save As**. An object with corrupted data is handled the same way, with its own message.

Copies carry this new data along: COPY, MIRROR, ROTATE and SCALE with copy, ARRAY (non-associative), COPYTOLAYER, grips with copy, COPYCLIP and PASTECLIP (also between tabs), WBLOCK and the ETRANSMIT package. When the copy goes to another drawing, the objects kept without being displayed that this data refers to go along, unchanged. What becomes part of a block (BLOCK, PASTEBLOCK, associative ARRAY, the Block Editor's BSAVE and REFEDIT) also carries this data, and EXPLODE gives it back to the objects.

Undoing (U) a block redefinition (BSAVE, a saved REFEDIT, BATTMAN, ATTSYNC, -BLOCK over an existing one) also restores this data to what it was before, and REDO redoes it. In REFEDIT, the blocks that contain the edited block follow the change, and an object that leaves the working set (REFSET Remove) goes to the drawing with the data it had.

Comparing drawings (DWGCOMPARE)

DWGCOMPARE compares the current drawing with another one (`.dwg`, `.dxf` or `.bbd`), like AutoCAD's. Reading and comparing run in the background (comparing tens of thousands of objects takes less than a second) and the result opens in a **new tab**, the **comparison**:

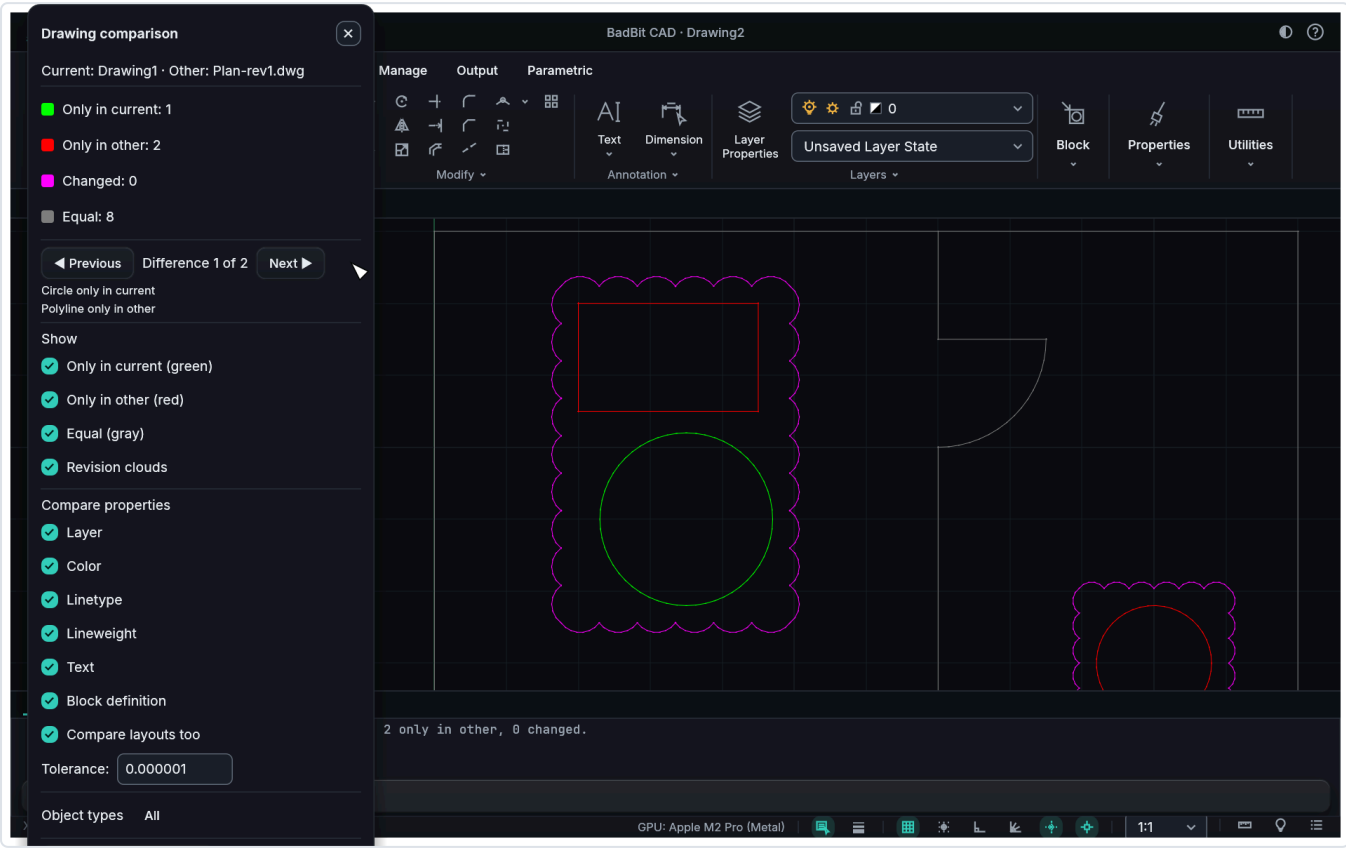
- **green**: only in the current drawing; **red**: only in the other; **gray**: the same in both;
- **changed**: the same geometry with a different layer, color, linetype, lineweight or **text**, or a reference to a **block with the same name and a different definition** in the two drawings; both versions appear (green and red) and the bar says what changed (**Block definition**, in the case of the block), with the text before and after. The bar also lists the blocks with the same name and a different definition;
- **revision clouds** (on their own layer) around each group of nearby differences.

The **navigation bar** (floating palette) shows the counts, ◀ **Previous** and **Next** ▶ (the zoom goes to the region and the list says what is in it), what to **show** (current, other, equal, clouds), the **compared properties** (unchecking color, for example, makes different colors count as equal; **Block definition** turns comparing definitions on and off), **Compare layouts too**, the **Tolerance** (in drawing units; the default is 0.000001, like AutoCAD's: two objects that differ by less than it count as equal) and the compared **object types**. Changing the filters, the tolerance or the layouts redoes the comparison in the same tab.

DWGCOMPAREEXPORT (or **Export the comparison...** on the bar) saves the comparison drawing as a file (Save As, in `.bbd`, DXF or DWG); **DWGCOMPARECLOSE** (or the bar's ×) closes the comparison tab.

Model space and the **layouts** (each layout's paper space against the layout with the same name in the other drawing) are compared. The clouds for a layout's differences stay in it, and **Previous**/**Next** switches to the tab of the region's layout (or of model), with the line `Layout <name>:` at the start of the list.

Layout **viewports** are included in the comparison, as in AutoCAD: a viewport with the same boundary and the same view is equal; with the same boundary and a different view (a different model center, a different scale, on or off) it is **changed**, and the bar says **Viewport view**; a viewport that exists in only one of the drawings is green or red. And what changed **in model** and shows through a viewport also gets a region (with its cloud) **in the layout**, around the place where the difference appears in the viewport (what is outside the view, or on a layer frozen in the viewport, does not). In the comparison drawing, the viewports show the model already with the comparison colors; with equal objects hidden, the viewport is still there (without the gray color).



DWGCOMPARE: the circle only in the current drawing (green), the new table and circle only in the revision (red), with the clouds and the navigation bar.

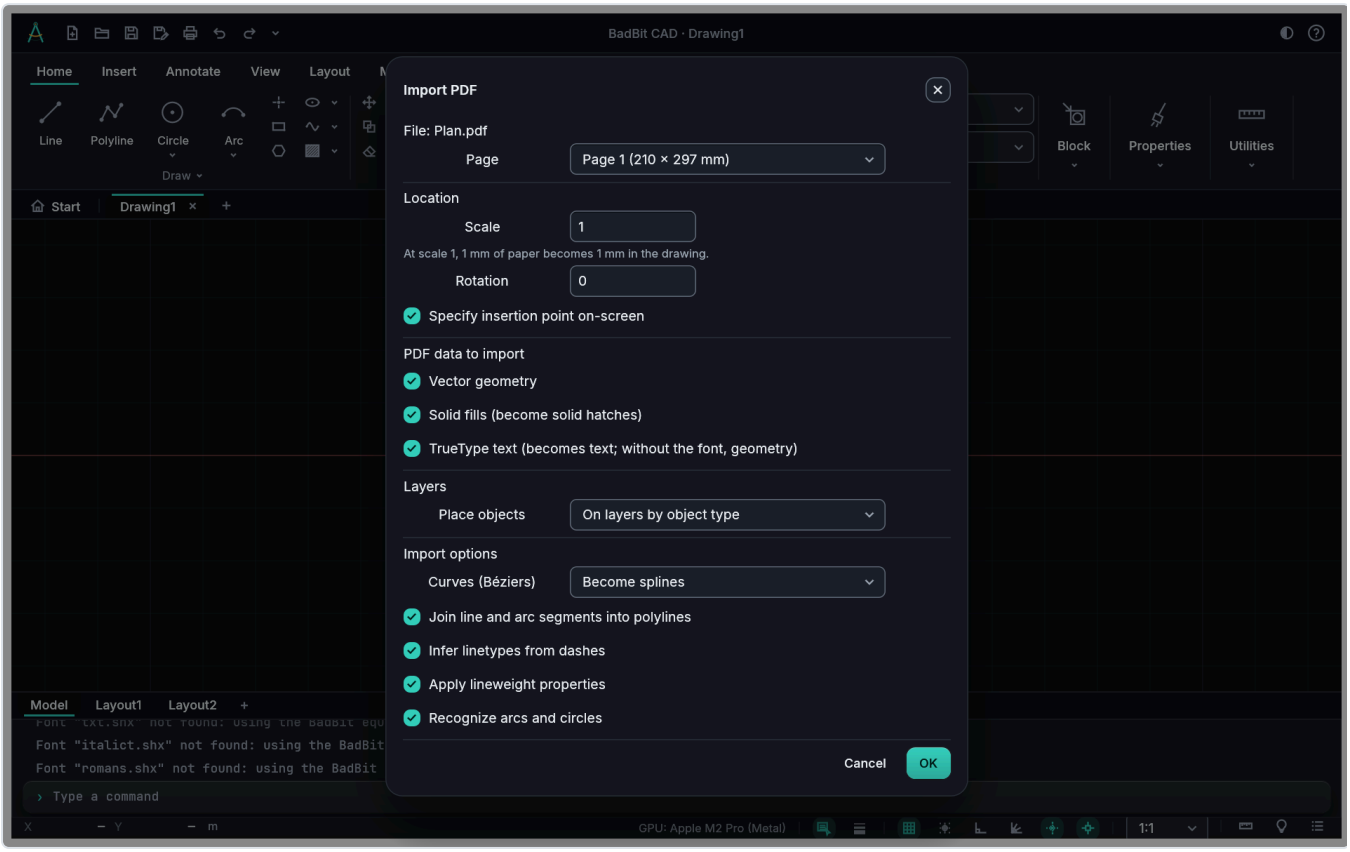
Importing PDF (PDFIMPORT)

PDFIMPORT brings a **vector PDF** in as drawing geometry, like AutoCAD's. Choose the file; the **Import PDF** dialog box shows:

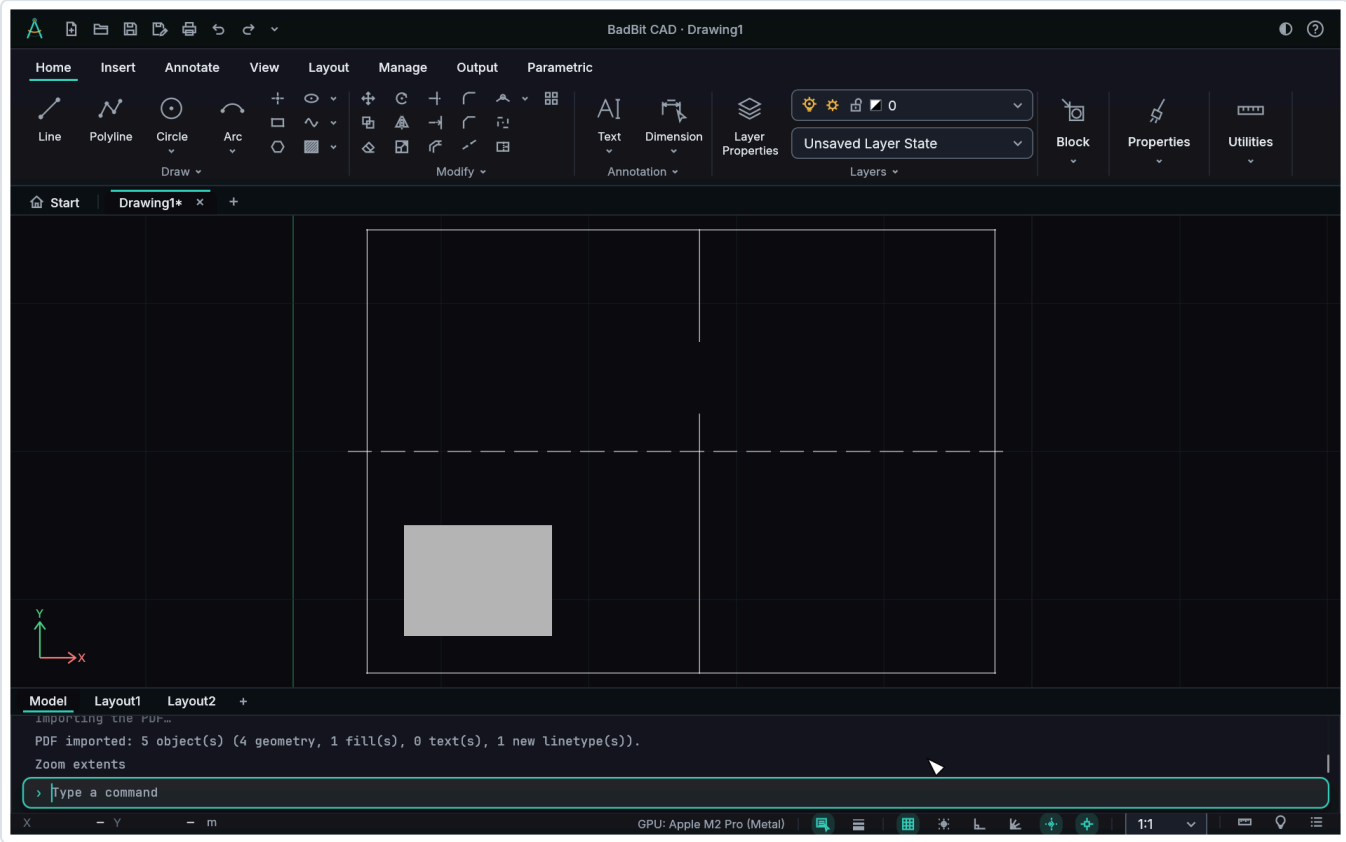
- **Page** (with its size in mm), **Scale** (at 1, 1 mm of paper becomes 1 mm in the drawing, in its unit), **Rotation** and the **insertion point** (the lower left corner of the page), typed or **specified on-screen** after OK;
- **PDF data to import**: **vector geometry** (recognized lines, polylines, arcs and circles; Bézier curves become a **spline** or a **polyline**), **solid fills** (they become **solid hatches**, with their holes) and **TrueType text** (it becomes text; when the PDF's font does not give the characters, it becomes **geometry** from the outlines of the embedded font);
- **Layers**: those of the **PDF** (the OCG layers, when the PDF has them), **by object type** (PDF_Geometry, PDF_Solid Fills, PDF_Text) or the **current layer**;

- **Import options**: **join** line and arc **segments** that touch into polylines, **infer linetypes from dashes** (the PDF's dash pattern and short collinear dashes, like those from plotting, become a single line with the linetype `PDF_DASH_n`), **apply lineweights** (the closest standard lineweight) and **recognize arcs and circles**.

The conversion runs in the background; everything goes in as a single undo step, and the command line summarizes what came in (and reports what was left out: images, patterns and gradients, protected PDF). The PDF reader is BadBit CAD's own (ADR 0021).



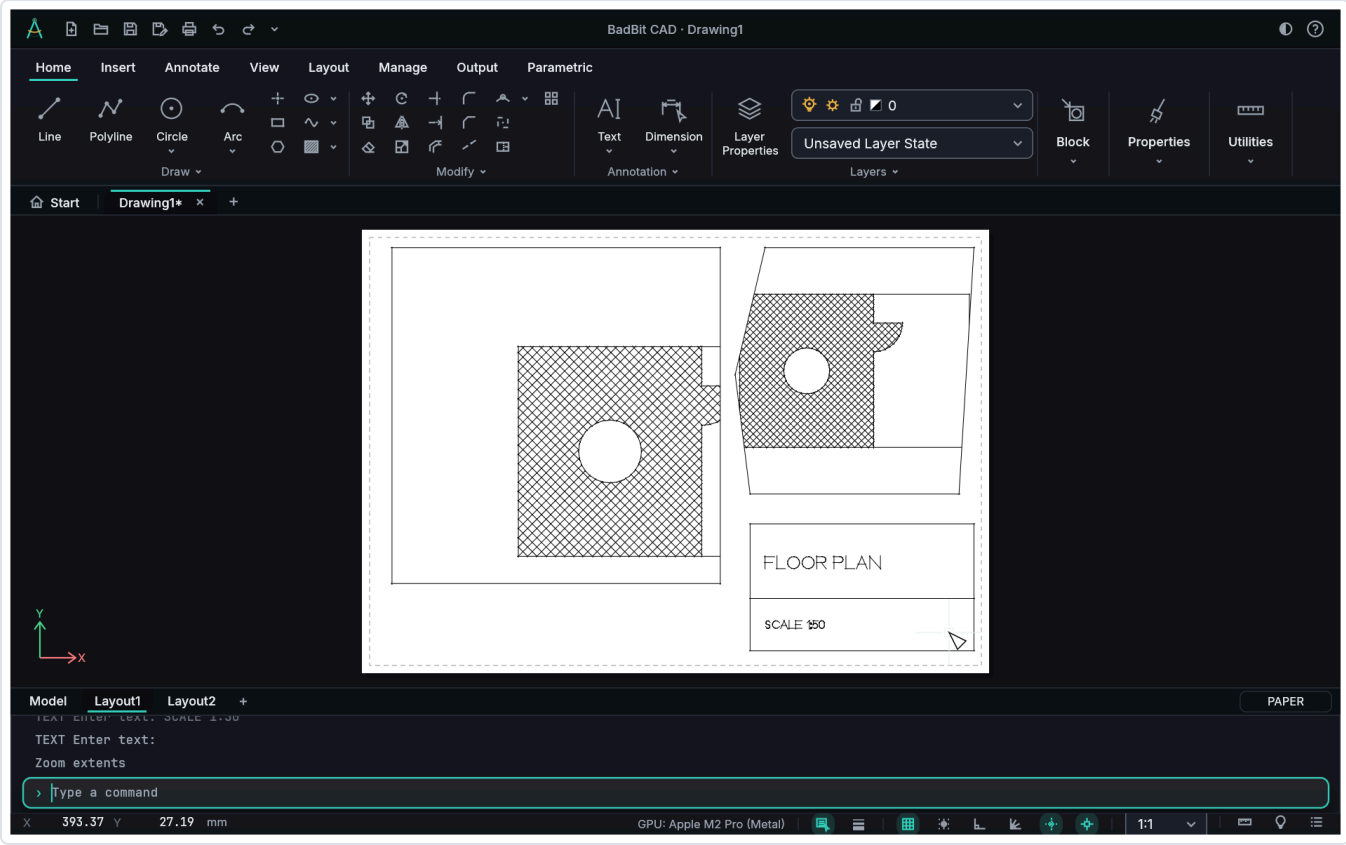
PDFIMPORT: the page, the scale, the rotation, the point on screen, the data, the layers and the options.



A floor plan plotted to PDF by BadBit CAD itself and imported back: walls, the slab as a solid hatch and the dashed axis with the inferred linetype.

Layouts and plotting

The drawing is made at full size in **model** space (in meters). To print, you set up a **sheet** in a **layout** (paper space): the paper size, the orientation, the margins, the title block and the **viewports**, which show the model inside the sheet at the chosen scale (1:50, 1:100...). As in AutoCAD, the **Model**, **Layout1**, **Layout2** and **+** tabs sit below the drawing area; every new drawing already has Layout1 and Layout2.



Layout1: the rectangular viewport with the plan, the polygonal one with a detail, and the title block drawn on the paper. The dashed line is the printable area of the paper.

The layout tabs

- **Click** a tab: goes to it (the Model tab returns to the view it had).
- **Double-click**: renames it (Enter confirms, Esc cancels).
- **Right-click**: New layout, Copy, Delete, Rename, Page Setup Manager and Plot.
- **Dragging** a tab over another changes the order.
- **+**: a new layout at the end, with the page setup of the current one.

The first time a layout is opened, it gets a viewport in the printable area showing the whole model (AutoCAD's LAYOUTCREATEVIEWPORT). On the right of the ribbon, the **PAPER**/**MODEL** button shows where you are drawing and switches from one to the other.

On paper, measurements are in **sheet millimeters** (an A3 landscape sheet goes from 0,0 to 420,297): what you type, the status bar coordinates and the inquiries (DIST, AREA, ID) are in mm. A dimension on paper measures in mm (a 100 mm line gets the dimension "100,00"). Styles (the default text, the

dimension arrows and text, the hatch pattern, the multiline width, the table and leader texts) come out on the sheet at the size they print: in a drawing in meters, like BadBit's, the styles are made for the model at 1:100, so the default 0.25 m text is 2.5 mm on the sheet; in a drawing in mm from AutoCAD, the styles are already the sheet's (2.5 is 2.5 mm). A height typed on paper is in mm. CHSPACE moves a dimension between model and paper without changing the measured value or its size on the sheet. Lines, text, dimensions, hatches and blocks drawn on paper stay only in that layout (the title block, the notes); the editing commands, selection, snaps and Select All look only at the space you are in. Tables, attributes and associative dimensions also work on paper (the dimension stays linked to the paper object it snapped to).

As in AutoCAD, the viewport does not go into a block: BLOCK and WBLOCK of objects leave the viewport in the layout (WBLOCK takes only the objects of the current space, into the model of the new drawing). INSERT of a file and an external reference (XATTACH) bring only the model of the other drawing, and the block editor (BEDIT) has no layouts.

LAYOUT (alias LO)

`Enter layout option [Copy/Delete/New/Template/Rename/SAveas/Set/?] <Set>:`

- **New** (`N`): the name (the default is the next free Layout*n*).
- **Copy** (`C`): the source layout (the current one with Enter) and the name of the copy ("Layout1 (2)" with Enter). The copy takes the viewports and everything on its paper.
- **Delete** (`D`): the layout and what is on its paper. The drawing always keeps at least one layout.
- **Rename** (`R`): the layout and the new name (unique names, without `< > / \ " : ; ? * | , =` `` and different from "Model").
- **Set** (`S`): the layout that is shown on screen (or `Model`).
- **?**: lists the layouts.
- **Template** (`T`) and **SAveas** (`SA`) are not available yet.

Each option is one undo step.

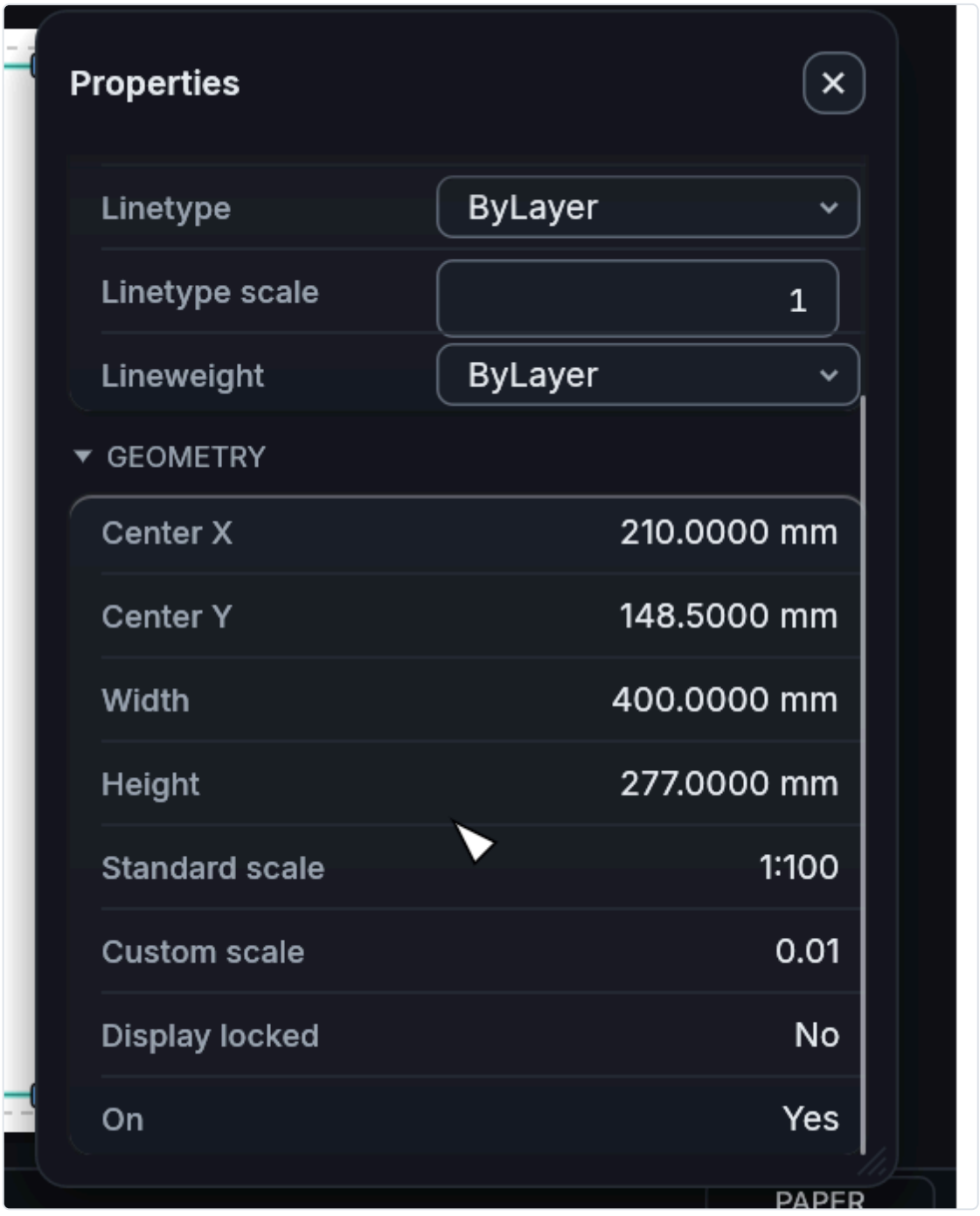
Viewports: MVIEW (alias MV)

On paper (not inside a viewport), MVIEW creates a viewport:

`Specify corner of viewport or [ON/OFF/Fit/Lock/Object/Polygonal] <Fit>:`

- **Two corners**: a rectangular viewport. It starts out showing the whole model.
- **Fit** (`F` or Enter): a viewport the size of the printable area.
- **Polygonal** (`P`): the vertices of the outline, one by one; `C` (or Enter) closes it, `U` removes the last one. The **Polygonal** button on the Layout viewports panel starts directly here.
- **Object** (`O`): click a closed polyline, a circle or an ellipse on paper; its outline becomes the viewport's.
- **ON** and **OFF** (`ON`, `OFF`): click the viewport; when off, it does not show the model (not even in the plot).
- **Lock** (`L`): `ON` or `OFF` and the click on the viewport. With the scale locked, zooming inside it does not change the scale.

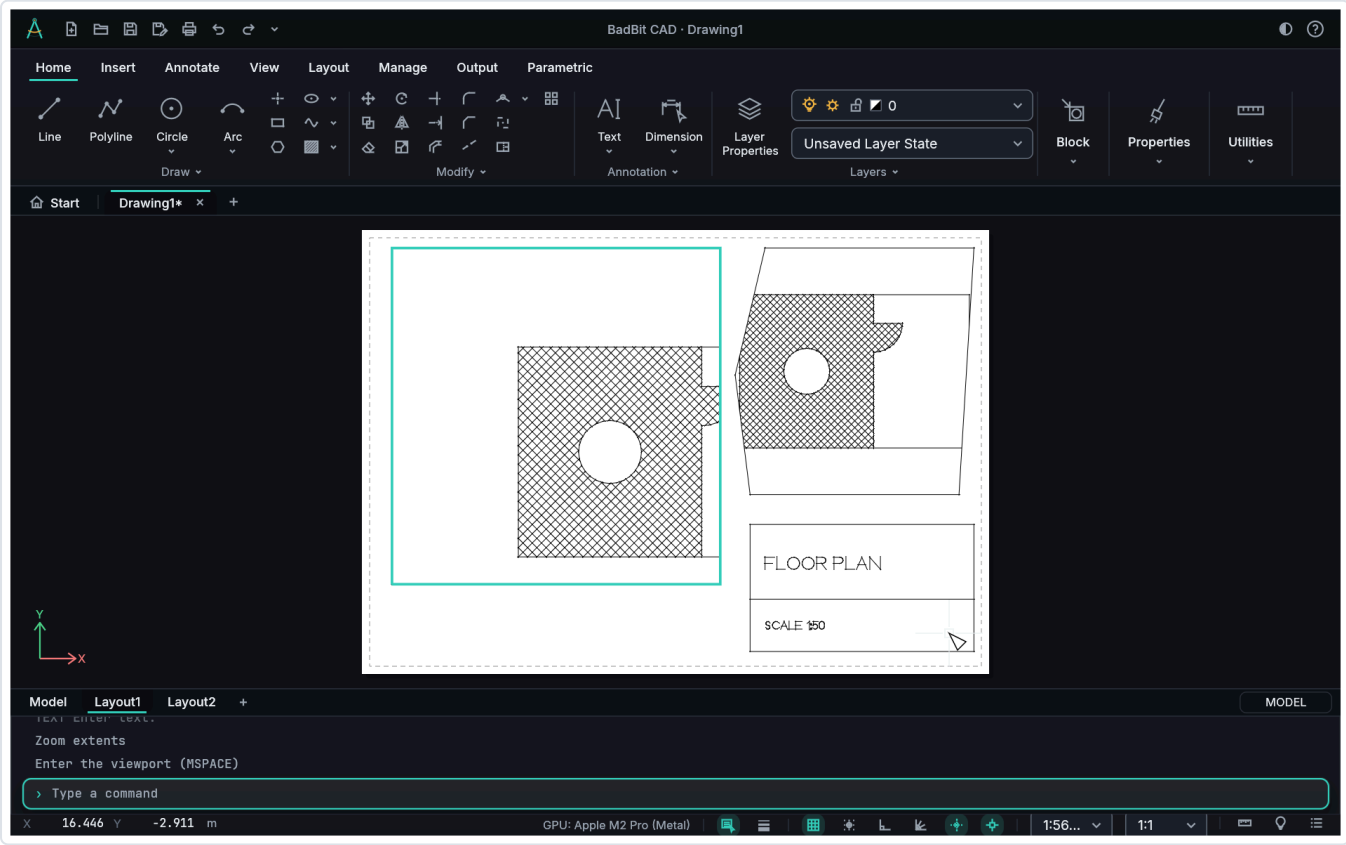
The viewport is a paper object: it moves, copies, erases, rotates (the outline only) and scales (the model inside grows with it) like the others, has grips at the corners (on a rectangular one, dragging a corner keeps the rectangle) and appears in the properties palette with the scale, the lock and whether it is on. The scale is shown as in AutoCAD, in any drawing unit: the **Standard scale** is the name in the drawing's scale list (`1:100`) or **Custom**, and the **Custom scale** is the paper-to-drawing ratio (`0.01` at 1:100). In a drawing in mm, a viewport at 1:100 shows `1:100` and `0.01`, not the internal factor (100,000 mm of model per meter of paper). The center, width and height of the viewport are also shown in the drawing unit (on an A3 sheet in a drawing in mm, the viewport that fills the sheet is 400 mm wide, not 0.4). It goes on the current layer; put viewports on a non-plotting layer so the border does not come out on paper.



In a drawing in mm, the viewport at 1:100: Standard scale 1:100 and Custom scale 0.01.

Inside and outside the viewport: MSPACE and PSPACE (MS, PS)

- **MSPACE** (`MS`), a **double-click inside** a viewport or the PAPER button on the ribbon: enters the viewport. Its border becomes thick, in the highlight color, and drawing now happens in **model** space, through the viewport: everything you draw goes to the model and also appears in the other viewports. A click in another viewport makes it the active one.
- Inside the viewport, the **mouse wheel** and the **middle button** change the view **of the viewport** (the paper stays still), unless it is locked; zoom extents (middle double-click) fits the whole model in the viewport. The change goes into the drawing when you leave the viewport, as a single undo step ("ZOOM").
- **ZOOM** and **PAN** typed inside the viewport also change its view: ZOOM Window fits the two corners by the shape **of the layout viewport** (not the screen), Center puts the point at its middle with the given height, `2X` zooms around its center and ZOOM Previous goes back to **its** previous view (each layout viewport, the paper of each layout and the model have their own list). They also work right after MSPACE, at the same instant (a script, automation), and never change the viewports of another layout, nor those of a copied layout.
- On the status bar, next to the annotation scale, the **viewport scale** appears: choosing 1:50 in the list sets the viewport to 1:50 (1 m of model becomes 20 mm on paper). With the viewport locked, the list only shows it.
- **PSPACE** (`PS`), a **double-click outside** the viewport or the MODEL button on the ribbon: goes back to paper.



MSPACE: the active viewport with the thick border, MODEL on the ribbon and the viewport scale on the status bar.

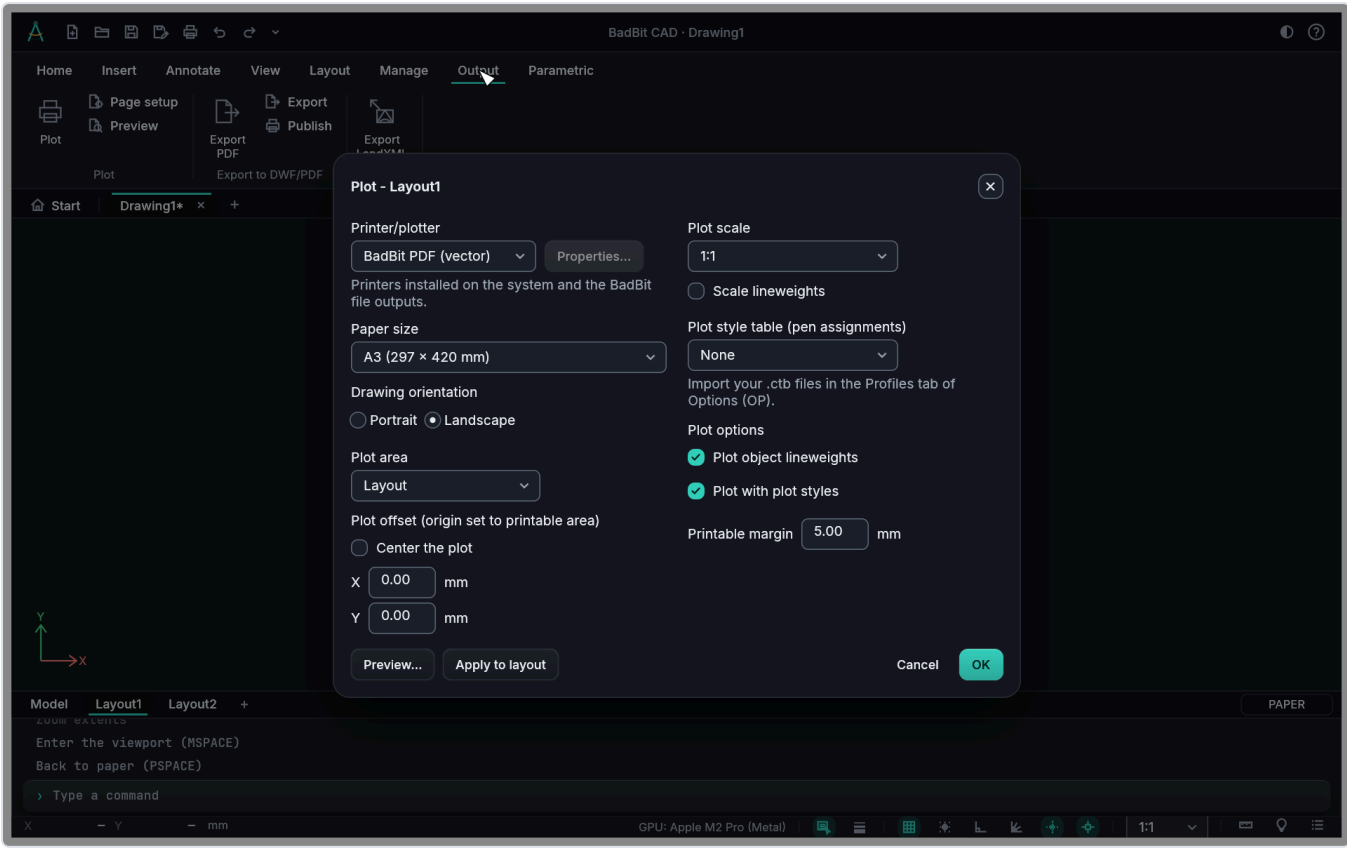
Layers per viewport: VPLAYER

`Enter an option [Freeze/Thaw]:`, then the layer names (separated by commas; `*` matches anything, as in `PAR*`), and `Enter an option [All/Current] <Current>:` — only in the active viewport (MSPACE) or in all those of the layout. A layer frozen in a viewport disappears only there (on screen and in the plot); in the others and in the model it remains.

Plot: PLOT (Ctrl+P)

The **Plot** dialog has the same organization as AutoCAD's:

- **Printer/plotter**: the **printers installed on the system** (the list is looked up when the dialog opens; meanwhile, "Looking for the system printers..." appears) and the BadBit CAD file outputs: **BadBit PDF** (vector), **PNG image** and **BadBit SVG** (vector). A printer that is not ready appears with its status: "queue stopped", "offline", "not accepting jobs". See Printing to system printers.
- **Paper size**: with a system printer, **the papers it reports** (with their dimensions); with the file outputs, A4 to A0 and the US sizes (Letter, Legal, Tabloid, ANSI C to E).
- **Orientation**: Portrait or Landscape.
- **Plot area**: **Layout** (the whole sheet, at 1:1), **Extents** (everything drawn), **Display** (what is on screen) and **Window** (the **Window<** button asks for the two corners in the drawing). You can also plot from the Model tab (Extents, Display or Window).
- **Offset**: **Center the plot** or the X and Y from the corner of the printable area.
- **Scale**: **Fit to paper** or one of the scales in the drawing's list (SCALELISTEDIT, see the *Grid and units* chapter); **Scale lineweights**. The scale is the true ratio in the drawing unit: in a drawing in mm, 1:100 takes a 10,000 × 5,000 mm rectangle to 100 × 50 mm on paper, just like 10 × 5 m in a drawing in meters. In the Model tab, below the list, the same scale appears the way AutoCAD writes it, **1 mm = N units** (1:100 is `1 mm = 100 mm` in a drawing in mm and `1 mm = 0.1 m` in one in meters; in a unitless drawing, as in AutoCAD, `1 mm = 100 units`).
- **Plot style table**: the `.ctb` pen tables imported into the profile (Options, Profiles tab). Each color of the drawing comes out with the color, lineweight and screening (fading) that the table specifies: for example, color 1 at 0.50 mm in black. True colors (RGB) do not go through the color-dependent table.
- **Options**: plot object lineweights and plot with plot styles.
- **Printable margin**: the part of the sheet the output does not print. With a system printer, these are **the margins it reports** for the paper (they appear below, for reference only); for the file outputs (and for a printer that does not report them), the margin you type (5 mm by default).
- **Preview...**: the preview. **Apply to layout**: saves the setup in the layout (one undo step). **OK**: chooses the file and plots (or, with a system printer, sends it to the printer).



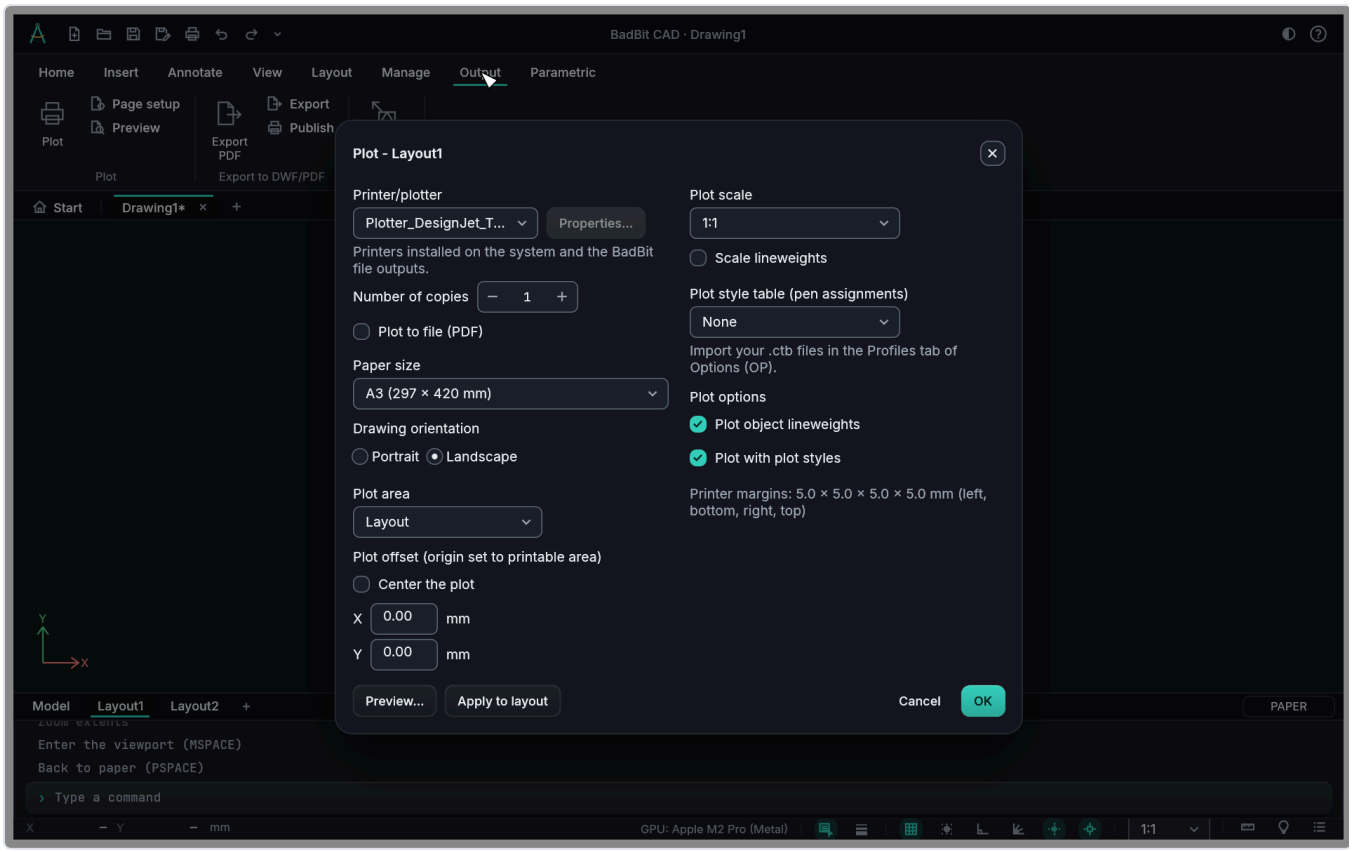
The Plot dialog for Layout1.

Plotting runs **in the background**: a card in the corner shows the progress and a **Cancel** button, and you can keep drawing. At the end, the command line says where the file was saved.

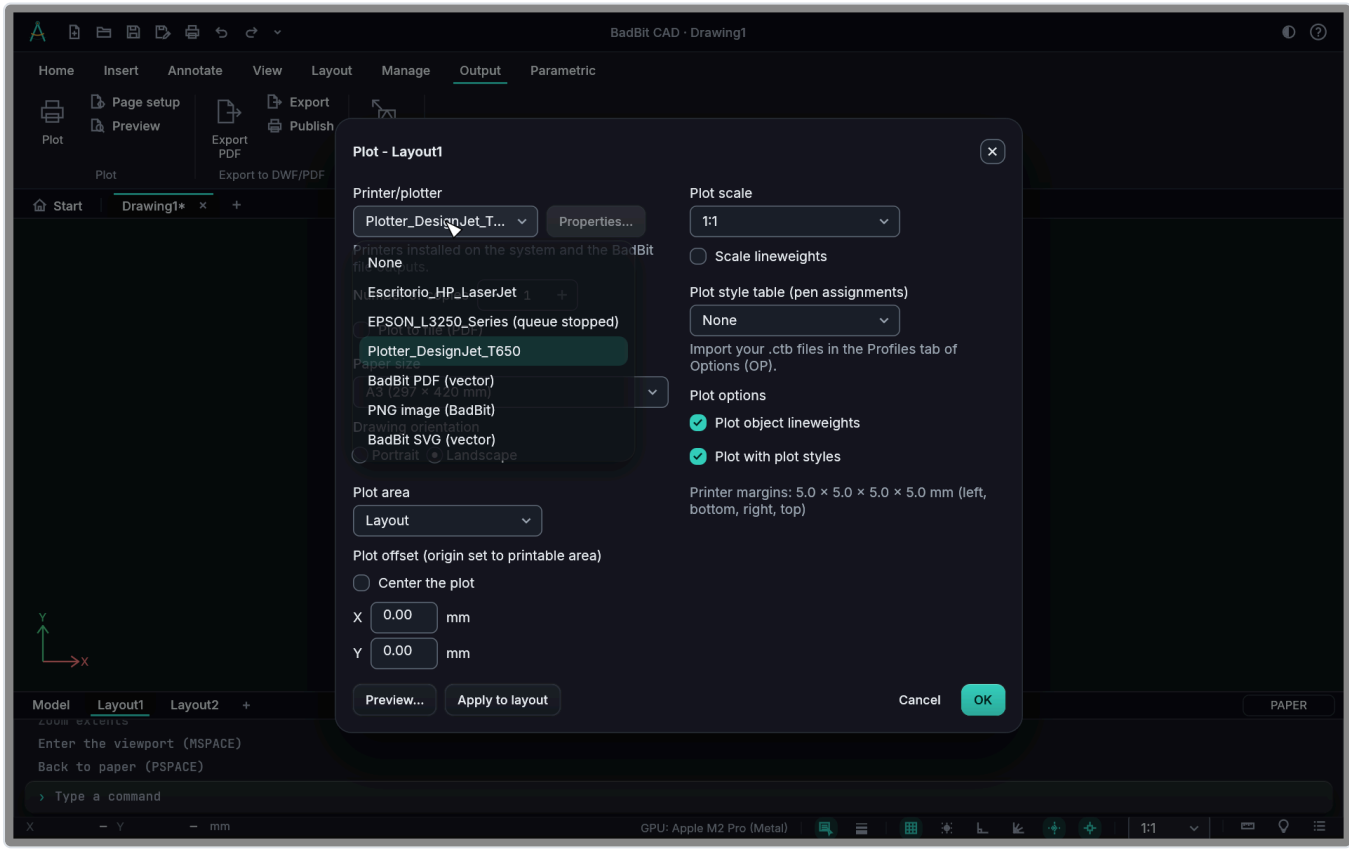
Printing to system printers

As in AutoCAD, the **Printer/plotter** list of the Plot dialog shows the printers installed on the computer before the BadBit CAD file outputs. When you choose one:

- **Paper size** becomes the list of papers **of the printer** (a plotter shows the large sheets; an office printer, A4, Letter, Legal...), and the **margins** become the ones it reports for the paper, already in the chosen orientation.
- **Number of copies**: from 1 to 999.
- **Plot to file (PDF)**: instead of sending it to the printer, saves the PDF of the sheet (the same one that would go to it).
- **Properties...**: the printer options beyond paper and orientation. On macOS and Linux, a dialog with the options the printer accepts (quality, paper type, color or grayscale, duplex, tray), each with the printer's default on the first line; on Windows, the **printer driver's own dialog**. What you choose applies to the next print jobs on that printer while BadBit CAD is open.
- **Images** (attached images and the images inside attached PDFs; the PDF's lines go as lines): they go to the printer **with the original pixels**, with no resolution reduction, on every system; the printer itself fits the image to its print resolution. The same applies to the PDF and SVG saved to a file.



Layout1 on the plotter: its papers, the margins it reports, the copies, Plot to file and Properties. (In the manual's screenshots the printers are examples.)



The Plot dialog list: None, the system printers and, at the end, the BadBit CAD file outputs.

OK sends the sheet to the printer **in the background**, with the same progress card and **Cancel** button as plotting to a file; you can keep drawing. At the end, the command line says which printer it went to and the job number in the queue. If the printer queue is **stopped** or the printer is **offline**, the job waits in the queue and the command line warns you; if the system **refuses** it the warning appears there, without freezing anything: "printer X is not accepting jobs", in the program's language, when the printer is refusing jobs; the system's own message for any other reason (for example, the print service does not respond).

The sheet comes out **to scale**: BadBit CAD sends the sheet at the paper size and asks the system not to fit it (1:50 stays 1:50 on paper).

The printer is stored in the page setup. Chosen in **Page Setup** (PAGESETUP, Modify...), it goes into the layout, in the `.bbd` file, by name. When the drawing is opened on another computer that does not have that printer, the Plot dialog warns you ("The printer ... does not exist on this computer: using the default, ...") and uses the **system default printer**, with the paper of the same name or the same dimensions (otherwise, its default paper), as AutoCAD does. With no printer installed at all, the BadBit CAD PDF is used.

PUBLISH also prints: in **Publish to**, choose **The printer of each page setup**. Each checked layout goes to the printer of its page setup (those set to a file output go to the system default printer); consecutive layouts on the same printer, paper and orientation go as a single job.

On each system:

- **macOS and Linux**: the **CUPS** printers (the same ones as in System Settings on the Mac and in the printer manager on Linux). BadBit CAD sends the PDF it generates itself. Without CUPS installed, or with the service stopped, the list shows only the file outputs and the command line gives the reason. On Linux, if the printer description file (PPD) cannot be read, the papers come by their standard name (A4, Letter...) and the margin is the typed one.
- **Windows**: the printers installed in Windows, with the driver's papers and margins; the sheet is drawn directly to the driver. Transparencies come out blended with the white of the paper.
- **Browser**: there are no system printers; the list shows only the file outputs.

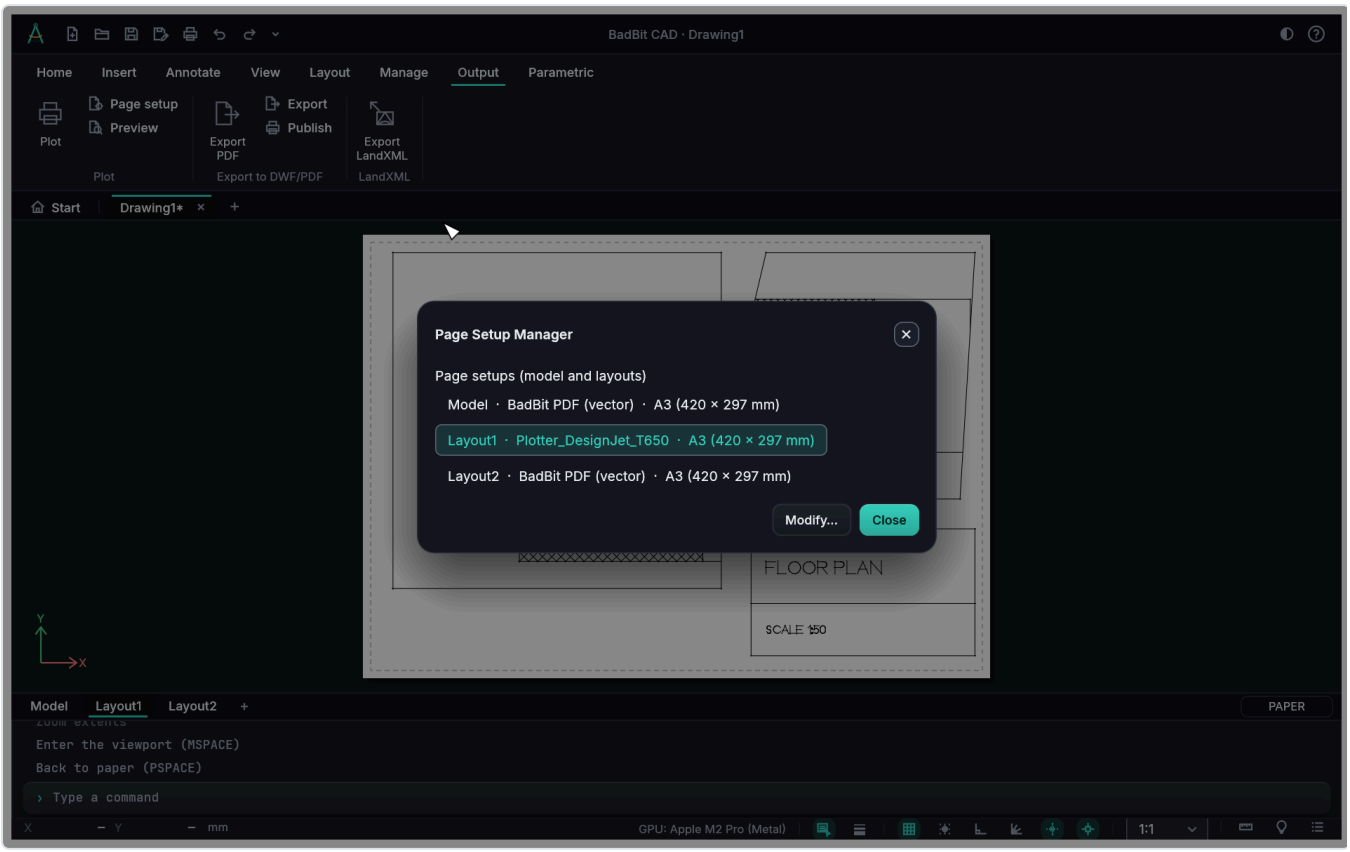
What comes out on the sheet:

- **Vector lines** in PDF and SVG, with round ends and corners; **linetypes** (dashed) as on screen.
- **Text as geometry** (the strokes of the BadBit CAD font; with TrueType fonts, solid letters, with the inside of the "O" hollow): the PDF opens the same on any computer, without fonts. This also applies to the text of dimensions, leaders, tolerances, attributes and tables.
- **Hatches** (the pattern lines and the background), **solids** and **filled arrowheads** as filled areas, with **transparency**; **gradients** in small bands of color.
- **Tables** with the background, borders and text of each cell in its own color and lineweight.
- **Blocks and external references** with everything inside them (hatches, tables, attributes, other blocks), by the same rule as on screen (ByBlock takes the color, lineweight and linetype of the insertion; layer "0" takes the insertion's layer), also in nested blocks.
- **Viewports** clipped by their outline (rectangular or polygonal), with the layers frozen in them left out and **annotative objects at the viewport scale** (those that support its scale).
- Only the layers that are on, not frozen and set to plot.

- **PNG** comes from the same rasterizer as the viewport without a GPU, at the chosen **DPI** (A3 at 150 DPI: 2480 × 1754 pixels).

Page setup: PAGESETUP

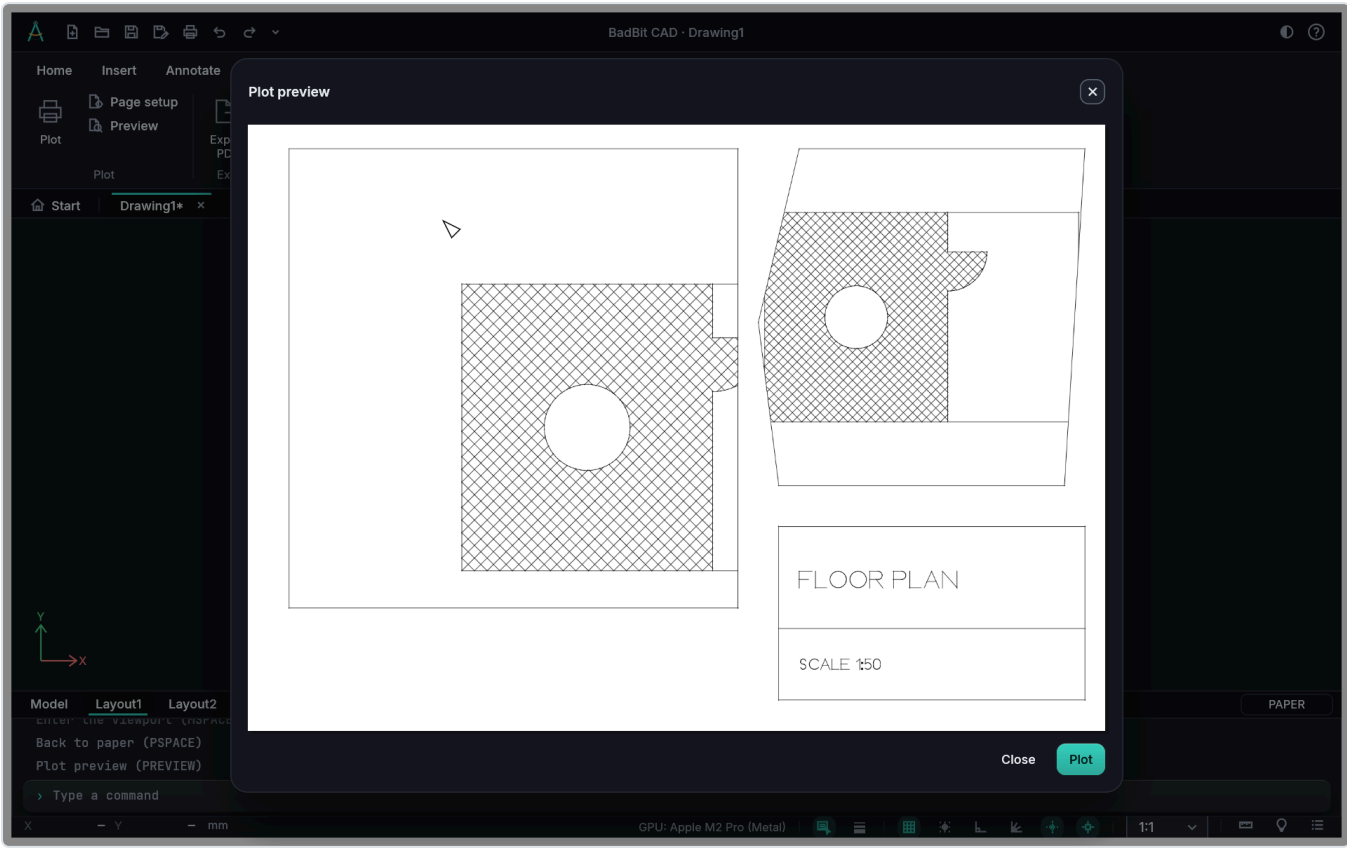
The **Page Setup Manager** lists the model and the layouts with the output and paper of each one; **Modify...** opens the same dialog as plotting, which here saves the setup in the layout (OK) without plotting. Each layout's setup goes into the `.bbd` file; the model's lasts while the drawing is open.



The model and the two layouts, with output and paper.

Preview: PREVIEW (PRE)

The sheet as it will come out, with pens and lineweights, in a window over the drawing; **Plot** from there goes straight to the file.

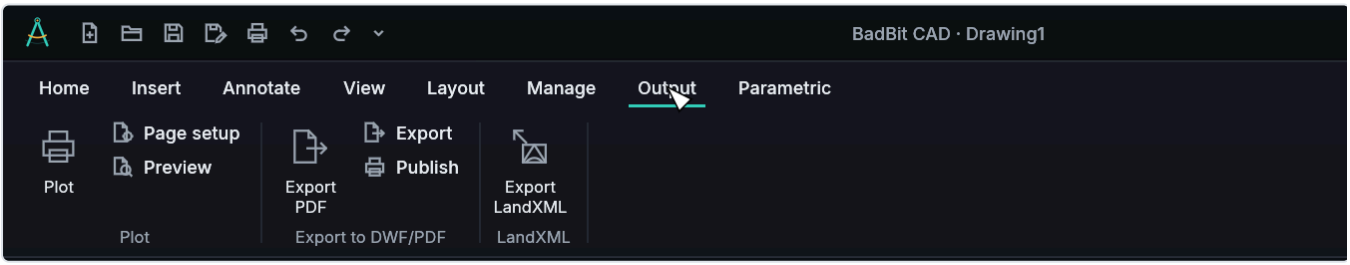


PREVIEW of Layout1: the two viewports and the title block.

Export: EXPORT, EXPORTPDF, PUBLISH

- **EXPORTPDF**: the sheet on screen (the layout or the model) to a PDF, with its page setup.
- **EXPORT** (`EXP`): PDF, SVG or PNG, by the extension chosen in the file dialog. **EXPORTSVG** and **EXPORTPNG** go directly.
- **PUBLISH**: check the layouts; they go, each with its own page setup, into one **multi-page PDF**.

The **Output** tab of the ribbon has the **Plot** panel (Plot, Page setup, Preview) and the **Export to DWF/PDF** panel (Export PDF, Export, Publish); the **Layout** tab has the **Layout** panel (New layout, Page setup, Layout options) and the **Layout viewports** panel (Rectangular, Polygonal, Enter viewport, Back to paper and, under the small arrow, VPLAYER).



The Output tab of the ribbon.

Still missing

- The choices in the printer's Properties... only last while BadBit CAD is open (AutoCAD saves them in a `.pc3` file); printing from the browser.

- On screen, annotative objects appear in all viewports at the current annotation scale (CANNOSCALE); in the plot, each viewport uses its own scale.
- Layout from a template (`.dwt`), LAYOUT's SAveas, named page setups and the "VP Freeze" column in the layer manager (for now, VPLAYER).
- On the CPU path, panning and zooming on the sheet redraw the whole sheet (on the GPU path, the cached frame works as in the model).

Changing space: CHSPACE

Moves objects between model and paper through the viewport, keeping their appearance: from inside a viewport (MSPACE), the chosen model objects go to paper, at the place and size they appear in it (divided by the viewport scale); from paper (PSPACE), they go to the model through the layout's only viewport or, with several, through the one you activate (double-click inside it) before `Enter`. The command line says how many were moved and the scale factor. In the Model tab, CHSPACE does not apply (as in AutoCAD). It is in the expanded Modify panel (Home tab).

DXF and DWG files

BadBit CAD opens and saves both AutoCAD formats:

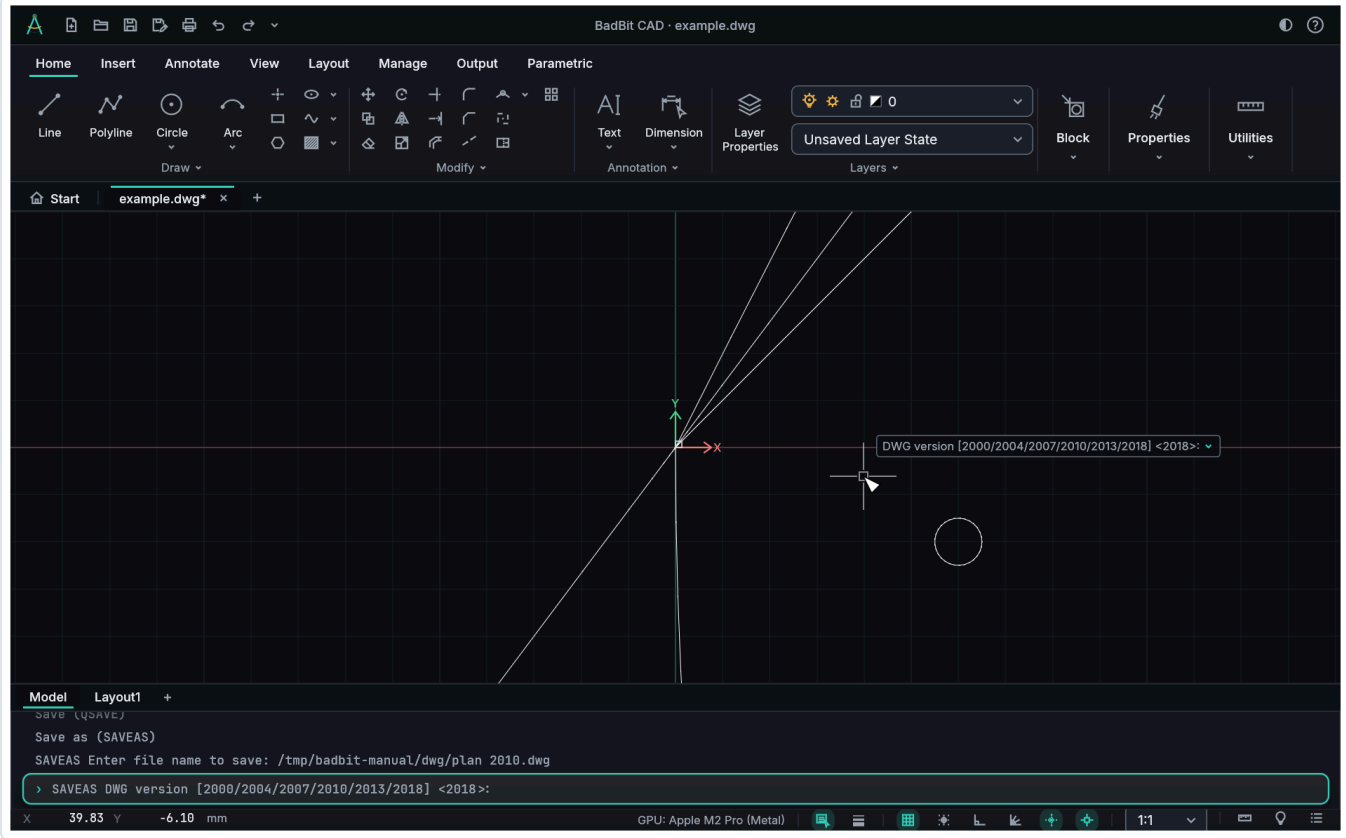
- **DWG**, AutoCAD's native format, from **AutoCAD 2000** to **AutoCAD 2018** (which is also the format of AutoCAD 2019 and later): 2000, 2004, 2007, 2010, 2013 and 2018;
- **DXF**, the exchange format, from **R12** to **2018**, in **text** and in **binary**.

The goal is complete compatibility: the drawing opens with everything BadBit CAD knows how to do, and what it does not understand yet **is not discarded** — it is kept and goes back unchanged when the drawing is saved again in the same format and version. DWG and DXF go through the same path internally: a DWG and the DXF of the same version open in BadBit CAD exactly the same.

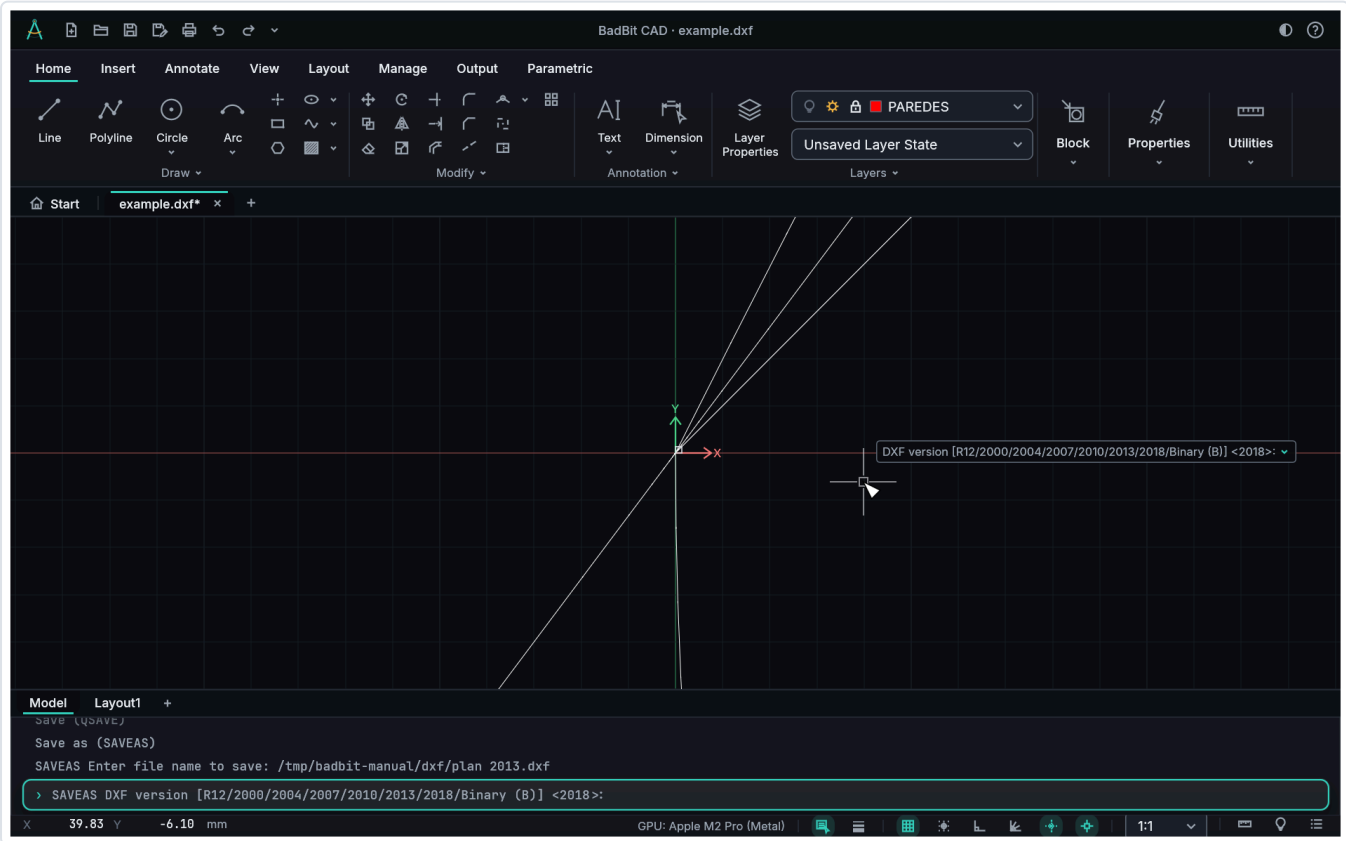
Commands

Command	What it does
OPEN	Opens a `.bbd`, a `.dwg` or a `.dxf` (the file dialog shows all three).
DXFIN	The same as OPEN, with the DXF filter first.
QSAVE	In a drawing opened from a DWG or a DXF, saves back to the same format, in its version (the first time it asks, see below).
SAVEAS	With the `.dwg` extension, asks for the DWG **version**; with `.dxf`, the **version** and whether it is **binary**.
DXFOUT	The same as SAVEAS, with the DXF filter first and the name ending in `.dxf`.
RECOVER	The same as OPEN, reporting the result of the check even when the file has no defects (reading DWG and DXF always repairs what it can, like AutoCAD's RECOVER).
IMPORTREPORT	Opens the **Import report** of the drawing opened from a DWG or DXF again (the command belongs to BadBit CAD: the name is the same in all three languages).
AUDIT	Checks the drawing as it would go into the AutoCAD file (handles, owners, references, blocks of inserts and dimensions, layers, linetypes, styles, extended data applications, layouts) and says what it would find.

With **FILEDIA** at 0, the version is asked on the command line: type `2000`, `2004`, `2007`, `2010`, `2013` or `2018` (for DXF also `R12`; `Enter` keeps the version of the open file, or 2018); for DXF, `B` toggles binary on and off.



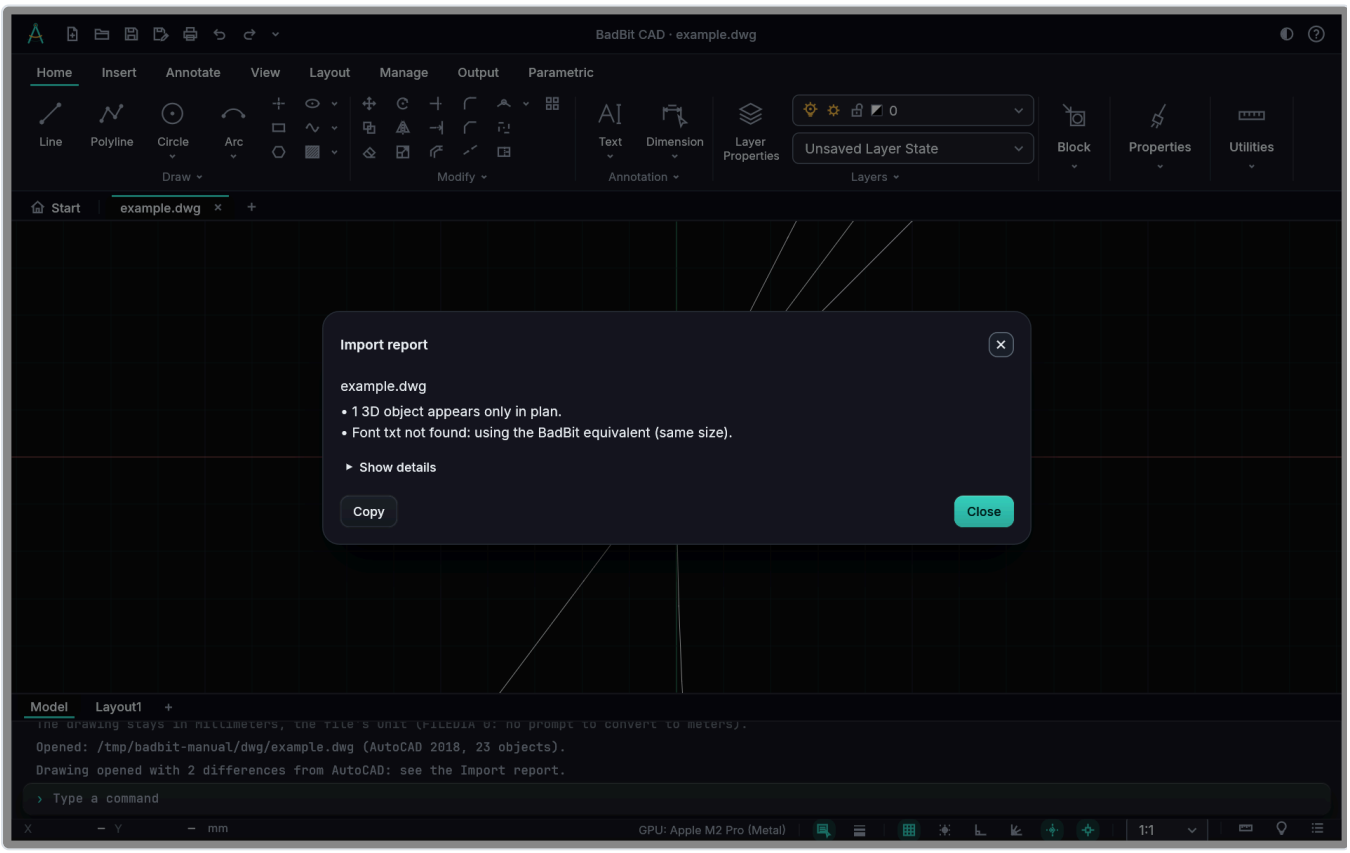
With FILEDIA 0, after the `.dwg` file name comes the DWG version.



With FILEDIA 0, after the file name comes the DXF version.

Opening a DWG

A DWG opens like a DXF (see the next section): in the background, with progress, in a new tab, with the same units, the same objects and the same **Import report**.



The DWG opened: the sentence at the top and what looks different from AutoCAD.

- **Versions.** 2000, 2004, 2007, 2010, 2013 and 2018 (and the formats of the AutoCAD releases that save in them: 2000i and 2002; 2005 and 2006; 2008 and 2009; 2011 and 2012; 2014 to 2017; 2019 and later).
- **Application objects** (Civil 3D, AutoCAD Map, Architecture: styles, settings, points, surfaces, alignments) are kept **byte for byte** and go back unchanged when saving as DWG **in the same version**. The entities of these applications appear through the proxy graphics the file carries.
- **Damaged file.** A wrong CRC, a missing section, a page that does not decompress and an unreadable object never prevent the file from opening: whatever can be read is read (like AutoCAD's RECOVER) and the report says what was repaired. Password-protected drawings open without the encrypted data (with a warning).
- **Performance.** A DWG with one million objects opens in about 3 to 4 seconds (measured on an Apple M2 Pro), without freezing the window.

Opening a DXF

Reading runs in the background, with progress in the corner of the screen: the window keeps responding even with a file of one million objects (a few seconds). The drawing opens in a new tab, with the file name.

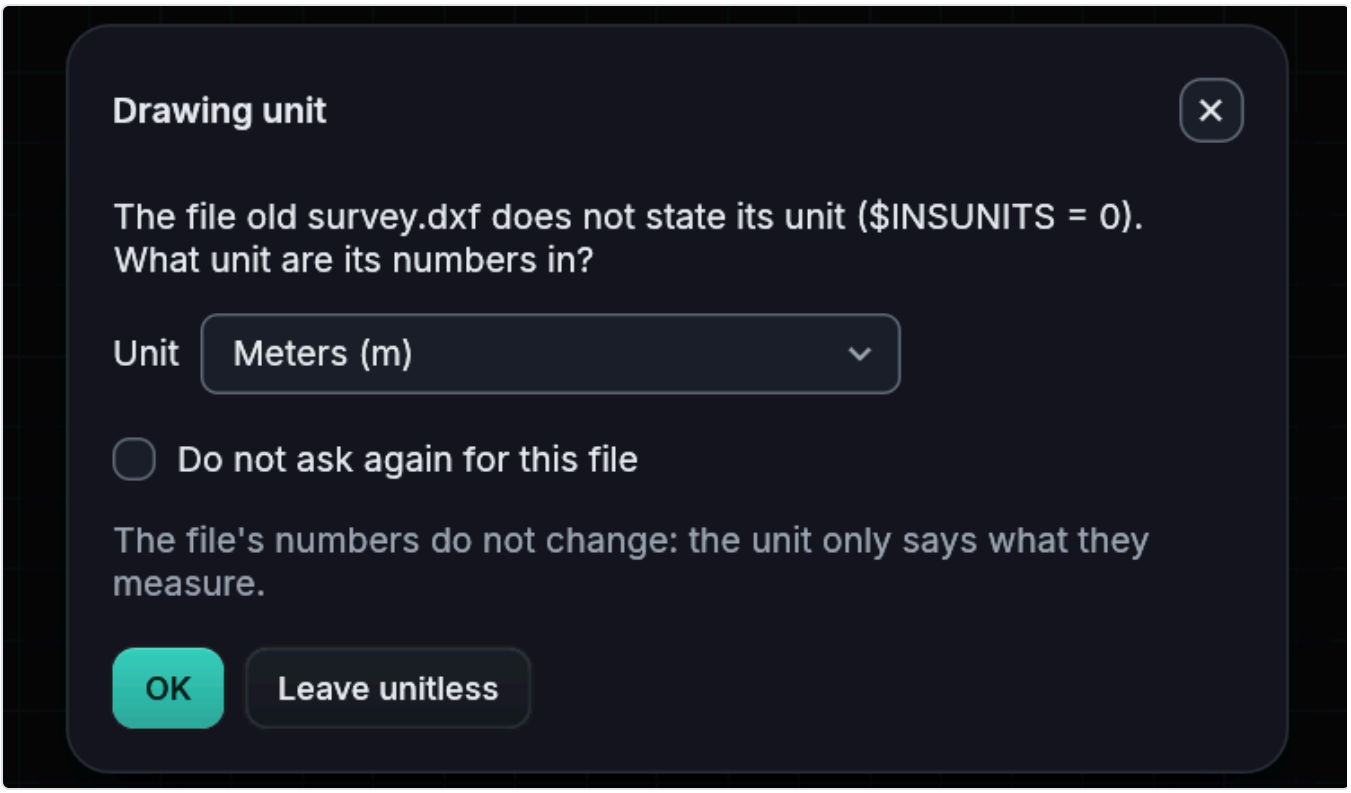
- **Units.** An AutoCAD drawing is in drawing units (`INSUNITS`). BadBit CAD keeps **exactly the numbers in the file** and uses the file's unit and formats in that drawing (see "Drawing unit and precision", below): a DXF in millimeters opens in millimeters, with the same numbers as AutoCAD on screen, and a UTM survey in meters opens with the coordinates intact down to the last digit.
- **The view.** As in AutoCAD, the drawing opens in the view in which it was saved (the active model view, the center and the height) and each layout, the first time it appears, in its paper view; with no saved view (or with a 3D view), it opens at zoom extents, which does not count frozen layers. When saving as DXF or DWG, the current model view goes into the file.
- **Text.** Accented characters and symbols come out right in any version: the code page of an older file (`ANSI_1252` and the others, including those for Japanese, Chinese and Korean), the UTF-8 of 2007 and later, and the `\U+` and `\M+` codes. A file that declares a code page but was written in UTF-8 by another program is read as UTF-8, with a warning.
- **What becomes a BadBit CAD object.** Lines, polylines (lightweight and old-style, with arcs and widths), circles, arcs, ellipses, splines, construction lines and rays, points, single-line and multiline text, blocks with attributes, inserts (MINSERT becomes an array, with the columns and rows along the insert's axes also when the normal is not the top one; a tilted MINSERT becomes a one-cell array with the copies, each at its own place in space; one that was not changed goes back as a MINSERT in every version, as AutoCAD saves it), dimensions of all types (with the style and overrides), leaders, tolerances, hatches (pattern, solid and gradient, with islands and associativity), four-sided solids, tables, layouts with their page setup and viewports, layers (ACI and true color, linetype, lineweight, on, frozen, locked, plottable), linetypes, text styles (TrueType and SHX; SHX fonts are read from the AutoCAD font folders on the computer, see *AutoCAD SHX fonts* in the text chapter), dimension styles, multileader styles and table styles.
- **AutoCAD's mirroring.** Objects with an inverted extrusion (the ones mirrored in AutoCAD) appear in the right place.
- **What BadBit CAD does not understand** (Civil 3D and other application objects, proxy objects, 3D solids, images, extended data) is kept **intact**. On screen you see the drawing the file carries for those who do not have the application (the proxy graphics), the plan projection of 3D objects and the outline of images. In Properties, the type is "Object from another program", with the original type. These objects can be moved, copied and erased; changing one of them makes BadBit CAD save its drawing in place of the original (with a warning).
- **Damaged file.** A truncated file, wrong group codes, duplicate handles, broken references and misplaced values never prevent the file from opening: whatever can be read is read, and the rest is counted in the report, as AutoCAD's AUDIT and RECOVER do.

Drawing unit and precision

Converting coordinates to another unit and back does not always return the same number (in millimeters, 2 out of every 100 coordinates would change in the last digit; in centimeters and in feet, more than 10). In surveying this is not acceptable. Therefore, when opening a DXF or DWG:

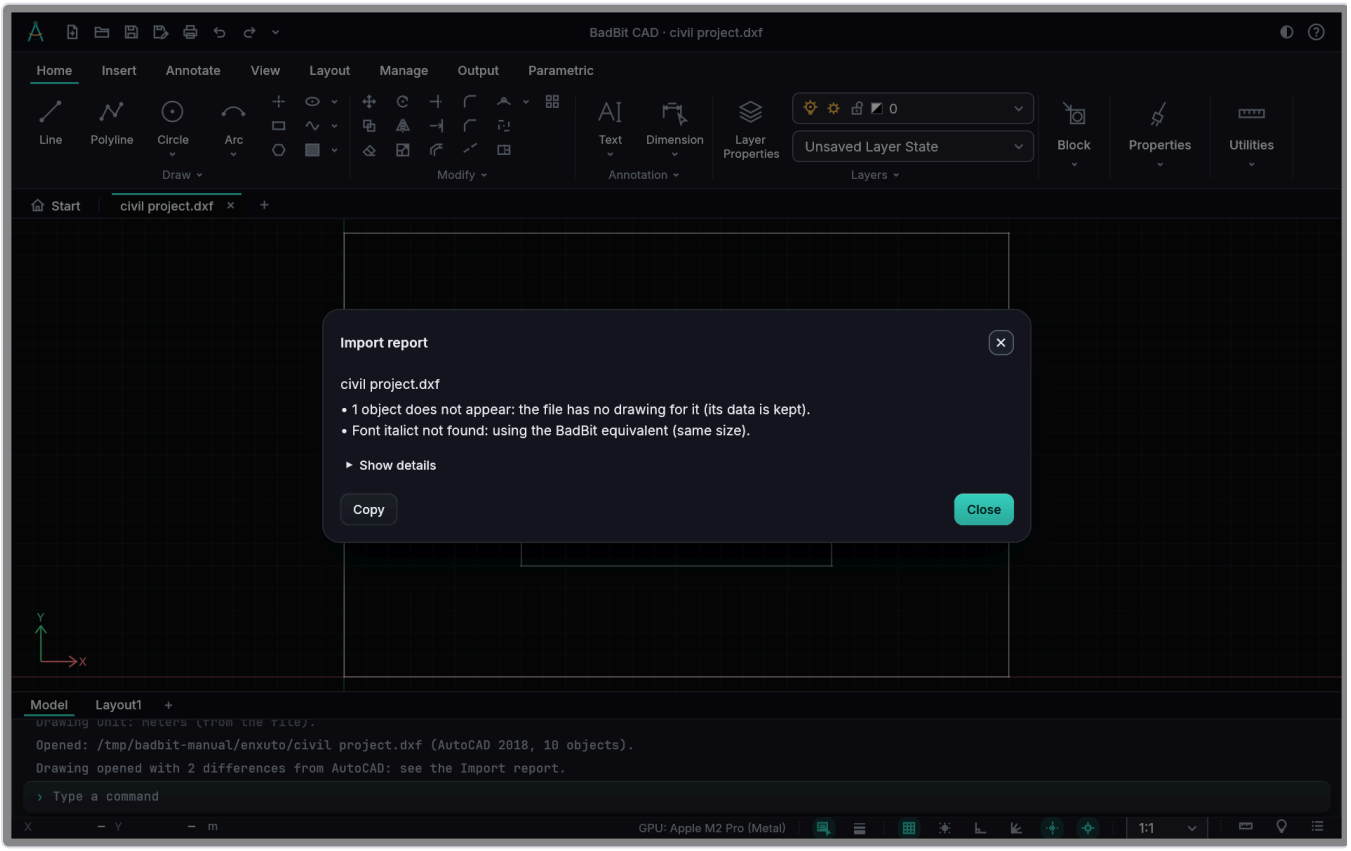
- **The numbers stay those of the file.** Coordinates are not converted: only the unit (`INSUNITS`) says what they measure. Saving back to DXF or DWG, in any version, whether or not through `.bbd`, writes the same numbers, bit for bit (tested with UTM coordinates such as 642776.2149028378 and 8127326.7306 and with files in mm, cm, km and feet).

- **The file's unit and formats apply to that drawing**, each tab with its own: the length unit (AutoCAD's 25: inch, foot, mile, mm, cm, m, km, microinch, mil, yard, ångström, nanometer, micrometer, decimeter, decameter, hectometer, gigameter, astronomical unit, light year, parsec and the US survey units), the type and decimal places of lengths (`LUNITS`, `LUPREC`) and of angles (`AUNITS`, `AUPREC`), the zero direction (`ANGBASE`), the direction (`ANGDIR`) and the metric or imperial system (`MEASUREMENT`). The status bar (with the unit next to the coordinates), dynamic input, LIST, DIST, AREA, ID and the UNITS dialog use that unit; typed coordinates also stay exactly as typed. Saving writes back the same variables.
- **Unitless file** (`INSUNITS` 0, common in older drawings and in surveying): the **Drawing unit** dialog asks what unit the numbers are in, with meters already selected; "Do not ask again for this file" stores the choice. "Leave unitless" shows the numbers as they are. With FILEDIA 0, INSUNITSDEFSOURCE applies (meters by default), stated on the command line. BadBit CAD never assumes the unit silently.
- **INSERT and XREF from another unit**: the block comes in with the numbers of its own file and the insert carries the factor between the units (a file in mm inserted into a drawing in meters comes in with the scale times 0.001), with a warning on the command line. With the same unit, nothing changes.
- **Areas and distances in UTM coordinates** are computed without losing precision: AREA, LIST and MASSPROP of a UTM polygon match the value computed by hand to within one millionth of a square meter.
- **Hatch with computed corners, round trip.** DXF and DWG only store coordinates, and an intersection point is rounded in the file (in UTM, to half a billionth of a meter): where the straight segment joins two curves (arc, circle, ellipse, spline), reading rebuilds the corner from the curve and the measurement comes back the same; an outline made only of lines whose corners are intersections, or with a trimmed polyline arc (which goes through the center, rounded), comes back with that rounding, which on a 1 cm figure in UTM changes the area by ~1e-7 (the limit of the format).
- **Scale of ANSI hatches in AutoCAD.** BadBit CAD's ANSI patterns (ANSI31 to ANSI38) have the standard's measurements, in meters; AutoCAD rebuilds the hatch from its own pattern file (in mm in a metric drawing) when it is edited. So the file carries the scale AutoCAD understands: an ANSI31 at scale 1 in a model in meters shows in AutoCAD with scale 0.0394, and the one on the sheet (1.25 mm between lines) with 0.3937, and editing the hatch in AutoCAD keeps the spacing. Opened back in BadBit CAD, the scale is 1 again (if AutoCAD did not change it). BadBit's other patterns are its own designs: in AutoCAD, editing the hatch switches to AutoCAD's pattern of the same name.
- Text DXF is written with the smallest number of digits that round-trips to the same value (up to 17; AutoCAD writes 16, which is not always enough).



A unitless DXF: meters is already selected.

Import report



A project with an object that does not appear and a missing font: the dialog opens by itself and the top says only what looks different from AutoCAD.

The top of the report is for the person using the drawing: it says, in short sentences, **only what does not appear or appears differently from AutoCAD**. One sentence per topic, for example:

- **"1,166 Civil 3D objects appear only as outlines, as in AutoCAD without Civil 3D; their data is kept."** — points, surfaces, parcels, daylight lines: BadBit CAD shows the drawing that Civil 3D saves for those who do not have Civil 3D (the proxy graphics).
- **"2 objects do not appear: the file has no drawing for them (their data is kept)."** — for example, an empty surface.
- **"2 objects from other applications show only the drawing stored in the file; their data is kept."**
- **"Font italict not found: using the BadBit equivalent (same size)."** — one per SHX font that is not in the font folders (the texts use FONTALT, if it is there, or the BadBit CAD equivalent, of the same size).
- **"3 3D objects appear only in plan."**, **"1 item appears simplified (linetype, hatch or dimension)"** — a linetype with text or shapes **in use** (the dashes remain), a hatch outline, a dimension style adjustment.
- File defects **that change the drawing**: **"The file is incomplete (cut off)"**, **"N objects with invalid data were left out"**, **"N objects point to a layer, linetype or style missing from the file and use the default"**, **"The DWG has N defects; what could be read appears, as with AutoCAD's RECOVER"**, **"N damaged parts of the file were ignored"**.

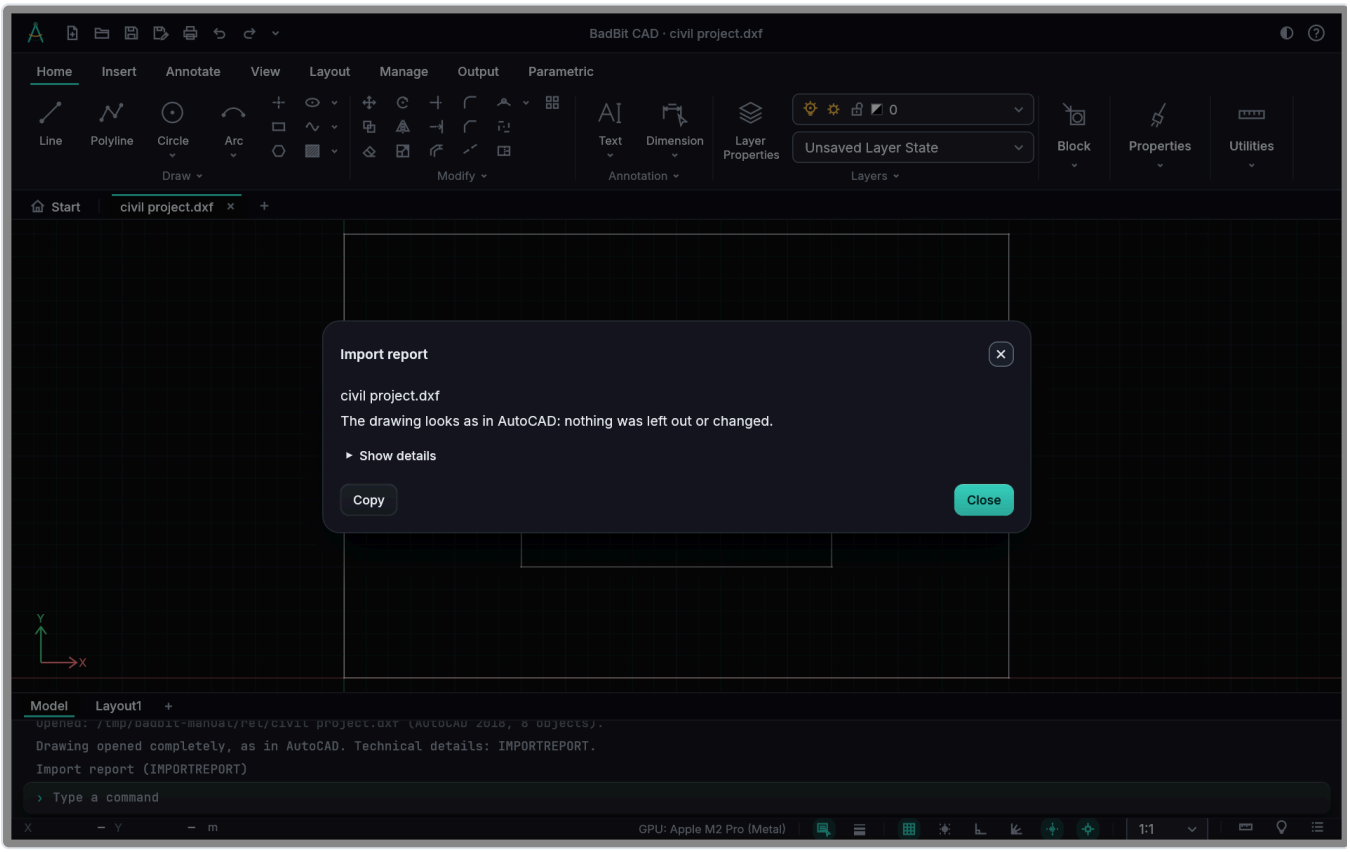
With no differences at all, the top says **"The drawing looks as in AutoCAD: nothing was left out or changed."**

What is technical or changes nothing on screen goes under **Show details** (collapsed):

- **Kept intact** — the **non-graphical** objects of the applications (label and table styles, settings nodes, display properties), summed up in one line per application: **Civil 3D** (`AECC_*`), **AutoCAD Architecture and MEP** (`AEC_*`, `AECB_*`), **AutoCAD Map and geolocation** (`DM*`, `MAP*`), **3D and rendering** and the **other applications**, by the name the file gives; and where they go back to (DWG or DXF).
- **The groups with their counts** — visible but not understood, simplified, repaired or ignored, adjustments with no effect on the drawing (renamed items, UCS, unused complex linetypes), kept.
- **The complete list, type by type** — each warning with the object type and how many times, including the repaired defects that do not change the drawing (duplicate handle, text encoding different from the declared one).

Kept intact means that BadBit CAD does not interpret the object, but does not lose it either: it stays with the drawing, byte for byte, and goes back unchanged when saving in the same format — in DWG, **in the same source version** (the Civil 3D and AutoCAD Map ones also go to the other DWG versions and to DXF — from a DWG 2007 or newer, converted or as proxies; from a DWG 2000 or 2004, as proxies with the data as it was —, as AutoCAD saves them; the architecture objects AutoCAD knows are left out of DXF, as in AutoCAD; in another DWG version or in DXF the others do not go, and the save report says which); in DXF, when saving as DXF. A Civil 3D project has thousands of these objects, and the drawing is complete even so.

The dialog **only opens by itself** when something appears differently from AutoCAD (or does not appear); the command line says how many differences: **"Drawing opened with 4 differences from AutoCAD: see the Import report."** When nothing changes on screen (only objects kept intact, adjustments and defects with no effect), the dialog stays closed and the command line says **"Drawing opened completely, as in AutoCAD. Technical details: IMPORTREPORT."**



Only application styles and settings: nothing different on screen; IMPORTREPORT opens the report, with the technical part under Show details.

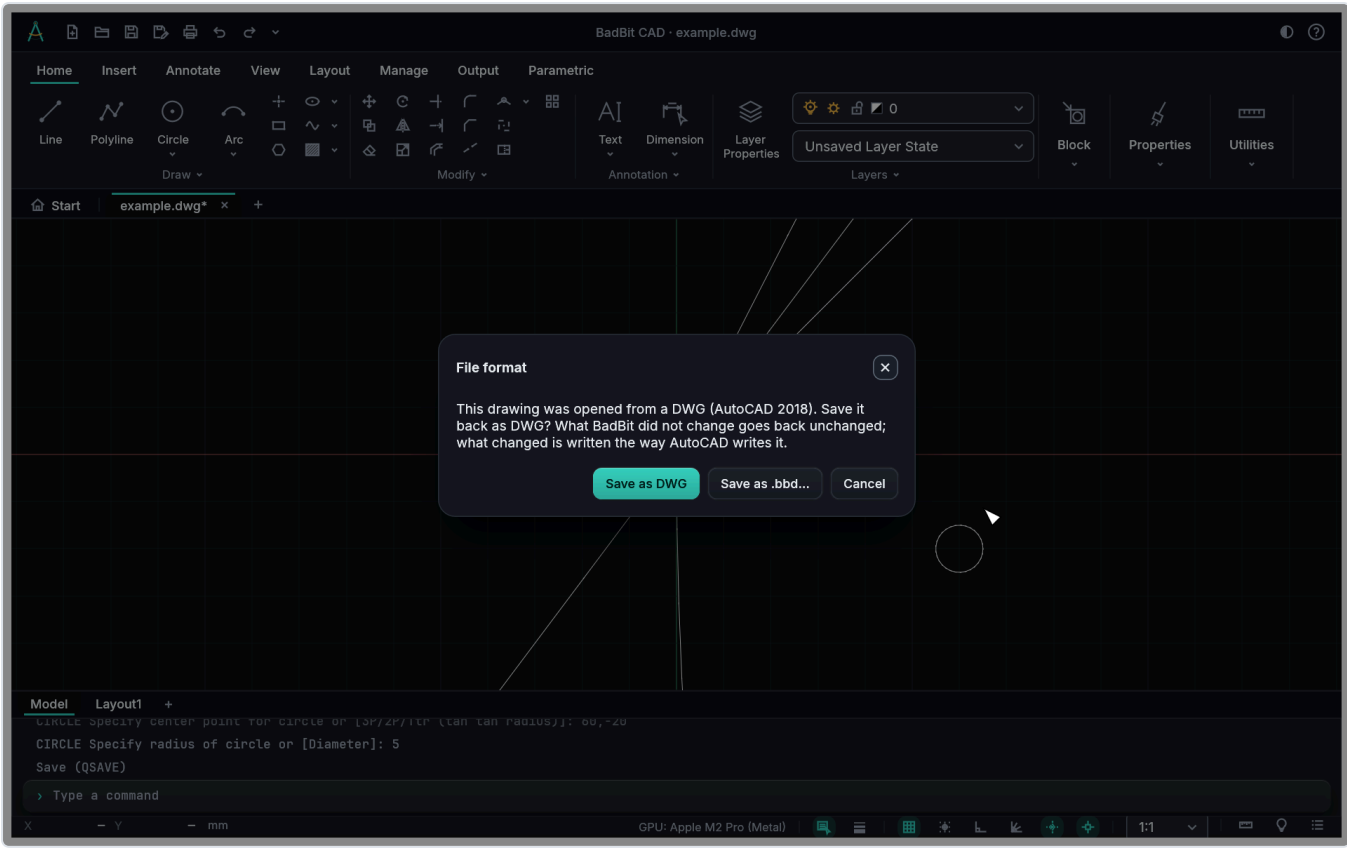
The **IMPORTREPORT** command opens the report later (also on the **Manage** tab, **DWG and DXF** panel). **RECOVER** always shows the result, even when there is nothing that needs attention; **AUDIT** gives the result on the command line.

Copy puts the whole report on the clipboard, with the details.

Anonymous blocks in DWG. AutoCAD writes into the DWG the name of anonymous blocks (`*U`, `*X`, `*D`, `*T`) without the number, which is stored only in the block definition; BadBit CAD reads the number from there (`*U9`, `*X12`), as in DXF, and writes it as AutoCAD does. That way Civil 3D volume tables ("Cut/Fill Summary") and the other anonymous blocks appear in place, next to the texts that go with them. The texts of these tables request the font in the text itself — **Courier New**, the typewriter one, in bold only in the title and the factors —, and BadBit CAD uses the Courier New installed on the computer (without it, the substitute, with the font warning).

Saving as DWG

A drawing opened from a DWG saves back to DWG, in its version. The first time, BadBit CAD asks (as it does with DXF):



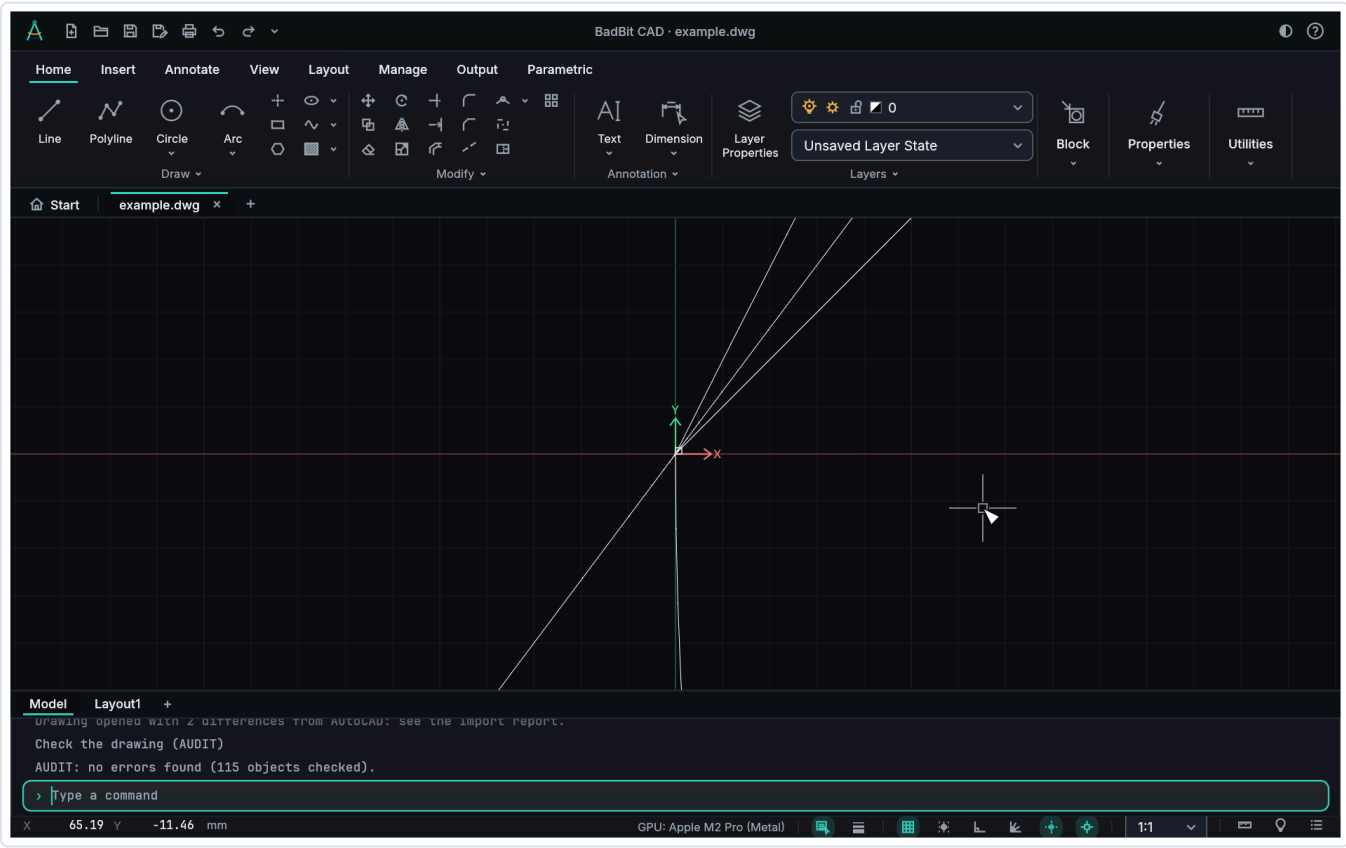
- **Save as DWG** writes to the same file, in its version (the next times it does not ask).
- **Save as .bbd...** opens Save As.

To write a DWG from any drawing (a `.bbd`, a DXF), use **SAVEAS** with the `.dwg` extension and choose the version (2018 is the default).

- **What did not change goes back as it came.** Each object of the source DWG that BadBit CAD did not change is rewritten with the same bits as the original file (only the links to the other objects are rebuilt), when the version written is the same as the one read.
- **What changed, or was created in BadBit CAD,** is written as AutoCAD writes it: the same fields, in the same order, with the same compression choices (checked object by object against AutoCAD files).
- **Another version.** Saving in another version converts everything through BadBit CAD; application objects and those that exist only as bytes go **only** to the source version — in the others they are left out, with the links tidied up, and the save report says which. The exception is **Civil 3D objects** (and AutoCAD Map objects), which also go to the **other DWG versions** — from 2000 to 2018 — and to DXF, whatever version they come from (see "Saving a Civil 3D project in another DWG version", below). The **render settings**, the **sun** and the AutoCAD Map **data table** go into every DWG and DXF version, as AutoCAD writes them (the sun does not go to DXF 2000 and 2004, as in AutoCAD). **Materials** go from DWG to DXF, from DXF to DWG and between all versions, with all their fields, and the entities keep their material; in 2000 and 2004, which do not have translucency, self-illumination, reflectivity and the illumination model, BadBit CAD keeps them in the material as AutoCAD does, and they come back when saving again in 2007 or newer. The material of each entity (ByBlock, Global or its own material), which a 2000 and 2004 entity does not have, goes with it in the extended data, as AutoCAD writes it, and comes back when that file is opened in BadBit CAD or in AutoCAD. **Visual styles** go from DWG

to DXF, from DXF to DWG and between all versions, from 2000 to 2018 (in 2010 and earlier, the properties the version does not have are kept in the style, as AutoCAD does, and come back when saving again in a newer version). When opening a DXF 2000, 2004 or 2007, AutoCAD loses the names of visual styles; BadBit CAD keeps them. **Table styles** go with their cell styles in every version and in DXF, as AutoCAD writes them (in DXF and in DWG up to 2007 they also go separately, in the cell style map; from 2010 on, only inside the table style), and DXF 2013 or newer carries the layout thumbnail the source file had. The file's named **page setups** and named **plot styles** come back in every version, and the drawing variables the version does not have in the header (in 2000 and 2004) go into the variable dictionary, as AutoCAD writes them. Also going to the other version are the **page setup of each layout** (the device, paper, scale and plot area the file had), the file's **extension dictionaries** (draw order, the geographic location in DWG, application data), the **associativity** and the **seed point** of hatches, **splines by fit points** (they remain fit-point splines in AutoCAD) and the **drawing properties** (title, subject, author, keywords, comments and the custom ones), which in 2000 go where AutoCAD keeps them and come back when saving again in a newer version. **Detail view and section view styles** also go from DWG to DXF, from DXF to DWG and between all versions (the display name, which only 2018 has, is kept in the style in earlier versions, as AutoCAD does; DXF 2000 does not carry their colors, as in AutoCAD).

- **Check.** Every DWG written is read back by BadBit CAD itself before it is delivered: an object that would not come back the same (corrupted data coming from the source file) is dropped from the file, with a warning, instead of going in damaged. **AUDIT** checks the drawing the same way, without saving:

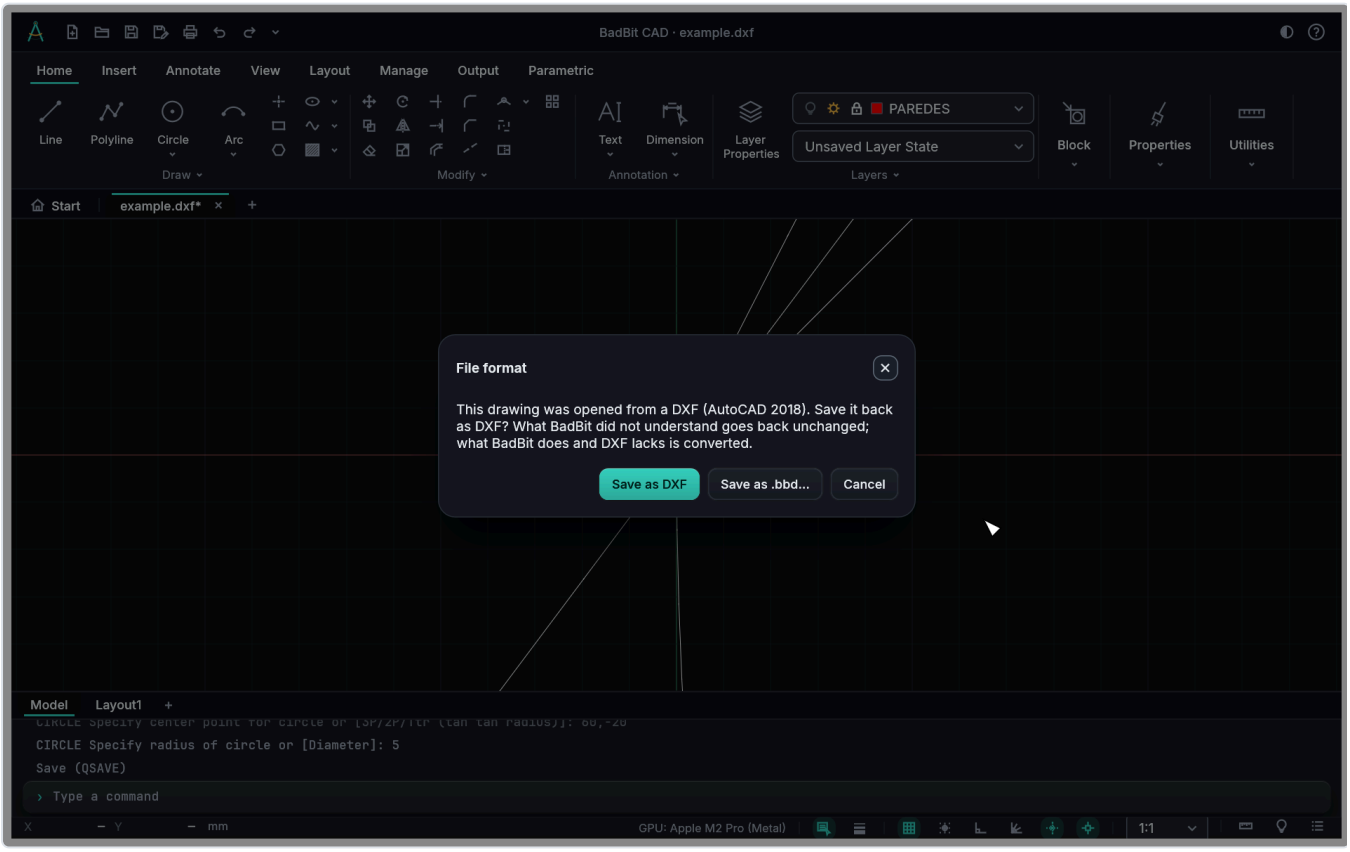


AUDIT: the drawing as it would go into the AutoCAD file, checked.

- **Performance.** One million objects are saved in about 3 seconds, in the background.

Saving as DXF

A drawing opened from a DXF saves back to DXF. The first time, BadBit CAD asks:



- **Save as DXF** writes to the same file, in its version (the next times it does not ask).
- **Save as .bbd...** opens Save As (`.bbd` keeps everything, including what BadBit CAD does not understand in the DXF).

What **did not change** in BadBit CAD goes back **exactly as it came** from the file (with the handles, the extended data, the Z of the points and everything BadBit CAD does not use), when the version written is the same as the one read. What changed, or was drawn in BadBit CAD, is written from BadBit CAD, carrying along from the original what it can (the extended data, the Z). Saving in another version converts everything through BadBit CAD.

What the chosen version does not have is converted into what it has, and the save report says what:

In BadBit CAD	In DXF
Multileader (MLEADER)	Classic leader (LEADER) and multiline text.
Table in 2000 and R12	An anonymous block with the lines, texts and backgrounds (looks the same; from 2004 on it goes as a table, see below).
Associative array	An anonymous block with the items.
Region	The polylines of the outline and the holes.
Gradient in 2000	Solid in the first color.
Background mask and defined height of multiline text in 2000	In the text's extended data, as AutoCAD writes them; they come back when opened in AutoCAD or in BadBit CAD.
Multiline attribute before 2018	One attribute per line (`OBS_001`, `OBS_002`…), with the original kept in the first one, as AutoCAD writes it; it becomes a multiline attribute again when opened in AutoCAD or in BadBit CAD (also those AutoCAD saved this way).
True color (RGB) in 2000	The nearest color index (ACI), by the AutoCAD palette. In DWG 2000 the RGB goes separately, as AutoCAD writes it, and comes back when opened in AutoCAD or in BadBit CAD; in DXF 2000 it is lost (AutoCAD loses it too).
In R12: lightweight polyline, ellipse, spline, multiline text, hatch, leader, tolerance, construction line	Old-style polyline, polylines, single-line texts, the pattern lines, the parts, long line.

Saving tidies up the links between objects as AutoCAD's AUDIT would: unique handles, the extended data applications registered, and references to objects that no longer exist (erased in BadBit CAD) nulled or removed from groups and dictionaries. The same check as AUDIT runs in BadBit CAD's tests on every file written.

What goes natively to AutoCAD

Besides the drawing objects, these drawing data go into DXF and DWG as AutoCAD writes them, and come back when opening:

- **Tables** (from 2004 on): they go as an **AutoCAD table** (ACAD_TABLE), with the graphics block that AutoCAD shows until it regenerates the table. Each cell carries the text it shows (the formula already computed, the number in its format), the text rotation and what it overrides from the style (alignment, background, text color and height, the four borders). The **table style** goes complete: the title, header and data rows with the six borders and, from 2007 on, all **cell styles** — yours too —, with the text rotation, the margins and the **border linetype**. In DWG from 2010 on, the table also carries the cell style of each row and each cell and the **table breaking** (the parts, the maximum height and the spacing). In 2000 (which has no tables) the block goes, as AutoCAD saves it.
- **Nothing is lost in the round trip.** What AutoCAD does not keep in the table (the minimum row heights, the format with the unit and separator, the breaking in DXF, the cell styles before 2007) goes along, in a BadBit CAD data record that AutoCAD carries without touching. When opened

back, the table and the style return **exactly** as they were, in any version. If the table was changed in AutoCAD (a text, a row, a column, the position) or the style was changed there, what AutoCAD saved prevails.

- **Scale list** (SCALELISTEDIT, from 2007 on): the drawing's list goes into the file's scale dictionary (`ACAD_SCALELIST`) and comes back when opening. When edited in BadBit CAD, scales with the same name stay the same for AutoCAD (the annotative objects that use them are not lost); an erased scale that some annotative object still uses stays, as AutoCAD does.
- **Active layout**: saved with a layout on screen, the file writes `$TILEMODE` 0 and that layout as the active one; AutoCAD (and BadBit CAD) open in it.
- **Current table style** (CTABLESTYLE) and **current multileader style** (CMLEADERSTYLE): they go into the DXF header and the variable dictionary (their place in DWG), and CTABLESTYLE is also kept in `.bbd`.
- **Color book colors**: the colors of an AutoCAD color book (RAL, PANTONE...) that the file carries in layers, objects and the background of multiline text come back with the **book and color name** next to the RGB. When you change the color of an object in BadBit CAD, the name goes away (the new color is not from the book). In DWG the name goes in the layers and the text background; in ordinary objects the color points to a color object (DBCOLOR) with the name, kept in the drawing's color dictionary (`ACAD_COLOR`), which BadBit CAD reads and writes (from 2004 on; in DXF both go, the name in the object and the color object). The form of this object in DWG has not yet been checked against an AutoCAD file with a color book color (comparison phase); a color object of another form is kept as it came. Inside multiline text, AutoCAD keeps only the RGB (so does BadBit CAD).
- **ANNOALLVISIBLE** (from 2007 on): it goes into the DXF header (`$ANNOALLVISIBLE`) and the variable dictionary (its place in DWG, which has no header field for it), and comes back when opening.
- **Unit of external references** (from 2007 on): the file unit of each external reference, stored by XATTACH, goes into the reference's block record (the `INSUNITS` of the BLOCK_RECORD) and comes back when opening; XREF uses it for the insert factor when reloading.
- **Thumbnail**: DWG and DXF files saved by BadBit CAD carry the drawing thumbnail (the preview that AutoCAD's Open dialog, Windows File Explorer and the Start tab show), drawn by the CPU at save time, in a thread, with the **current view** (the model as it is on screen, or the sheet of the current layout), as AutoCAD does: in DWG, the preview section with the BMP (up to 2010) or the PNG (from 2013 on); in DXF, the THUMBNAILIMAGE section. Without the screen ready (before the first frame), the whole drawing is used; in that case, very large drawings (more than 300,000 objects) are saved without a new thumbnail, so as not to slow down saving.
- **Associative dimensions** (DIMASSOC 2): the link of each dimension to the geometry goes in the DIMASSOC object, as AutoCAD writes it, in any version, also at the intersection of two objects, at the end of an elliptical arc and in the angular dimension between two lines (which AutoCAD links by the four ends of the lines and BadBit CAD by their intersection: writing and reading convert from one form to the other); AutoCAD's come back linked in BadBit CAD (see "Associative dimensions" in the dimensions chapter). Only the point on the curve of a radius dimension stays in BadBit CAD (AutoCAD keeps only the center).
- **Constraints and parameters** (from 2000 on; in DXF, 2010 and 2018): geometric and dimensional constraints and the variables of the parametric drawing go into AutoCAD's associative network, as it writes them in each version; AutoCAD recognizes them and keeps them when editing (also those on splines, on the end of the ellipse axis and those in paper space), and its own come

back as BadBit CAD constraints, editable (see "Constraints in DXF and DWG" in the parametric drawing chapter). Their reserved layer, `*ADSK_CONSTRAINTS`, goes without the `*` in DWG 2000 and 2004, as AutoCAD writes it. The dimensions of dynamic dimensional constraints, which AutoCAD keeps on that layer, stay with the constraint: they do not appear as ordinary dimensions in BadBit CAD and go back to the file only once, without a block, as it writes them.

- **Dimension variables**: the file header carries the dimension variables of the current dimension style **with its overrides** (DIMTXT, DIMZIN... typed or set with SETVAR), as AutoCAD writes them; when the file is opened (in AutoCAD or in BadBit CAD), those that differ from the style come back as overrides of the current style, and new dimensions come out like the earlier ones. Each dimension keeps the overrides it was created with, so reopening the file does not change existing dimensions. A dimension style with AutoCAD's default values (DIMZIN 0, DIMDEC 4, `.` separator), which AutoCAD does not write to the file, also comes back the same.
- **Hidden attributes of dynamic blocks**: the value of an attribute hidden by the visibility state goes into the file (the attribute comes out invisible in the insert) and comes back when opening.
- **Named filters** (FILTER): they go into a BadBit CAD record in the drawing dictionary and come back when opened in BadBit CAD. AutoCAD keeps its filters outside the drawing (in the user's `filter.nfl` file), so it does not see them; it only keeps them intact.
- **2D solid** (SOLID and TRACE, the four-corner fill): BadBit CAD shows it as a solid hatch, and it goes back as SOLID in any DXF and DWG version, also inside blocks, as AutoCAD writes it (before, saved in another version, it became a hatch).
- **Handles**: every object opened from a DWG or DXF keeps the file's handle when saved without editing — also those of paper space, the model layout and multiline styles —, which is what Civil 3D and other applications use to link one object to another.

Round trip through `.bbd`

Saving as `.bbd` a drawing opened from DWG or DXF keeps along everything that came from AutoCAD (including the bytes of the DWG objects); opening that `.bbd` and saving as DWG or DXF gives the same file as saving directly from the opened drawing.

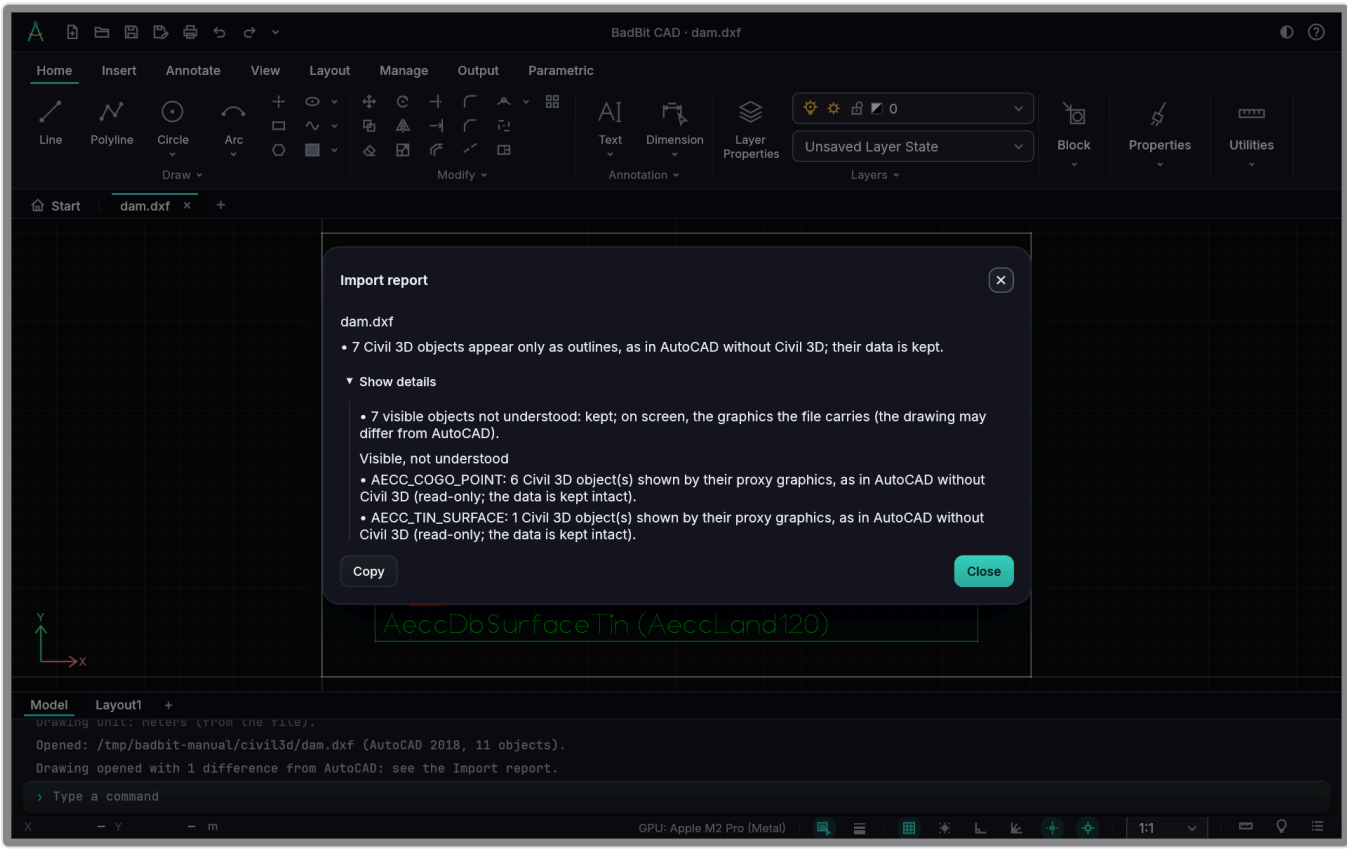
External references and file blocks

External references (XATTACH, XREF) and INSERT of a file accept `.dwg` and `.dxf`. As in AutoCAD, inserting a drawing also brings the used layers, linetypes and its text, dimension and table styles that do not yet exist in the current drawing: this is the way to bring in the styles of a template saved as DXF.

Civil 3D objects

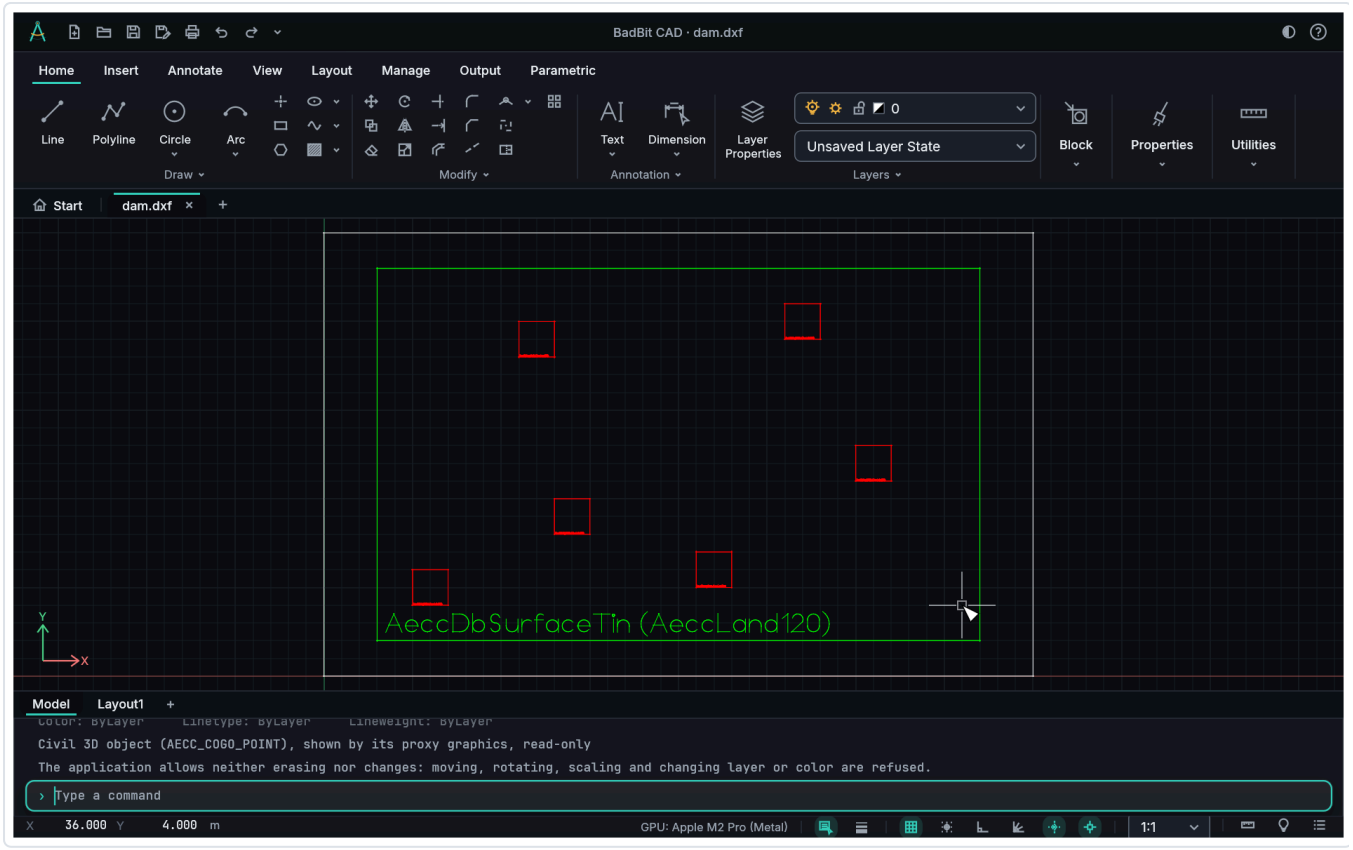
A project made in Civil 3D brings objects that only Civil 3D understands: COGO points, TIN surfaces (with contours), parcels, alignments, feature lines, gradings and their labels. BadBit CAD shows these objects **as AutoCAD without Civil 3D shows them**: through the **proxy graphics**, the drawing that Civil 3D saves with each object for those who do not have the program. What appears is exactly what the file carries. Many projects carry only the box of each object and the class name (for example,

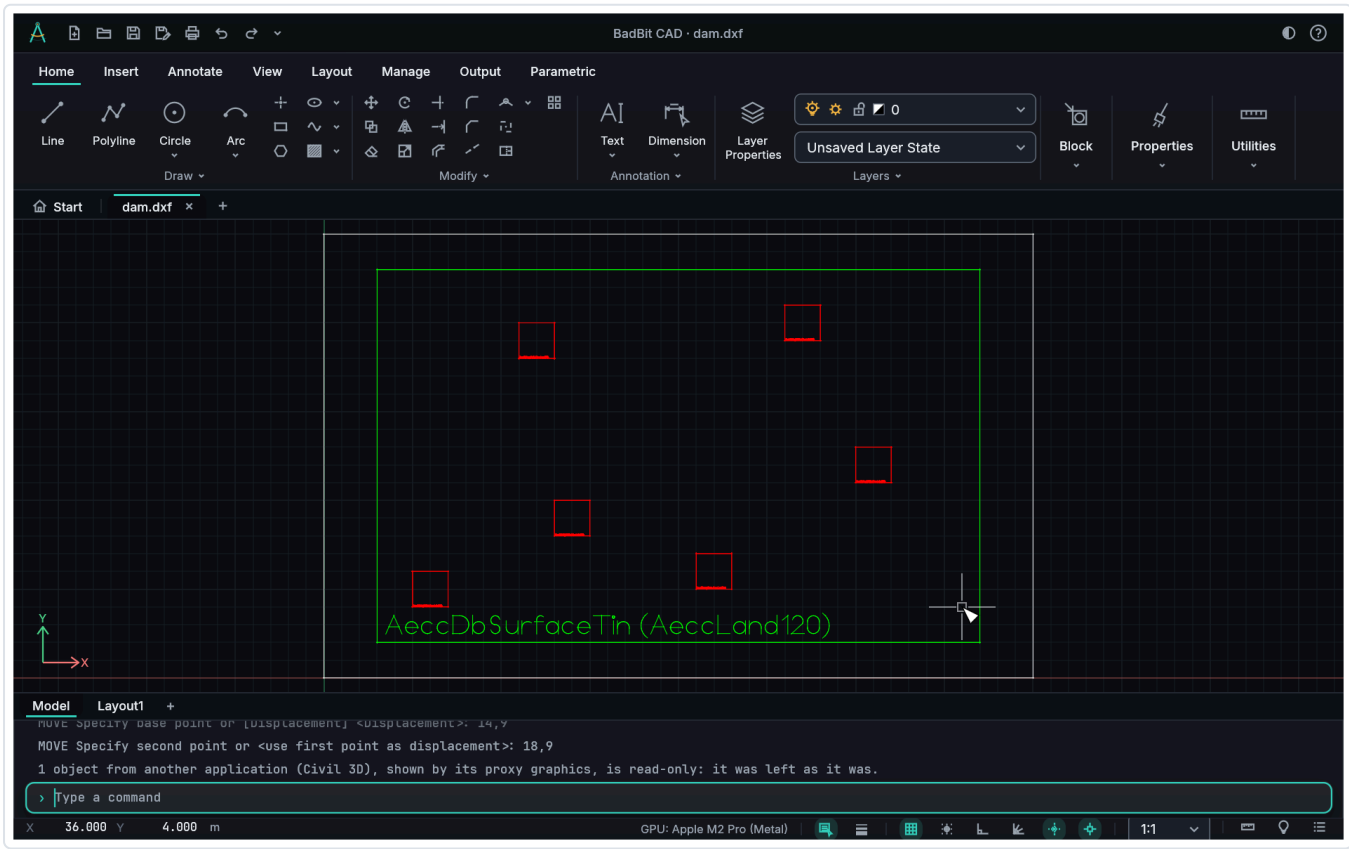
"AeccDbCogoPoint (AeccLand120)"): this is the same as what plain AutoCAD shows from that file. To see contours and labels as ordinary drawing objects, use the "Export to AutoCAD" command (EXPORTTOAUTOCAD) in Civil 3D before sending the file.



What you can do with them:

- **View**: colors, layers and linetypes follow AutoCAD's rule. What the proxy graphic does not define comes from the object (like ByBlock); what it defines (color, layer) prevails. Turning the layer off or freezing it hides the object. An empty object (a surface with no data, which Civil 3D saves with its box far away, at 1e20) does not appear and does not spoil zoom extents (ZE).
- **Select**: by click, by window and by crossing window, like any object.
- **LIST and Properties**: LIST says "Civil 3D object (AECC_COGO_POINT), shown by its proxy graphics, read-only" and whether Civil 3D allows erasing it. In the Properties palette, the "Shown by" row says "Proxy graphics, read-only".
- **Read-only**: move, rotate, scale, mirror, stretch and changing layer or color are refused. The command line warns you, and the drawing stays as it was. This is what AutoCAD does with a proxy object without "editing allowed": changing its drawing without Civil 3D would break the object. In a mixed selection, the other objects change and the Civil 3D ones stay.
- **Erase**: follows the permission that Civil 3D itself writes in the class. Surfaces, parcels, feature lines and labels can be erased; COGO points cannot.
- **Copy**: the copy takes only the drawing (the proxy graphics), without the Civil 3D data. It can be edited, and when saved as DWG or DXF it becomes ordinary lines and texts.





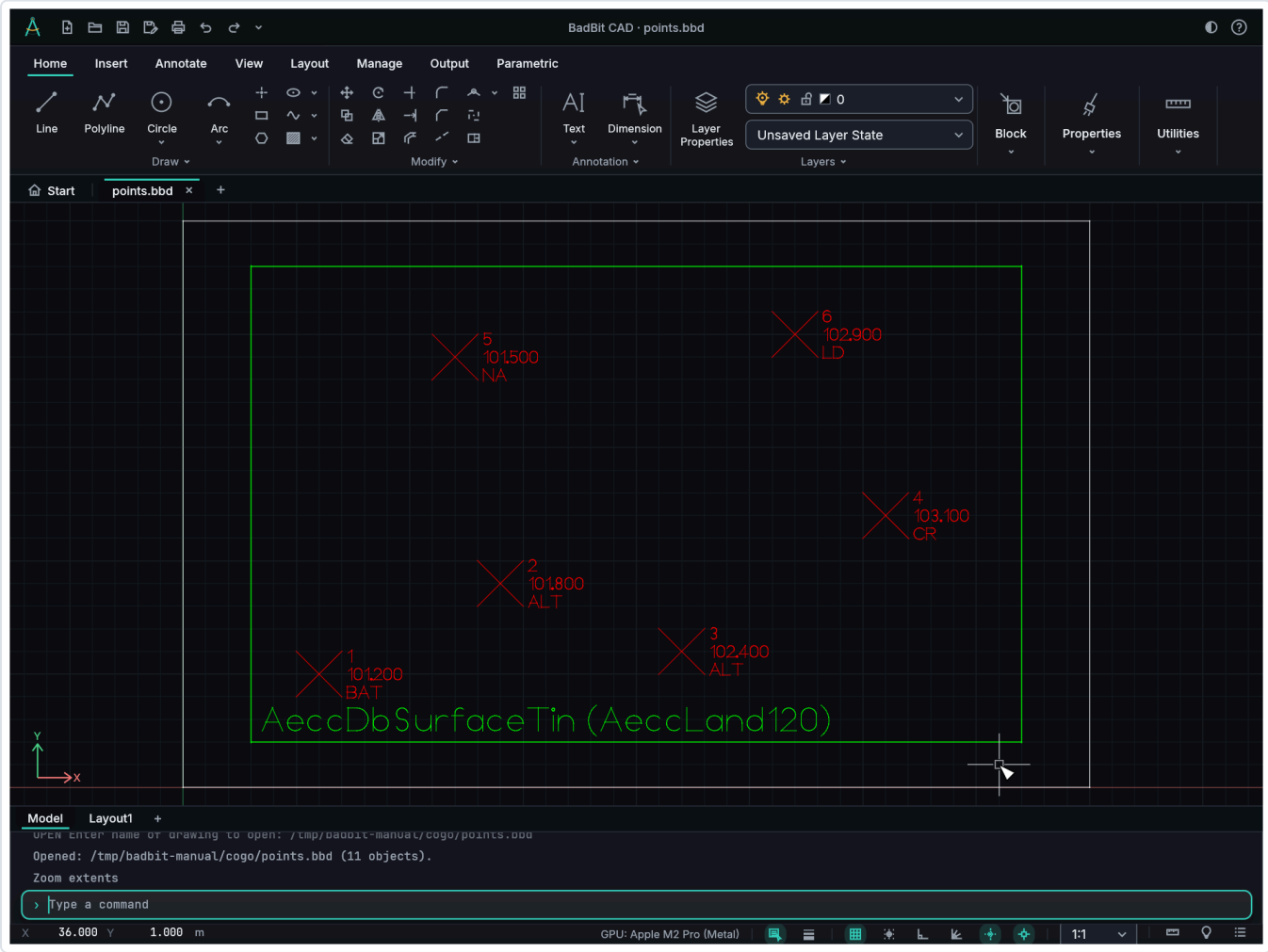
Saving. Saving as DWG in the version of the source file returns each Civil 3D object with the same bytes: Civil 3D opens the file with everything as it was. In another DWG version they go converted (below). In DXF (2000 to 2018) they go **as proxies**, as AutoCAD without Civil 3D writes them: each object with its data, texts and links, and the proxy drawing; Civil 3D's associative actions are left out and the associative network goes without them, also as in AutoCAD. When that DXF is opened and

saved as DWG 2018, the objects return to their class (as AutoCAD does); in the other DWG versions they remain proxies. The open drawing keeps them: saving later as DWG in the source version returns everything.

The Import report counts, by class, the "Civil 3D objects shown by their proxy graphics" (AECC_COGO_POINT, AECC_TIN_SURFACE...).

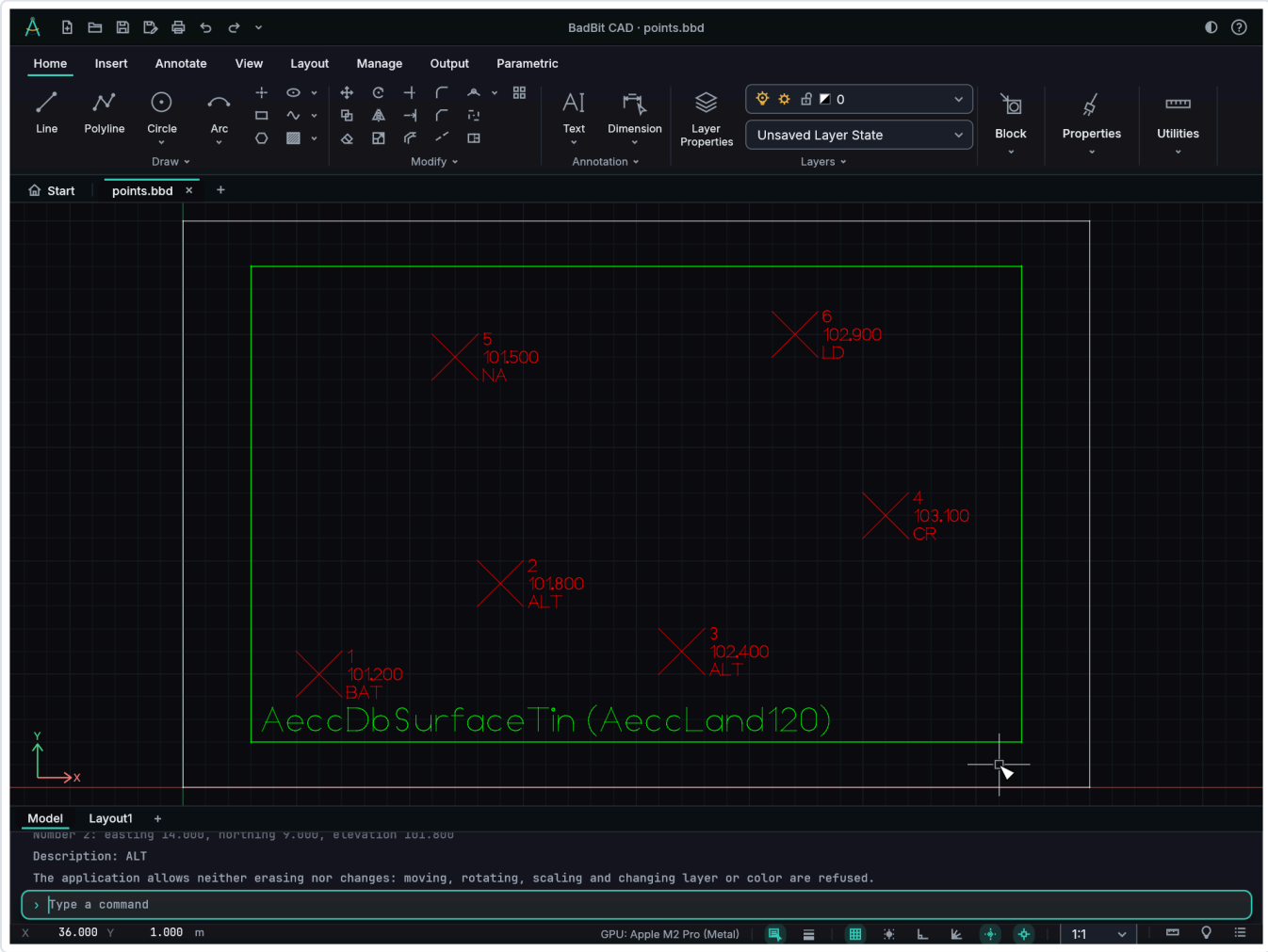
Civil 3D points

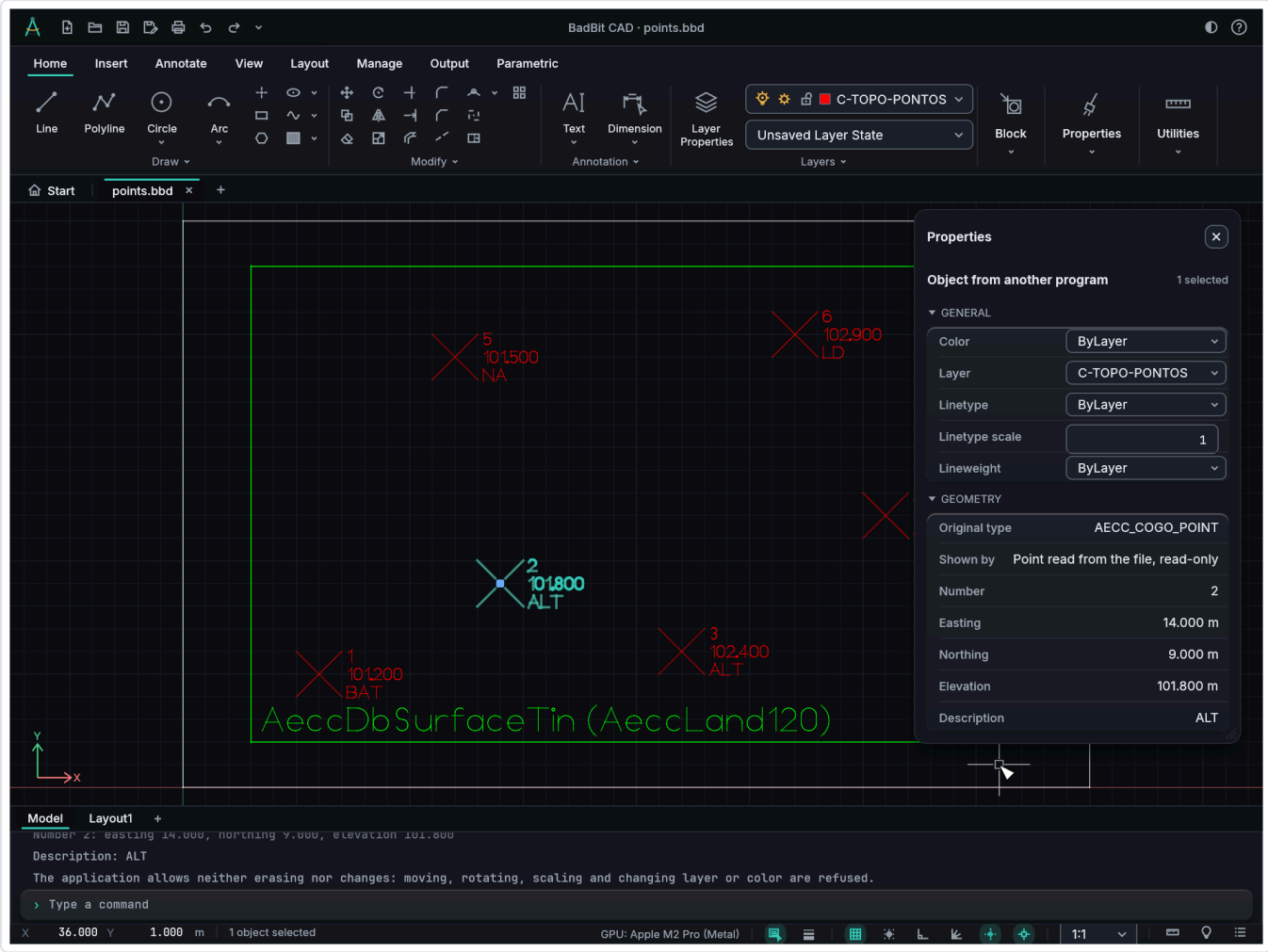
The **COGO points** of a Civil 3D DWG (2000 and later) are not limited to the proxy box: BadBit CAD reads from the file the **number**, **easting**, **northing**, **elevation** and **description** of each point, like AutoCAD with the Civil 3D Object Enabler. The point appears at its **exact position**, with the **marker** and **label** of the project's point style and label style (see "Civil 3D styles", below); with no readable style, with the default marker and label: number, elevation (three decimal places) and description.



- **Label**: only the components the label style **shows**, each with its own **height** and **color**. When the style the point group points to hides all components (the template "Standard" of the collection), BadBit CAD uses the next label style in order, the default of the point settings (in the dam project, **PONTO_COTA_DESC**: number, description in green and elevation in red, 0.2 m at 1:1000). This order is **assumed** and is being checked against Civil 3D.

- **Marker**: the one from the point style (shape, size, layer and color). If the style cannot be read, an X the size of the box Civil 3D saves around the point, with the point object at the center, drawn with the drawing's PDMODE and PDSIZE, and the label with letters half the size of the X.
- **Snap**: **Node** picks the exact point (Civil 3D's easting and northing, without rounding).
- **LIST**: says "Civil 3D COGO point (AECC_COGO_POINT), read from the file, read-only", the number, easting, northing, elevation and description.
- **Properties** (Ctrl+1): the rows Number, Easting, Northing, Elevation, Description, **Point style** and **Label style**, read-only; "Shown by" says "Point read from the file, read-only". With several points selected, what differs appears as "mixed". LIST also shows both styles.
- The point remains **read-only** and cannot be erased (as before), and goes back to the DWG with the same bytes.
- The description shown is the **raw** one (the one typed in Civil 3D); the full description from description keys is not read yet.





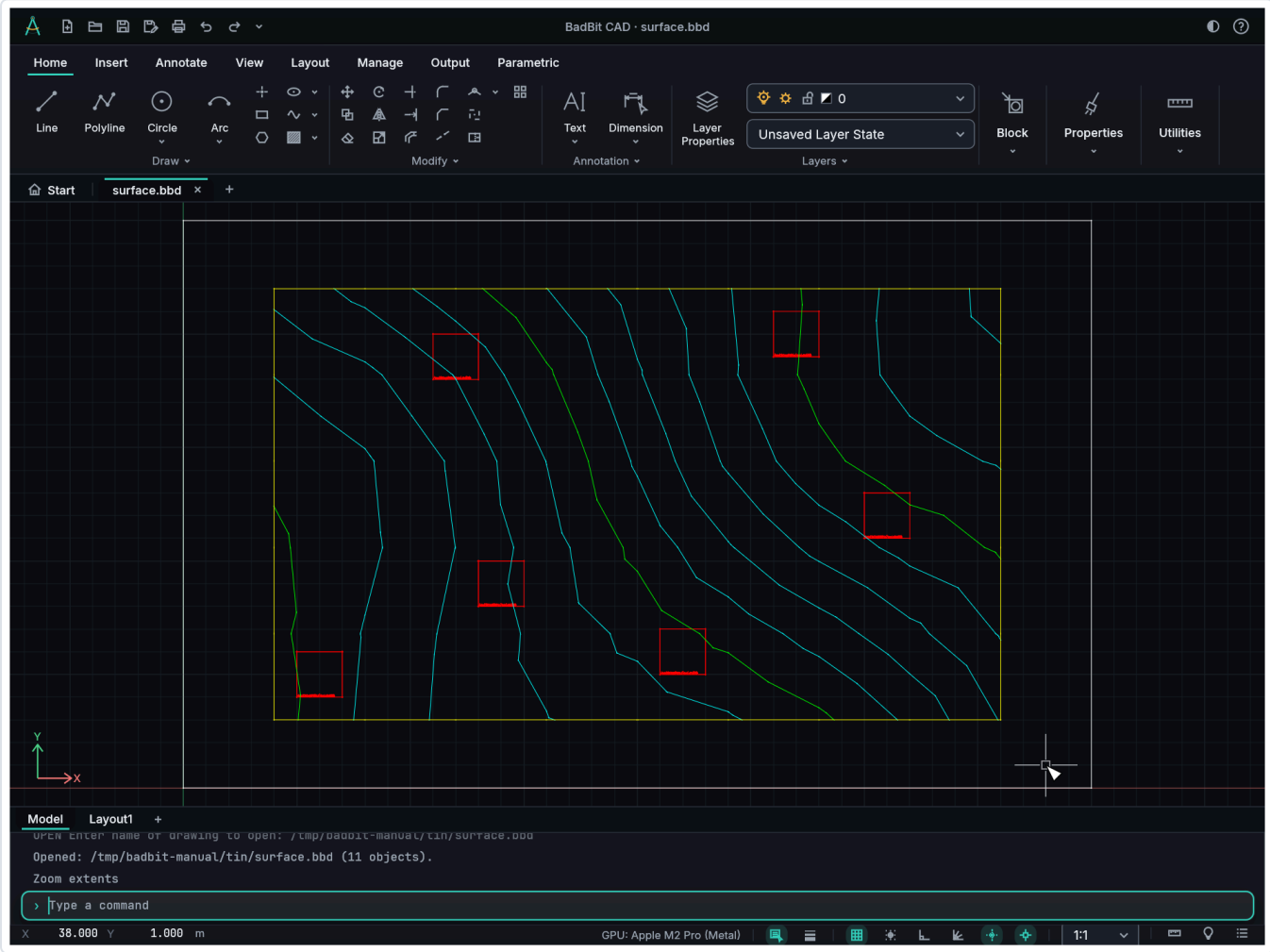
In the Import report, the points read count as "Civil 3D COGO points read from the file", among the objects kept intact, and not as visible but not understood. If a point cannot be read (a different format), it appears through the proxy graphics, like the other Civil 3D objects. In **DWG 2000 and 2004** the texts of Civil 3D objects sit in the middle of the data, in a place that has not yet been checked against a Civil 3D file: BadBit CAD looks for the description of each point and only accepts the table when everything matches; if it does not match, the point appears through the proxy.

Civil 3D surfaces

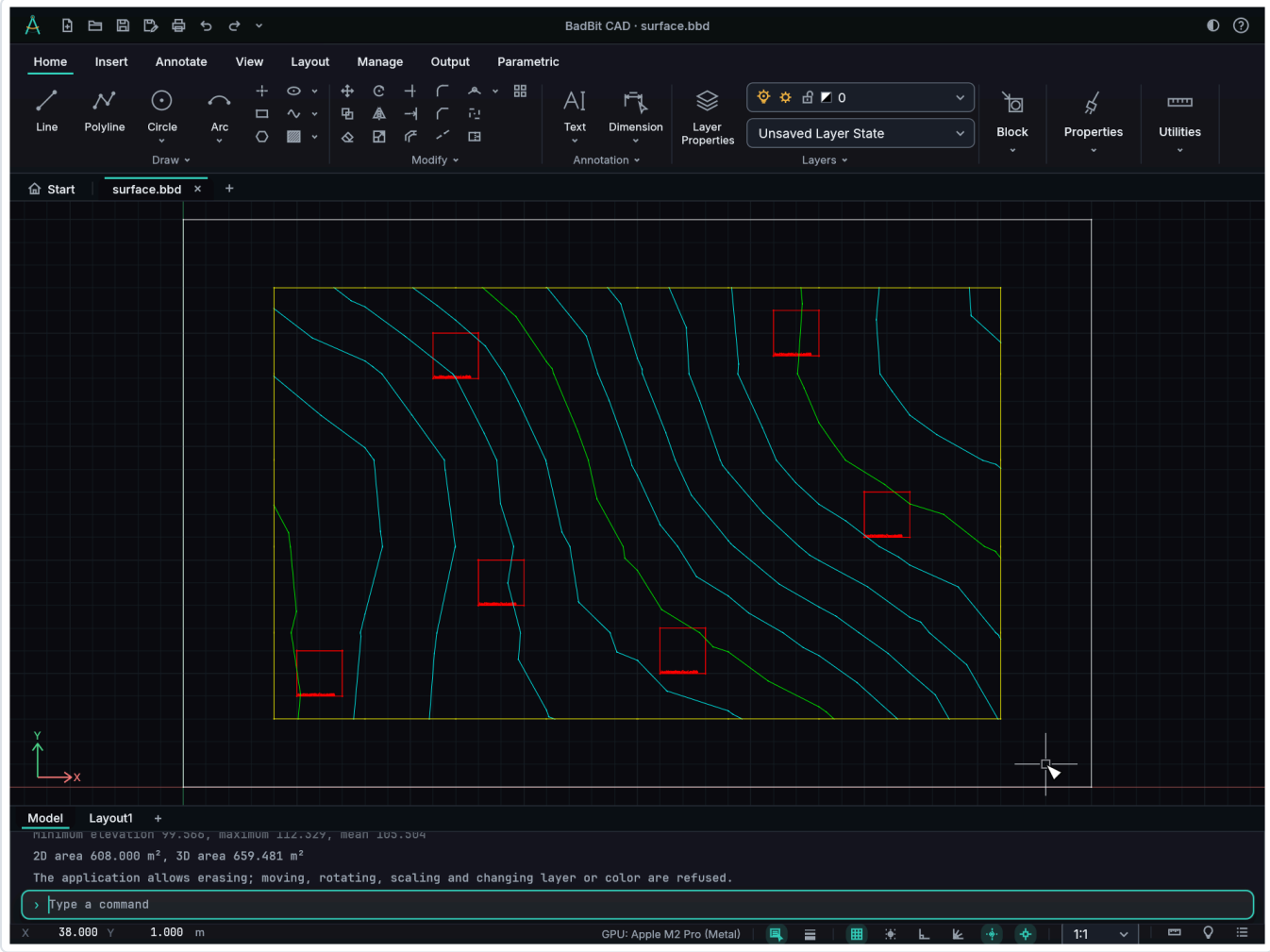
The **TIN surfaces** of a Civil 3D DWG (2000 and later) are not limited to the proxy box either ("AeccDbSurfaceTin (AeccLand120)"): BadBit CAD reads from the file the **vertices** and **triangles** of each surface, its **name** and **style**, and shows it as Civil 3D shows it in plan, with what its style turns on:

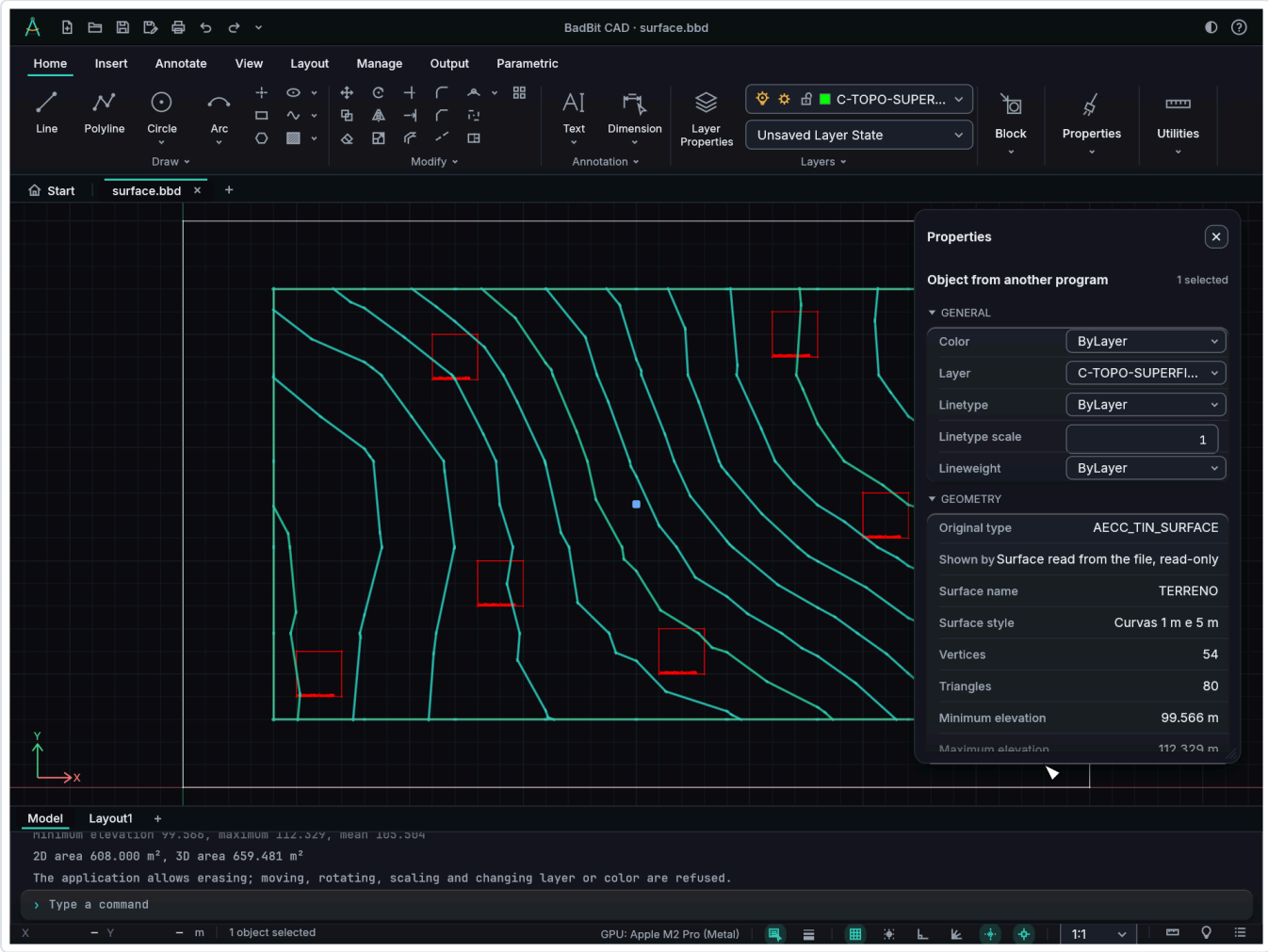
- **Contours**: **minor** and **major** at the **style interval** (for example, 0.1 m and 0.5 m, or 1 m and 5 m). If the style cannot be read, minor contours every 1 unit and major every 5, with a 0.35 mm lineweight. Contours are computed when the file is opened, in parallel, by the same computation as LANDXMLIN.
- **Border**: the outline of the visible triangles.
- **Triangles** and **points** (the vertices, drawn with the drawing's PDMODE and PDSIZE), when the style turns them on.

- A surface with a style that **turns everything off** (for example, volume surfaces with the "INVISIVEL" style) **does not appear**, as in Civil 3D; the data remains in the drawing (LIST and Properties). If the style cannot be read, the border and contours appear.
- Triangles hidden in Civil 3D (outside the border or deleted) are left out of the contours and border.



- **Colors and layers**: each component with the style's **layer**, **color**, **linetype** and **lineweight** (ByLayer, ByBlock or its own color), as in Civil 3D; the style's layer "0" is the surface's layer. With no readable style, everything follows the layer and color of the surface.
- **Snap**: **Endpoint** and **Nearest** pick the contours and border.
- **LIST**: says "Civil 3D TIN surface (AECC_TIN_SURFACE), read from the file, read-only", the name and style, how many vertices and triangles, the minimum, maximum and mean elevations and the 2D (in plan) and 3D (on the surface) areas.
- **Properties** (Ctrl+1): the rows Surface name, Surface style, Vertices, Triangles, Minimum elevation, Maximum elevation, Mean elevation, 2D area, 3D area, **Minor interval** and **Major interval**, read-only; "Shown by" says "Surface read from the file, read-only"; and **Show triangles** (see below).
- The surface remains **read-only** (it can be erased, as before) and goes back to the DWG with the same bytes.
- The **elevations** and the **vertex count** include all points of the surface, including those outside the border; the **areas** and **contours**, only the visible triangles.


BadBit CAD · surface.bbd
Home
Insert
Annotate
View
Layout
Manage
Output
Parametric
Line
Polyline
Circle
Arc
Draw
Modify
Text
Dimension
Annotation
Layer Properties
Unsaved Layer State
Layers
Block
Properties
Utilities
Start
surface.bbd
Model
Layout1
2D area 608.000 m², 3D area 659.481 m²
The application allows erasing; moving, rotating, scaling and changing layer or color are refused.
Type a command
38.000
1.000
m
GPU: Apple M2 Pro (Metal)
1:1

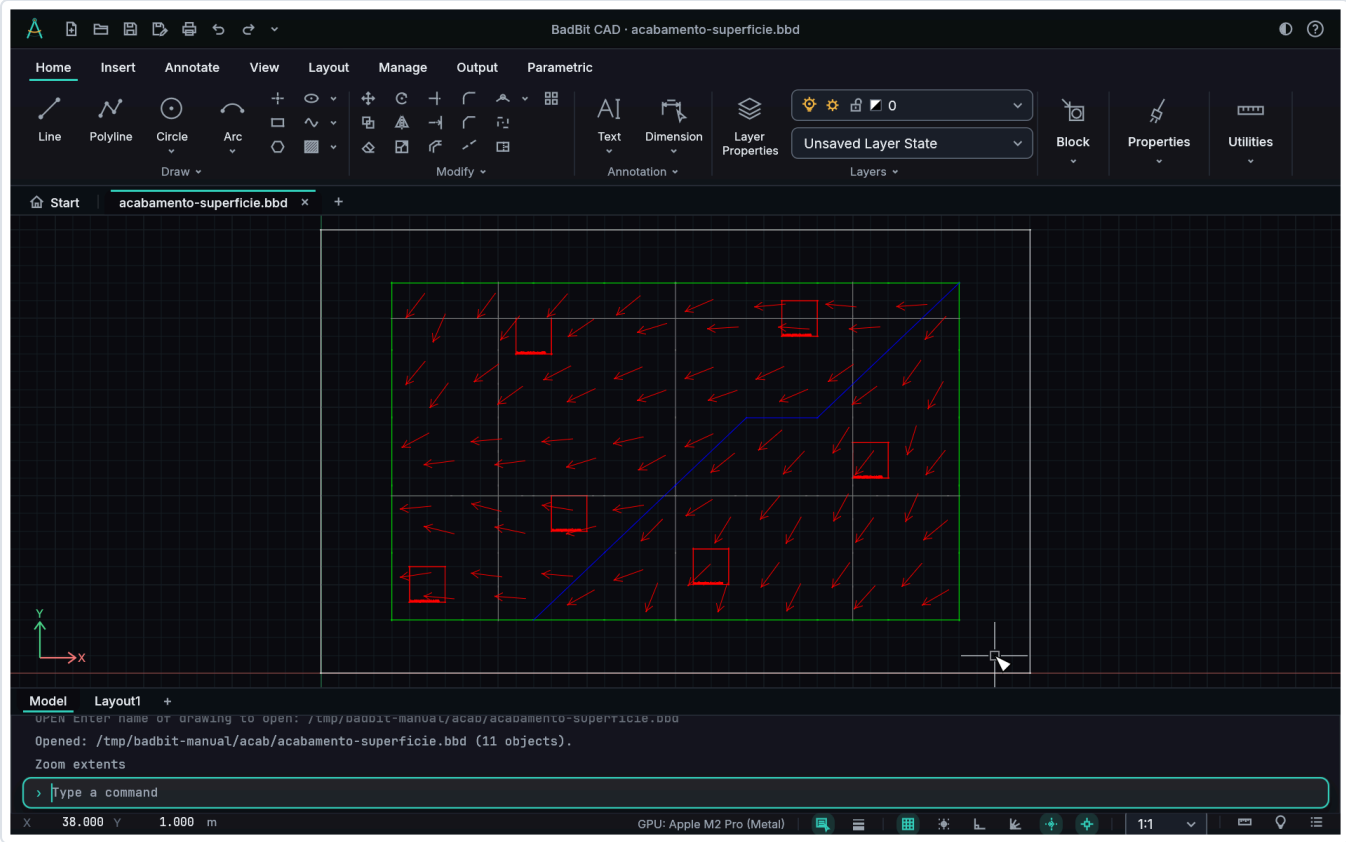


In the Import report, the surfaces read count as "Civil 3D TIN surfaces read from the file", among the objects kept intact, and not as "shown only as outlines". If a surface cannot be read (a different format), it appears through the proxy graphics, as before.

Grid, slope arrows and watersheds. When the surface style turns these components on, BadBit CAD draws them with the layer and color of each one. Civil 3D does not save these drawings in the file (it rebuilds them), so BadBit CAD does its own computations on the triangles:

- **Grid**: the lines of constant X and Y, clipped to the surface, every **10 intervals** of the minor contours.
- **Slope arrows**: one per triangle, at its middle, pointing downhill, the size of the triangle.
- **Watersheds**: water flows from triangle to triangle along the steepest slope; triangles that reach the same low point or the same stretch of the border form a watershed, and the drawing is the boundaries between them.

The grid interval comes from the style (the lines in the X direction and the Y direction with the two values the style stores and that BadBit CAD assumes to be the grid intervals; without them, every 10 minor contours). The arrow size the style stores is not read yet. The report mentions both under "Show details" (`grid-interval`, `slope-arrow-length`).



Grid surfaces are not read yet, and user contours, directions, elevations and slopes are not drawn: when the style turns one of them on, the report says which under "Show details".

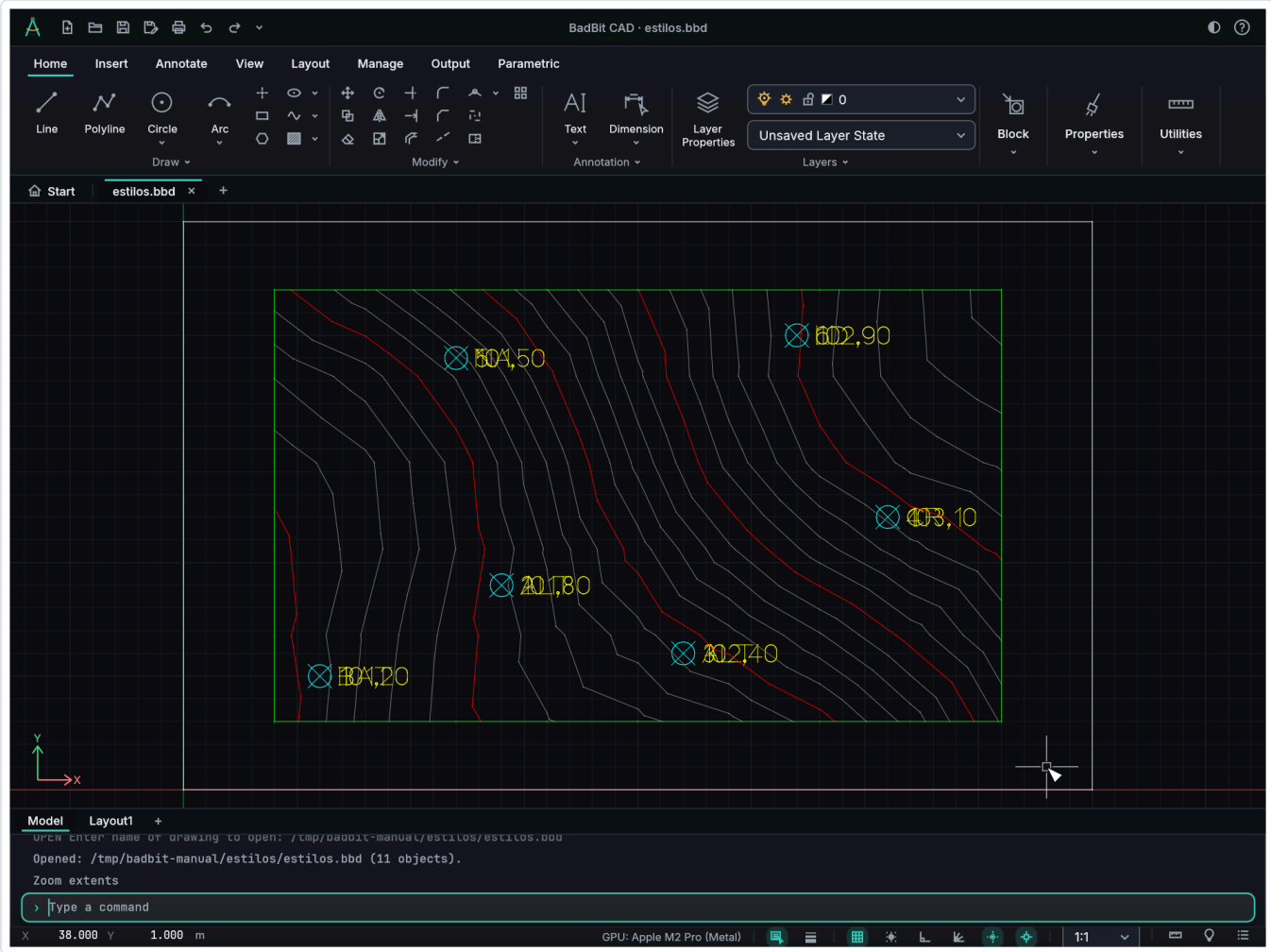
Civil 3D styles

BadBit CAD reads the project's **styles** and draws points and surfaces as the style specifies, like AutoCAD with the Civil 3D Object Enabler:

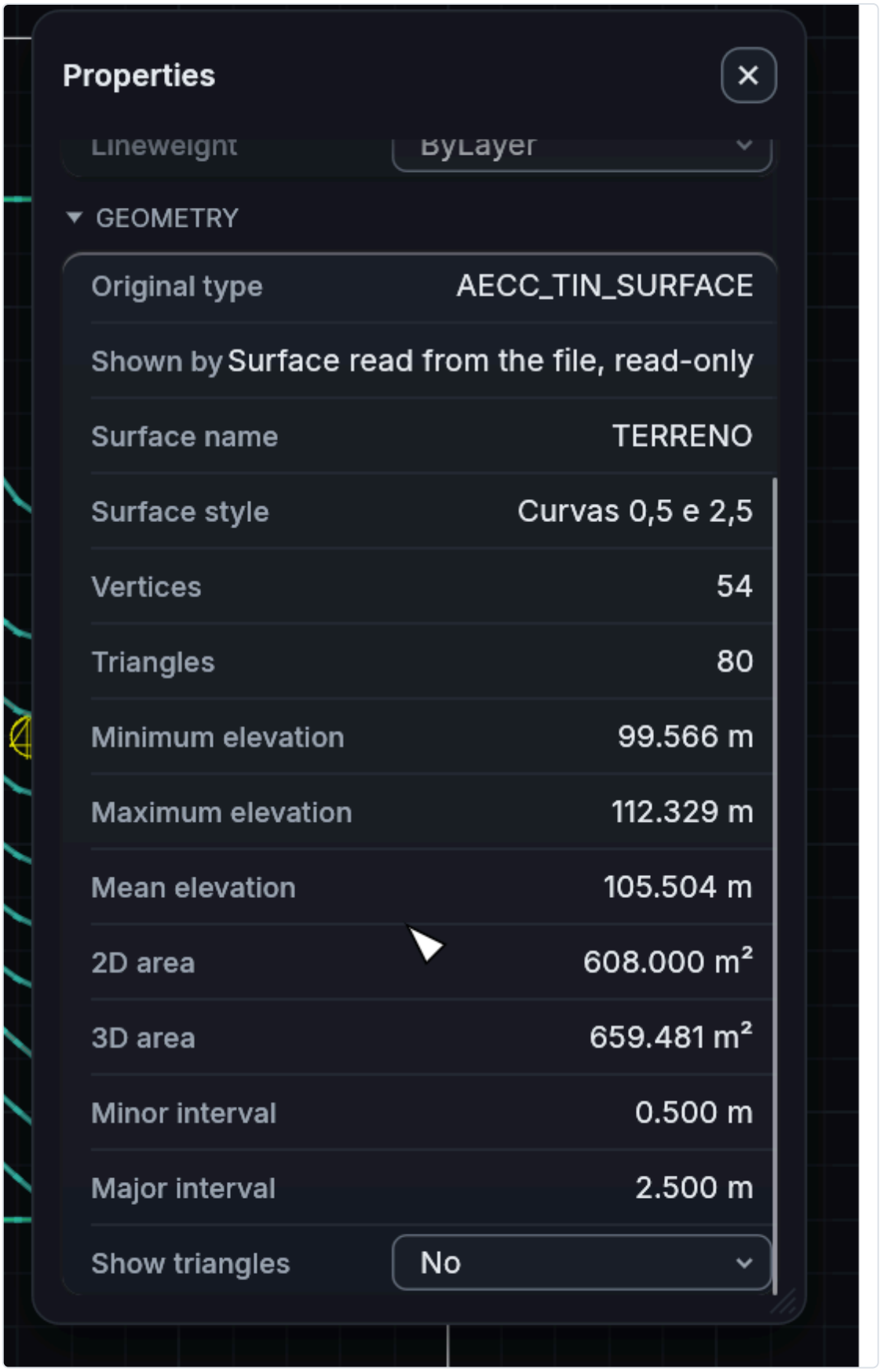
- **Surface style**: the components turned on in plan (points, triangles, border, major and minor contours), each with its own layer, color, linetype and lineweight, and the **interval** of minor and major contours.
- **Point style**: the **marker** — the AutoCAD point (by PDMODE and PDSIZE), a custom marker (dot, none, +, X or tick, with a circle and/or square around it, the same as PDMODE) or a **block** of the drawing —, the **size** (by the drawing scale, fixed or in units) and the layer and color of the marker and label.
- **Point label style**: the **text components** (number, elevation and description), with the **decimal places** and **decimal separator** (period or comma) of the elevation field, whether the component is **visible**, the **height** and **color** of each component's text, the component each one is **anchored** to, the **anchor point** on it (bottom left, middle right...) and the X and Y **offsets**, the **text style** and the **layer** of the label.

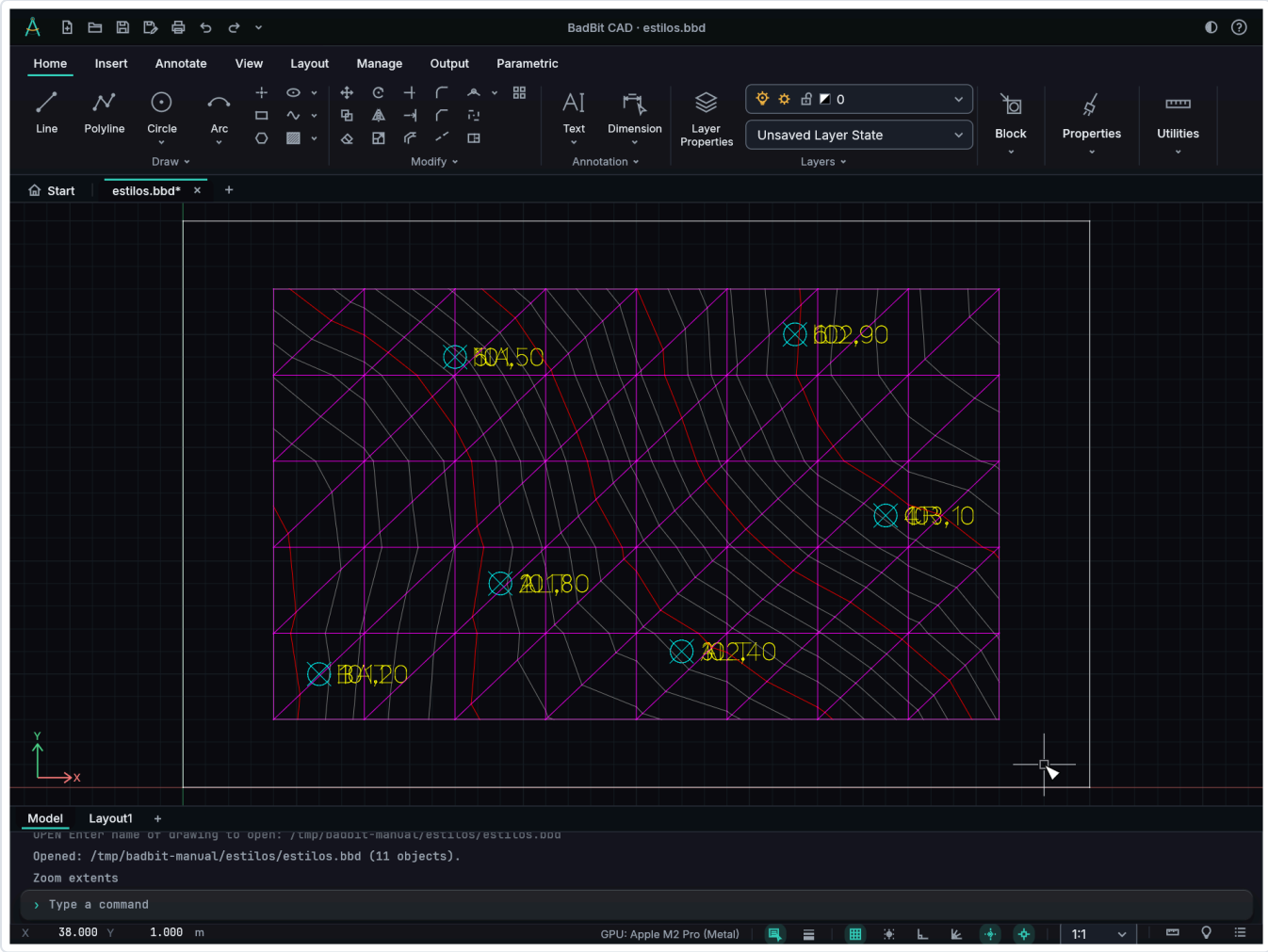
The size "by the drawing scale" (of the marker and the text, in millimeters on paper) uses the drawing's **annotation scale** (the one on the status bar, CANNOSCALE): 2 mm at 1:500 gives 1 m in the drawing. Without an annotation scale, 1:1000 applies.

Civil 3D points use the style of the **point group** (the first one, in display order, that has one) or the default of the drawing's point groups; points do not store their own style in this format.



Show triangles. In the Properties palette of a surface, the **Show triangles** row turns triangles on even when the style does not show them (Yes/No). It applies **only to the session**: it does not change the style, does not go into the DWG or the `.bbd`, and Undo (Ctrl+Z) reverts it. The triangles appear with the layer and color of the style's "triangles" component. When the style already shows the triangles, the row says "Yes (by style)".


Properties
Lineweight
ByLayer
GEOMETRY
Original type
AECC_TIN_SURFACE
Shown by
Surface read from the file, read-only
Surface name
TERRENO
Surface style
Curvas 0,5 e 2,5
Vertices
54
Triangles
80
Minimum elevation
99.566 m
Maximum elevation
112.329 m
Mean elevation
105.504 m
2D area
608.000 m²
3D area
659.481 m²
Minor interval
0.500 m
Major interval
2.500 m
Show triangles
No



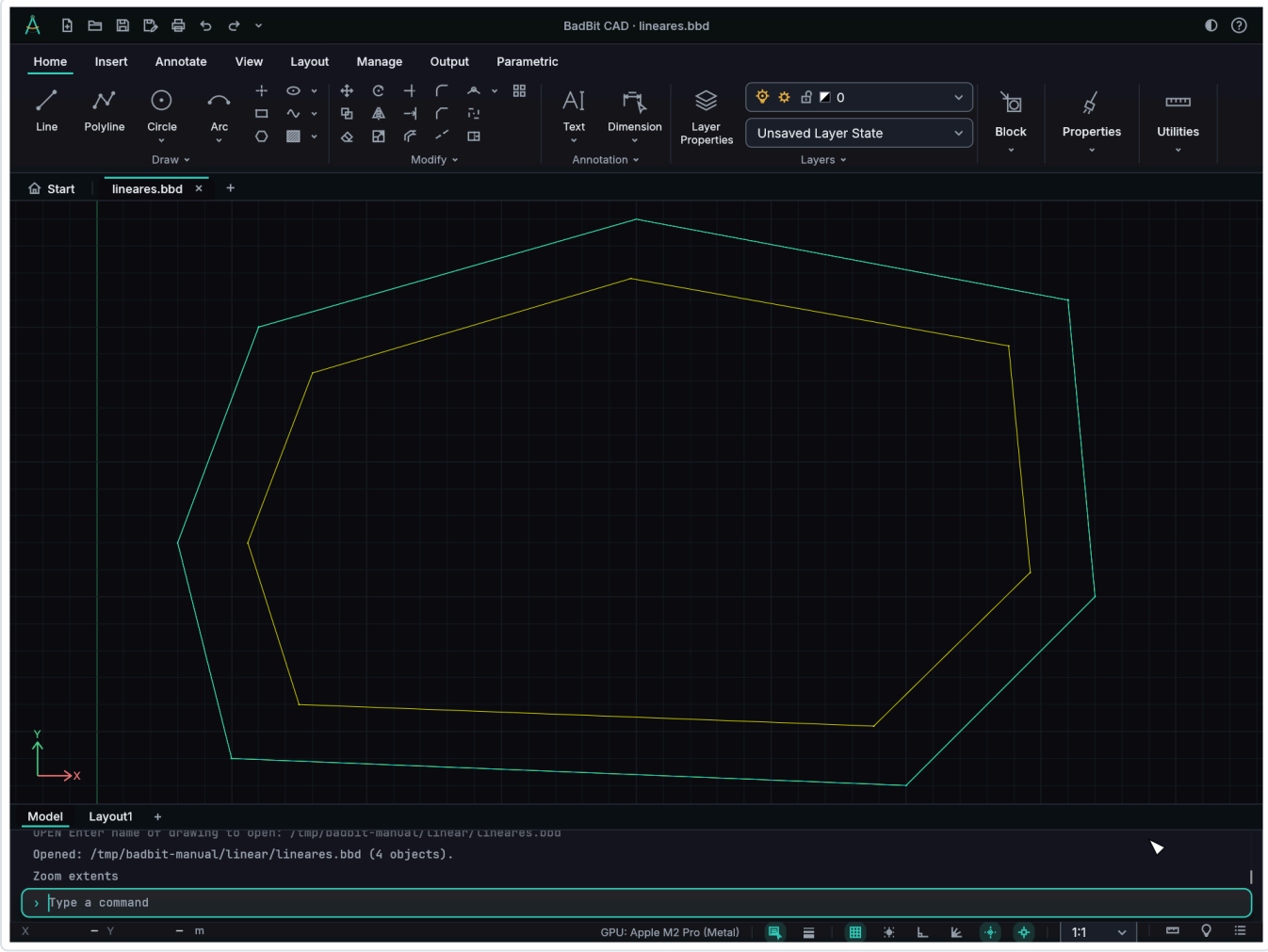
What is not read yet (the Civil 3D default applies; the Import report says which under "Show details", without changing the summary):

- the **attachment** of each label component (the corner of the text that sits at the anchor point): BadBit CAD assumes the one opposite the anchor point (attached below another component, the text hangs downward) and always on the left;
- the text **height** of a component that does not carry it (2.5 mm on paper applies);
- label fields other than number, elevation and description, and units other than meters in the field;
- the "relative to screen" marker size (used as drawing units).

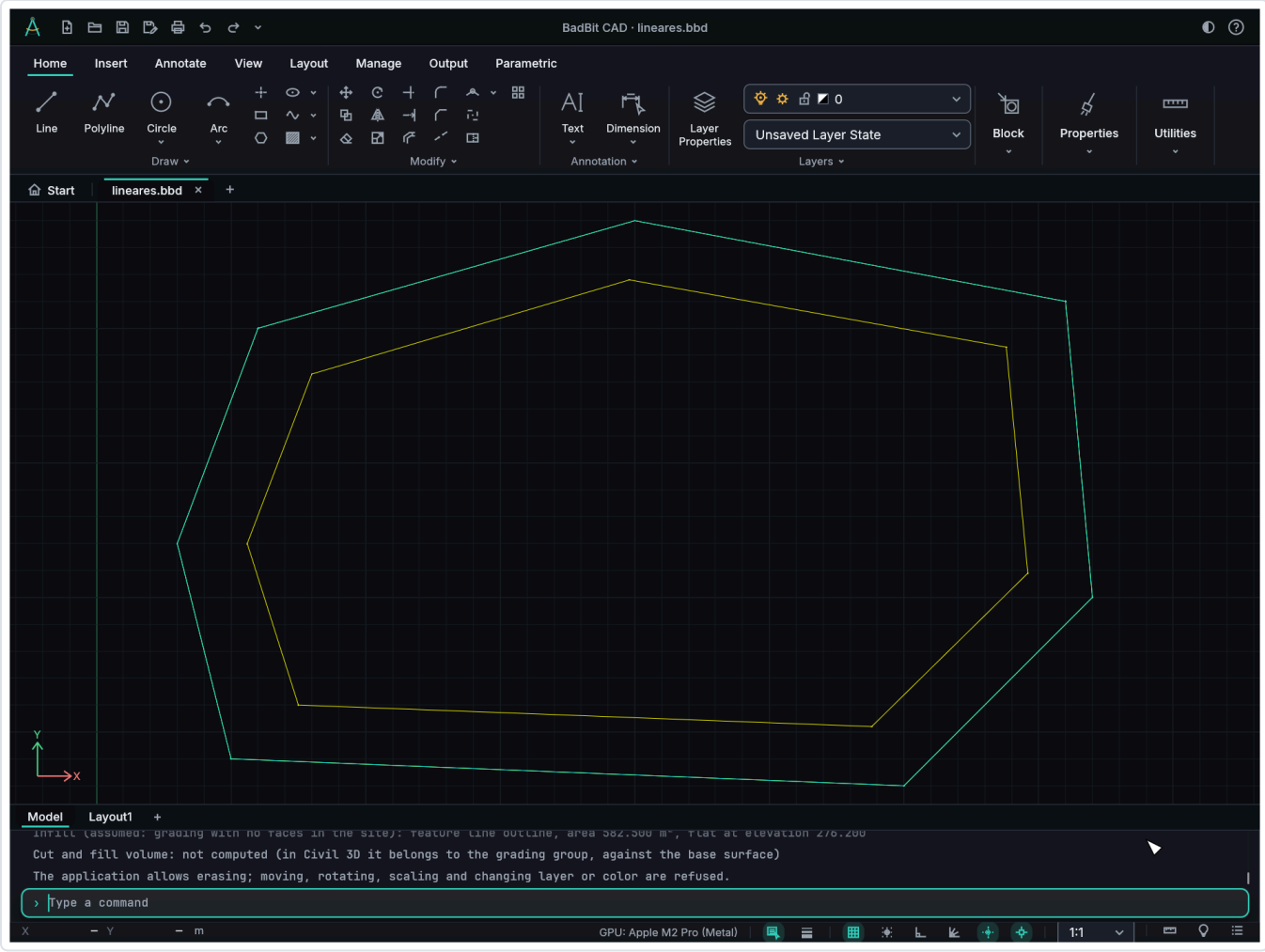
How BadBit CAD deciphered the styles (from the files alone) is described in `docs/formatos/civil3d-estilos.md`.

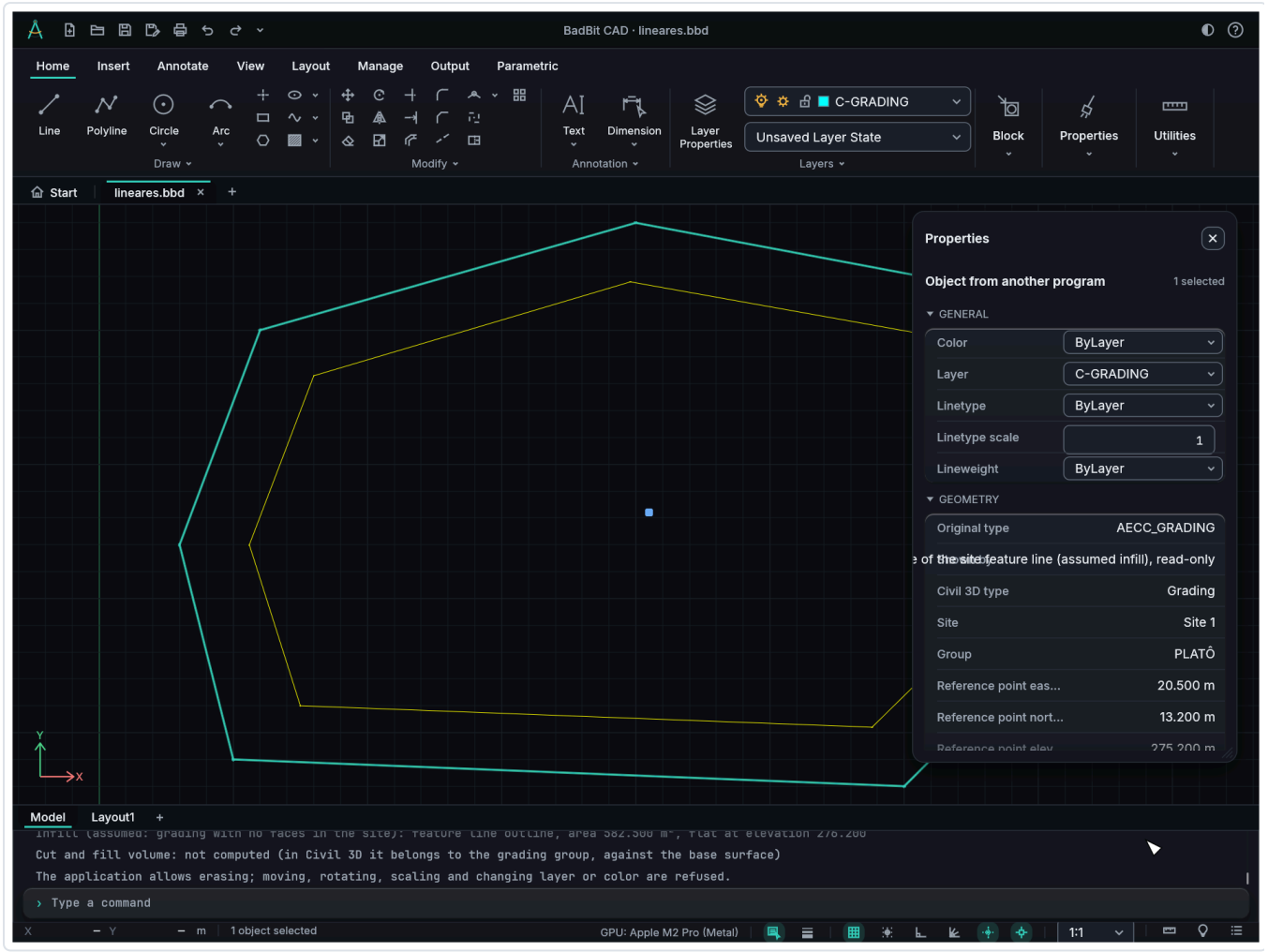
Civil 3D parcels, feature lines and alignments

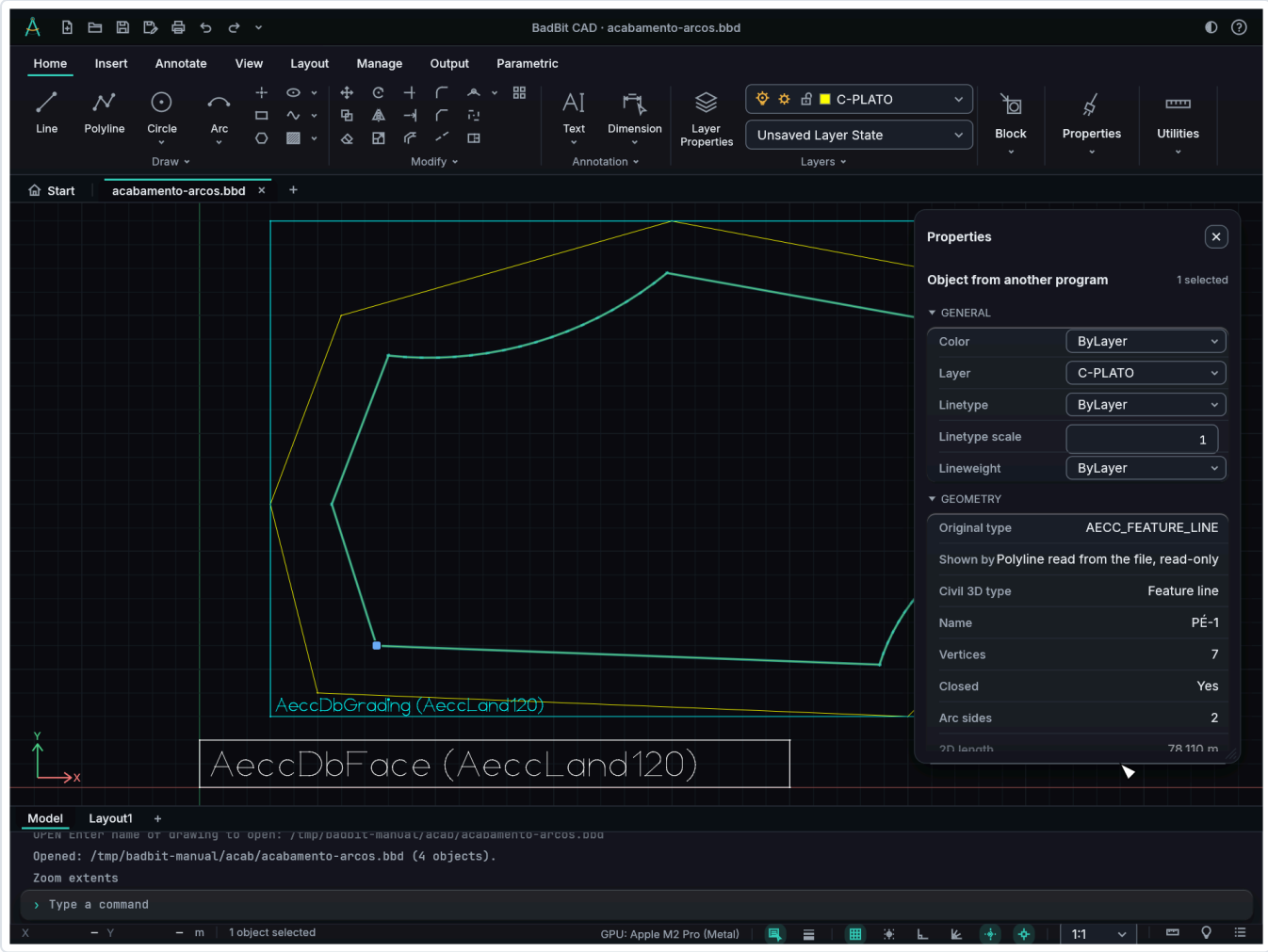
The **feature lines** of a Civil 3D DWG (2000 and later) are not limited to the proxy box ("AeccDbFeatureLine (AeccLand120)"): BadBit CAD reads from the file the **vertices** of each one, with the **elevation** of each vertex, the **arc sides**, the **name**, the **style** and the **site**, and shows it as the polyline it is in Civil 3D.



- **Drawing**: the closed polyline, with **arcs** on the arc sides, on the layer and in the color of the feature line (the Civil 3D feature line style is not followed yet).
- **Snap**: **Endpoint**, **Midpoint** and **Nearest** pick the vertices and sides.
- **LIST**: says "Civil 3D feature line (AECC_FEATURE_LINE), with the data read from the file, read-only", the name, style and site, how many vertices and whether it is closed, how many **arc sides** (when there are any), how many **PIs** and how many **elevation points** (when there are any: BadBit CAD calls an elevation point a vertex in the middle of a straight segment, with no change of direction; the file does not distinguish the two), the 2D and 3D lengths (arcs by their arc length), the minimum and maximum elevations and the plan area (with the arcs).
- **Properties** (Ctrl+1): the rows Civil 3D type, Name, Style, Site, Vertices, Closed, Arc sides, PI points and Elevation points (assumed), 2D length, 3D length, Minimum elevation, Maximum elevation and Area, read-only; "Shown by" says "Polyline read from the file, read-only".
- The feature line remains **read-only** (it can be erased) and goes back to the DWG with the same bytes.
- Each feature line is linked to its vertices by the site list (the same order in which the site stores the segments), no longer by the proxy box; the box only applies when the list does not match.


BadBit CAD · lineares.bbd
Home
Insert
Annotate
View
Layout
Manage
Output
Parametric
Line
Polyline
Circle
Arc
Draw
Modify
Text
Dimension
Annotation
Layer Properties
0
Unsaved Layer State
Layers
Block
Properties
Utilities
Start
lineares.bbd
Model
Layout1
Cut and fill volume: not computed (in Civil 3D it belongs to the grading group, against the base surface)
The application allows erasing; moving, rotating, scaling and changing layer or color are refused.
Type a command
GPU: Apple M2 Pro (Metal)
1:1


BadBit CAD · lineares.bbd
Home
Insert
Annotate
View
Layout
Manage
Output
Parametric
Line
Polyline
Circle
Arc
Draw
Modify
Text
Dimension
Annotation
Layer Properties
C-GRADING
Unsaved Layer State
Layers
Block
Properties
Utilities
Start
lineares.bbd
Properties
Object from another program
1 selected
GENERAL
Color
ByLayer
Layer
C-GRADING
Linetype
ByLayer
Linetype scale
1
Lineweight
ByLayer
GEOMETRY
Original type
AECC_GRADING
of the site feature line (assumed infill), read-only
Civil 3D type
Grading
Site
Site 1
Group
PLATÔ
Reference point eas...
20.500 m
Reference point nort...
13.200 m
Model
Layout1
Cut and fill volume: not computed (in Civil 3D it belongs to the grading group, against the base surface)
The application allows erasing; moving, rotating, scaling and changing layer or color are refused.
Type a command
1 object selected
GPU: Apple M2 Pro (Metal)
1:1

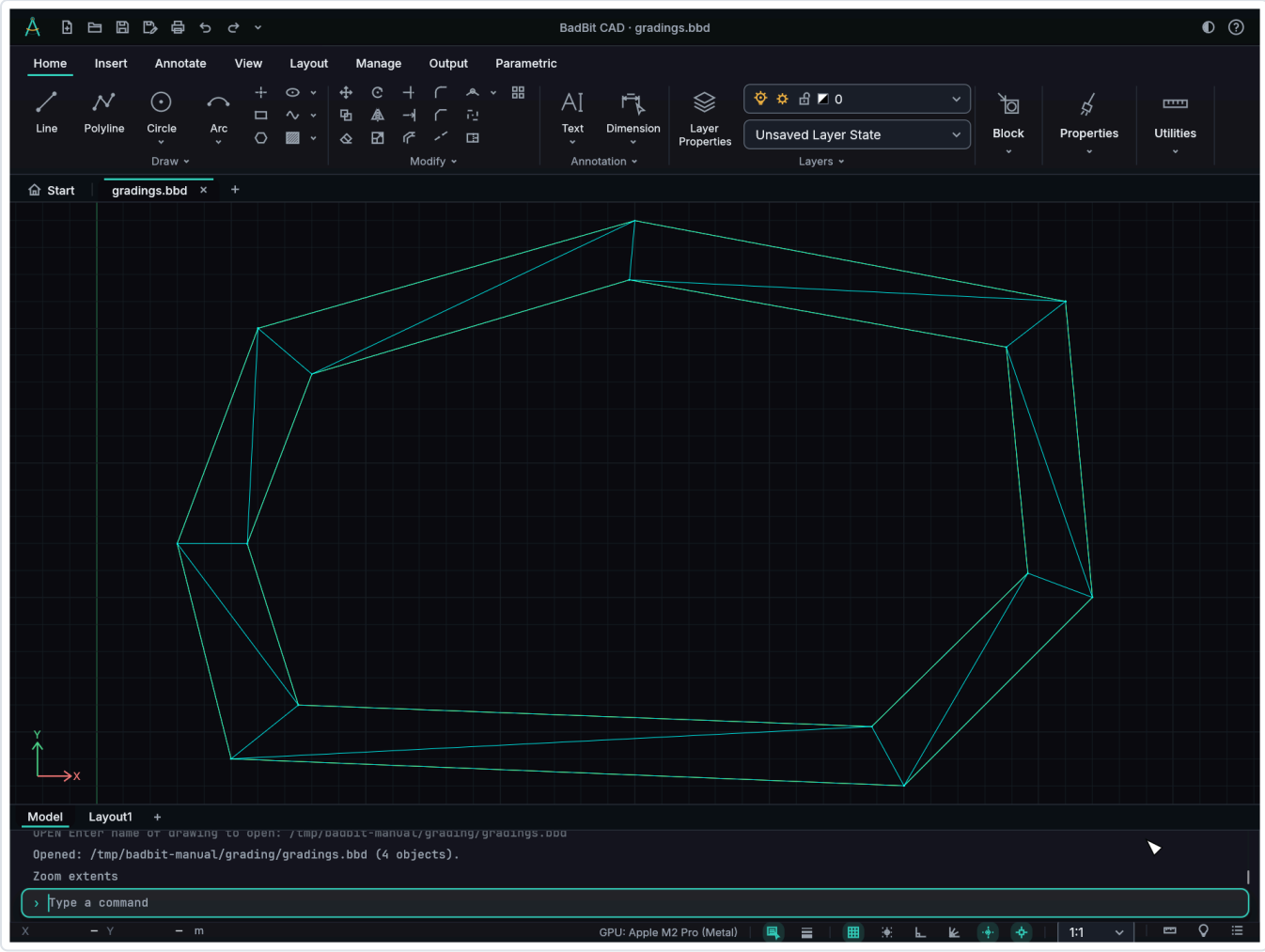


The other Civil 3D linear objects:

- **Parcels** (AECC_PARCEL): each Civil 3D **site** has an "outer face" saved as a parcel, with no outline, which Civil 3D does not draw. BadBit CAD recognizes it by the site name ("Site 1") and does not draw it either (before, the proxy's template box appeared at the drawing origin). LIST and Properties give the site name and the parcel style. Parcels with an outline are not read yet: they appear through the proxy graphics.
- **Gradings** (AECC_GRADING): see just below.
- **Alignments**: station labels (major and minor) and geometry point labels (PC, PT, TS...) of an alignment that **is not in the file** (the label group was left without an alignment) do not appear, as in Civil 3D; LIST gives the label style. The geometry of the alignment itself (tangents, curves, spirals, starting station) is not read yet: an alignment in the file appears through the proxy graphics.

Civil 3D gradings

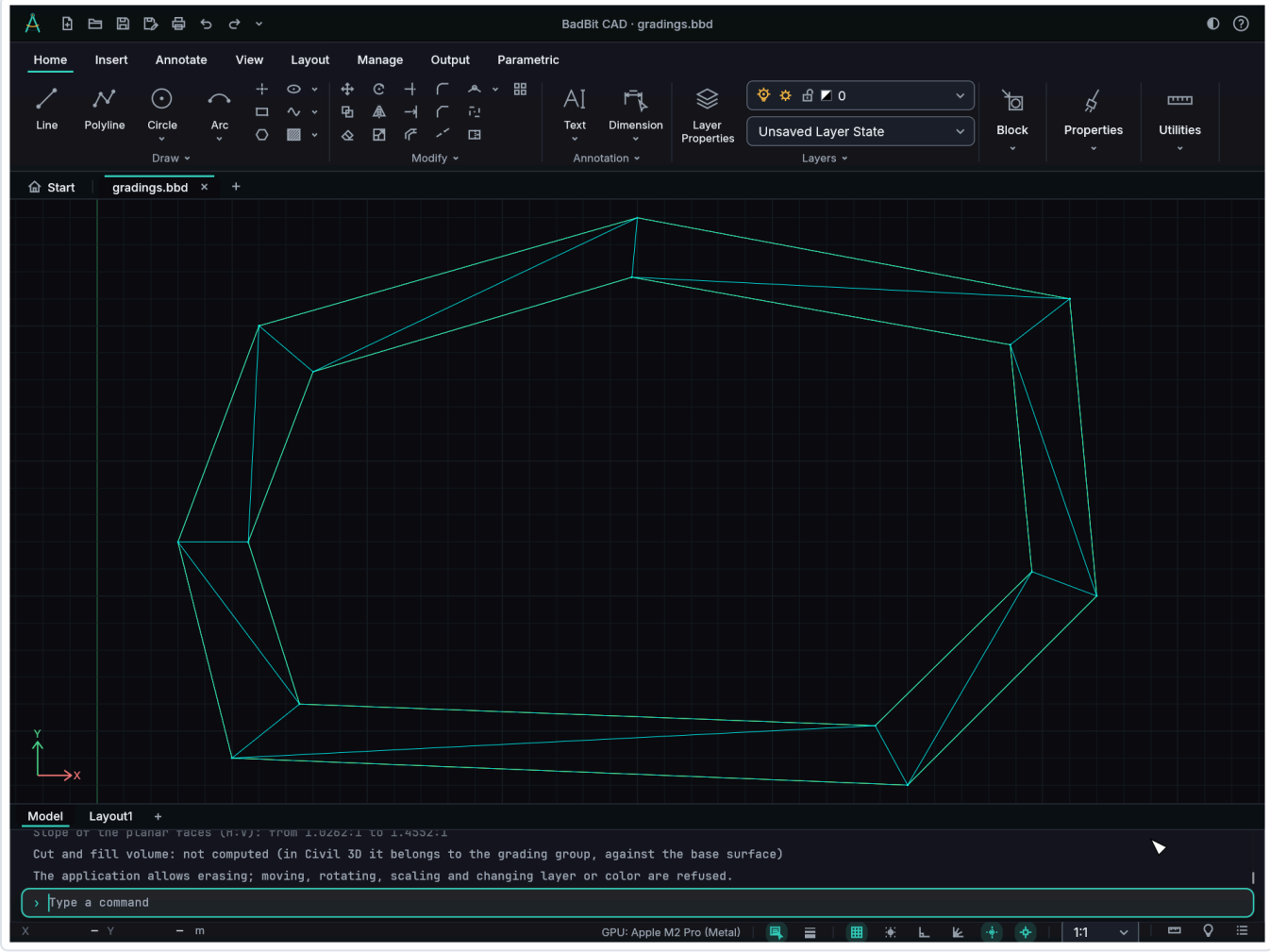
Civil 3D keeps the **face model** of each grading (the slopes and the platforms, as a sheet solid) inside the site. BadBit CAD reads that model and draws the grading **as its lines**, in place of the proxy box ("AeccDbGrading (AeccLand120)"):

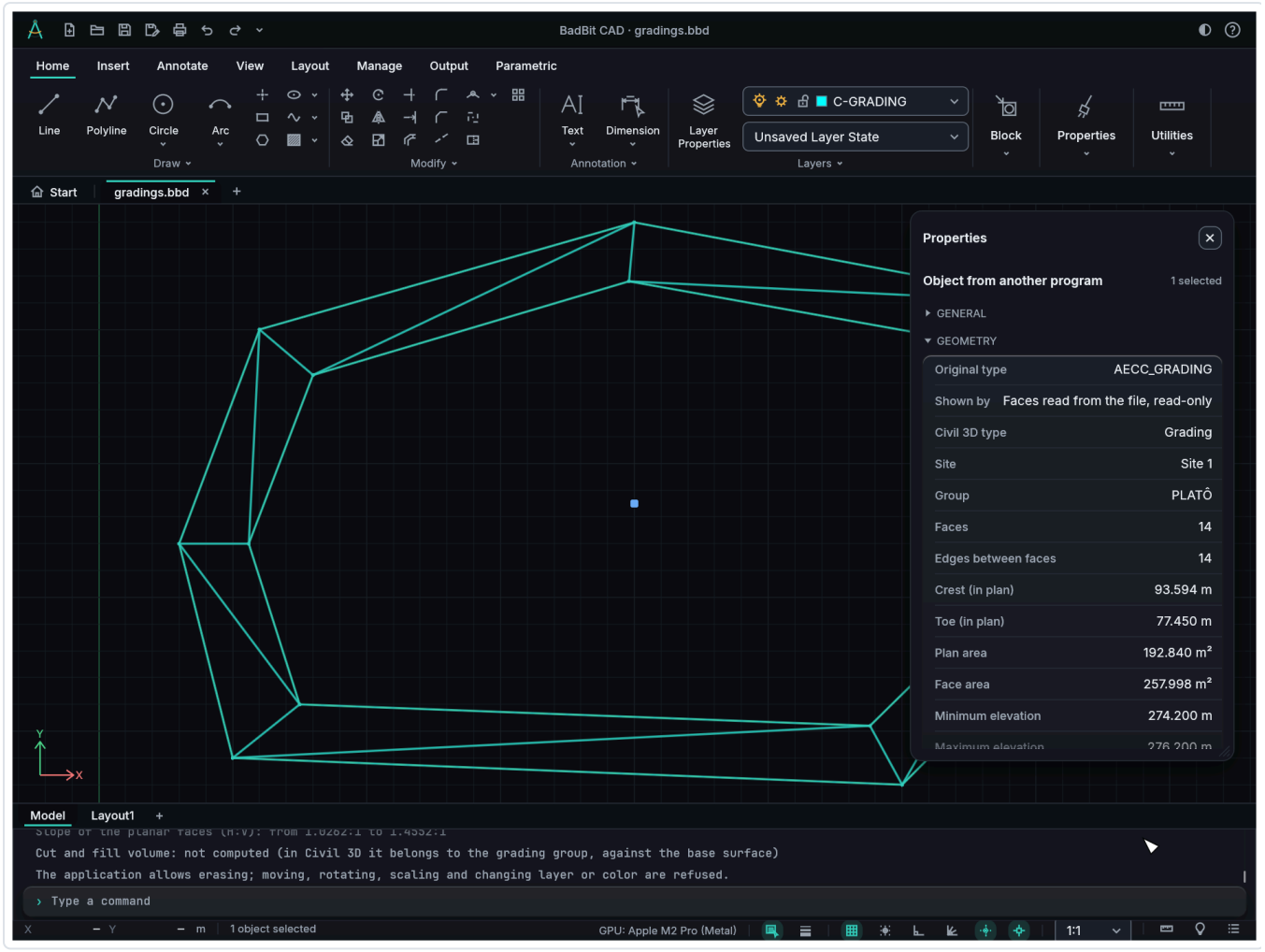


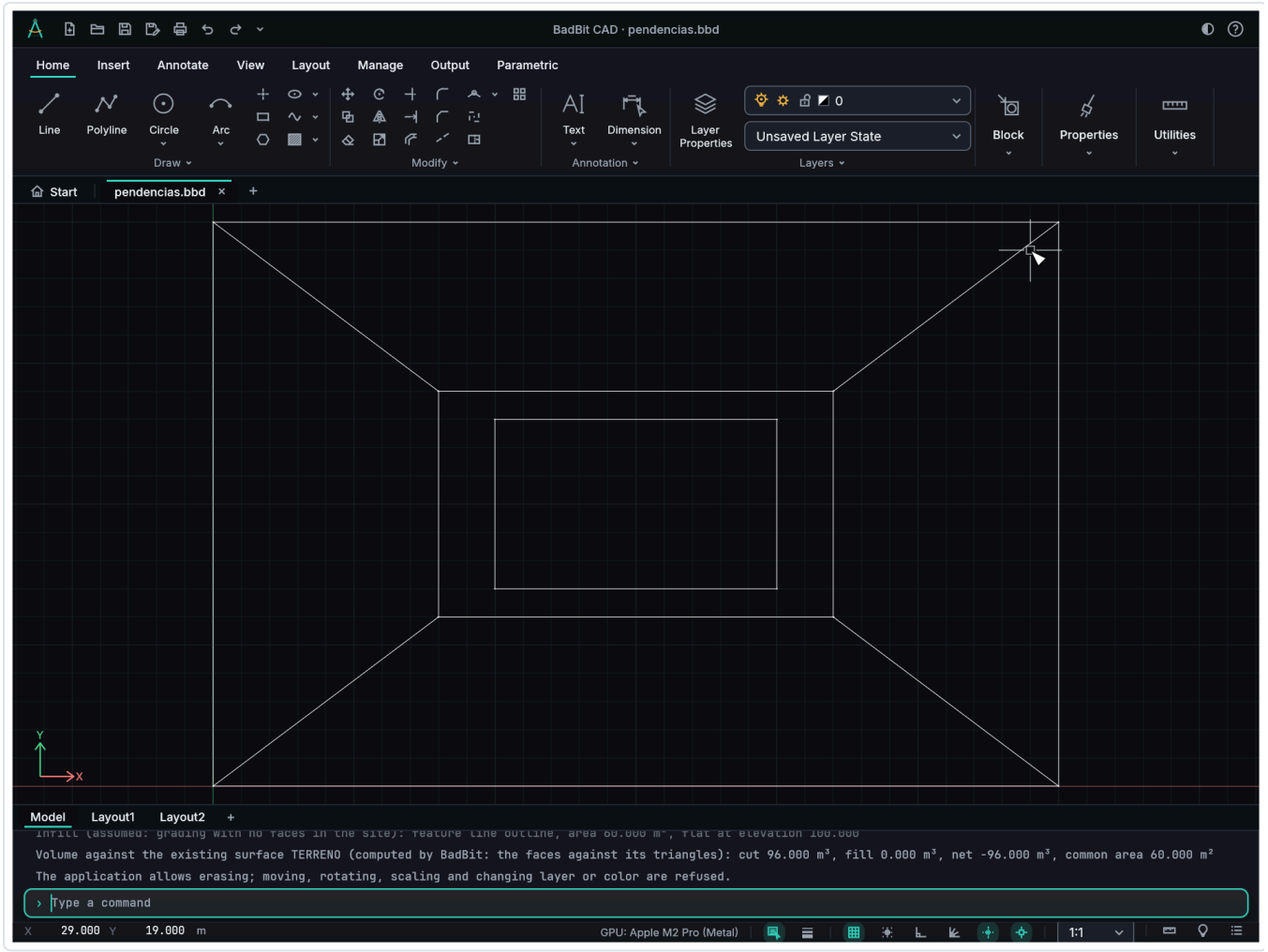
- **Drawing**: the edges of the faces in plan, each one only once: the **crest** (the highest border), the **toe** (the lowest border) and the **lines between faces** (the projection lines and the face diagonals), with the rounded corners as arcs, on the layer and in the color of the grading. The Civil 3D grading style is not followed yet (slope patterns, for example, do not appear).
- **LIST**: says that the lines come from the face model kept in the site, the site, the grading group, the reference point, how many **faces** and **edges between faces**, the plan length of the **crest** and the **toe**, the **plan area** and the **face area** (sloped), the minimum and maximum elevations, the **slope** of the planar faces (horizontal to vertical, "4:1") and the **cut and fill volume** against the existing surface (below).
- **Properties** (Ctrl+1): the rows Faces, Edges between faces, Crest (in plan), Toe (in plan), Plan area, Face area, Minimum elevation, Maximum elevation, Slope (H:V), Existing surface (volume), Cut volume, Fill volume and Net volume, read-only; "Shown by" says "Faces read from the file, read-only".
- The grading remains **read-only** (it can be erased) and goes back to the DWG with the same bytes.
- When Civil 3D saved the proxy "without graphics" (the small box at the drawing origin), the site's face model is enough on its own: the grading appears through its lines.
- A grading **with no model in the site** is treated as an **infill** (assumed: it fills the closed area without projecting anything) and appears through the **outline** of the smallest closed feature line of the same site that contains the grading's **reference point** (assumed: the point clicked inside the area when creating the infill), checked against the proxy box; without the point, the feature

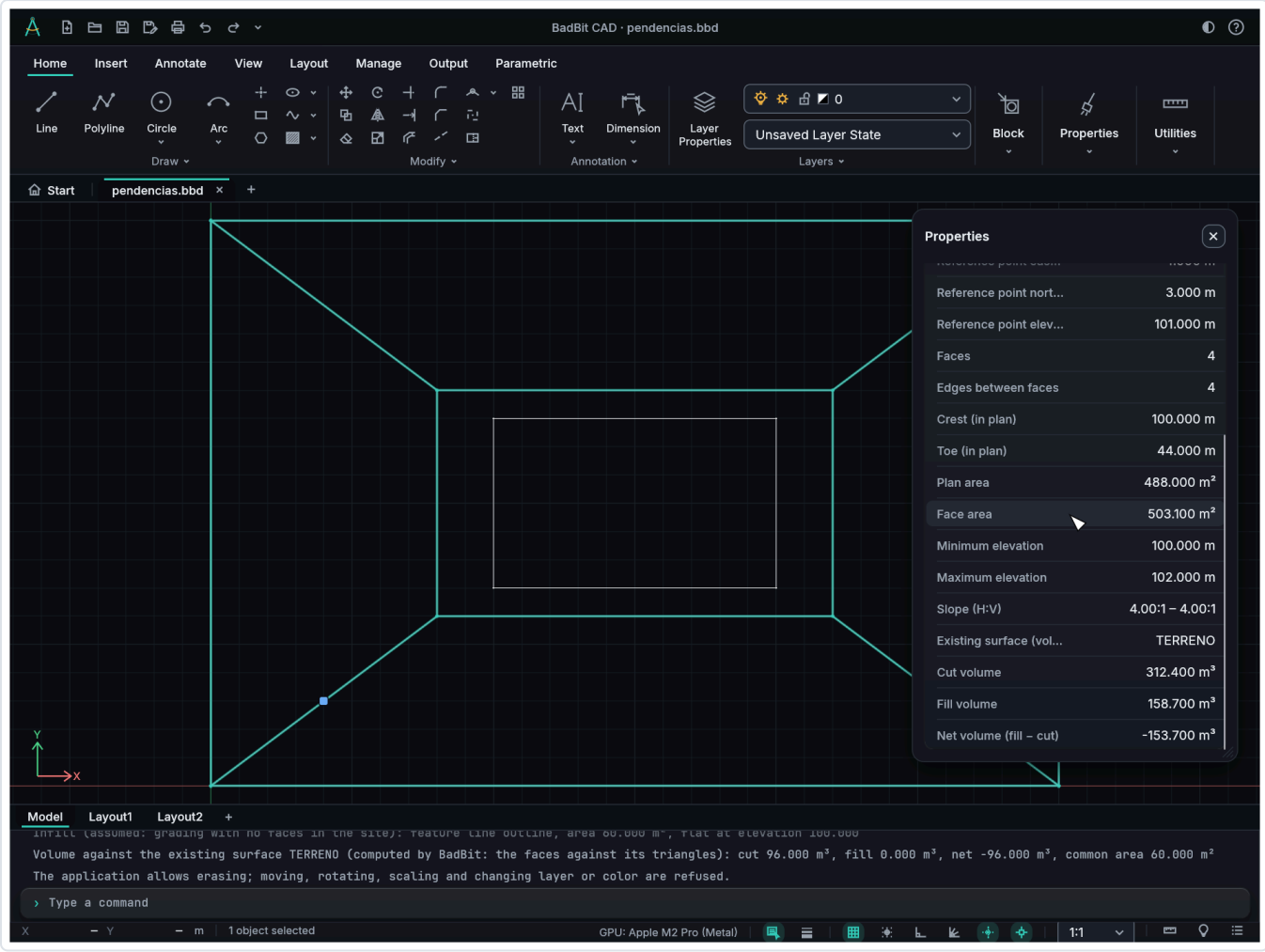
line with the proxy box. It is flat at that feature line's elevation. LIST says "Infill (assumed...)", the area and the elevation; Properties have the Infill (assumed) row and "Shown by" says "Outline of the site feature line (assumed infill), read-only". Without that feature line, the grading stays as its **proxy graphics**, with the site, the group and the reference point.

- **Cut and fill volume**: the computation is BadBit CAD's. The faces of the grading (or the outline of the infill, flat) are intersected with the triangles of the **existing surface**, and in each piece the elevation difference gives the exact volume: **cut** where the face is below the surface, **fill** where it is above, the **net** (fill minus cut) and the **common area** (the part of the face outside the surface does not count). The existing surface is the last one the grading group lists (in a typical project: the group surface and then the existing ground); a grading outside a group uses the one of the only group in the same site; and a grading in a site **with no group** uses the **only existing surface that covers the whole grading** in plan (assumed: Civil 3D only computes grading volume within a group; volume and group surfaces do not count). With no existing surface, LIST says the volume was not computed. It is a quantity: it does not say whether the design meets any criterion.
- **Computation trace**: LIST shows, right below the volume, how many face **triangles** were used against how many of the existing surface, how many **common pieces** in plan, and the tessellation **chord** (the largest deviation between the curved faces and the mesh triangles). The chord is **zero** when the computation does not use the mesh, which is the normal case: planar faces with straight sides go in through their own triangles (exact); curved faces (cones, NURBS and the others) and faces with curved edges go in through the model's **exact geometry** (the exact elevation of the face integrated in strips against each triangle of the existing surface, split where the elevation difference becomes zero, including where the face only touches the base, with a requested relative error of 10^{-12}); and in an infill with arcs each arc of the outline goes in as the exact circular segment. Only a face whose exact computation does not close (for example, a face that folds over itself in plan) goes in through the mesh, and then the chord appears, along with **how the base was chosen** and the **method** (prisms between the overlapping triangles, exact integral of the elevation difference, cut and fill separated along the line where the difference becomes zero). Properties have the rows Base chosen by, Triangles (faces / base), Common pieces (volume) and Tessellation chord (volume).
- **Grading group total**: for a grading in a group, LIST and Properties also show the **group total** — the **group surface** against the existing surface, as Civil 3D computes the group volume —, with cut, fill, net and common area (Properties: Group surface (volume), Group cut, Group fill, Group net). When the drawing has a Civil 3D **volume surface** that compares the same two surfaces, its name appears (Properties: Civil 3D volume surface (same surfaces)): its "Cut/Fill Summary" table is the one to check against. In the test project, the group total gives 10,666.00 m³ of cut and 1,172.34 m³ of fill, the same numbers as the "VOL ESCAVAÇÃO" table that Civil 3D saved in the drawing.
- The computation runs in parallel, while the file is being opened (off screen: BadBit CAD keeps responding).


BadBit CAD · gradings.bbd
Home
Insert
Annotate
View
Layout
Manage
Output
Parametric
Line
Polyline
Circle
Arc
Draw
Modify
Text
Dimension
Annotation
Layer Properties
0
Unsaved Layer State
Layers
Block
Properties
Utilities
Start
gradings.bbd
Model
Layout1
Cut and fill volume: not computed (in Civil 3D it belongs to the grading group, against the base surface)
The application allows erasing; moving, rotating, scaling and changing layer or color are refused.
Type a command
GPU: Apple M2 Pro (Metal)
1:1


BadBit CAD · gradings.bbd
Home
Insert
Annotate
View
Layout
Manage
Output
Parametric
Line
Polyline
Circle
Arc
Draw
Modify
Text
Dimension
Annotation
Layer Properties
C-GRADING
Unsaved Layer State
Layers
Block
Properties
Utilities
Start
gradings.bbd
Properties
Object from another program
1 selected
GENERAL
GEOMETRY
Original type
AECC_GRADING
Shown by
Faces read from the file, read-only
Civil 3D type
Grading
Site
Site 1
Group
PLATÔ
Faces
14
Edges between faces
14
Crest (in plan)
93.594 m
Toe (in plan)
77.450 m
Plan area
192.840 m²
Face area
257.998 m²
Minimum elevation
274.200 m
Maximum elevation
276.200 m
Model
Layout1
Cut and fill volume: not computed (in Civil 3D it belongs to the grading group, against the base surface)
The application allows erasing; moving, rotating, scaling and changing layer or color are refused.
Type a command
1 object selected
GPU: Apple M2 Pro (Metal)
1:1


BadBit CAD · pendencias.bbd
Home
Insert
Annotate
View
Layout
Manage
Output
Parametric
Line
Polyline
Circle
Arc
Draw
Modify
Text
Dimension
Annotation
Layer Properties
0
Unsaved Layer State
Layers
Block
Properties
Utilities
Start
pendencias.bbd
Model
Layout1
Layout2
Volume against the existing surface TERRENO (computed by BadBit: the faces against its triangles): cut 96.000 m³, fill 0.000 m³, net -96.000 m³, common area 60.000 m²
The application allows erasing; moving, rotating, scaling and changing layer or color are refused.
Type a command
29.000
19.000
m
GPU: Apple M2 Pro (Metal)
1:1



In the Import report, the feature lines read count as "Civil 3D feature lines read from the file", the gradings drawn from the model as "Civil 3D gradings drawn from the face model", the infills as "Civil 3D gradings with no faces in the site shown as infill" and the objects Civil 3D does not draw as "Civil 3D objects that Civil 3D does not draw", among those kept intact; only what remains as proxy graphics (gradings with no model) counts as "shown only as outlines". If an object cannot be read (a different format), it appears through the proxy graphics, as before. Parcels, gradings and alignment labels are also read from a DWG 2000 or 2004 (their texts, which in these versions come in the middle of the data, are looked for there).

Saving a Civil 3D project in another DWG version

A Civil 3D project opened from a DWG 2007 or newer can be saved in another DWG version — **2013, 2010, 2007, 2004 or 2000** (**SAVEAS**, `.dwg` extension, the version) — **with the Civil 3D objects**: COGO points, surfaces, feature lines, gradings, parcels, labels, styles and settings go along, and BadBit CAD reads them back the same (same points, same coordinates and elevations, same triangles).

In DWG 2013, 2010 and 2007 the Civil 3D and AutoCAD Map objects go **as proxies**, with the 2018 data **whole**, as AutoCAD without Civil 3D writes them (checked against the same drawing saved by AutoCAD in each version: the same data, the same texts and the same links); the architecture objects

AutoCAD knows (the styles and display properties `AEC_*`) go in their own format, with the same data (the save report counts these as "written with the same data"). BadBit CAD, like Civil 3D, takes the objects out of these proxies when opening.

- Civil 3D's **associative actions** (for example, offset alignments that follow the main alignment) are left out in another version, and AutoCAD's **associative network** goes without them, as AutoCAD writes it; in the same version (2018) everything goes back as it came.

Project saved in DWG 2000 or 2004. A Civil 3D project opened from a DWG 2000 or 2004 can also be saved in any other DWG version (2000 to 2018) and in DXF: the Civil 3D and AutoCAD Map objects go **as proxies, with the source version's data whole**, as AutoCAD without Civil 3D writes them (checked against the same drawing saved by AutoCAD in each version: the same counts, the same data), and the architecture objects AutoCAD knows go in the chosen version's own format. Civil 3D (and BadBit CAD) take the objects out of these proxies when opening; saved again in the source version, the drawing comes back with them as they were. Before, a project saved in DWG 2000 lost all Civil 3D objects in any other version, and one in 2004 lost part of them (and all of them in DXF). **AutoCAD Map** objects that come in a **DXF** (the Map display manager) also go to the other DXF versions and to DWG as proxies, with the data table and the geographic location, as AutoCAD writes them.

In DWG 2004 and 2000 (from a 2007 or newer project) the conversion is larger.

In these versions the DWG keeps the texts of each object **in the middle of the data** (from 2007 on they are kept separately), in a place that changes from class to class. BadBit CAD writes each object in one of these ways (the save report shows them as "template", "no texts" and "proxy"):

- **template**: the object is identical to one AutoCAD wrote in DWG 2004 (compared with reference files): the texts go in the place AutoCAD used (mostly the settings and styles that come from the Civil 3D template);
- **no texts**: the object has no texts and goes as it is (COGO points, for example, in 2004);
- **proxy**: all the others go **wrapped in a proxy**, with the Civil 3D data **whole, as it was**. This is exactly what **AutoCAD without Civil 3D** writes when it saves a Civil 3D project in DWG 2004 or 2000 (checked with AutoCAD itself): AutoCAD opens the file without errors, and Civil 3D, when opening, takes each object out of the proxy, as it does with any drawing that went through plain AutoCAD.

BadBit CAD also reads these proxies: a Civil 3D project saved in DWG 2004 or 2000 by plain AutoCAD opens with the points, surfaces and other Civil 3D objects, like the original.

AutoCAD 2027 opens the 2004 and 2000 files written this way without AUDIT errors. Reading by **Civil 3D** has **not been checked** yet: open the saved file in Civil 3D to confirm before delivering it to someone.

- In **DWG 2004** and **DWG 2000** all Civil 3D objects go, except the associative actions, which depend on AutoCAD 2010's associative network (which 2004 and 2000 do not have).
- In **DWG 2000** the colors inside Civil 3D objects have another format: only the color number (ByLayer, ByBlock or the color from 1 to 255), without the RGB. In **template** objects, BadBit CAD converts the colors by the **rule of each class**, taken from the reference drawings saved by

AutoCAD in 2004 and in 2000; the others go as proxies (without conversion: the proxy carries the 2007-and-later data). The report counts, by class, the objects "with their colors converted to DWG 2000".

- The architecture objects AutoCAD writes in the **DWG 2004** native format (a Civil 3D project saved in 2004 by plain AutoCAD has hundreds of them) return to the 2007-and-later format when that 2004 file is saved in 2007, 2010, 2013 or 2018, and go to 2000, as AutoCAD does (the texts taken from the middle of the data by the same templates). The settings object (`AEC_VARS_DWG_SETUP`) also goes back from AutoCAD's 2004 to 2007 and later.
- The settings object (`AEC_VARS_DWG_SETUP`) also goes in its **native format** from 2018 (or 2007 to 2013) to 2004 and 2000, as AutoCAD writes it; before, it went as a proxy. It ends with a stamp of the program that wrote it (two numbers: AutoCAD 2027 always writes 9 and 0, Civil 3D writes its own): BadBit CAD keeps the one from the source file; the rest comes out bit for bit like AutoCAD's.
- The display properties of structural members (`AECS_DISP_PROPS_MEMBER_PLAN`, `_PLAN_SKETCH` and `_PROJECTED`) written by an older Civil 3D are in an **old form**, which AutoCAD 2027 updates when opening and writes in the new form in all versions. BadBit CAD does the same when saving in another version (2000 to 2018): they come out bit for bit like AutoCAD's, and in 2000 they go in their native format, with the colors converted (before, they went as proxies). In the same source version they go back as they came.
- In **DWG 2000** every proxy object carries, as in AutoCAD, the `ACAD` extended data with the DXF name of the class (`AcRTDxfName`).
- Section and detail view styles, visual styles, materials, render settings, the sun and the data table go converted.
- The **save report** says, by class, how many Civil 3D objects were written and in which way ("template", "no texts" or "proxy"), how many had their colors converted in 2000, and how many were left out and why. An object that could not be written safely is never written halfway: it is left out and goes into the report.
- Saving again in DWG 2004 the DWG 2004 written by BadBit CAD returns the Civil 3D objects with the same bytes.
- A Civil 3D project saved in DWG 2004 or 2000 by **plain AutoCAD** (with the objects as proxies) also goes to DWG 2018, 2013, 2010 and 2007 and back to 2000 with the Civil 3D objects (their data is the 2018 data, which the proxy keeps).

Proxies between DXF and DWG and between versions

Objects that AutoCAD without the owner application writes as **proxies** (ACAD_PROXY_OBJECT and ACAD_PROXY_ENTITY; in a DXF saved from a Civil 3D project these are all the Civil 3D objects) go **whole** to any DXF and DWG version from 2000 on, from a DXF or from a DWG: the same objects, the list of classes they depend on, the data and the texts in the form of each version (in 2000 and 2004 the texts are inside the data, as AutoCAD writes them) and the entities by their record, without exploding into the graphics. The ownerless proxies Civil 3D writes (label styles) also go. **Checked with AutoCAD 2027**: the Civil 3D project saved by it in DXF 2010 and written by BadBit CAD in DWG and DXF 2000 to 2018 opens without AUDIT errors, with the same proxies as the file AutoCAD itself writes from the same DXF.

When going to another version with the proxies, AutoCAD objects whose form changes from version to version go converted (detail and section view styles, visual styles, materials).

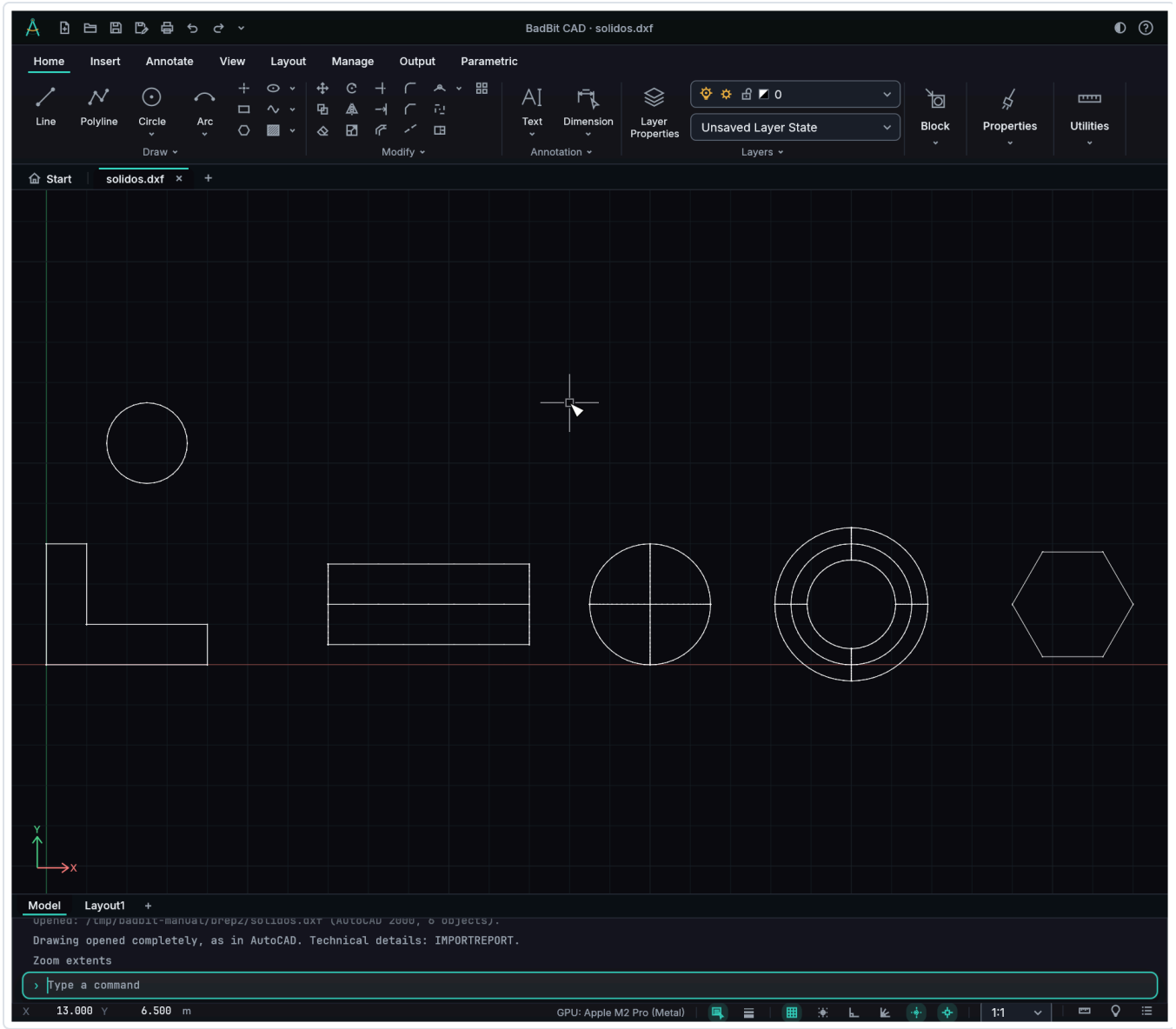
An entity on a **layer that is not in the table** of the DXF gets that layer (color 7, Continuous), as AutoCAD does when opening the file, and goes back through the stored record. An **arc or circle on a vertical plane** (perpendicular to the XY plane) appears in plan as the segment it is when seen from above and goes back unchanged when saving in the same version.

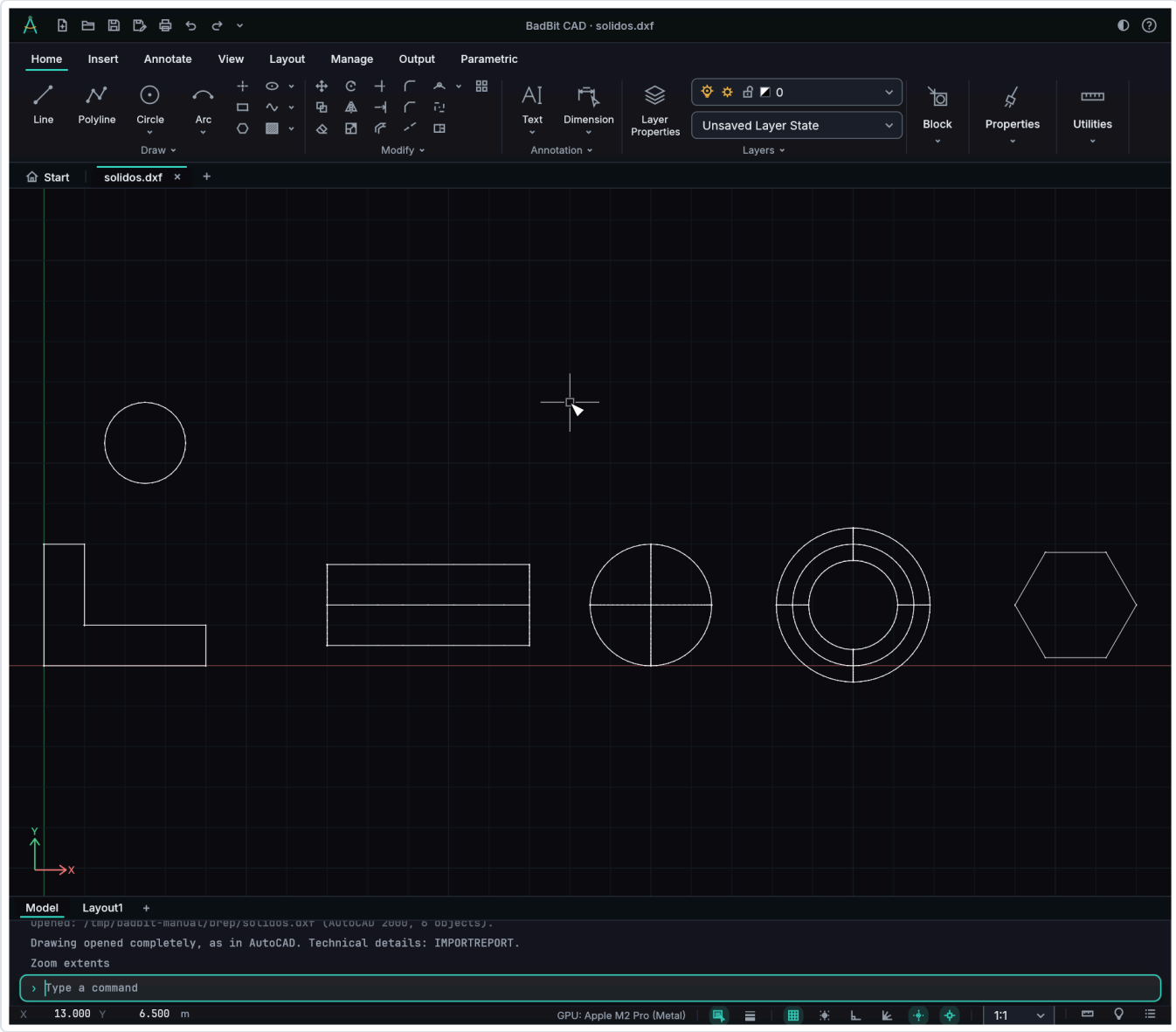
3D solids (3DSOLID, REGION, BODY) and surfaces

AutoCAD **3D solids** (3DSOLID), **regions** (REGION), **bodies** (BODY) and **surfaces** (SURFACE, PLANESURFACE, EXTRUDEDSURFACE, LOFTEDSURFACE, REVOLVEDSURFACE, SWEPTSURFACE and NURBSURFACE) are read by BadBit CAD's own B-rep kernel: the exact geometry of each face (planes, cylinders, cones, spheres, tori, NURBS surfaces and the **procedural** surfaces — offset, revolved, extruded, sum — evaluated from their definition even when the file does not carry the approximation) and of each edge comes from the ACIS data the file carries, in any DXF and DWG version (text SAT from 2000 to 2010 and binary SAB in the newer versions).

Checked with AutoCAD 2027. Solids made in AutoCAD itself (box, sphere, cylinder, cone, torus, wedge, pyramid, polysolid; extrusions of a polyline with arcs and of a spline, with and without taper; partial and full revolutions; sweep and loft; union, subtraction and intersection; edge fillet and chamfer; slice; rotated, mirrored and scaled solids), regions (from a circle, from a polyline with an arc and with a hole) and surfaces (planar, extruded and revolved), saved in **DWG 2000, 2004, 2007, 2010, 2013 and 2018** and in **DXF 2000 and 2018**, open in BadBit CAD in the same place and with the same bounding box and the same centroid that AutoCAD's MASSPROP gives; the volume (or the area and perimeter) matches to the decimal places AutoCAD prints, with a difference of a few ten-thousandths on some curved faces. In a DWG 2013 or newer, the solid data sits in a separate section of the file, and BadBit CAD reads it from there.

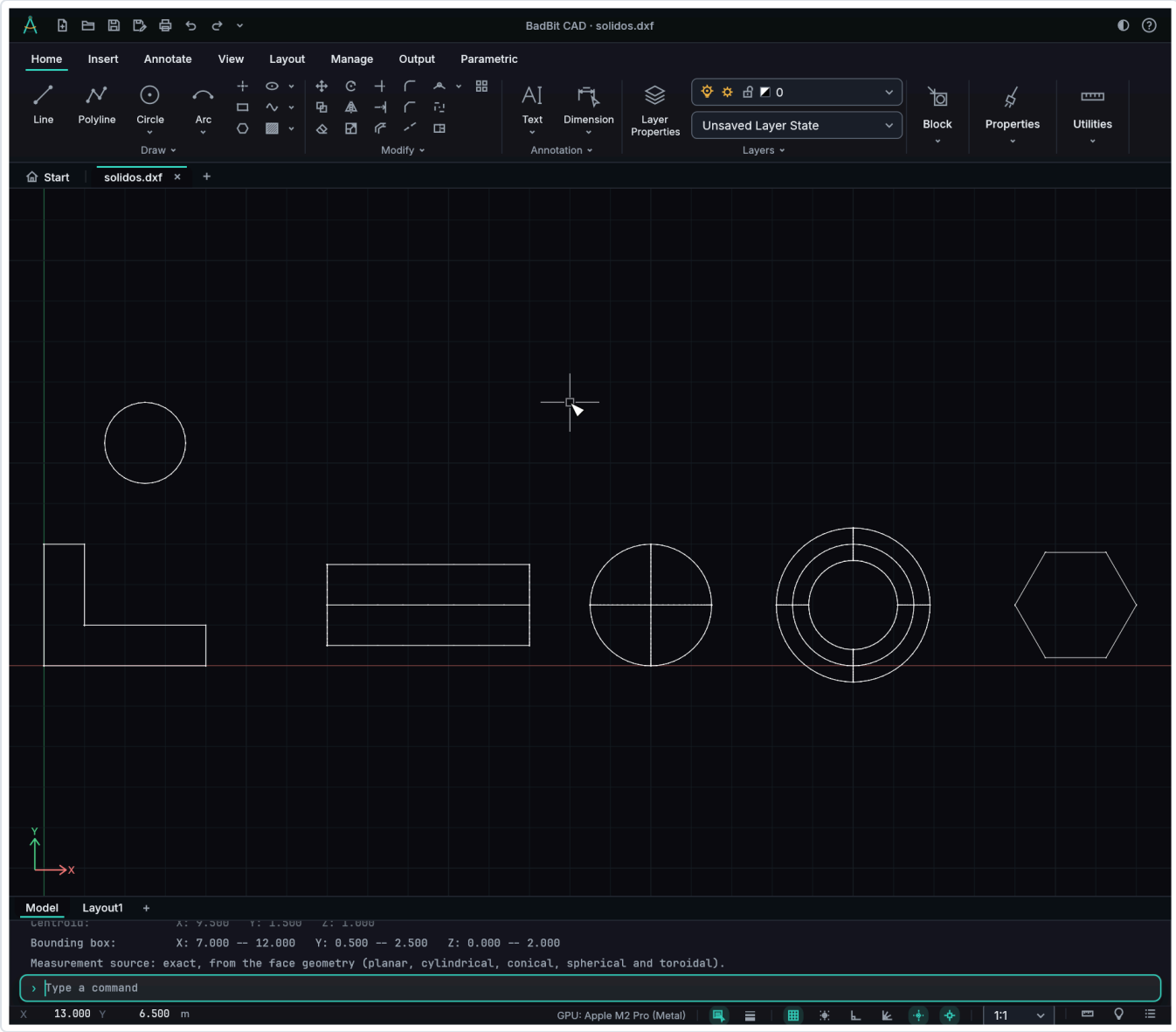
On screen, each solid appears **as AutoCAD shows it in top view in 2D wireframe**: the **projected edges** (exact lines, arcs, circles and ellipses; NURBS curves also exact), the **silhouettes** of curved faces (the outline of a sphere, the two generator lines of a lying cylinder, the two circles of a torus) and the **isolines** of curved faces, in the number set by the drawing's **ISOLINES** variable (4 if the file does not say): the generator lines of the cylinder and cone, the meridians and equator of the sphere, the circles of the torus in both directions, the internal lines of NURBS. Planar faces have no isolines.


BadBit CAD · solidos.dxf
Home
Insert
Annotate
View
Layout
Manage
Output
Parametric
Line
Polyline
Circle
Arc
Draw
Modify
Text
Dimension
Annotation
Layer Properties
Unsaved Layer State
Layers
Block
Properties
Utilities
Start
solidos.dxf
Model
Layout1
Drawing opened completely, as in AutoCAD. Technical details: IMPORTREPORT.
Zoom extents
Type a command
13.000
6.500
m
GPU: Apple M2 Pro (Metal)
1:1

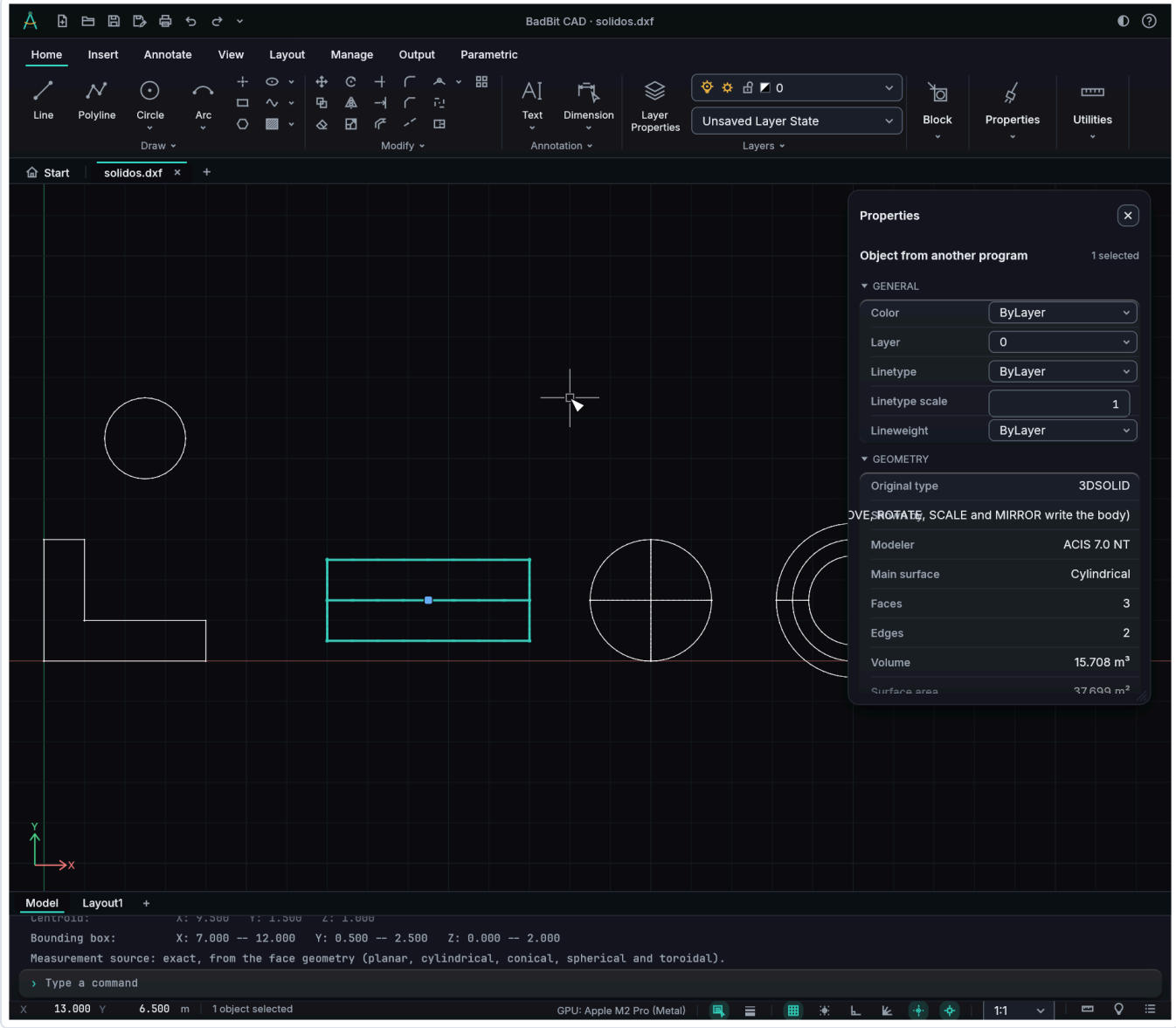


A solid is **selected** by its edges, is included in **ZE** (zoom extents) and has **snaps** on its edges and vertices: endpoint and midpoint of straight edges, center of circular edges (a standing cylinder gives the center through the circle of its cap) and the points of the silhouettes.

LIST gives the type, the modeler that wrote it (ACIS or ASM and the version), the lumps, faces, edges and vertices, the surfaces by type, the **volume** and surface area (for a region, the area and perimeter), the **centroid** and the **bounding box** in 3D and, on the last line, the **measurement source** (see below):



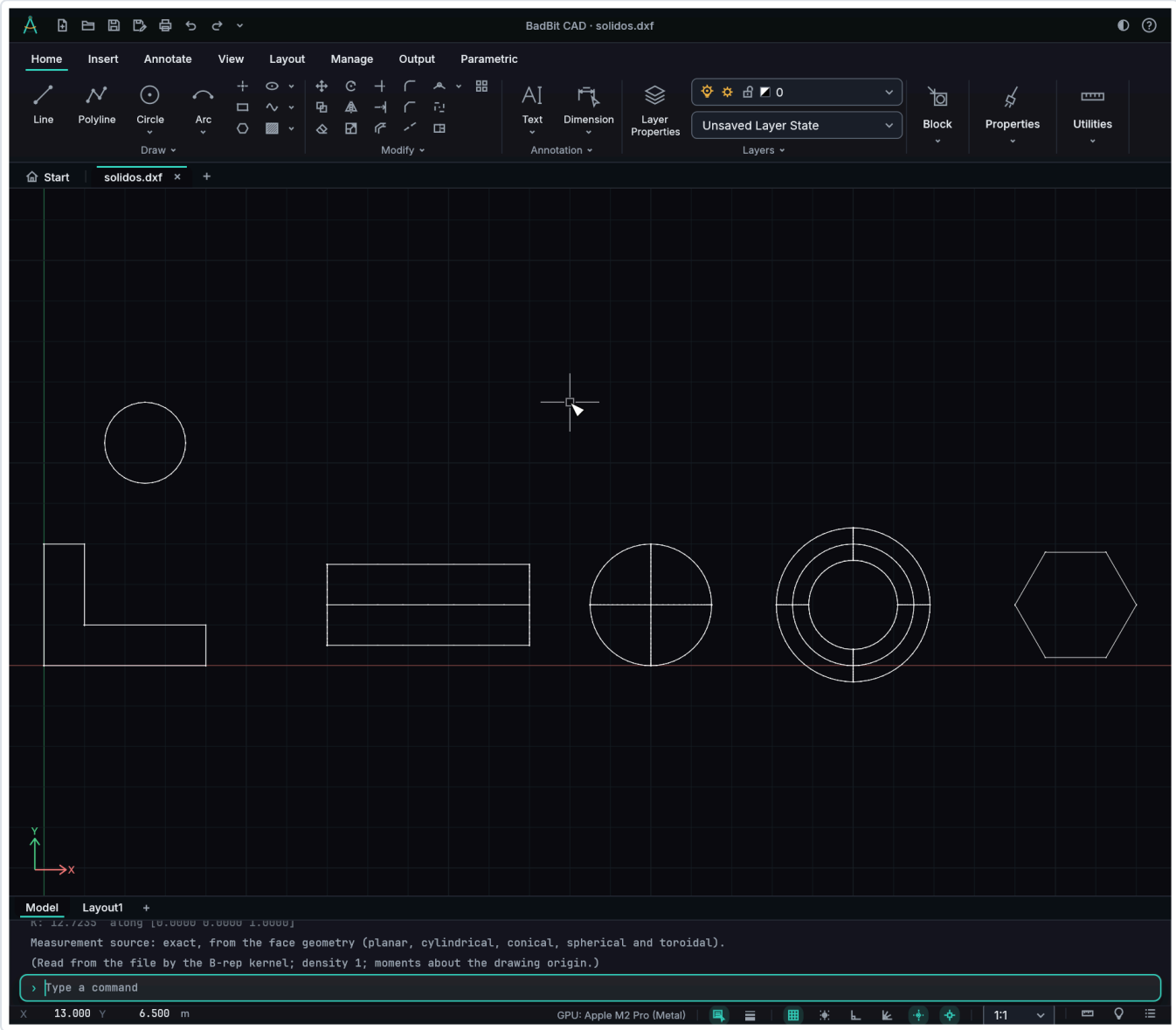
Properties show the same summary (modeler, main surface, faces, edges, volume, area, measurement source and centroid), read-only:

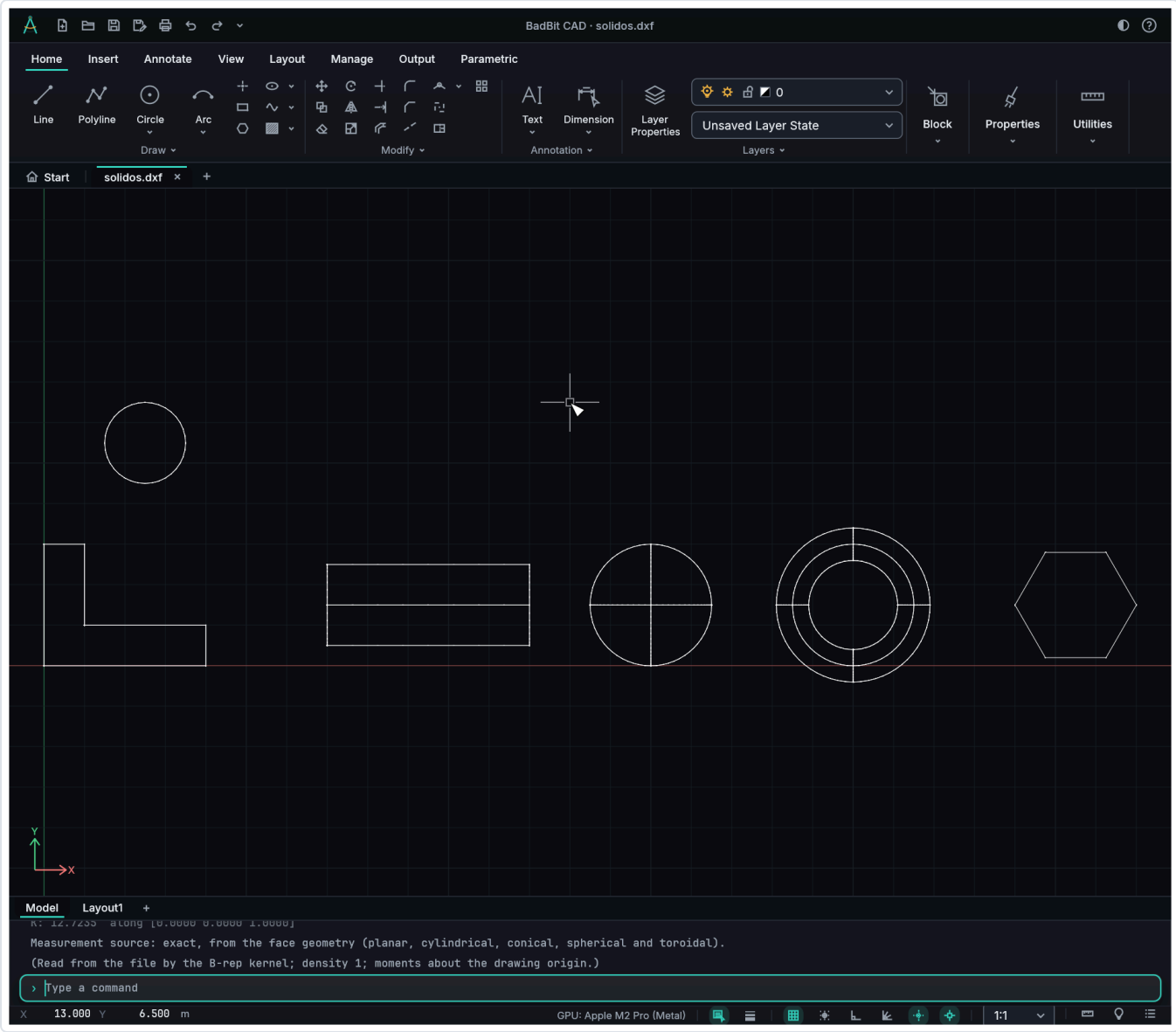


MASSPROP accepts the solids and regions read: for a solid, mass (density 1, as in AutoCAD) and volume; for a region, area and perimeter; for both, the bounding box and the centroid. For a solid, as AutoCAD lists them, there are also the **moments of inertia** about the drawing's X, Y and Z axes, the **products of inertia** XY, YZ and ZX, the **radii of gyration** and the **principal moments** with their directions about the centroid; for a planar region, the area moments in the plane (X, Y, the XY product, the radii and the principal ones with the angle). The values come from the **exact geometry** of the faces, not from the display mesh: on planar, cylindrical, conical, spherical and toroidal faces (whole or trimmed by lines, arcs, ellipses and intersection curves) the volume, area, centroid and moments match the formula to about 0.000000000001% (a sphere of radius 5 gives 523.5988, as in AutoCAD), also in UTM coordinates; on NURBS faces and construction surfaces (loft, sweep, spline revolution) the computation is numerical, with the estimated error well below the decimal places shown. Only a face that BadBit CAD cannot integrate this way (a surface it does not understand) goes in through the mesh, with an error of about 0.001%.

Where each number comes from. LIST and MASSPROP end with the **Measurement source** line, and Properties have a row with the same name, as the trace BadBit CAD leaves in every computation: **exact** (all faces planar, cylindrical, conical, spherical or toroidal, integrated over their geometry); **numerical integration, estimated relative error X** (there are NURBS or construction faces);

approximate edge stored in the file, estimated relative error X (a face bounded by an intersection curve the file stores only as an approximation: the measurement is that of the region of this approximation, and the estimated error includes how far it deviates from the face); **from the mesh, estimated relative error X** (or **no error estimate**), when some face cannot be integrated over its geometry. The error is relative, in scientific notation: 3.2×10^{-10} means 0.000000032% of the volume (or of the area, for a region and a surface). In a `.bbd` saved by an earlier version, the line says the source was not recorded; opening the DWG or DXF again measures it again.



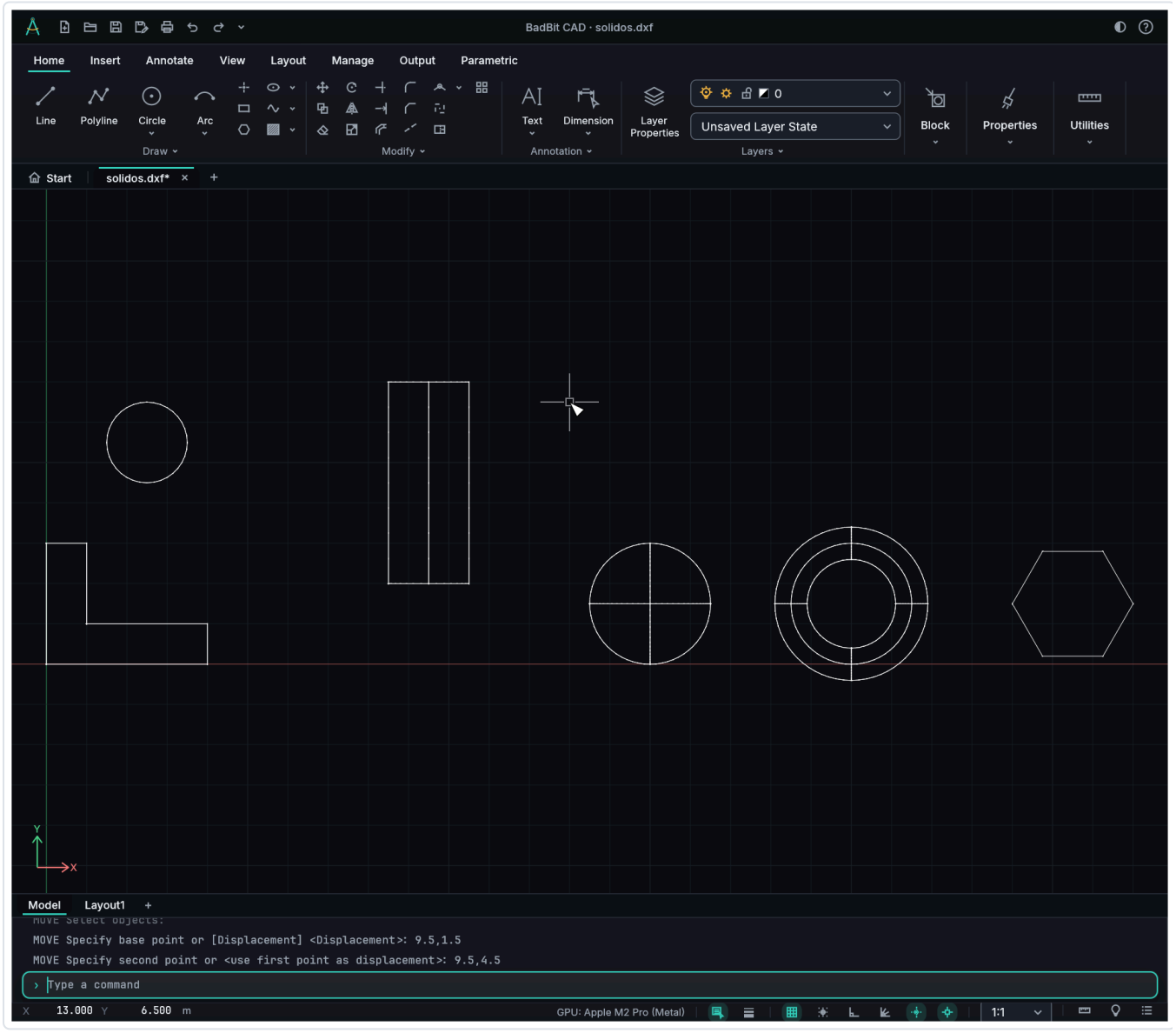


MOVE, ROTATE, SCALE and MIRROR work on solids, regions, bodies and surfaces read from DXF (and on solids, regions and bodies from DWG): the plan drawing and the numbers go along (volume, centroid, box and moments) and, when saving, BadBit CAD writes the AutoCAD object itself with the **transformed ACIS body**, in the same format and version it came from (the encoded SAT of DXF or the SAB of DWG). The transformation goes into the body's transform record, as ACIS itself does: the face geometry stays intact, including what BadBit CAD does not understand. Changing the layer, color or linetype also writes the original object, with the ACIS data unchanged.

Solids inside blocks. A solid, region or surface inside a block goes to the place of the insert by the **full 3D transformation**: the X, Y and Z scales, the rotation, the insertion point with its Z and the normal of the insert. The attributes of a tilted insert also go onto its plane when saving (the point and the angle in the insert's system, with the normal), including after moving, rotating or scaling the insert. The other shapes of the block appear in plan by the projection of the 3D transformation (see the blocks chapter); what is written for them does not change. When exploded (EXPLODE), the solid comes out with the volume, area, bounding box, centroid and moments of the insert: in the test file `corpo_2018.dwg`, the block with a 4×6×8 box inserted with scale 2 and 30° gives volume 1536, like AutoCAD's MASSPROP on the exploded copy, and inserted with scale (2, 1, 3) and 45° gives volume 1152, area 768 and the box of the 8×6×24 box rotated (this insert, as in AutoCAD, cannot be exploded:

EXPLODE warns **Cannot explode that block**; see the blocks chapter). With non-uniform scale (or a tilted normal), volume, centroid and moments remain exact, and the surface area comes from the mesh of the transformed body: the **Measurement source** line says so, with the estimated error.

Copies of solids. COPY, **MIRROR** with copy, **ARRAY**, **COPYCLIP/PASTECLIP** (and paste as block, BLOCK and WBLOCK) carry the **body** along: the copy of a 3DSOLID, a REGION, a BODY or a surface has the same volume (or the same area), appears in LIST and in Properties as a solid and, when saving, is written as a **real solid** in DXF and DWG (before, it became the lines of the plan drawing), with the accumulated transformation (the place of the copy, the rotation of the array, the mirror). The copy's body also goes into the `.bbd`. The copy of an EXTRUDEDSURFACE, LOFTEDSURFACE, REVOLVEDSURFACE, SWEPTSURFACE or NURBSURFACE comes out as the **generic surface** (SURFACE) with the same body: the construction parameters (the path, the sections) stay only in the original. Copying between drawings with different units converts the body by the unit of each one. `.bbd` files saved before this version do not have the body of the copies made in them: those copies remain the plan drawing.



When unchanged, the ACIS data goes back **intact** when saving, and in any DWG or DXF version the body goes in the form AutoCAD requires there: as **text** in DXF up to 2010 and in DWG 2000 (the binary of the newer versions is converted to text), inside the object in DWG 2004 to 2010 and, in **2013 or newer**, in the separate section of the file, which BadBit CAD builds with the solids read (the source one is never copied: if it is corrupted, its solids are left out with a warning, and the file opens). **Checked with AutoCAD 2027**: AutoCAD solids and regions opened and saved by BadBit CAD in DWG 2000 to 2018 and DXF 2000 to 2018 open in AutoCAD, with AUDIT reporting no errors and the same volume, box and centroid in MASSPROP (except what is left out, below). The drawing with 301 solids (one of them with 900 holes and 1.4 MB of ACIS data) also opens and measures the same in DWG 2013 and 2018 and in DXF 2018.

Surfaces. AutoCAD surfaces (PLANESURFACE, EXTRUDEDSURFACE, REVOLVEDSURFACE, LOFTEDSURFACE, SWEPTSURFACE and NURBSURFACE), read from a DXF or a DWG, go as the **same type** to any DWG and DXF version, from 2000 to 2018, with the ACIS body and the construction data (the profile, the path, the sections, the options) in the order AutoCAD writes them. **Checked with AutoCAD 2027**: the six surfaces made in it, saved by BadBit CAD in all versions, open in AutoCAD with AUDIT reporting no errors, in the source type and with the same area. AutoCAD writes NURBSURFACE as SURFACE in 2010 and earlier and shows it as NURBSURFACE; BadBit CAD writes the NURBSURFACE itself there, which AutoCAD accepts and shows the same. A surface whose construction data is missing from the file (or a LOFT with guides, not checked yet) goes as the generic surface (SURFACE), with the same body, and the save report counts the "surfaces written as generic surface". In the Import report the solids read count as "objects read by the B-rep kernel", among those kept intact; a solid whose data cannot be read (unknown format, procedural surface with no stored approximation) stays as before, with no drawing, counted as "3D in plan", and the detail says what was missing.

Known limits

- **DWG R13 and R14** (AutoCAD 13 and 14): reading follows the specification but has not yet been checked with real files; writing is from 2000 on.
- **Application objects in DWG** go back only in the source version (the Civil 3D ones also go to the other DWG versions).
- **Civil 3D objects in DWG 2000 and 2004**: read through what does not depend on the texts (points, surfaces, feature lines, styles), and those that come as proxies (written by plain AutoCAD or by BadBit CAD) whole; the place of the texts in these versions has not yet been checked against a Civil 3D file with points and surfaces. AutoCAD 2027 opens those written by BadBit CAD without errors; their reading by Civil 3D (the proxy, "modelo" and "sem textos" ones) has not been checked yet. In DWG 2010 and 2013 the associative network goes in the format of each version, checked in AutoCAD 2027.
- **ACIS solids, regions and surfaces**: reading and writing have been checked with AutoCAD 2027 (see above). When saving to DXF up to 2010 or to DWG 2000, a solid with a geometry type the text conversion does not know yet **is left out**, with a warning in the save report ("not written", with the reason). A solid inside a **block** appears in the right place in plan, but the Z scale and the elevation of the insert do not enter the measurements. The copy of a construction surface still comes out as the generic surface (see above). The SURFACE AutoCAD writes in place of NURBSURFACE (DWG and DXF up to 2010) goes back as SURFACE. Rolling-ball, loft/skin, net

and rail-sweep procedural surfaces without the stored approximation are evaluated by an assumed form of the data or, when not even that can be read, approximated by the edges of the face (the import report says "approximated by the edges").

- **User coordinate systems (UCS)**: kept, but BadBit CAD always draws in the world system.
- **MULTILEADER** remains MULTILEADER in DXF and DWG of any version from 2000 on (one created, changed or from another version; a changed one keeps the block, labels and colors of the original). Only in **DXF R12**, which does not have it, does it go as lines and text, with a warning.

LandXML (Civil 3D)

LandXML 1.2 is the open exchange format of surveying and road design programs: Civil 3D exports and imports project data through it. BadBit CAD reads and writes LandXML to bring from a Civil 3D project (or from another program) the **points**, the **surfaces** (the terrain model in triangles or in a grid, with boundaries and watersheds), the **alignments** (centerlines, with stations, profile and cross sections), the **parcels**, the **feature lines** (feature lines and breaklines), the **corridors** and the **pipe networks**, and to send back to Civil 3D the points and polylines drawn in BadBit CAD.

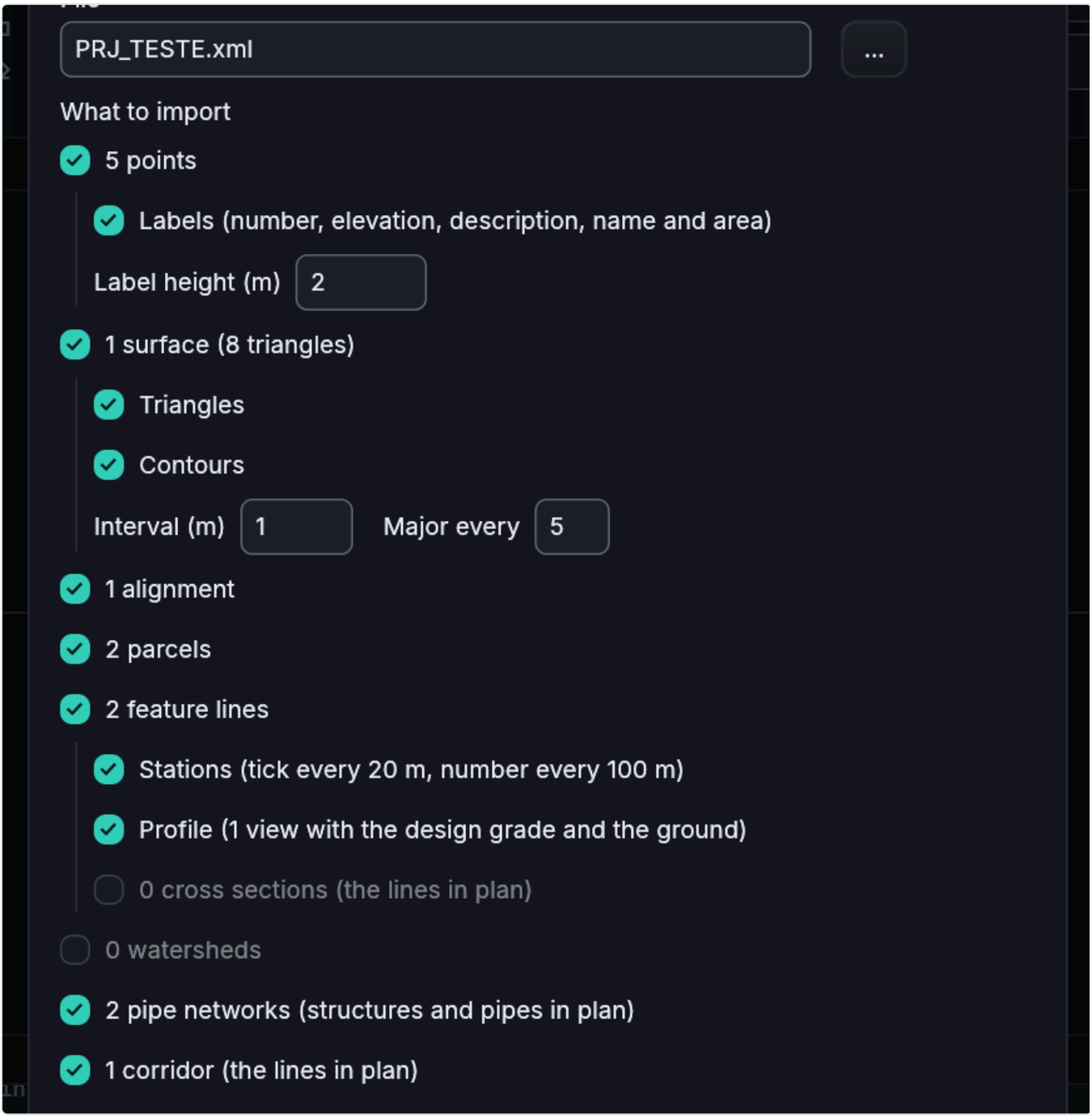
Commands

Command	What it does
LANDXMLIN	Imports a LandXML file into the **current drawing** (the command names are Civil 3D's, the same in all three languages).
LANDXMLOUT	Exports the selected objects to a LandXML file; with no selection, it asks `Select objects:` like AutoCAD commands (`Enter` with nothing chosen means the whole model space).

Both are on the ribbon: **Insert → LandXML → Import LandXML** and **Output → LandXML → Export LandXML**.

Importing (LANDXMLIN)

LANDXMLIN opens the **Import LandXML** dialog and, with FILEDIA 1, the system dialog to choose the file (`.xml`). The path can also be typed in the **File** field (Enter or **Read**). The file is read in the background, with progress in the corner of the screen; then the dialog shows what it contains:

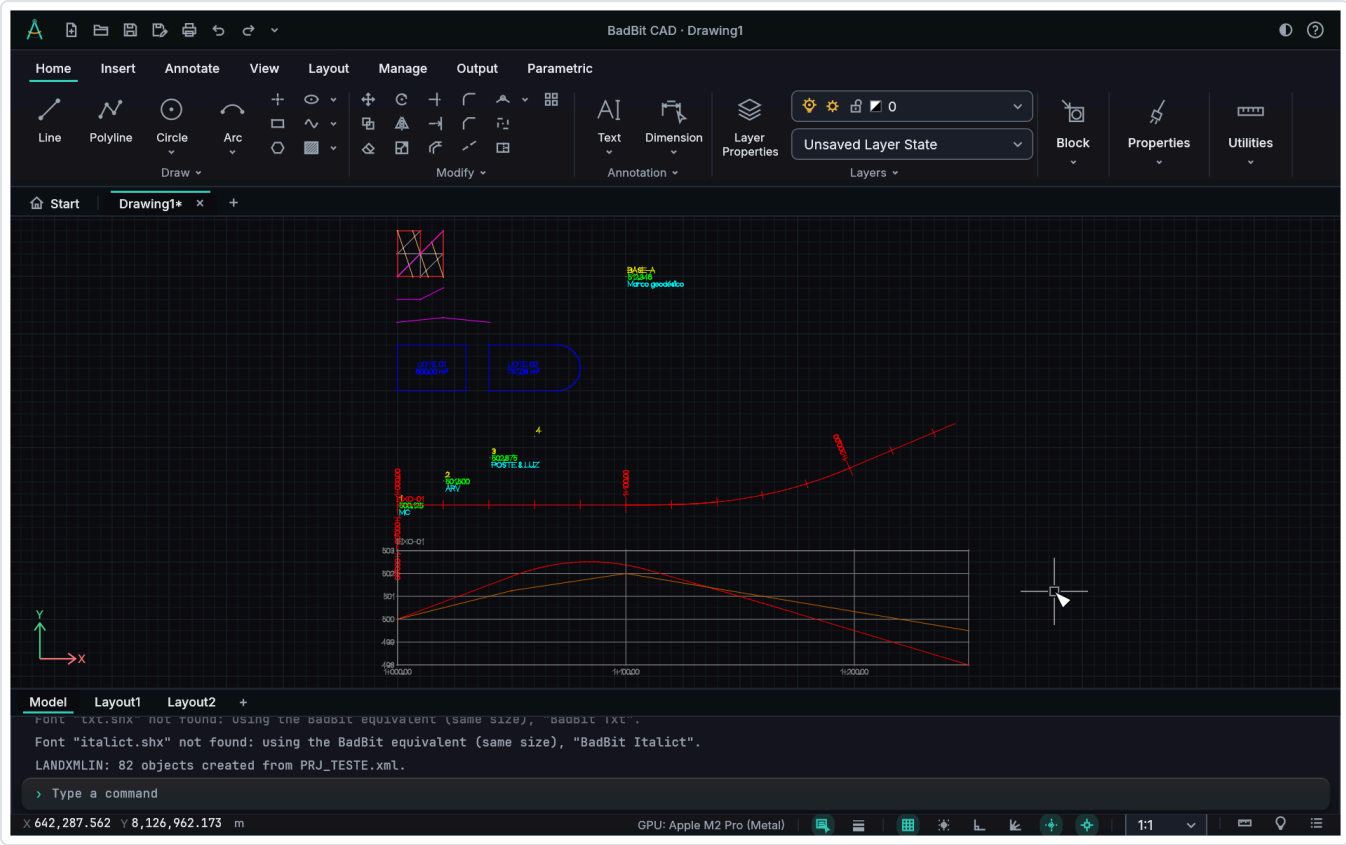


What the file contains, with the counts, the contour interval and the units.

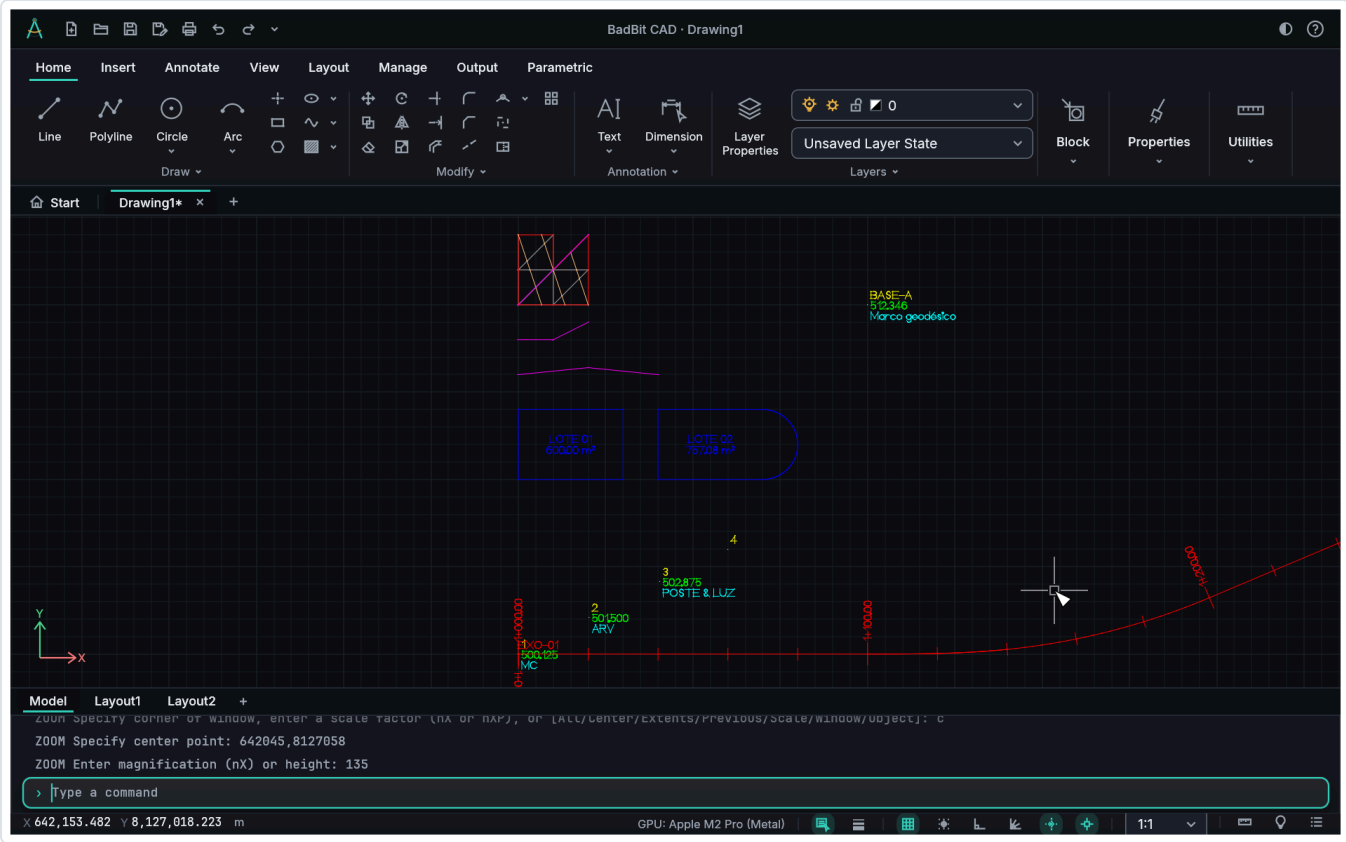
- **What to import**: one checkbox per type, with the count in the file (points, surfaces with the number of triangles, alignments, parcels, feature lines, pipe networks and corridors). For alignments, **Stations**, **Profiles** and **Cross sections**; for surfaces, **Watersheds**. A type the file does not have is grayed out.
- **Labels** and **Label height**: the number, elevation and description of points, the name and area of parcels, the name of the alignment and the elevation of major contours. The suggested height is a round number (1, 2, 2.5 or 5 times a power of 10) and proportional to the size of the survey.
- **Surfaces**: **Triangles** (the terrain model, one triangle per visible face) and **Contours**, with the **Interval** (default 1 m) and **Major every** (default 5: a contour whose elevation is a multiple of 5 m is a major one).

- **File units**: the system (metric or imperial) and the length unit of the file (LandXML's `<Units>`). If it is the same as the drawing's, the coordinates come in **unconverted, bit for bit**; otherwise, they are converted to the drawing unit (the factor appears in the dialog; the US survey foot is 1200/3937 m).

Import builds the drawing in the background (the window keeps responding) and creates everything **as a single undo step**: one U undoes the whole import. If the drawing was empty, the zoom goes to extents.



The whole file: the surface (top left), the feature lines, the parcels, the points and the alignment with the spiral and the arc.



Close up: the points with number, elevation and description; the surface with the border, triangles, breakline and contours; the parcels with name and area at the center.

What becomes a BadBit CAD object

LandXML	In the drawing	Layers
Points (`CgPoints`)	Point (POINT) at the file's easting and northing, with the labels	`LXML-PONTOS`, `LXML-PONTOS-NUMERO`, `LXML-PONTOS-COTA`, `LXML-PONTOS-DESCRICAO`
TIN surface (`Surface`: `Pnts` and `Faces`)	Triangles (closed three-vertex polylines), the border (closed polyline) and the contours	`LXML-SUPERFICIE-TRIANGULOS`, `LXML-SUPERFICIE-BORDA`, `LXML-CURVAS-MESTRAS`, `LXML-CURVAS-INTERMEDIARIAS`, `LXML-CURVAS-ROTULOS`
Surface breaklines (`Breaklines`)	Polylines	`LXML-LINHAS-QUEBRA`
Alignment (`Alignment`: `Line`, `Curve`, `Spiral`)	Polyline with the exact arcs; each spiral as its own object (a spline through the points of the clothoid, with the tangents at the ends)	`LXML-ALINHAMENTOS`, `LXML-ALINHAMENTOS-ROTULOS`
Parcel (`Parcel`)	Closed polyline, with the name and area at the center; with a spiral, the lines and arcs in polylines and each spiral in a spline, as in the alignment	`LXML-LOTES`, `LXML-LOTES-ROTULOS`
Feature line (`PlanFeature`)	Polyline; with a spiral, the lines and arcs in polylines and each spiral in a spline, as in the alignment	`LXML-LINHAS-CARACTERISTICAS`
Alignment stations	Perpendicular tick every 20 m and, every 100 m, the larger tick with the number (`0+100.00`)	`LXML-ALINHAMENTOS-ESTACAS`
Station equation (`StaEquation`)	Stations follow the ahead numbering after it; at its place, the label `0+250.00 = 0+300.00` (back = ahead)	`LXML-ALINHAMENTOS-ESTACAS`
Profile (`ProfAlign` and `ProfSurf`)	Frame below the drawing with the design grade (vertical curves) and the ground	`LXML-PERFIL-QUADRO`, `LXML-PERFIL-GREIDE`, `LXML-PERFIL-TERRENO`
Cross sections (`CrossSects`)	The section line in plan, perpendicular to the centerline at the station, with the station at the end; and one frame per section below the profile, with the ground and the design	`LXML-SECOES`, `LXML-SECOES-QUADRO`, `LXML-SECOES-TERRENO`, `LXML-SECOES-PROJETO`
Superelevation (`Superelevation`)	Diagram below the profile: the slope of the outside lane and of the inside lane along the stations, with the curve side and the pivot type in the label of each region	`LXML-SUPERELEVACAO-QUADRO`, `LXML-SUPERELEVACAO-EXTERNA`, `LXML-SUPERELEVACAO-INTERNA`

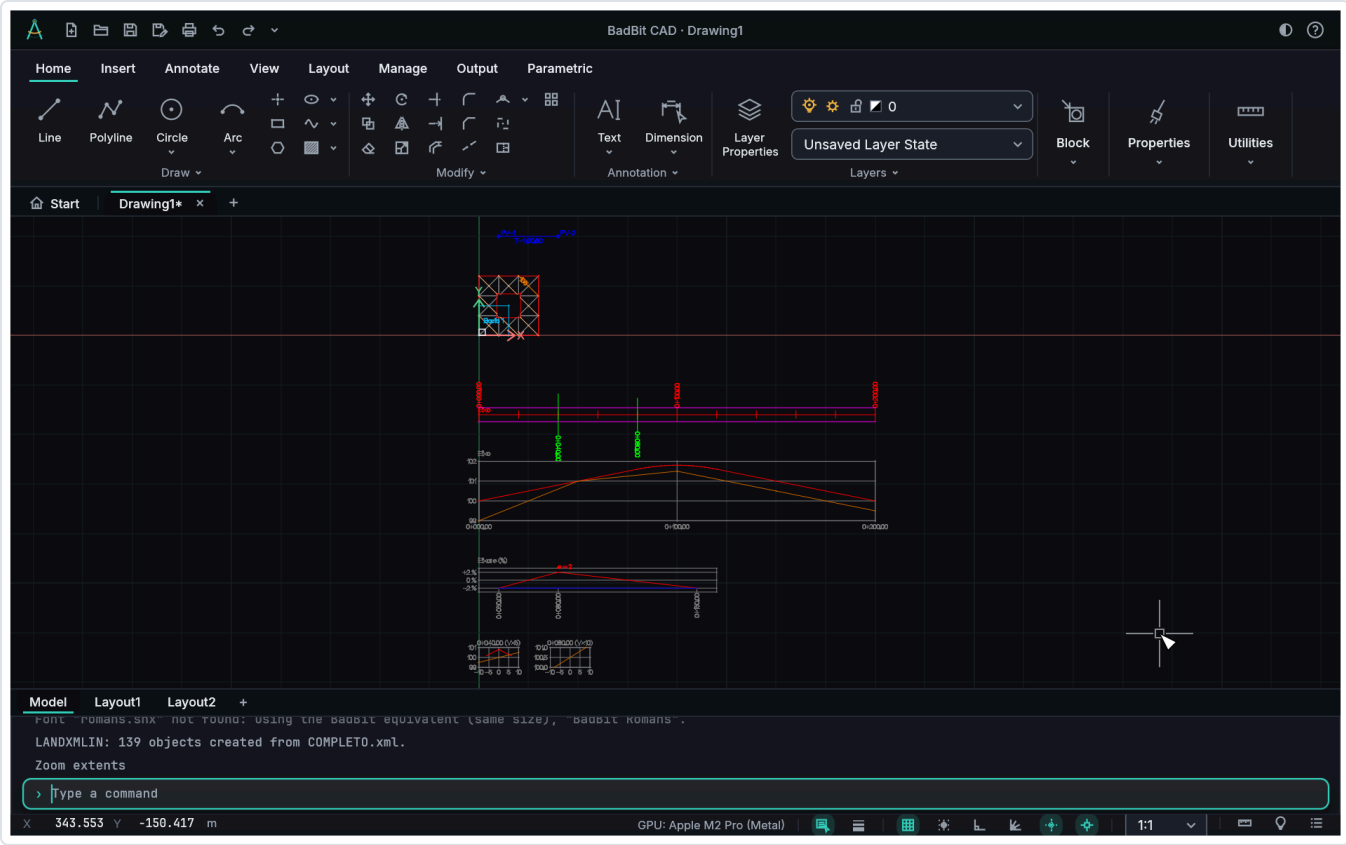
LandXML	In the drawing	Layers
Surface boundaries (`Boundaries`)	Exact clipping of the TIN: the triangles, contours and border are cut at the line of the outer boundary and of the voids (islands come back); a triangle that is entirely outside disappears	—
Watersheds (`Watersheds`)	Closed polyline with the name	`LXML-BACIAS`
Pipe network (`PipeNetwork`)	Structures as circles at their diameter, pipes from edge to edge of the structures, the names (the pipe with its diameter)	`LXML-REDES-POCOS`, `LXML-REDES-TUBOS`, `LXML-REDES-ROTULOS`
Corridor (`Roadway`)	The lines it contains (edges, feature lines)	`LXML-CORREDORES`

- **Coordinate order.** LandXML writes **Northing, Easting, elevation** ("N E Z"). BadBit CAD puts the easting in X and the northing in Y: a point `8127326.73 642776.21 512.41` goes to X = 642776.21 and Y = 8127326.73.
- **The elevation.** The BadBit CAD drawing is 2D: the elevation of each point (and of the vertices of feature lines and of the surface) is **kept with the object** and appears in the label; LANDXMLOUT writes it back.
- **Contours.** Computed by BadBit CAD from the triangles (each triangle cut by the plane of each elevation, the pieces chained into polylines, in parallel). They always come out **closed** or **ending at the border** of the surface; a contour that passes exactly through a surface point goes through it. Only visible faces count (those marked as invisible in Civil 3D are left out), and with a boundary the contours are cut at its line.
- **Spirals.** Each spiral of the alignment is **an object of its own**: a spline that passes through the points of the clothoid (curvature varying with length) and leaves tangent to the neighboring line and arc, ending exactly at the end point in the file (it stays within one micrometer of the clothoid on a 100 m spiral). An alignment with spirals becomes pieces: consecutive lines and arcs in a polyline and each spiral in a spline; the first piece carries the whole alignment for LANDXMLOUT. In feature lines and parcels (rare) it is the same: each spiral becomes a spline, the first piece carries the whole object and the report says how many (the parcel area is that of the outline, with the spiral through the points of the clothoid). The spiral spline **can be edited** (MOVE, ROTATE, grips): LANDXMLOUT recomputes the spiral from it (see **Exporting**, below).
- **Grid surfaces.** A `grid` surface becomes triangles: each four-point cell into two (and also a regular grid that comes only with the points, without faces).
- **Parcel area.** Computed from the geometry (lines and arcs), in the drawing unit squared.

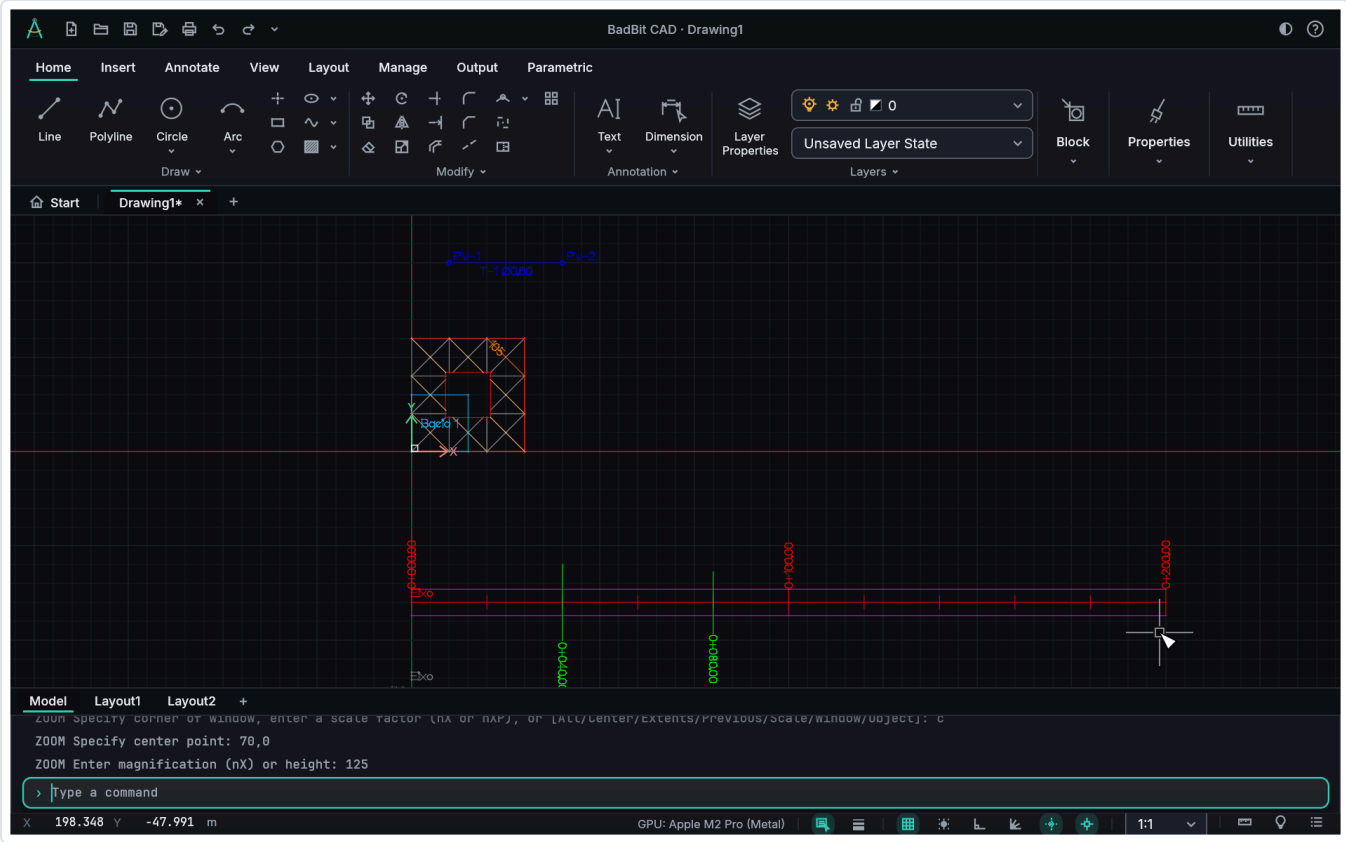
Stations, profile and cross sections

Stations follow Civil 3D's default style: a tick perpendicular to the centerline every 20 m and, at stations that are multiples of 100 m, the larger tick with the number in kilometers + meters (`0+100.00`, with the drawing's decimal separator), from the starting station of the alignment. The **profile** comes

out in a frame below everything that was imported: the stations horizontally, at the drawing scale, and the elevations vertically with a **10× exaggeration** (the Civil 3D default); the grid at the 100 m stations and at round elevations, the elevations on the left, the stations at the bottom and the profile name at the top. The **design grade** (design profile) carries the vertical curves (parabolas, tangent to both grades; a circular curve comes out as the parabola of the same length) and the **ground**, the points in the file. **Cross sections** appear in plan as the line perpendicular to the centerline at each one's station, from the smallest to the largest sampled offset, and each one in a frame below the profile (see below). **Superelevation** comes out in a diagram below the profile and goes back through LANDXMLOUT as it came.

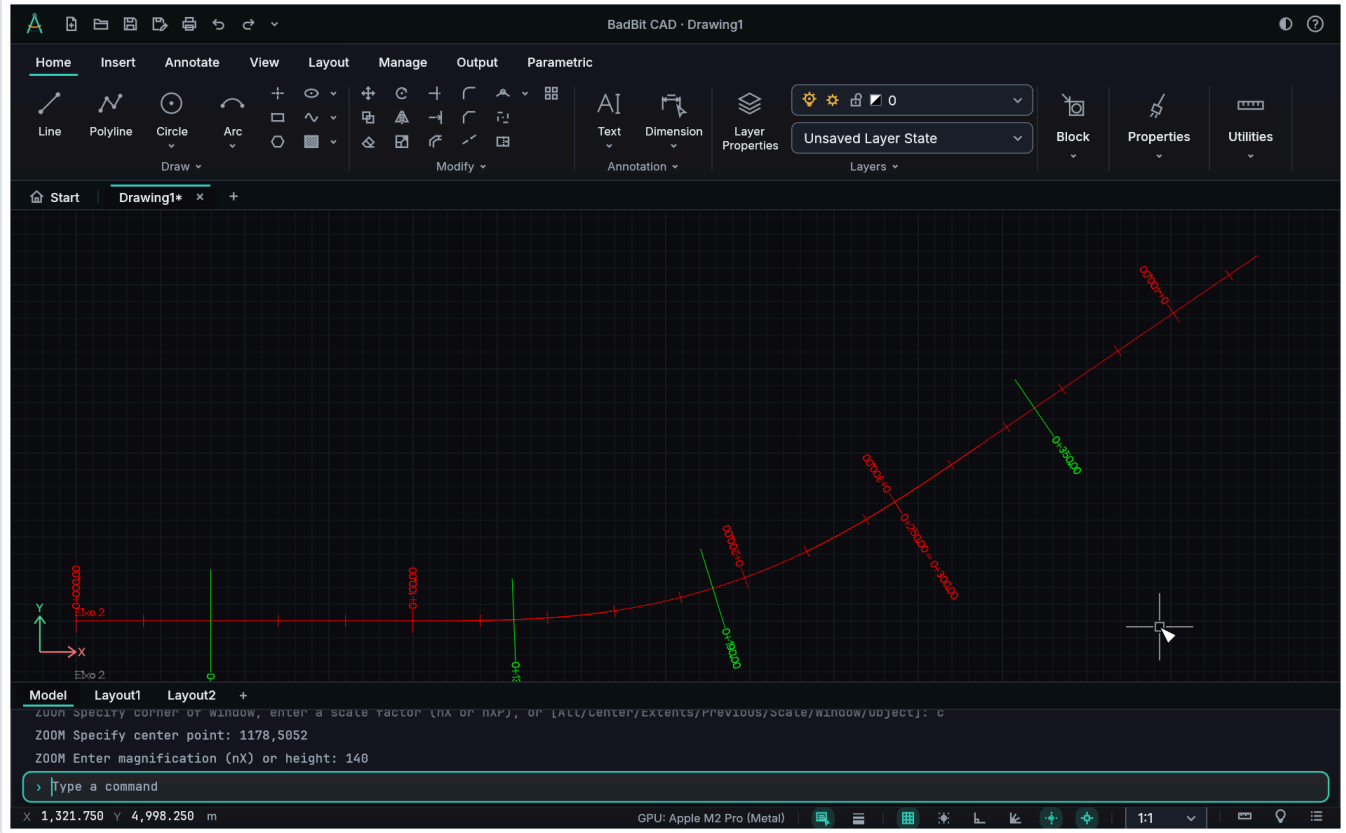


The grid surface with the void in the middle and the watershed, the pipe network at the top, the centerline with stations, the cross sections and the corridor, and the profile frame at the bottom.



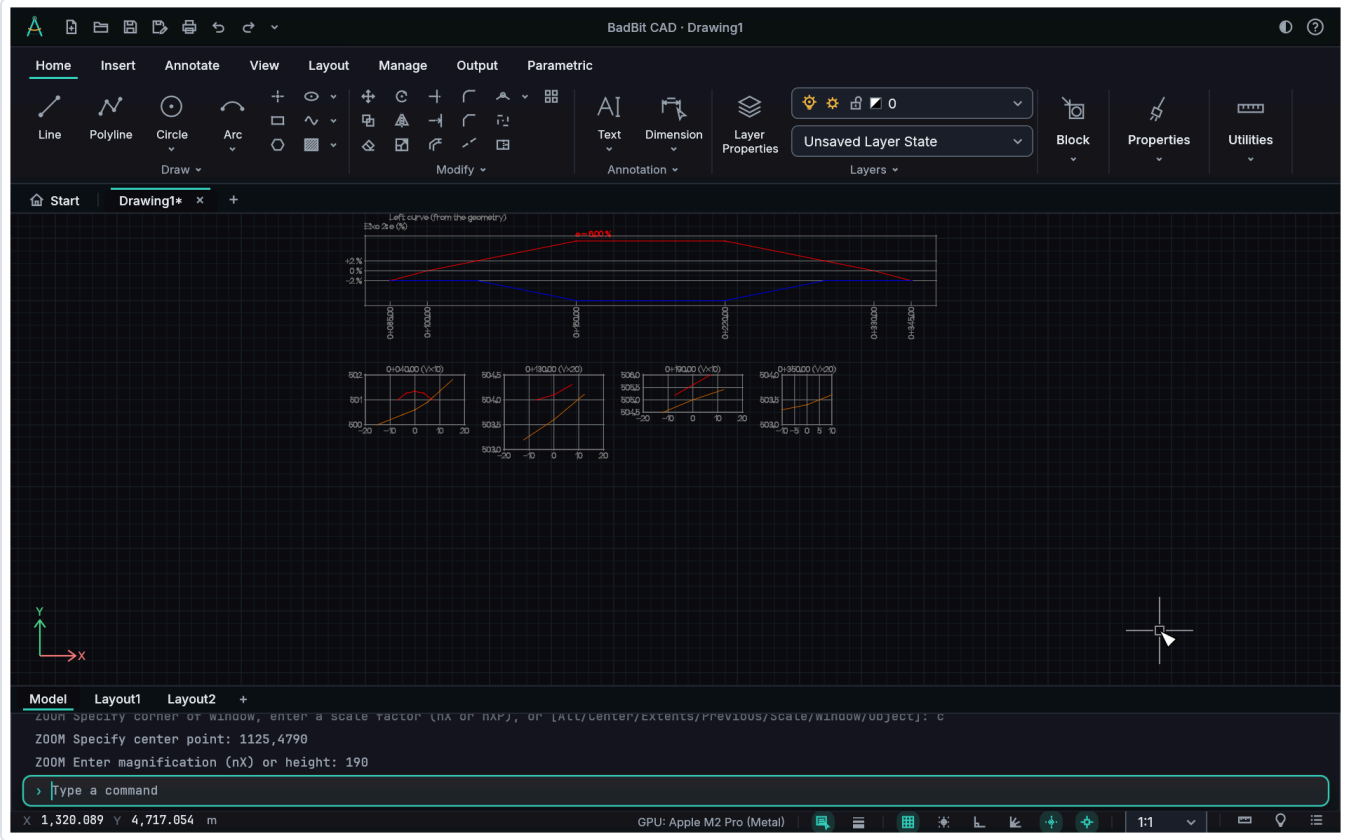
The stations every 20 m with the number every 100 m, the cross sections with their station, the corridor edges and the structures and the pipe with its diameter.

Spirals, station equations, superelevation and cross sections

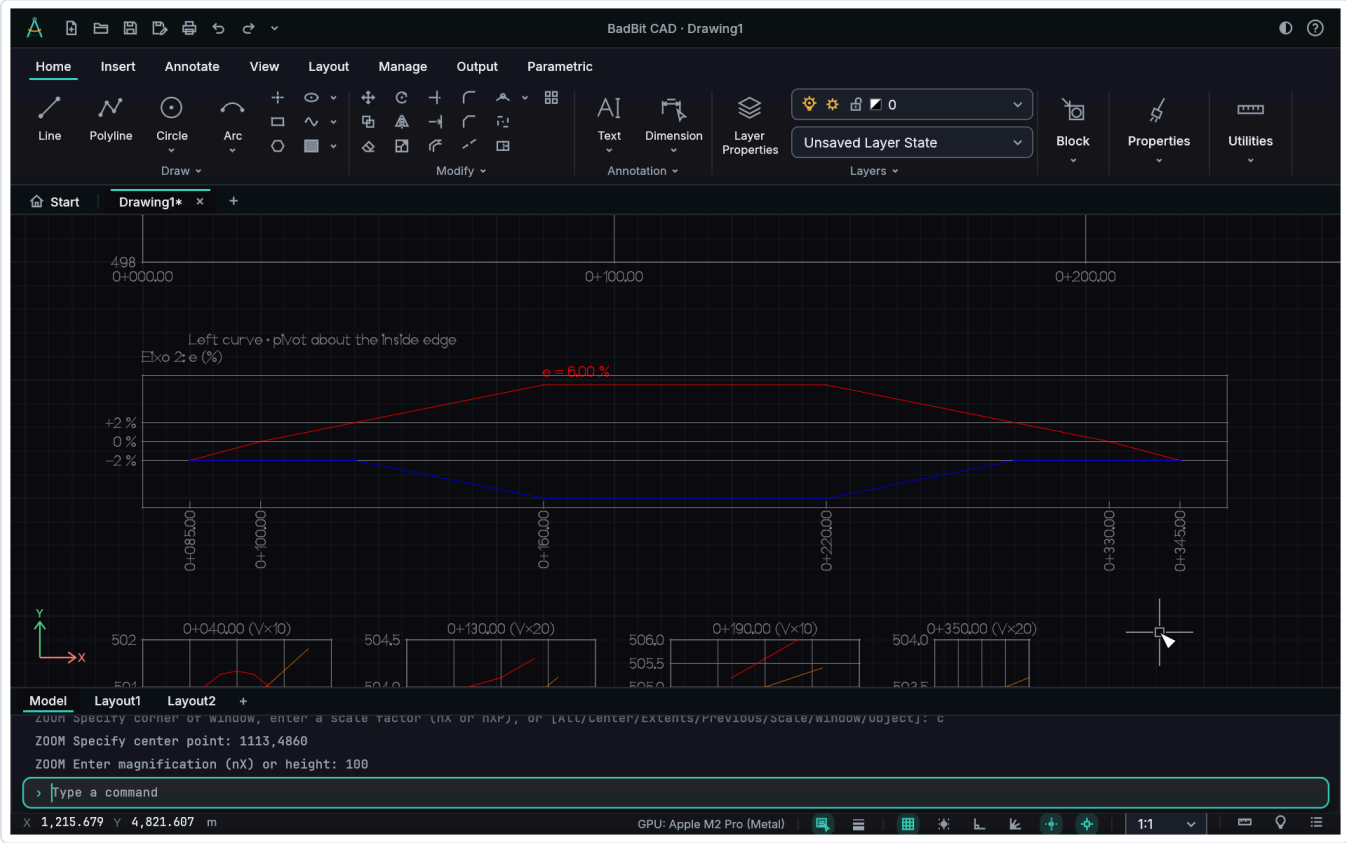


The line, the entry spiral, the arc, the exit spiral and the line: the spirals are splines (objects of their own). At internal station 250 is the equation `0+250.00 = 0+300.00`, and the stations follow the new numbering.

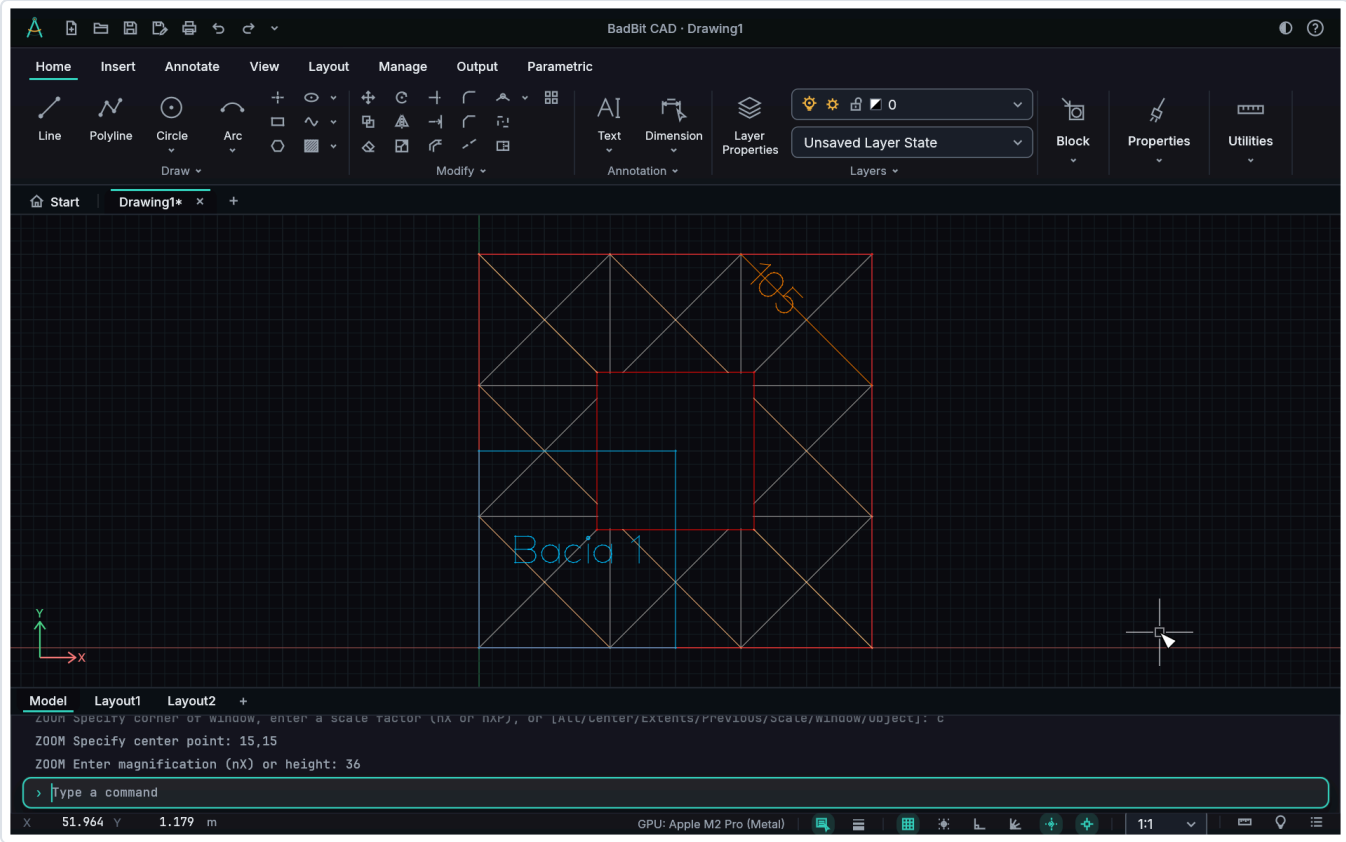
- **Station equation** (`StaEquation`). At the internal station of the equation, the numbering shown restarts at the **ahead** station (the **back** station is the one it came from). The ticks every 20 m and the numbers every 100 m fall on the round stations of the new numbering; at the place of the equation there is the label "back = ahead". The stations of the cross sections and superelevation and the numbers in the profile frame come out in the numbering shown. Without `staInternal` in the file, the internal station is taken from the back station (assumed).
- **Superelevation** (`Superelevation`). One diagram per alignment, below the profile: the stations horizontally and the cross slope vertically, with the 0 % and normal crown (±2 %) lines, the **outside lane** (red) and the **inside lane** (blue), pivoting about the centerline: the outside lane goes from the normal crown (−2 %) to 0 %, to full superelevation and back; the inside lane stays at −2 % until the outside lane passes +2 % and then follows it. The key points have the station below and the full superelevation is written (`e = 6.00 %`). LandXML does not carry the normal crown: Civil 3D's −2 % applies (assumed). The point names are those of the 1.2 schema (`BeginRunoutSta`, `BeginOfRunoffSta`, `FullSuperSta`, `FullSuperelev`, `RunoffSta`, `StartofRunoutSta`, `EndofRunoutSta`); without the full superelevation, the diagram says `e = ?`. At the top of the frame, at the start of each region, go the **curve side** and the **pivot type** ("Right curve · pivot about the inside edge"). The 1.2 schema has no field for either; BadBit CAD reads (assumed) the attribute of `Superelevation`, a child element with the same name or the `Property` (`label`/`value`) of a `Feature` inside it, with the names `pivotType`, `pivot`, `pivotMethod`, `pivotPoint`, `rotationPoint`, `rotationPivot` or `superelevationPivot` for the pivot (values with `center`/`crown`/`baseline`/`axis` = centerline, `inside`/`inner` = inside edge, `outside`/`outer` = outside edge, `left`/`right` = left/right edge; any other text appears as it came) and `curvature`, `side`, `curveSide`, `curveDirection`, `direction` or `rot` for the side (`left`/`L`/`ccw` = left, `right`/`R`/`cw` = right). Without the side in the file, the one from the geometry at the full superelevation station applies (the direction of the arc or spiral there), marked "(from the geometry)". The pivot does not change the slopes in the diagram (only the edge elevations). Both go back through LANDXMLOUT as they came, together with the superelevation kept whole.
- **Cross sections** (`CrossSects`). One frame per station, four per row, below the superelevation diagram: the offsets horizontally (1:1, the centerline at zero) and the elevations vertically, with the smallest exaggeration of 1, 2, 5, 10 or 20 times that keeps the frame legible (written in the title, `0+040.00 (V×10)`); the round grid with the numbers, the **ground** (orange) and the **design** (red).



At the top, the superelevation (the outside lane up to +6 % and the inside lane down to −6 %); at the bottom, the four cross sections, the last one at station 0+350.00 (after the equation).

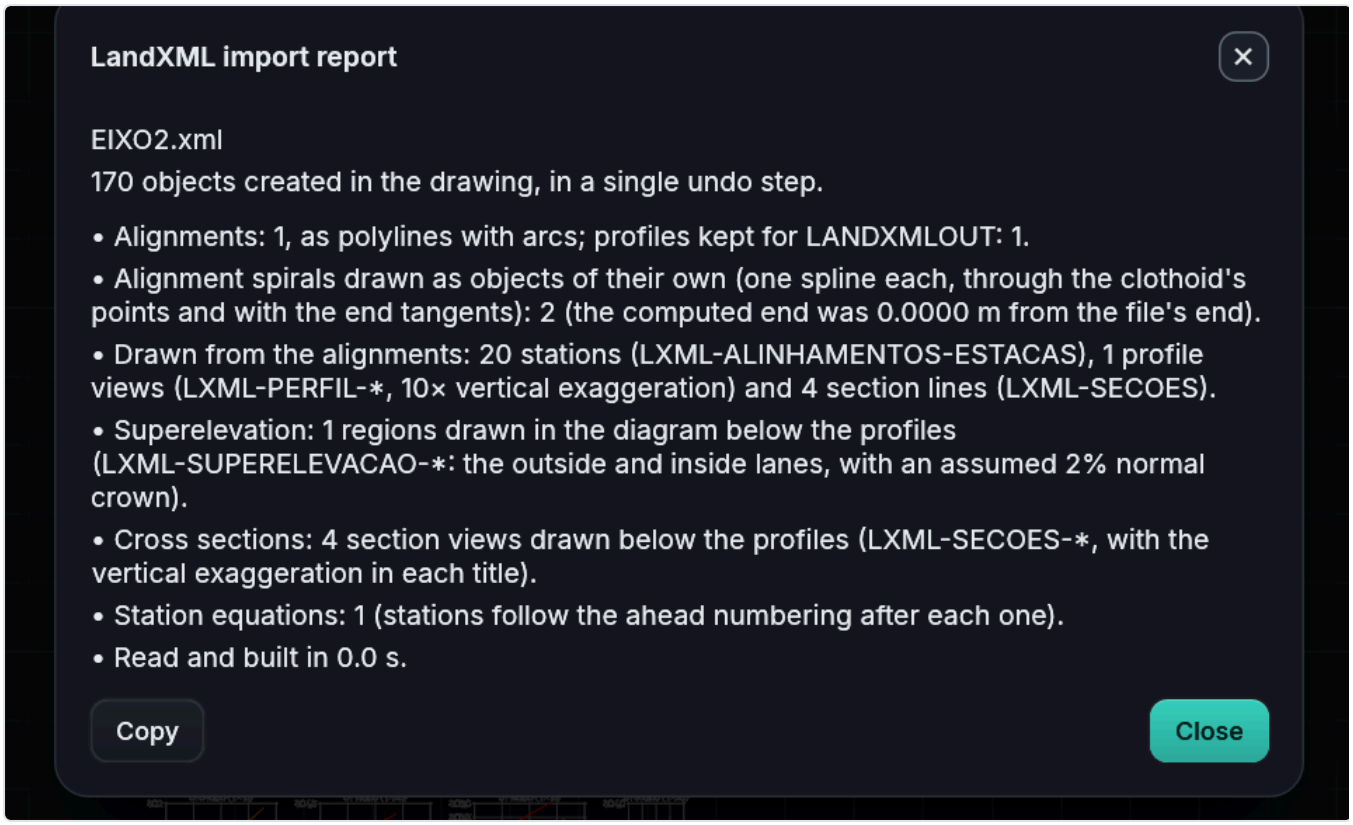


The superelevation with `pivotType="insideEdge"` and `curvature="left"` in the file: the label "Left curve · pivot about the inside edge" above the diagram title.



The void in the middle of the grid: the triangles and contours are cut exactly at the line of the void, and the border shows the outline and the void.

Clipping by the boundary is exact, like the Civil 3D boundary: each triangle edge, each contour and the outline of the TIN are cut at the point where they cross the outer boundary, a void or an island (the elevation of the cut point is that of the edge, the face is planar); what is outside disappears, and a triangle that is entirely outside does not appear. The border drawn is the outline of the TIN inside the boundaries and the boundaries inside the TIN. The report says how many triangles were removed and how many were cut.



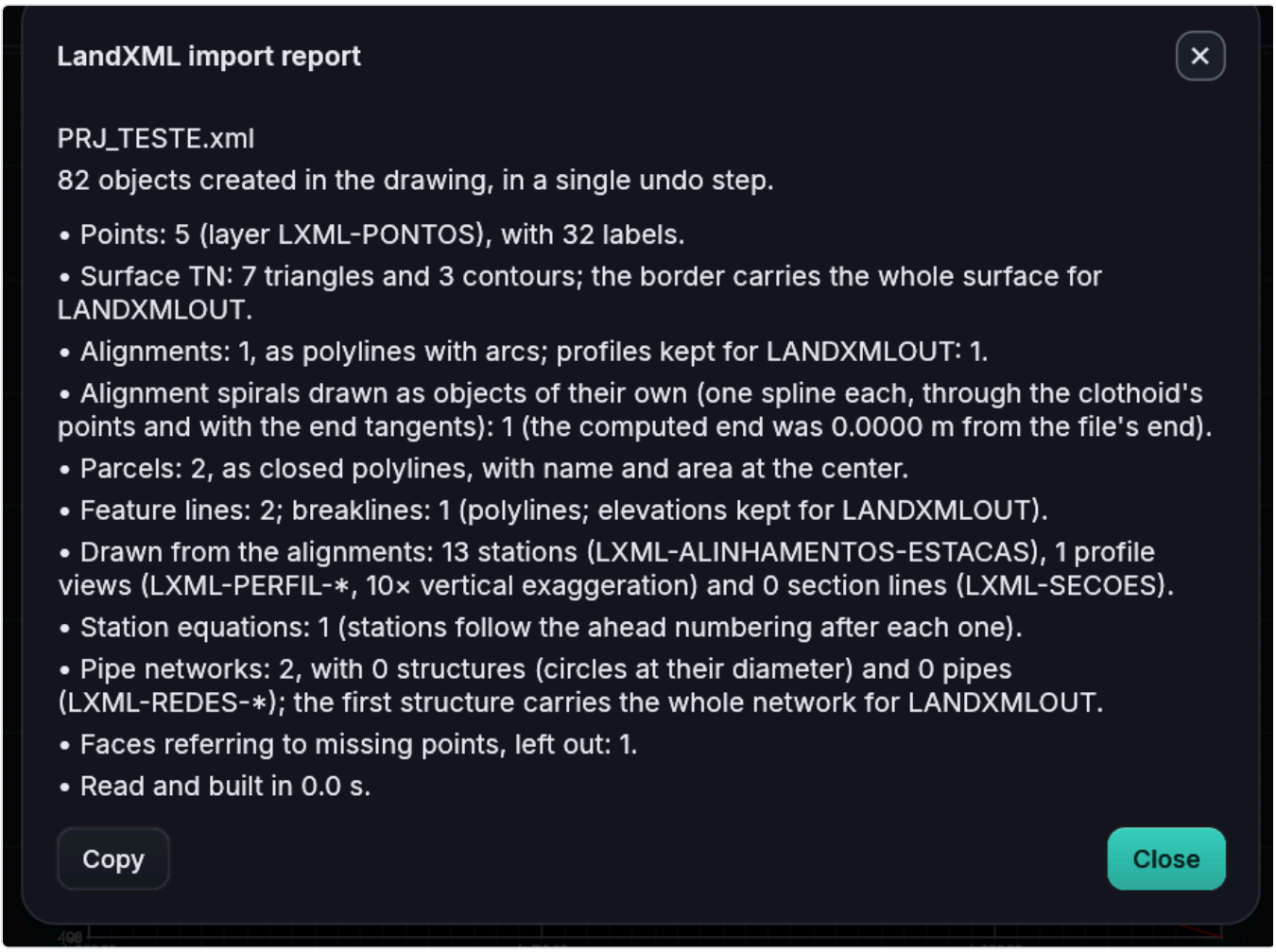
The spirals as splines, the stations, the superelevation drawn, the cross section frames and the station equation.

Pipe networks and corridors

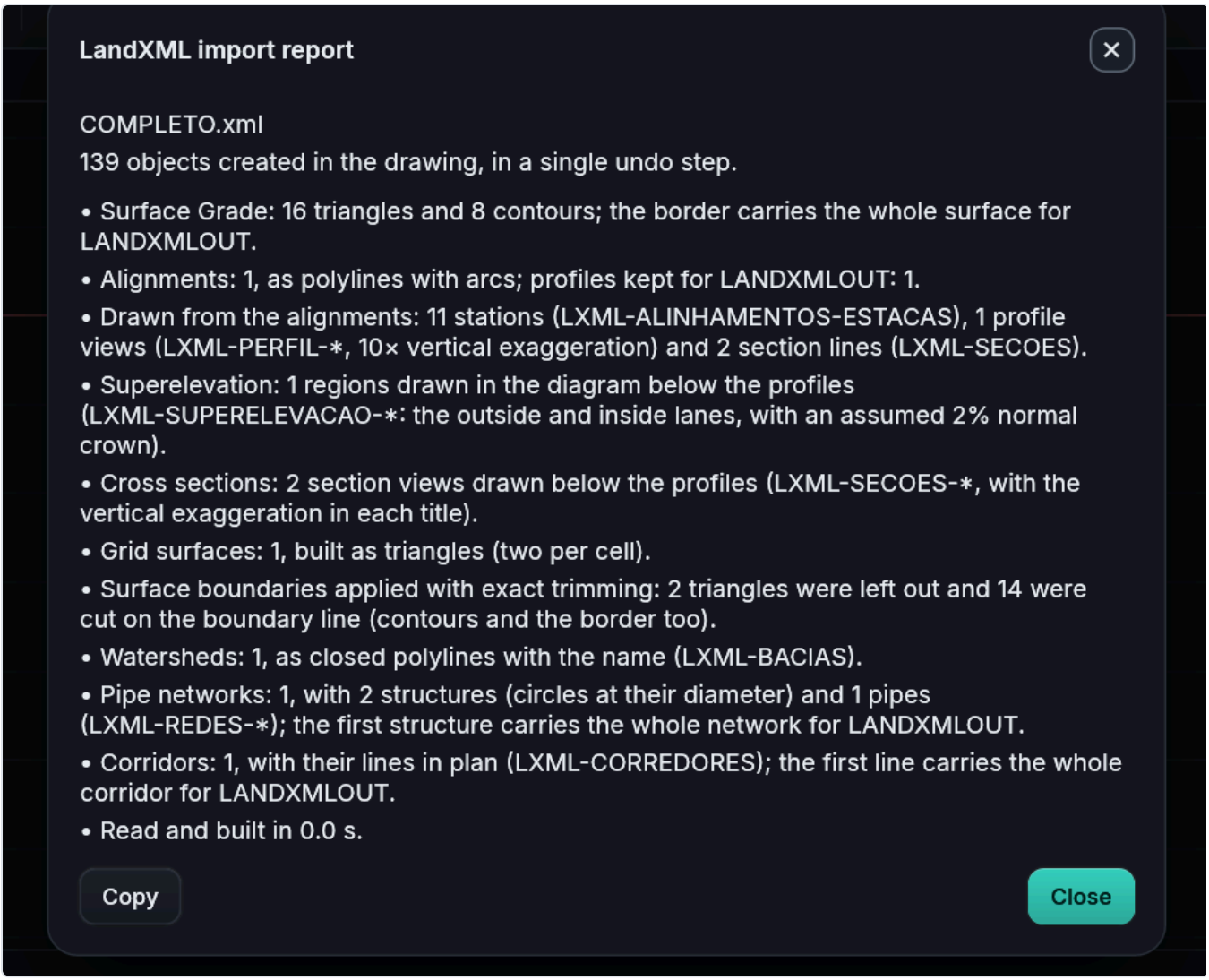
Each **structure** (`Struct`) becomes a circle at the diameter in the file (the label height, if the file does not say), with the name beside it; each **pipe** (`Pipe`) goes from the edge of the start structure to the edge of the end structure, with the name and diameter below (`T-1 Ø0.60`). Rim and invert elevations, slopes and the rest of the network are kept: the **first structure** carries the whole network for LANDXMLOUT. From the **corridor** come the lines in plan it contains (the edges and the feature lines); the first one carries the whole corridor.

The LandXML import report

At the end the report appears, short: how many objects were created, what came out of each type, the spirals (splines) and what was left out.



What was created and what was left out; Copy puts the text on the clipboard.

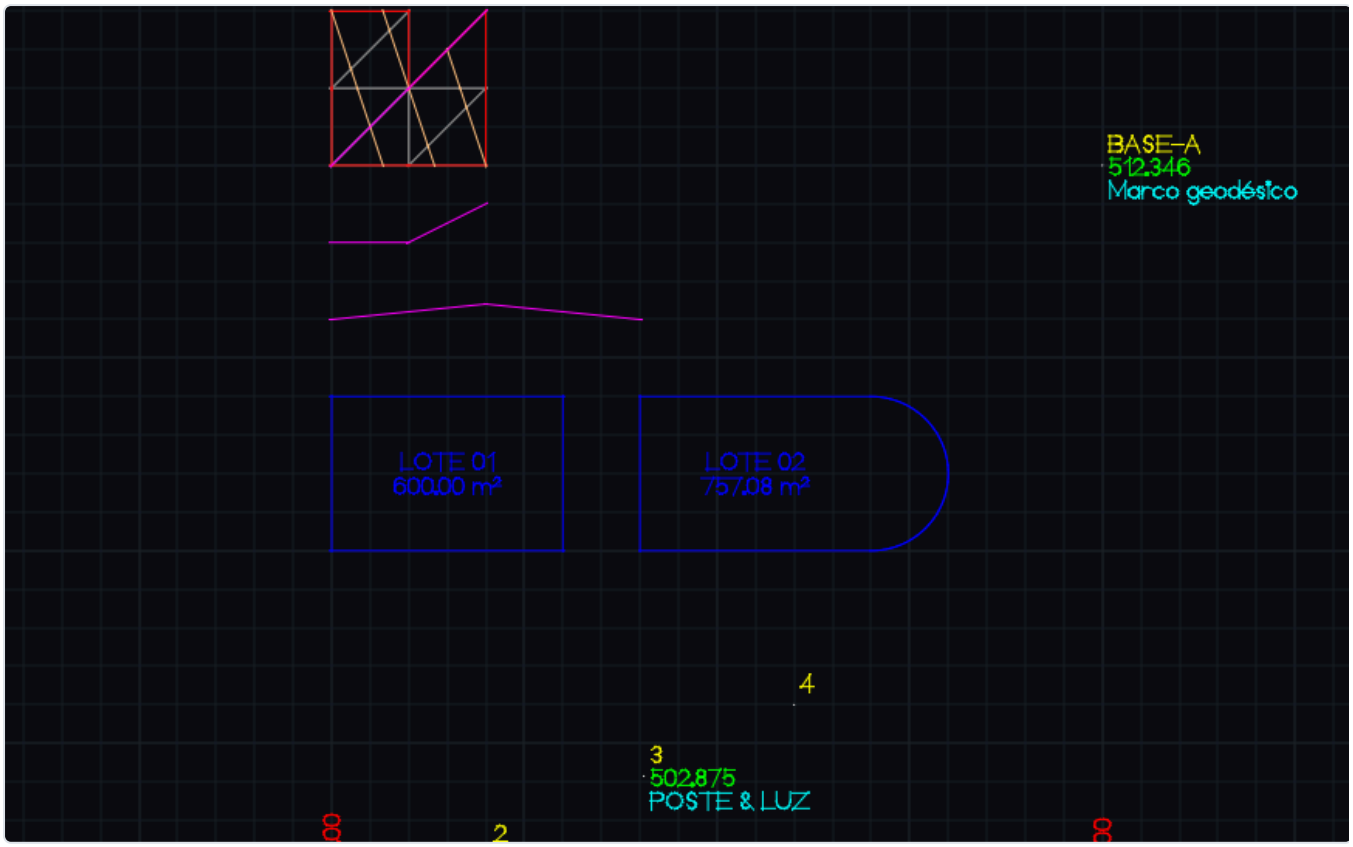


The complete file: the grid, the boundaries applied by exact clipping, the stations, the profile, the cross sections, the superelevation, the watershed, the network and the corridor.

Reading is **tolerant**: what BadBit CAD does not read yet is skipped and appears on a single line, with the count; unreadable numbers, faces with points that do not exist and references to points that do not exist are left out with a warning; a file that ends in the middle is read up to that point.

Exporting (LANDXMLOUT)

LANDXMLOUT opens the **Export LandXML** dialog with what goes into the file and the file name (next to the drawing, with its name and the `.xml` extension; **...** opens the system dialog).



The selected objects (or, with no selection, the whole model), the count by type and the unit.

- **Points**: those that came from a LandXML go back with the stored name, description, code and elevation, at their current position (if they were moved, with the new coordinates). Those drawn in BadBit CAD get the next free numbers, without elevation.
- **Surfaces**: the **border** of the surface carries the whole surface (points, triangles and breaklines) exactly as it came. Select the border to export the surface.
- **Alignments, parcels and feature lines** that came from a LandXML: if the polyline did not change, the original object goes back (with the spirals, the starting station and the profiles); if it changed, the elements come from the lines and arcs of the polyline, with the stored name, station and profiles. In an object with spirals, the first piece carries the object: if it changed, only its elements come from the drawing.
- **Spirals** (the spline of each one): if the spline **did not change**, the spiral goes back as it came, bit for bit. If it **changed** (moved, rotated, stretched by the grips), the spiral comes out **recomputed from the spline**: its start and end, the PI at the intersection of the end tangents, the length, the end radii from the change of direction along it (an end that was tangent remains tangent if the clothoid fits that way), the direction and the `clothoid` type; the command line says how many. If the spline **is no longer a clothoid** (the best clothoid is more than 1/10,000 of its length away, 1 cm on a 100 m spiral — assumed tolerance), it goes as an **irregular line** (`IrregularLine`, through the points of the spline), with a warning on the command line and in the Export LandXML dialog with the name of the object and the deviation. Only a spline whose object (the first piece) also goes counts; a spline selected on its own exports nothing.
- **Polylines and lines drawn in BadBit CAD**: feature lines (PlanFeature), with the layer name and a number.
- **Pipe networks and corridors**: the first structure (or pipe) carries the whole network and the first line of the corridor, the whole corridor, exactly as they came (converted to the drawing unit).

- **Boundaries, watersheds, cross sections, superelevation and station equations** go back with the surface and the alignment, as they came; a grid surface goes back as a TIN (its triangles).
- What LANDXMLIN generated and has no data of its own (triangles, contours, stations, profile, cross section and superelevation frames, section lines and labels) does not go into the file.

The file is valid LandXML 1.2 (the namespace and version of the public schema, the drawing units) and the numbers go with all their digits: importing and exporting again gives the same points and the same faces, bit for bit.

Performance

A LandXML with **100,000 points and a 200,000-triangle surface** (21 MB) opens without freezing the window: reading (0.2 s), the 1 m contours (0.03 s) and building the objects run in the background, with progress in the corner; the window only receives the finished drawing (measured on an Apple M2 Pro).

Not done yet

- An edited spiral always comes out as a clothoid (or an irregular line): the `bloss`, `sinusoid` and other types go back as they came only if the spline did not change. The neighboring lines and arcs do not follow the edited spiral (if its end moves, the file is left with a step).
- The superelevation normal crown is assumed (−2 %); the pivot type does not change the diagram (which shows the slopes, not the edge elevations).
- Pipe profiles and the network profile view.
- Exporting hand-drawn contours and triangles as a surface.

Images, PDF underlays, transmittal and password

BadBit CAD attaches to the drawing, as a **reference**, an **image** (photo, map, scan) or a **PDF page** (a supplier's plan, the scanned survey): the file stays outside the drawing and appears in it in **draw order** (an image attached later covers what was already there, as in AutoCAD; **DRAWORDER** changes it), like external references. The drawing keeps the name, path, position, clipping and adjustments; the pixels and the page are read from the file at each opening, in the background. **ETRANSMIT** gathers the drawing with everything it uses into a `.zip` or a folder, and **SECURITYOPTIONS** saves the DWG with a password.

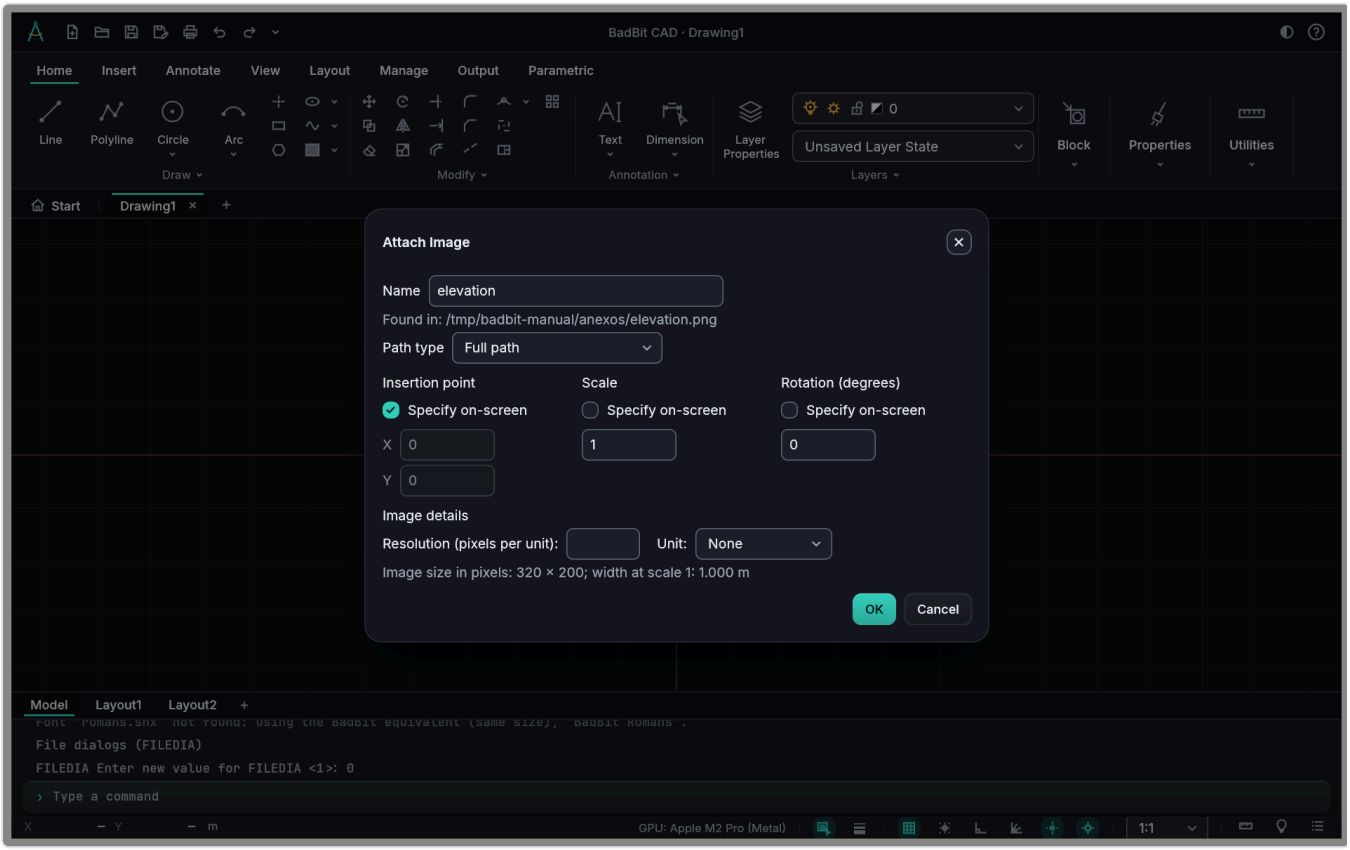
Commands

Command	What it does
IMAGEATTACH (IAT)	Attaches an image (PNG, JPG, TIFF, BMP, GIF, WebP).
IMAGE (IM)	Opens the **External References** manager, with the images and PDFs.
IMAGECLIP (ICL)	Clips the image: rectangular, polygonal, inverted; turns the clipping on and off and deletes it.
IMAGEADJUST (IAD)	Brightness, contrast and fade of the image.
TRANSPARENCY	Turns the image transparency on or off (transparent pixels let you see what is underneath).
IMAGEFRAME	The image frame: 0 hidden, 1 displayed and plotted (default), 2 displayed but not plotted.
PDFATTACH	Attaches a page of a PDF.
PDFCLIP	Clips the PDF underlay (the same options as IMAGECLIP).
PDFLAYERS	Turns the layers of the PDF underlay on and off.
PDFADJUST	Fade, contrast, monochrome and adjust to background of the PDF underlay.
PDFOSNAP	1 (default): snaps see the geometry of the PDF underlay; 0: only the frame.
PDFFRAME	The frame of PDF underlays (0, 1 or 2, like IMAGEFRAME).
PDFIMPORT	With PDF underlays in the drawing, asks `Select PDF underlay or [File] <File>:` and converts the underlay into geometry.
ETRANSMIT	Creates the transmittal package (zip or folder) with the report.
SECURITYOPTIONS	The password to open the drawing (DWG 2004, 2007, 2010, 2013 and 2018).
DRAWORDER (DR)	The draw order of any object, images too: Above objects (A), Under objects (U), Front (F), Back (B).

The command names are AutoCAD's, the same in all three languages.

Attaching an image (IMAGEATTACH)

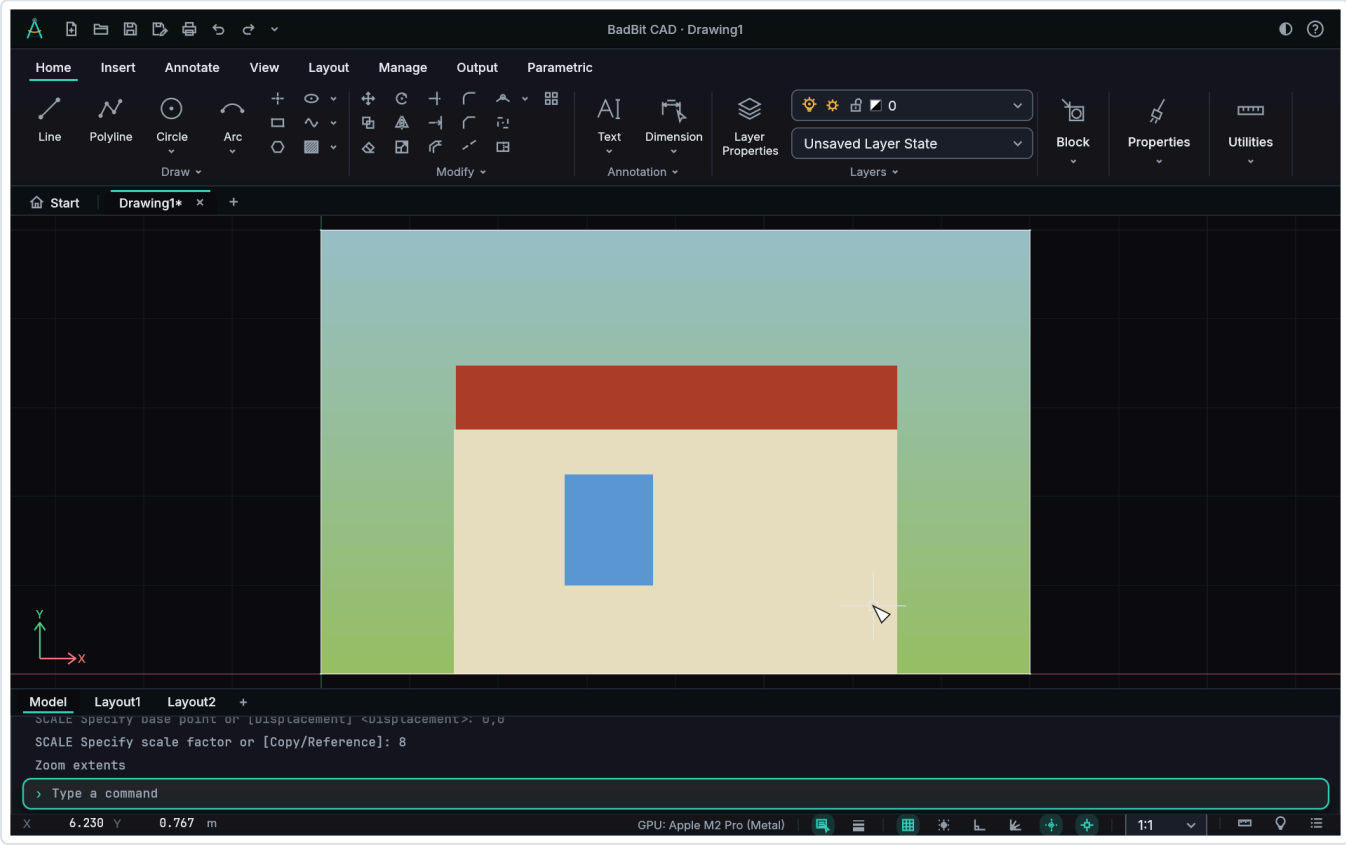
IMAGEATTACH opens the system dialog to choose the file and then the **Attach Image** dialog:



The definition name, the path type, the insertion point, the scale and the rotation (typed or on screen) and the image details.

- **Name**: the name of the definition (the file name, without the extension). Two images with the same name share the file, the state and the path; a name already used by another file is not accepted.
- **Path type**: **Full path**, **Relative path** (to the drawing folder; needs the drawing to be saved) or **No path** (only the name: the file is looked for in the drawing folder). Windows paths (`C:\projeto\foto.jpg`) are also found by name on other systems.
- **Insertion point**, **Scale** and **Rotation**: **Specify on-screen** asks on the command line (`Specify insertion point <0,0>:`, `Specify scale factor <1>:`, `Specify rotation angle <0>:`), with a click or a typed value; a clicked scale is the distance to the insertion point and the rotation, its angle.
- **Image details**: the **Resolution** (pixels per unit) and its **Unit** (None, Inch, Centimeter), already **read from the file** when it has them (the PNG `pHYs`, the JPEG JFIF and EXIF, the TIFF resolution tags), as AutoCAD does. Without resolution, the image comes in **1 unit wide** at scale 1 (as in AutoCAD); with resolution, at its real size (the DPI per inch) converted to the drawing unit. The dialog shows the size in pixels and the width at scale 1.

The insertion point is the **lower left corner** of the image. The image is read in a thread: it appears as soon as it is ready (the drawing keeps responding). Large images use **levels of detail** (each level half the size of the previous one): zoomed out, the screen uses the smallest level; zoomed in, the pixels appear as squares (without blurring), as in AutoCAD. The stored pixels have a memory cap (1.5 GB): beyond it, the pixels of the images that have been off screen the longest are dropped. The graphics card also has a cap (512 MB) and, when the image exceeds its largest texture size, the first level that fits is used.



The image underneath the drawing, with the frame.

Move, copy, rotate, scale and mirror take the image along (mirroring flips it). The grips at the four corners move the whole image. **Properties** show the name, the path, the page (for a PDF), the width, the height and the rotation.

With FILEDIA 0 (command line)

With the **FILEDIA** variable at 0, IMAGEATTACH opens no dialog and asks everything on the command line, as AutoCAD does: `Enter image file name to attach:` (the full path, or relative to the drawing folder), `Specify insertion point <0,0>:`, the scale and the rotation. The definition name is the file name and the resolution comes from the file. PDFATTACH asks `Enter path of PDF file to attach:` and, with more than one page, `Enter page number <1>:`.

Draw order (DRAWORDER)

The image sits in the **draw order** like any object: what comes after it appears on top, and what comes before is **hidden** by it (an image attached after the lines covers them, as in AutoCAD). **DRAWORDER** (DR) chooses the objects and asks `Enter draw order [Above objects (A)/Under objects`

`(U)/Front (F)/Back (B)] <Back>:`. **Back** (Enter) puts the image underneath everything; **Front**, on top of everything; **Above objects** and **Under objects** ask `Select reference objects:` and put the chosen objects just above (or just below) them. It works for images, hatches, lines and everything else, as a single undo step, and the order goes into DXF and DWG (the order of the entities in the file and AutoCAD's `SORTENTSTABLE`, which is also read when opening). An image with **TRANSPARENCY** on does not hide what is underneath.

An image **inside a block** appears with its pixels at the place of the insert (before, only the frame) and follows the same draw order inside the block: the block objects that come **before** it (and what came before the insert in the drawing) are hidden by its area, and those that come **after** appear on top, on screen and in the plot. An image whose file was not found hides nothing, at any block level (also in a block inside another block). An image seen through a **concave polygonal viewport** in a layout is clipped by its exact outline.

The missing image

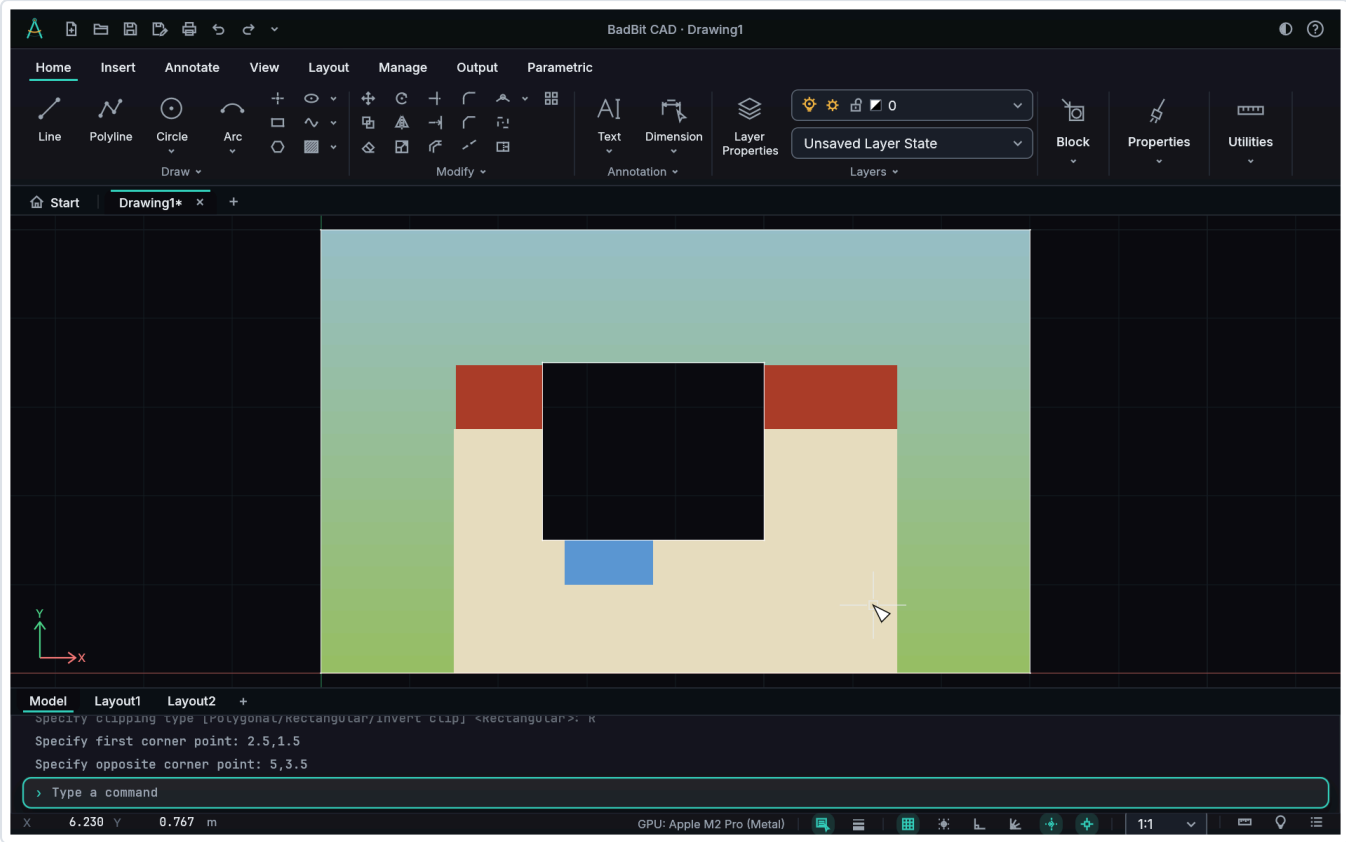
If the file is not found (or the image was **unloaded** in the manager), the **frame with the name** of the definition appears, even with IMAGEFRAME 0. A file that exists but cannot be read (unknown format, damaged file, image too large) also shows only the frame.

Clipping (IMAGECLIP and PDFCLIP)

IMAGECLIP uses the selected image (or asks `Select image to clip:`) and then:

`Enter clipping option [ON/OFF/Delete/New boundary] <New>:`

- **New boundary**: if there is one already, `Delete old boundary? [Yes/No] <Yes>:`. The command line states the mode (**Outside mode**: what is outside the boundary disappears). `Specify clipping type [Polygonal/Rectangular/Invert clip] <Rectangular>:` — **Rectangular** asks for the two corners; **Polygonal**, the points (U undoes the last one, Enter closes); **Invert clip** switches to **Inside mode** (what is inside disappears) and asks for the type again.
- **ON** and **OFF** keep the boundary; **Delete** removes the clipping.



The inverted rectangular clipping: the middle of the photo disappears and the drawing underneath shows.

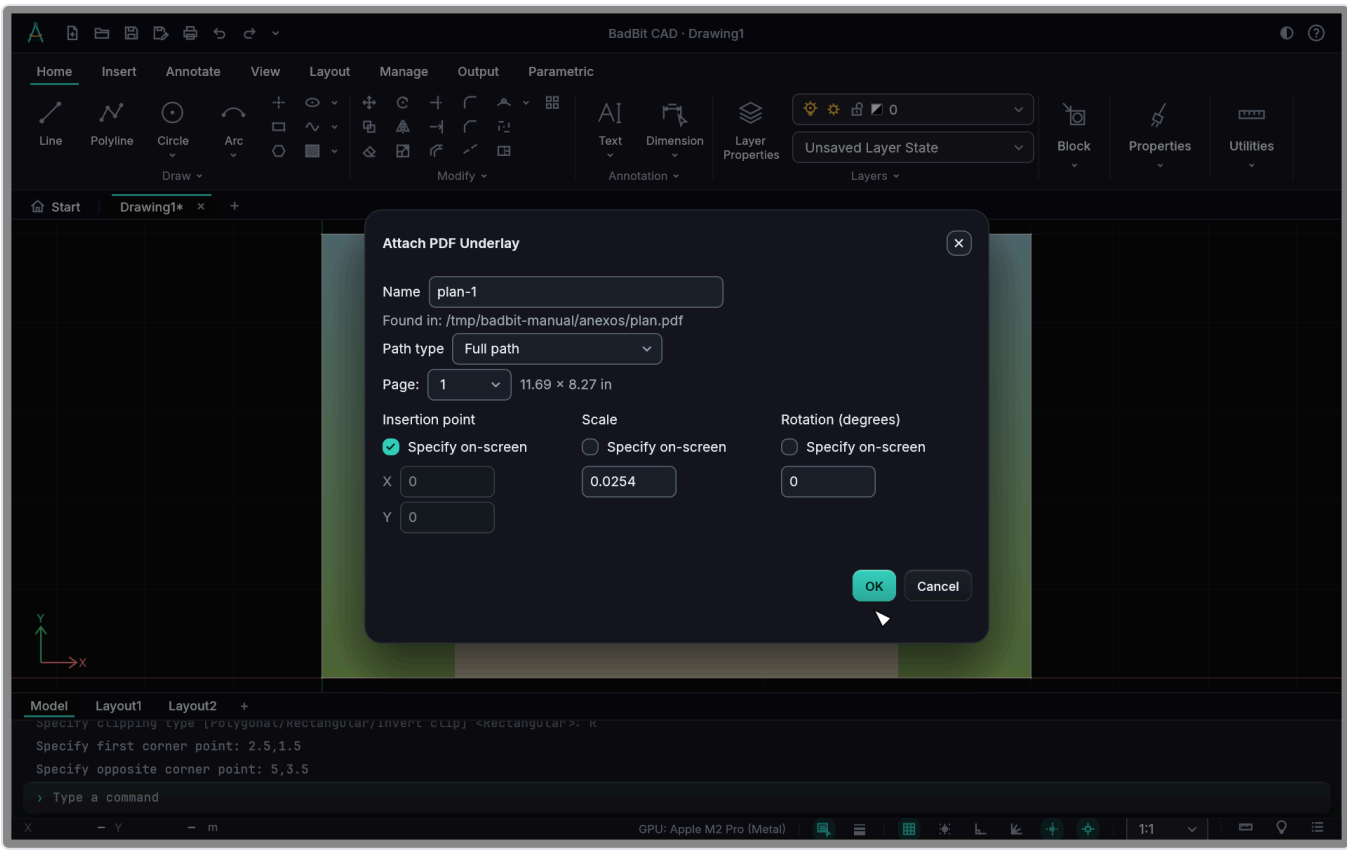
PDFCLIP is the same, for PDF underlays.

Adjustments (IMAGEADJUST, TRANSPARENCY, PDFADJUST)

- **IMAGEADJUST**: the **Image Adjust** dialog with **Brightness** (default 50), **Contrast** (default 50) and **Fade** (default 0, blends with the background color); **Reset** returns to the defaults.
- **TRANSPARENCY**: `Enter transparency mode [ON/OFF] <OFF>:`. When off, the transparent background of the image takes the screen background color (or white on the sheet).
- **PDFADJUST**: `[Fade/Contrast/Monochrome/AdjustColorForBackground] <Fade>:`. Fade goes from 0 to 100 (default 25) and contrast from 0 to 100 (default 75, which shows the PDF as it is; below that the colors move toward gray, above it they move away from it); fade and contrast apply equally to the lines and to the photos inside the PDF. Monochrome removes the color and leaves lines and photos in shades of gray by luminance, starting from black on a light background (on a dark background, inverted: the darkest becomes light). With **AdjustColorForBackground** on, on a dark background the lightness of the PDF colors is inverted, keeping the hue (black becomes white, dark blue becomes light blue, pure red stays), so that a drawing made for white paper shows up; it applies equally to the lines and to the photos inside the PDF. On a light background and in the plot (white paper) nothing changes.

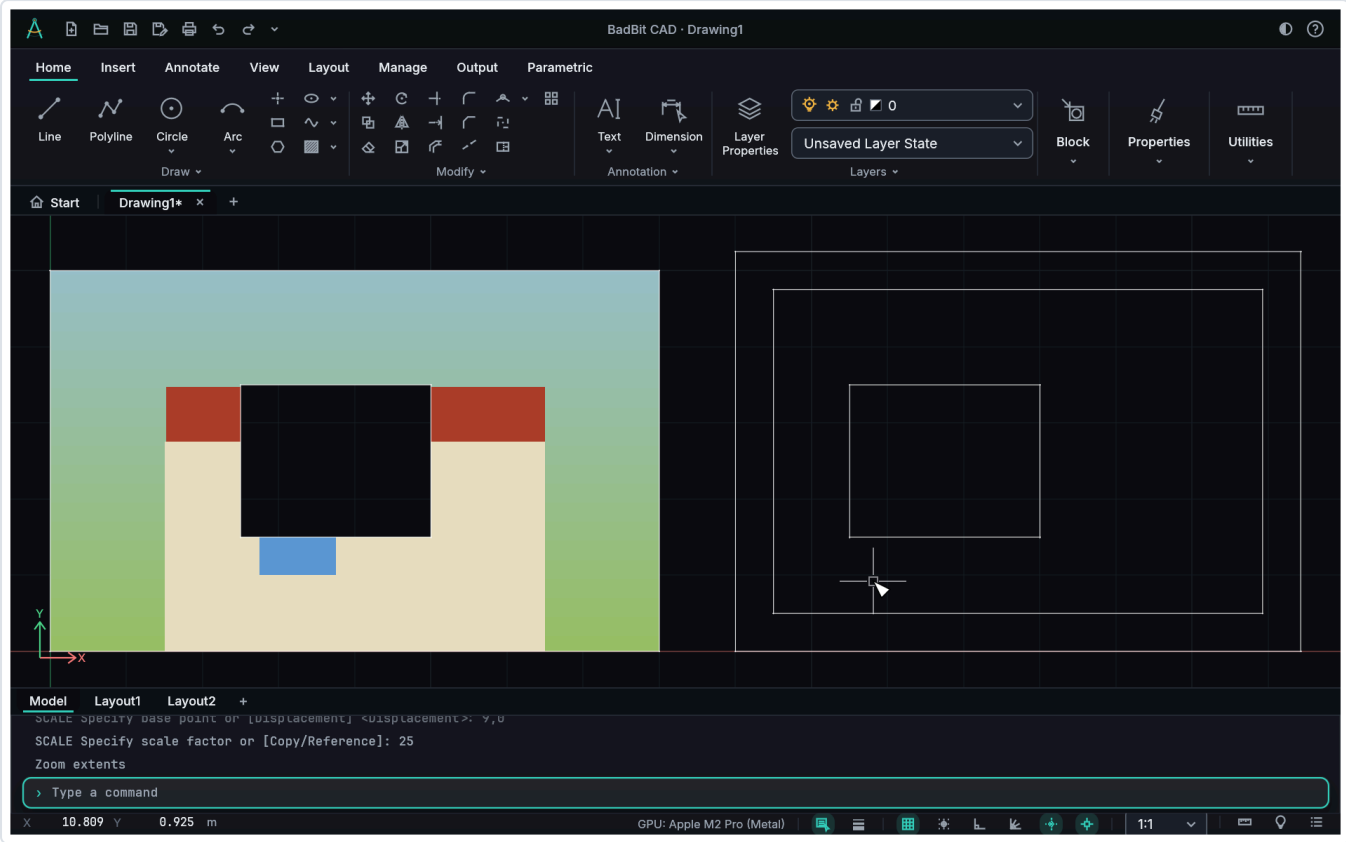
Attaching a PDF (PDFATTACH)

PDFATTACH opens the **Attach PDF Underlay** dialog: the name (the file name with the page number), the path type, the **page** (with its size in inches), the point, the scale and the rotation.



The chosen page and its size.

The default scale puts the page at **real size**: one PDF inch is 25.4 units in a drawing in millimeters and 0.0254 in a drawing in meters. The page is read by the BadBit CAD PDF reader (the same one as PDFIMPORT) in a thread and drawn as geometry, with fade and contrast; the **raster images inside the PDF** (photos, logos, the scanned plan) also appear, underneath its lines. **Fade** takes the colors toward the **background color**, as in AutoCAD (on a dark background the lines get darker; on a light one, lighter). Photos inside the PDF can be in JPEG, **JPEG 2000** (common in maps and scanned plans), **JBIG2** or **CCITT** (Group 3 and 4 fax, common in black-and-white scanned documents): all of them appear. The **inverted clipping** of PDFCLIP (or XCLIP) removes from the photos inside the PDF what is inside the boundary, as it already did with the lines. **PDFLAYERS** turns the layers of the PDF on and off (the layers turned off are kept in the reference). With **PDFOSNAP** 1, snaps (endpoint, midpoint, center, intersection...) pick the geometry of the PDF.



The PDF page next to the photo (here at scale 25 so they appear together).

Converting the PDF underlay into geometry (PDFIMPORT)

With a PDF underlay in the drawing, **PDFIMPORT** asks which one (or **File**, Enter, to choose another PDF as before). The underlay's page is converted with its scale, rotation and position, and the lines, texts and areas come into the drawing as a single undo step, on the PDF layers. The underlay remains in the drawing.

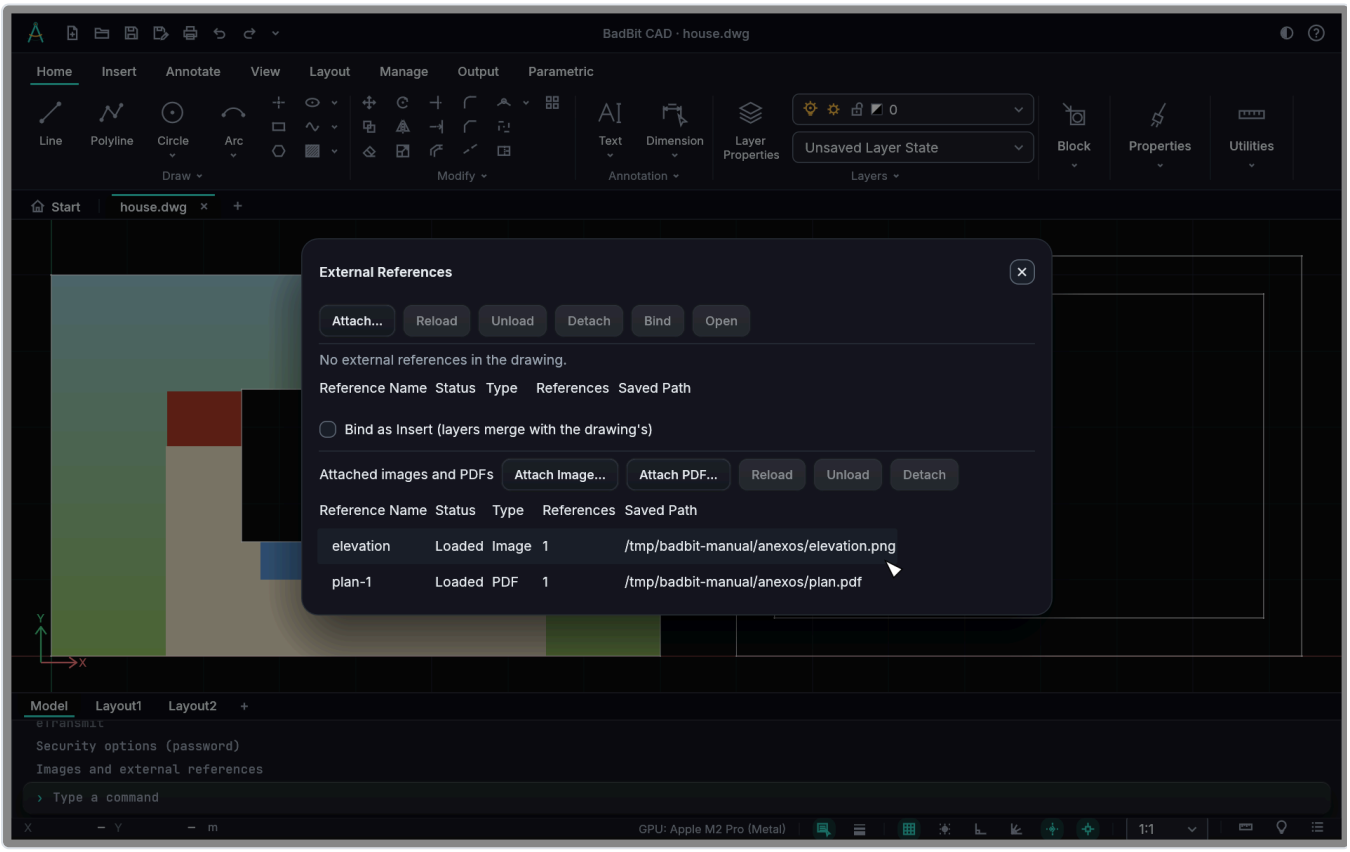
PDFIMPORT (from the underlay or from a file) also brings:

- the PDF **raster images**: each one is saved as a **PNG next to the drawing** (in the drawing folder, named `<PDF name>_imagem1.png`, `..._imagem2.png`; with no saved drawing, in the PDF folder) and comes in as an **attached IMAGE** at its place, on the `PDF_Imagens` layer (or on the PDF layer), as AutoCAD does. The command line says how many were saved. Photos in JPEG 2000, JBIG2 and fax (CCITT) come in like the others; an image the decoder cannot read (a rare feature it does not support, or a damaged file) is left out, with a warning stating the format;
- **simple shadings** (two-color gradients, linear and radial) as a **gradient hatch**;
- an **encrypted PDF** (password protected) is not read, with the message `This PDF is password protected or encrypted, and BadBit does not read encrypted PDFs. Open it in the program that created it, save an unprotected copy and try again.`

The External References manager (IMAGE)

IMAGE (or XREF) opens the **External References** manager, which also lists the **attached images and PDFs**: the name, the status (**Loaded**, **Unloaded**, **Not Found**, **Unreferenced**), the type (Image or PDF), how many references and the path. A definition whose last image was erased stays in the list as

Unreferenced (with 0 references), as in AutoCAD, and goes into DXF and DWG when saving; **Detach** removes it for good. The **Attach Image...** and **Attach PDF...** buttons attach; with one chosen, **Reload** (reads the file again), **Unload** (disappears from the screen and from memory, the frame remains), **Detach** (removes all references) and the **Saved path** (change the path of all references of the definition).



The external references at the top and the images and PDFs at the bottom.

Plotting

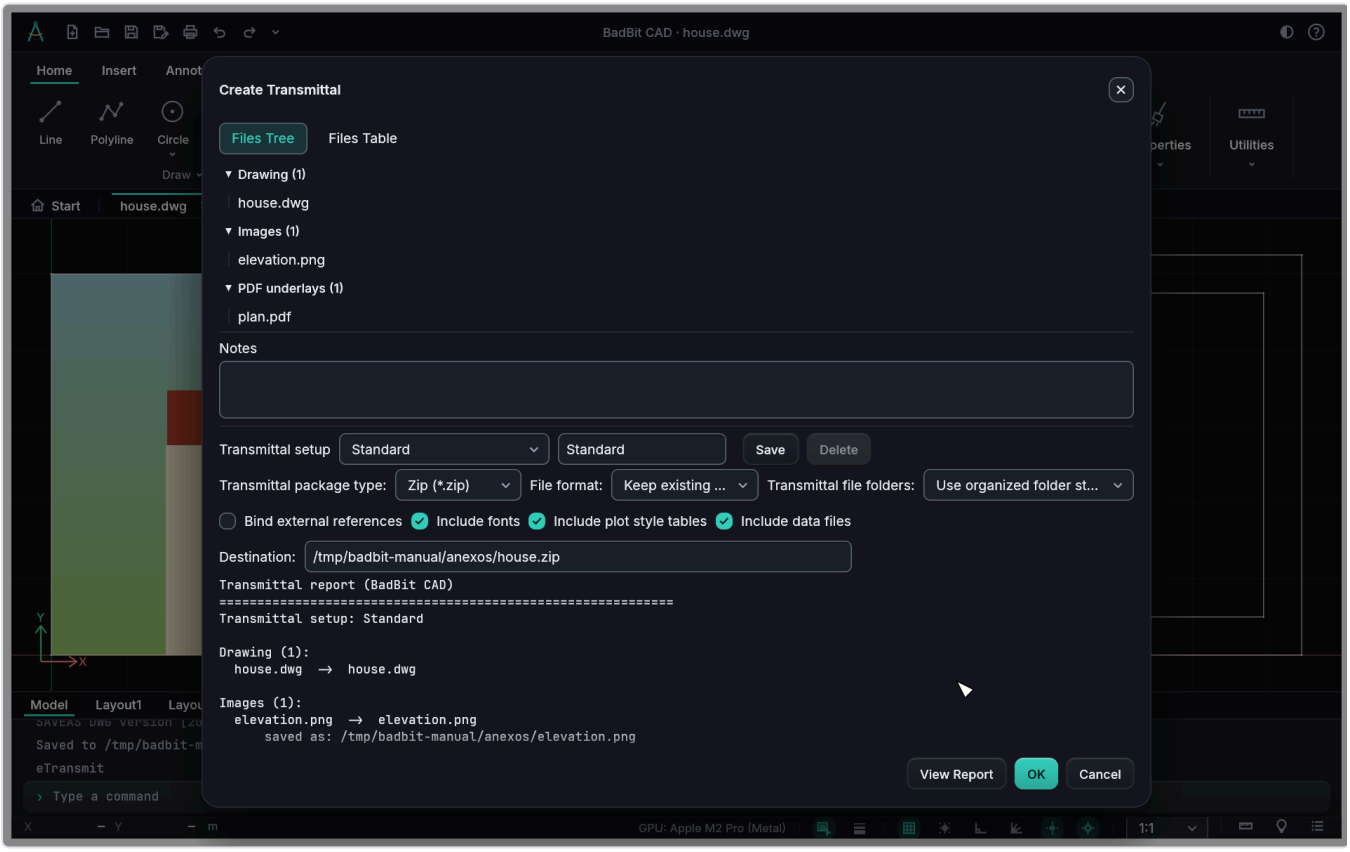
Images come out in the plot (PLOT, EXPORTPDF, EXPORTSVG, EXPORTPNG) with their clipping and adjustments: in PDF as an image (with transparency), in SVG as an embedded PNG and in PNG through the same rasterizer as the screen. The frame with IMAGEFRAME 2 does not plot. A PDF underlay plots as geometry.

DXF and DWG

The image goes into DXF and DWG like AutoCAD's: the **IMAGE** with the **IMAGEDEF** (in the `ACAD_IMAGE_DICT` dictionary), the **IMAGEDEF_REACTOR** and the **RASTERVARIABLES**; the PDF as a **PDFUNDERLAY** with the **PDFDEFINITION** (in `ACAD_PDFDEFINITIONS`). Inverted clipping before DXF/DWG 2010 and the PDF layers turned off go in BadBit CAD's XDATA (AutoCAD in those versions shows the normal clipping). The PDF page size is not in the file: when opening, BadBit CAD reads it from the PDF. **IMAGEFRAME** goes into `RASTERVARIABLES` and comes back when opening; **unreferenced definitions** come back to the dictionary; **draw order** goes into the order of the entities and into `SORTENTSTABLE`.

Transmittal package (ETRANSMIT)

ETRANSMIT (the drawing must be saved) opens the **Create Transmittal** dialog:



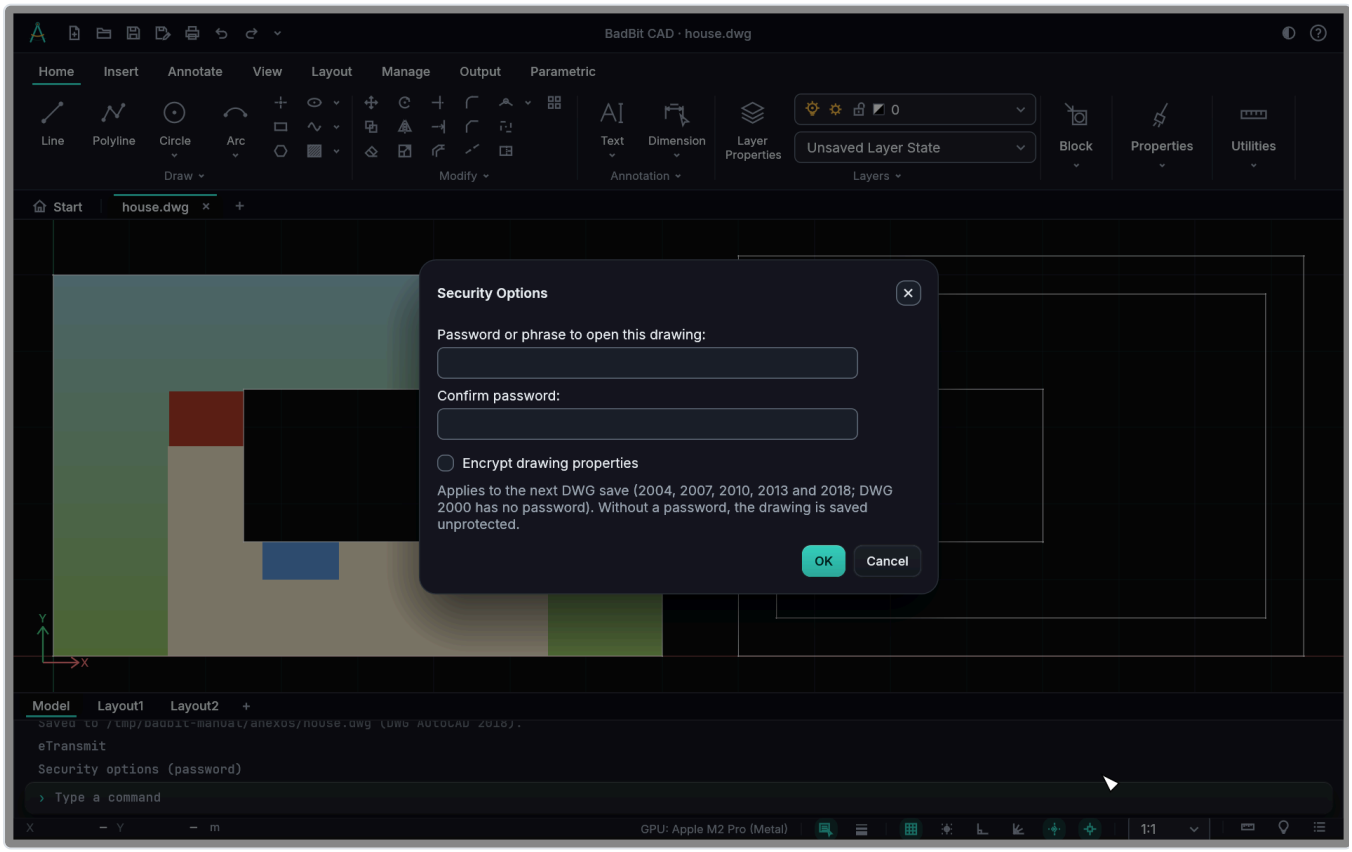
The tree of files by type, the notes, the transmittal setup and the report.

- **Files Tree** and **Files Table**: the drawing, the **external references** (and those inside them), the **images**, the **PDF underlays**, the **SHX fonts** (and big fonts) and **TrueType fonts** of the text styles, the **pen tables** (the `.ctb` / `.stb` files themselves, found in the drawing folder, in its `Plot Styles` folder or in the `Plot Styles` folders of AutoCAD on the computer; only one that is not found is regenerated from the profile), the **sheet sets** (`.dst`, when the drawing came from an AutoCAD sheet set), the **data link** spreadsheets (the `.xlsx` / `.xls` / `.csv` files of DATALINK) and the **data files** (the `.pat` files of hatch patterns that are not BadBit CAD's, in the drawing folder). What is not found is shown in red and goes into the report.
- **Notes**: go into the report.
- **Transmittal setup**: named setups are kept in the **profile** (Save, Delete). **Transmittal package type**: Zip or Folder. **File format**: **Keep existing file formats** (DWG and DXF in the source version; `.bbd` as it is) or DWG 2018/2013/2010/2007/2004/2000 or DXF 2018. **Folders**: **Use organized folder structure** (what is in the drawing folder keeps its relative path; the rest goes to `xrefs/`, `imagens/`, `fontes/`, `penas/`, `folhas/`, `dados/`) or **Place all files in one folder**. **Bind external references** (BIND: they come into the drawing and do not go as files).
- **View Report** shows the report; **OK** builds the package in a thread (the `.zip` or the folder, with the drawing saved again with the package's **relative paths** and the `.txt` report with the drawing's name). The path of the **sheet set** (`.dst`) and that of the **data link spreadsheets** stored in the drawing are also rebuilt for their place in the package (relative to the drawing), and the package drawing carries the **thumbnail** of the current view, as when saving (in `.bbd` too).

The package's `.bbd` is a new file, but it carries everything "Save As" carries: the objects, components and preferences from a newer BadBit CAD version that this version kept without displaying come back unchanged in the package.

Password (SECURITYOPTIONS)

SECURITYOPTIONS opens the **Security Options** dialog: the **password to open the drawing**, the **confirmation** and **Encrypt drawing properties** (the preview and the summary too). The password applies at the next save in **DWG 2004, 2007, 2010, 2013 and 2018**; with no password, the drawing is saved unprotected. **DWG 2000** has no password (AutoCAD 2000 did not protect drawings): it is saved without protection, with the warning `DWG 2000 has no password (AutoCAD 2000 did not protect drawings): the file will be saved without protection.` on the command line.



The password and the confirmation.

When opening a password-protected DWG, BadBit CAD asks for the password in a dialog; a wrong one shows `Invalid password. Please enter the password again.` and asks again. A drawing opened with a password keeps it when saving. The protection format follows the ODA open specification; what it does not describe is assumed (see `docs/formatos/dwg-senha.md`) and still needs to be checked in AutoCAD.

System variables

Type the name of the variable (or **SETVAR**, alias SET, and then the name); the command line shows the current value between `< >` and asks for the new one. `Enter` keeps the value; a value out of range is reported and the question is repeated. In SETVAR, `?` lists all variables with their current values.

Variable	What it controls	Values	Default
CURSORSIZE	Size of the cursor crosshairs, in % of the screen	1 to 100	5
GRIDUNIT	Grid spacing	Positive length	Follows the unit
OSMODE	Object snap modes (sum of the bits)	0 to 16383	4133
POLARANG	Polar increment, in degrees (in the profile)	Greater than 0 up to 360	90
POLARMODE	Polar mode (in the profile): 1 measurement relative to the last segment, 2 tracking along all polar angles, 4 use the additional angles, 8 acquire only with Shift	0 to 15	0
POLARADDANG	Additional polar angles, separated by semicolons (in the profile); `.` clears	Up to 10 angles from 0 to 360	(empty)
PDMODE	Point symbol	0–4, adding 32 and/or 64	0
PDSIZE	Point size	Positive (units), negative (% of the screen) or 0; comma or period as decimal separator	0
FILEDIA	File dialogs: 1 opens the system dialog; 0 asks for the name on the command line	0 or 1	1
INSUNITSDEFSOURCE	Unit of an incoming unitless file (INSERT, XREF, or opened with FILEDIA 0), by the `INSUNITS` code	0 to 24	6 (meters)
INSUNITSDEFTARGET	Unit of the unitless drawing that receives an INSERT or XREF	0 to 24	6 (meters)
SNAPUNIT	Grid snap spacing (F9)	Positive length	That of the grid (GRIDUNIT)
TEXTSIZE	Height of the next TEXT and MTEXT	Positive length	0.25 m
SAVETIME	Minutes between automatic saves (0 turns it off)	0 to 600	10
LAYLOCKFADECTL	Fading of objects on locked layers, in % (in the profile); negative turns it off while keeping the value	-90 to 90	50

Variable	What it controls	Values	Default
TEXTFILL	TrueType letters filled (1) or outline only (0), on screen, in the plot and in PDF (in the profile)	0 or 1	1
MIRRTEXT	Text in MIRROR: 0 stays readable; 1 mirrors the letters	0 or 1	0
PICKADD	Picks: 0 replace the selection (Shift adds); 1 add (Shift removes); 2 add and SELECT leaves the objects selected	0 to 2	2
PICKFIRST	Commands use what was selected before them	0 or 1	1
PICKBOX	Size of the cursor pickbox	0 to 50	4
PICKDRAG	1: dragging makes the rectangular window	0 to 2	2
PICKAUTO	4: dragging in empty space makes the lasso (without 4, the rectangular window)	0 to 7	5
SELECTIONCYCLING	List of overlapping objects on click (0 turns it off; Ctrl+W toggles)	−2 to 2	0
SELECTSIMILARMODE	What SELECTSIMILAR compares besides the type: 1 color, 2 layer, 4 linetype, 16 lineweight, 64 text style	0 to 255	130
TRIMEXTENDMODE	Mode of TRIM and EXTEND: 1 quick (every object is an edge); 0 standard (choose the edges)	0 or 1	1
EDGEMODE	Extended edges in TRIM and EXTEND (infinite line, circle of the arc)	0 or 1	0
SPLINETYPE	PEDIT's Spline: 5 quadratic, 6 cubic	5 or 6	6
SPLINESEGS	Straight segments per span in PEDIT's Spline	1 to 200	8
LTSCALE	Global linetype scale	Positive number	1
CELTSCALE	Linetype scale of new objects	Positive number	1
CELTYPE	Linetype of new objects	ByLayer, Continuous, a loaded one or one from the standard library (DASHED, HIDDEN, CENTER...; it	ByLayer

Variable	What it controls	Values	Default
		loads into the drawing by itself)	
PLINEGEN	The linetype runs along the whole new polyline (1) or restarts at each vertex (0)	0 or 1	0
LWDISPLAY	Lineweights appear on screen (the status bar button)	0 or 1	0
CELWEIGHT	Lineweight of new objects, in hundredths of a millimeter (in the drawing)	−1 (ByLayer) or 0, 5, 9 … 211	−1
LWDEFAULT	Lineweight of new layers, in hundredths of a millimeter (in the profile)	0, 5, 9 … 211	25
FILLETRAD	FILLET radius	Length	0
CHAMFERA / CHAMFERB	First and second CHAMFER distances	Length	0
CHAMFERC / CHAMFERD	Length and angle of CHAMFER by angle	Length / degrees	0
CHAMMODE	CHAMFER by distances (0) or by length and angle (1)	0 or 1	0
TRIMMODE	FILLET and CHAMFER trim the lines (1) or not (0)	0 or 1	1
OFFSETDIST	OFFSET distance (negative: Through)	Length	Through
OFFSETERASE	OFFSET erases the source	0 or 1	0
OFFSETLAYER	The offset goes to the current layer (1) or the source's layer (0)	0 or 1	0
OFFSETGAPTYPE	Outer corners of a polyline offset: 0 extends, 1 fillets, 2 chamfers	0 to 2	0
PLINEWID	Width of the new polyline	Length	0
PELLIPSE	ELLIPSE creates an ellipse (0) or a polyline (1)	0 or 1	0
SPLMETHOD	SPLINE starts by fit (0) or by control vertices (1)	0 or 1	0
SPLKNOTS	Spline knots: 0 chord, 1 square root, 2 uniform	0 to 2	0
SPLDEGREE	Degree of a spline by control vertices	1 to 5	3

Variable	What it controls	Values	Default
ARRAYASSOCIATIVITY	ARRAY creates an associative array (a single object, editable with ARRAYEDIT)	0 or 1	1
DELOBJ	Commands that create from objects erase the source ones (REGION too)	0 or 1	1
GRIPS	Show grips on selection	0 or 1	1
GRIPWELD	BadBit CAD's own: dropping an end grip on the end of another object welds the two (see Welding by grip)	0 or 1	1
GRIPSIZE	Side of the grips, in points	1 to 255	8
APERTURE	Half width of the snap box, in points	1 to 50	8
ORTHOMODE	Ortho (F8)	0 or 1	0
DYNMODE	Dynamic input (F12); 0 turns it off	−3 to 3	3
DYNPROMPT	Typed command and prompts next to the cursor (with F12 on); 0 only on the command line	0 or 1	1
QPMODE	Quick Properties	0 or 1	1
QPLOCATION	Where Quick Properties appear: 0 next to the cursor, 1 floating	0 or 1	0
CLAYER	Current layer	Layer name	0
TEXTSTYLE	Current text style (saved in the drawing)	Name of a text style in the drawing	Standard
DIMSTYLE	Current dimension style (saved in the drawing; only through SETVAR, since typed DIMSTYLE is the command)	Name of a dimension style in the drawing	Standard
DIMASSOC	How new dimensions are created: 2 associative (follow the object), 1 non-associative, 0 exploded into their parts	0 to 2	2
CANNOSCALE	Current annotation scale (saved in the drawing; also in the status bar list)	1:1, 1:50, 1:100, 2:1...	1:1
ANNOAUTOSCALE	Positive: when CANNOSCALE changes, the new scale is added to all annotative objects; 0 or negative turns it off	−4 to 4	−4

Variable	What it controls	Values	Default
ANNOALLVISIBLE	1 shows all annotative objects; 0 only those that support the annotation scale: in the model, the current one (CANNOSCALE); in each layout viewport, its own scale (as in the plot). Saved in the drawing (also in DXF and DWG)	0 or 1	1
CECOLOR	Color of new objects	BYLAYER, 1 to 255 or r,g,b	BYLAYER
OBJECTISOLATIONMODE	Objects hidden by ISOLATEOBJECTS and HIDEOBJECTS: 0 only in this session; 1 saved in the `.bbd` and hidden when opening (in the profile; DXF and DWG do not carry them)	0 or 1	0

Dimension variables

AutoCAD's dimension variables show the value of the **current dimension style** (with the overrides). Changing one of them does not change the style: it puts an **override** on the current style, and new dimensions are created with it (see Style overrides, in the Text and dimensions chapter). `-DIMSTYLE` Status lists all of them with their values; Save writes the overrides into a style. The overrides of the current style belong to the drawing, as in AutoCAD: they go into the DWG and DXF (the dimension variables of the header) and come back when the file is opened. Sizes are in drawing units; the defaults are those of Standard.

Variable	What it controls	Values	Default
DIMTXT	Text height	Positive length	0.25
DIMTXSTY	Text style of the dimensions	Text style name	Standard
DIMASZ	Arrowhead size	Length	0.25
DIMBLK, DIMBLK1, DIMBLK2	Both arrowheads, the first and the second	`.` or empty (closed filled), `_OPEN`, `_ARCHTICK`, `_DOT`, `_NONE`	Closed filled
DIMLDRBLK	Leader arrowhead (QLEADER, LEADER and DIMARC's leader)	The same names as DIMBLK	Closed filled
DIMEXO	Offset of the extension line from the origin	Length	0.0625
DIMEXE	Extension of the extension line beyond the dimension line	Length	0.125
DIMGAP	Gap between the text and the dimension line (and in the tolerance frame)	Length	0.0625
DIMDLI	Baseline spacing (DIMBASELINE, QDIM)	Length	0.375
DIMCEN	Center mark of radius and diameter	Length	0.25
DIMJOGANG	Angle of the DIMJOGGED jog, in degrees	Degrees	45
DIMSCALE	Overall scale: multiplies all sizes	Positive number	1
DIMLFAC	Measurement factor (100: meters dimensioned in centimeters)	Positive number	1
DIMLUNIT	Unit of linear dimensions: 1 scientific, 2 decimal, 3 engineering, 4 architectural, 5 fractional	1 to 5	2
DIMDEC	Decimal places (or the denominator of fractions)	0 to 8	2
DIMDSEP	Decimal separator	One character	`,`
DIMZIN	Zero suppression: 4 leading, 8 trailing (sum)	0, 4, 8 or 12	0
DIMPOST	Prefix and suffix: `<>` is the measurement (`<> m` gives `3,00 m`)	Text	Empty
DIMAUNIT	Angle unit: 0 decimal degrees, 1 degrees, minutes and seconds, 2 gradians, 3 radians	0 to 3	0
DIMADEC	Decimal places of angles	0 to 8	0

Variable	What it controls	Values	Default
DIMTAD	Text above the dimension line (1) or centered, with the line broken (0)	0 or 1	1
DIMTIH, DIMTOH	Text horizontal (inside and outside the extension lines); 0 aligned with the dimension line	0 or 1	0

Parametric drawing

Variable	What it controls	Values	Default
CCONSTRAINTFORM	Form of new dimensional constraints: 0 dynamic (on screen only), 1 annotational (prints)	0 or 1	0
CONSTRAINTBARDISPLAY	Show constraint bars	0 or 1	1
CONSTRAINTBARMODE	Types shown in the bars (sum: horizontal 1, vertical 2, perpendicular 4, parallel 8, tangent 16, smooth 32, coincident 64, concentric 128, collinear 256, symmetric 512, equal 1024, fix 2048)	0 to 4095	4095
CONSTRAINTINFER	Infer constraints when drawing and editing	0 or 1	0
CONSTRAINTNAMEFORMAT	Text of constraint dimensions: 0 name, 1 value, 2 name and expression	0 to 2	2
CONSTRAINTRELAX	Relax the constraints that prevent an edit (0: the edit does not happen)	0 or 1	0
DYNCONSTRAINTDISPLAY	Show dynamic constraints	0 or 1	1

Object groups

Variable	What it controls	Values	Default
PICKSTYLE	Group and associative hatch selection: 0 none, 1 a member brings the whole group, 2 an associative hatch brings its boundary, 3 both (kept in the profile; `Ctrl` + `Shift` + `A` turns groups on and off)	0 to 3	1

Properties and text scale

Variable	What it controls	Values	Default
SETBYLAYERMODE	Properties that SETBYLAYER changes to ByLayer (sum: 1 color, 2 linetype, 4 lineweight, 8 material, 16 plot style, 128 transparency; BadBit CAD changes the first three and keeps the others; kept in the profile)	0 to 255	255

CECOLOR (the color of COLOR) also accepts BYBLOCK and the name of one of the first seven colors (red, yellow, green, cyan, blue, magenta, white), and goes into DXF and DWG with the true color.

Center marks and centerlines

All of them belong to the drawing (they go into the file, DXF and DWG) and apply only to new marks and lines.

Variable	What it controls	Values	Default
CENTERCROSSSIZE	Size of each dash of the center mark cross	length, number with `x` (relative to the diameter) or ByLineType	0.1x
CENTERCROSSGAP	Gap between the cross and the extensions	length, number with `x` or ByLineType	0.05x
CENTEREXE	How far the extensions go past the circle or the ends of the lines	positive length	0.35 m (metric AutoCAD's 3.5 in proportion to BadBit CAD's Standard)
CENTERLAYER	Layer of new marks and lines (created if it does not exist)	name, or `.` for the current one	`.`
CENTERLTYPE	Linetype of new marks and lines	name, `.` (the current one) or ByLayer	CENTER2
CENTERLTSCALE	Linetype scale of new marks and lines	nonzero number	1
CENTERMARKEXE	Extensions on new center marks	0 hidden, 1 visible	1

User coordinate system (UCS)

Those of the drawing go into the file (`.bbd`, DXF and DWG); those of the profile stay with the user, as in the AutoCAD registry.

Variable	What it controls	Values	Default
UCSNAME	Name of the current UCS (empty: unnamed). Read-only.	text	empty
UCSORG	Origin of the current UCS, in world coordinates. Read-only.	x,y,z	0,0,0
UCSXDIR, UCSYDIR	X and Y directions of the current UCS, in world coordinates. Read-only.	x,y,z	1,0,0 and 0,1,0
UCSBASE	Base UCS for orthographic UCSs (BadBit CAD does not use it: read-only).	text	empty
UCSICON	UCS icon (drawing): bit 1 on, bit 2 at the origin.	0 to 3	3
UCSVP	Each model viewport keeps its own UCS (drawing).	0 or 1	1
UCSFOLLOW	Update the view to plan when the UCS changes (drawing; no effect in plan, which is already the top view).	0 or 1	0
UCSAXISANG	Default angle of UCS X, Y and Z (profile).	5, 10, 15, 18, 22.5, 30, 45, 90, 180	90
UCSSELECTMODE	The UCS icon can be selected (profile).	0 or 1	1
UCSORTHO	Restore the orthographic UCS with the orthographic view (profile; no effect in 2D).	0 or 1	1
UCSVIEW	A new named view saves the current UCS (profile).	0 or 1	1

Multilines

All three belong to the drawing (they go into the file, DXF and DWG) and apply to new multilines; MLINE's options also change them.

Variable	What it controls	Values	Default
CMLJUST	Justification of new multilines	0 Top, 1 Zero, 2 Bottom	0
CMLSCALE	Width scale of new multilines (negative reverses the order of the elements; 0 collapses them into one line)	number	2 (metric AutoCAD's 20 in proportion to BadBit CAD's Standard)
CMLSTYLE	Style of new multilines	name of a loaded style	STANDARD

LISP

Variable	What it controls	Values	Default
SECURELOAD	Load `.lsp` files from outside the trusted folders (profile): 0 without asking, 1 asks, 2 only from trusted ones.	0 to 2	1
TRUSTEDPATHS	Trusted folders for `.lsp` files, separated by `;` (profile; BadBit CAD's `lisp` folder always counts).	text	empty
CMDECHO	Echo of the commands of `(command ...)` (kept for routines; BadBit CAD always shows them in the history).	0 or 1	1
USERI1 to USERI5, USERR1 to USERR5, USERS1 to USERS5	Free values for routines (integers, reals and texts; per session).	—	0, 0.0 and empty

Wipeout

Variable	What it controls	Values	Default
WIPEOUTFRAME	Frame of wipeouts (saved in the drawing): 0 hidden (appears only in the selection highlight), 1 displayed and plotted, 2 displayed but not plotted.	0 to 2	1

View and layouts (read only)

They show the current view, as in AutoCAD; SETVAR shows the value and does not change it (`VIEWCTR = 210,148.5 (read only)`), and `?` also lists them.

Variable	What it shows	Values
VIEWCTR	The view center: in model space, in drawing units; on paper, in mm; inside a layout viewport (MSPACE), the model point at its center.	x,y
VIEWSIZE	The view height, in the same units as VIEWCTR (inside a layout viewport, the height of the model it shows).	number
CVPORT	The current viewport: 1 on paper; inside a layout viewport, its number (2 onward); 2 in model space.	integer
CTAB	The tab on screen: the layout name, or Model.	text

Performance and large drawings

BadBit CAD was built not to freeze with large drawings:

- Each change rebuilds only the objects touched; a LINE in a drawing with one million lines takes tenths of a millisecond.
- Layers, theme and selection highlighting are resolved without rebuilding objects.
- Circles and arcs gain detail on zoom gradually, a few milliseconds per frame.
- Selection by click and window queries only the objects near the cursor (spatial index).

Drawing on the processor (no GPU)

Without a graphics card, BadBit CAD draws on the processor: it keeps the image and redraws only what changes (when dragging the view, only the strip that comes in; in an edit, only the region touched; in continuous zoom, it shows the previous image scaled and redraws when the gesture stops). To test this mode on a computer with a GPU:

```
BADBIT_RENDER=cpu ./badbit-shell
```

The status bar shows "CPU" in place of the graphics card name.

Memory

BadBit CAD returns memory when a drawing is closed and does not grow while the app is idle:

- **Idle**, the command line cursor blinks for 30 seconds and then stays solid: with no input, BadBit CAD draws nothing (not even one frame) until the mouse or keyboard come back. Leaving the app open for hours, forgotten, does not increase memory or use the processor.
- **When a drawing is closed**, what belonged to it (the screen scene, the graphics card buffers, the spatial index, the undo history) is released in a separate thread and goes back to the system right away. On macOS, BadBit CAD turns off the system allocator's large block cache (`MallocLargeCache=0`), which held hundreds of MB for minutes after closing a large drawing: Activity Monitor shows the memory coming back right away.
- **With the drawing open**, the scene stores lines and triangles without slack once built (vector slack and the holes left by rebuilt objects are given back) and the graphics card gets only one eighth of slack on the first upload.

Measured in the automated test (with the real GPU) with a Civil 3D DWG of 1,442 objects (surfaces with contours, points and labels) on an Apple M2 Pro: open, from 560 down to 182 MB; after closing, from 459 down to 83 MB (and without growing at each open and close); empty, from 90 down to 41 MB.

Test neighborhood

The **DEMO** command loads, as a reference background, a test neighborhood with one million lines, to see performance in practice. The neighborhood is in real UTM coordinates (near X = 330,000 m and Y = 7,390,000 m) and is for viewing only: it cannot be selected or edited and is not saved in the file. It is a background of the open drawing: zoom extents (Z) frames the neighborhood together with whatever is drawn (something near 0,0 leaves the neighborhood small, in a corner), and NEW (`Ctrl` + `N`) or OPEN start without it. To see the neighborhood alone, run DEMO in a new drawing.

Command and shortcut reference

Commands

Command	Aliases	What it does
LINE	L	Straight lines, one object per segment.
PLINE	PL	Polyline (a single object).
CIRCLE	C	Circle by center and radius or diameter, 2P, 3P, Ttr and Tan, tan, tan.
ARC	A	Arc by three points or by the other methods (center, end, angle, chord length, direction, radius).
POINT	PO	Point object.
RECTANG	REC	Rectangle (Dimensions, Rotation).
POLYGON	POL	Regular polygon (Inscribed, Circumscribed, Edge).
MTEXT	T, MT	Multiline text in a rectangle, typed in the drawing, with per-run formatting from the Text Editor palette.
TEXT, DTEXT	DT	Single-line text (Justify, Style).
TEXTEDIT, DDEDIT	ED, double-click	Edits the contents of a text, in the drawing itself.
STYLE, DDSTYLE	ST	Text Style dialog: font (the stroke font or an installed TrueType font, with the font style), height, width factor, oblique angle and effects; Set Current, New, Delete, Apply.
-STYLE	-ST	The text style from the command line.
DIM	—	Smart dimension: the type comes from the object under the cursor (Angular, Baseline, Continue, Ordinate, Undo).
DIMLINEAR	DLI	Horizontal, vertical or rotated linear dimension (Mtext, Text, Angle, Horizontal, Vertical, Rotated; Enter selects the object).
DIMALIGNED	DAL	Dimension aligned with the origins (Mtext, Text, Angle).
DIMRADIUS	DRA	Radius dimension of an arc or circle.
DIMDIAMETER	DDI	Diameter dimension of an arc or circle.
DIMANGULAR	DAN	Angular dimension between two lines, of an arc, of a circle or by three points.

Command	Aliases	What it does
DIMARC	DAR	Arc length dimension (Partial, Leader).
DIMJOGGED	DJO, JOG	Jogged radius, for arcs with a distant center: the override center, the position and the jog.
DIMORDINATE	DOR	Ordinate dimension, X or Y from the origin (Xdatum, Ydatum).
DIMCONTINUE	DCO	Continued dimensions from the last one (Select, Undo).
DIMBASELINE	DBA	Baseline dimensions from the last one (Select, Undo).
DIMEDIT	DED	Edits dimensions: Home, New text, Rotate text, Oblique.
DIMTEDIT	DIMTED	Moves or rotates the dimension text (Left, Right, Center, Home, Angle).
QDIM	—	Quick dimensions from a selection (Continuous, Baseline, Ordinate, Radius, Diameter, datumPoint).
DIMBREAK	—	Breaks the dimension lines where objects cross them (Multiple, Auto, Manual, Remove).
DIMSPACE	—	Spaces or aligns parallel linear or aligned dimensions from a base one (Auto).
DIMREASSOCIATE	DRE	Links dimensions to object points (Select object; Enter skips the point).
DIMDISASSOCIATE	DDA	Detaches dimensions from objects.
DIMOVERRIDE	DOV	Dimension variable overrides on the chosen dimensions (Clear overrides).
DIMSTYLE, DDIM	D, DST	Dimension Style Manager: Set Current, New, Modify (Lines and Arrows, Text, Primary Units), Override, Delete.
-DIMSTYLE	—	The dimension style from the command line (Save, Restore, Status, Variables, Apply, ?).
MLEADER	MLD	Multileader: arrowhead, landing and text (leader Landing first, leader arrowHead first).

Command	Aliases	What it does
MLEADEREDIT	MLE	Adds leader lines to a multileader (Remove leaders).
MLEADERSTYLE	MLS	Multileader Style Manager: Set Current, New, Apply, Delete.
QLEADER	LE	Quick leader: up to three points, the width and the text lines beside it.
LEADER	LEAD	Classic leader by points (Annotation, Undo) and the text.
TOLERANCE	TOL	Geometric tolerance frame through the dialog (or typed: `POS;%%c0,05(M);;A;B;C`).
TABLE	TB	Table: the Insert Table dialog (style, empty or from a CSV, point or window, columns and rows) and the cell text typed in the drawing.
-TABLE	-TB	The table from the command line: columns, data rows and the point (Style, Width, Height).
TABLEDIT	double-click on the cell	Edits the text of a cell in the drawing; Tab, Enter and the arrow keys move between cells.
TABLEEXPORT	—	Writes the table to CSV with the values shown.
TABLESTYLE	TS	Table Style Manager: direction, Title, Header and Data; Set Current, New, Apply, Delete.
HATCH, BHATCH	H, BH	Hatch by internal points (with a preview under the cursor) or closed objects, with the Hatch Creation panel (Select objects, Remove boundaries, Properties, Origin, Islands, Inherit properties, Draw order, Annotative, Undo, Settings).
-HATCH	-H, -BH	Hatch from the command line only (also Advanced and Transparency).
GRADIENT	GD	Gradient hatch of one or two colors.
HATCHEDIT	HE, double-click	Changes pattern, scale, angle, islands, origin, transparency and draw order; Disassociate.
DRAWORDER	DR	Draw order of any object: Above objects, Under objects, Front, Back.

Command	Aliases	What it does
HATCHTOBACK	—	All hatches behind the lines.
XLINE	XL	Infinite construction line (Hor, Ver, Ang, Bisect, Offset).
RAY	—	Ray: infinite in one direction.
SPLINE	SPL	Spline by fit points or control vertices (Method, Knots, Degree, Tangency, Close).
ELLIPSE	EL	Ellipse and elliptical arc (Arc, Center, Rotation, Parameter, Included angle).
MOVE	M	Moves the selection (base point and second point, or Displacement).
COPY	CO, CP	Copies the selection: Multiple or Single, Array, Undo, Displacement.
ROTATE	RO	Rotates the selection (Copy, Reference).
SCALE	SC	Scales the selection (Copy, Reference).
MIRROR	MI	Mirrors the selection.
OFFSET	O	Parallel copies at a distance (Through, Erase, Layer).
TRIM	TR	Removes the clicked piece up to the intersections (Shift extends).
EXTEND	EX	Takes the clicked end to the next object (Shift trims).
FILLET	F	Rounds the corner between two lines (Radius, Trim, Multiple).
CHAMFER	CHA	Cuts the corner between two lines straight (Distance, Angle, Trim, Multiple).
BREAK	BR	Removes the piece between two points (`@` splits at the point).
BREAKATPOINT		Splits the object at a single point.
PEDIT	PE	Edits the polyline: Close, Open, Join with fuzz distance.
JOINENDS	—	Joins two ends with a segment, in a polyline (BadBit CAD's own).
JOIN	J	Joins lines on the same straight line and arcs of the same circle (even with a gap) and chains into a polyline; closes a polyline broken at a point.

Command	Aliases	What it does
EXPLODE	X	Breaks polylines into lines; TrueType text, into the outlines of the letters.
STRETCH	S	Moves only the points inside the crossing window.
ARRAY	AR, -AR	Copies in a grid (Rectangular) or around a center (Polar).
ERASE	E	Erases the selection.
OOPS	—	Brings back what the last ERASE removed.
LENGTHEN	LEN	Measures and changes the length of lines, arcs, open polylines, elliptical arcs and splines (Delta, Percent, Total, Dynamic).
ALIGN	AL	Aligns by pairs of points (moves, rotates and adjusts the scale).
MATCHPROP	MA, PAINTER	Copies the properties of one object to others (Settings).
OVERKILL, -OVERKILL	—	Removes duplicates, merges overlaps and optimizes polyline segments.
SPLINEDIT	SPE	Edits the spline (Close, Join, Fit data with Kink, Polyline, Edit vertex with Elevate order, Reverse).
DONUT	DO	Donut (ring of a wide polyline) or disk.
REVCLOUD	—	Revision cloud (Rectangular, Polygonal, Freehand, Object, Style, Modify).
BOUNDARY, -BOUNDARY	BO, -BO	Polyline around the clicked region, with islands.
DIVIDE	DIV	Points that divide the object into equal parts.
MEASURE	ME	Points at every given distance.
ID	—	Coordinates of a point.
DIST	DI	Distance, angle and deltas between points (Multiple points).
AREA	AA	Area and perimeter by points or object (Add, Subtract).
MEASUREGEOM	MEA	Distance, radius, angle, area, volume and quick measure, repeating.

Command	Aliases	What it does
OPTIONS	OP	Options, Profiles tab: set current, export and import user profiles, including AutoCAD's (`.pgp` , `.lin` , `.ctb` , `.stb` , `.arg`).
BLOCK	B	Creates a block from the objects (they become an insert); an existing name redefines it.
-BLOCK	-B	Creates a block removing the objects from the drawing (OOPS brings them back).
INSERT, -INSERT	I, -I	Inserts a block: name, point, X and Y scales and rotation (Scale and Rotate before the point).
REGION	REG	Closed objects and chains that close become regions.
ARRAYEDIT	—	Changes the associative array: rows, columns, items, angle, rotation and alignment.
LIST	LI, LS	Describes the selected objects.
PURGE, -PURGE	PU, -PU	Purges unused layers, linetypes, zero-length geometry and empty texts.
REGEN, REGENALL	RE, REA	Regenerates the drawing image.
UNDO	U	Undoes the last command.
REDO	—	Redoes what U undid.
NEW	`Ctrl` + `N`	New drawing in a new tab.
OPEN	`Ctrl` + `O`	Opens a `.bbd` , a `.dwg` or a `.dxf` in a new tab (or goes to its tab).
DXFIN	—	Opens a DXF (OPEN with the DXF filter first).
RECOVER	—	Opens a file reporting the result of the check, even with no defects.
AUDIT	—	Checks the drawing as it would go into the AutoCAD file.
CLOSE	—	Closes the drawing tab (asks whether to save).
—	`Ctrl` + `Tab` , `Ctrl` + `Shift` + `Tab`	Next tab and previous tab.

Command	Aliases	What it does
QSAVE, SAVE	`Ctrl` + `S`	Saves (only what changed).
SAVEAS	`Ctrl` + `Shift` + `S`	Saves with another name; with `.dwg` or `.dxf`, asks for the version (DWG 2000 to 2018).
DXFOUT	—	Saves as DXF (R12 to 2018, text or binary).
LAYOUT	LO	New, Copy, Delete, Rename and Set layouts; `?` lists them.
MVIEW	MV	Viewport: two corners, Fit, Polygonal, Object, ON/OFF, Lock.
MSPACE	MS, double-click inside	Enters the viewport (draws in the model).
PSPACE	PS, double-click outside	Goes back to paper.
VPLAYER	—	Freezes or thaws layers in the current viewport or in all of them.
PLOT, PRINT	`Ctrl` + `P`	Plot dialog: PDF, PNG or SVG, paper, area, scale, pens.
PAGESETUP	—	Page Setup Manager.
PREVIEW	PRE	Plot preview.
EXPORTPDF	—	The current sheet to a PDF.
EXPORT	EXP	The current sheet to PDF, SVG or PNG (by the extension); EXPORTSVG and EXPORTPNG directly.
PUBLISH	—	Several layouts in one multi-page PDF.
ZOOM	Z, ZE	Zoom extents.
UNITS	UN	Drawing units dialog.
OSNAP	OS	Drafting settings, on the Object snap tab.
DSETTINGS	DS, SE, DDRMODES	Drafting settings: Polar tracking and Object snap tabs.
LWEIGHT, LINEWEIGHT	LW	Lineweight settings: current (CELWEIGHT), default for new layers (LWDEFAULT), display (LWDISPLAY) and display scale.
LAYER	LA	Layer Properties Manager.

Command	Aliases	What it does
PROPERTIES	PR, PROPS, CH, MO, `Ctrl` + `1`	Properties palette.
—	`Ctrl` + `Shift` + `P`	Turns Quick Properties (QP) on and off.
FILTER, 'FILTER	FI	Selection filters: tests by type and property, Begin/End AND, OR, XOR and NOT, named filters; transparent inside a command.
QUICKPROPERTIES	—	Shows the Quick Properties of the selection (even with QPMODE 0).
CUI	—	Customizes the Quick Properties rows by object type (kept in the profile).
DWGCOMPARE	COMPARE	Compares the current drawing with another DWG, DXF or `.bbd` in its own tab, with the navigation bar.
DWGCOMPAREEXPORT	—	Saves the comparison drawing.
DWGCOMPARECLOSE	—	Closes the comparison tab.
PDFIMPORT	-PDFIMPORT	Imports a vector PDF as geometry (lines, curves, solid hatches, texts), with page, scale, rotation, position and layers.
THEME	—	Toggles between the light and dark themes.
DEMO	—	Loads the test neighborhood as a background of the open drawing (removed with NEW and OPEN).
CURSORSIZE, GRIDUNIT, OSMODE, POLARANG, POLARMODE, POLARADDANG, PDMODE, PDSIZE, FILEDIA, SAVETIME, SNAPUNIT, TEXTSIZE, MIRRTEXT, LWDISPLAY, LWDEFAULT, CELWEIGHT	—	System variables (see the dedicated chapter).
DIMASSOC, CANNOSCALE, ANNOAUTOSCALE, DIMTXT, DIMASZ, DIMDEC…	—	Dimension and annotation scale variables (see the System variables chapter).
GEOMCONSTRAINT	GCON	Geometric constraint: asks for the type and the objects (Parametric drawing chapter).
GCHORIZONTAL, GCVERTICAL, GCPERPENDICULAR, GCPARALLEL, GCTANGENT, GCSMOOTH, GCCOINCIDENT, GCCONCENTRIC,	—	Each geometric constraint directly.

Command	Aliases	What it does
GCCOLLINEAR, GCSYMMETRIC, GCEQUAL, GCFIX		
AUTOCONSTRAIN	—	Applies the constraints the geometry almost satisfies (Settings).
DIMCONSTRAINT	DCON	Dimensional constraint: asks for the type, or converts associative dimensions.
DCLINEAR, DCHORIZONTAL, DCVERTICAL, DCALIGNED, DCANGULAR, DCRADIUS, DCDIAMETER	—	Each dimensional constraint directly.
DCCONVERT	—	Converts associative dimensions into dimensional constraints.
DCFORM	—	Form of the next dimensional constraints (dynamic or annotational).
DCDISPLAY	—	Shows or hides the dynamic constraints of objects.
DELCONSTRAINT	DELCON	Removes the constraints from the chosen objects.
CONSTRAINTBAR	CBAR	Shows or hides constraint bars (of the objects or of all).
CONSTRAINTSETTINGS	CSETTINGS	Constraint settings (Geometric, Dimensional, AutoConstrain).
PARAMETERS	PAR, PARAM	Parameters Manager (dimensional constraints and user variables).
-PARAMETERS	-PAR, -PARAM	Parameters from the command line (New, Edit, Rename, Delete, List).
CCONSTRAINTFORM, CONSTRAINTBARDISPLAY, CONSTRAINTBARMODE, CONSTRAINTINFER, CONSTRAINTNAMEFORMAT, CONSTRAINTRELAX, DYNCONSTRAINTDISPLAY	—	Parametric drawing variables (see the System variables chapter).
ABOUT	—	About dialog: version, build, system, modules, Copy information and third-party licenses.
HELP	'HELP	Opens the PDF manual (also `F1`); the first time, downloads the one for the installed version.

Command	Aliases	What it does
ISOLATEOBJECTS	—	Hides all objects except the chosen ones (on screen only; also through the Isolate objects button on the status bar).
HIDEOBJECTS	—	Hides the chosen objects (on screen only).
UNISOLATEOBJECTS	—	Shows again the objects hidden by ISOLATEOBJECTS and HIDEOBJECTS.
CUTCLIP	—	Cuts to the clipboard (copies and erases, a single undo step; `Ctrl` + `X`).
CUTBASE	—	Cuts with a base point (`Ctrl` + `Shift` + `X`).
OBJECTISOLATIONMODE	—	Variable for hidden objects (see the System variables chapter).
GROUP	G	Groups the objects (Name, Description; without a name, an unnamed group).
UNGROUP	—	Ungroups a group (by object, by name or all).
GROUPEDIT	—	Adds and removes objects from a group and renames it.
LAYERSTATE	LAS	Layer States Manager: saves, restores (choosing the properties), edits, renames, deletes, imports and exports (`.las`).
LAYERSTATESAVE	—	Saves the current layers as a new layer state.
LAYMCUR	—	Makes the layer of the chosen object current.
LAYERP	—	Undoes the last change to layer settings (does not touch the objects).
LAYERPMODE	—	Turns LAYERP tracking on or off.
COLOR	COL, COLOUR, DDCOLOR	Color of new objects (CECOLOR) in the Select Color dialog; **-COLOR** (-COL) asks on the command line.
SETBYLAYER	—	Changes the color, linetype and lineweight of objects to ByLayer (Settings: SETBYLAYERMODE; asks about ByBlock and blocks).
RENAME	REN	Rename dialog: layers, styles, blocks, linetypes, layouts, UCSs, views and

Command	Aliases	What it does
		viewport configurations, with wildcards; **-RENAME** (-REN) on the command line.
SCALETEXT	—	Scales texts, MTEXTs and attributes without changing their position (base point, model or paper height, match object, scale factor).
TSCALE	—	Express Tools: scales texts by a factor or to a final height, from the justification of each one.
CENTERMARK	—	Associative center mark on each chosen circle or arc (Layer).
CENTERLINE	—	Associative centerline between two lines (Layer).
CENTERDISASSOCIATE	—	Detaches center marks and centerlines from objects.
CENTERREASSOCIATE	—	Links a center mark to another circle or arc, or the centerline to two other lines.
CENTERRESET	—	Resets the extensions of center marks and centerlines to CENTEREXE.
VIEW	V	View Manager: named views of the model and the layouts (Set Current, New, Update Layers, Edit Boundaries, Delete, Rename).
-VIEW	-V	Named views from the command line: ?, Delete, Orthographic, Restore, Save, Settings and Window.
VIEWGO	—	Restores the named view by name (goes into ZOOM Previous).
CLOSEALL	—	Closes all tabs, asking about each changed drawing (Cancel for all).
UCS	—	User coordinate system: origin, 3 points on the plane, World, Previous (10), Z, X and Y (180°), Object, View, ZAxis, Named (Restore, Save, Delete, ?) and Apply.
UCSICON	—	UCS icon: ON, OFF, All, Noorigin, ORigin, Selectable and Properties.
UCSMAN	UC, DDUCS	UCS dialog: named UCSs, orthographic UCSs of the plan view and settings (icon, UCSVP, UCSFOLLOW).

Command	Aliases	What it does
SCRIPT	SCR	Runs a `.scr` script file (the text feeds the command line; space and end of line act as Enter). Dragging a `.scr` onto the window also runs it.
RSCRIPT	—	Goes back to the start of the script (on the last line, repeats the script until Esc).
DELAY	—	Waits, in the script, for the given time in milliseconds (0 to 32767).
RESUME	—	Continues an interrupted script (Esc, Backspace or error); `'RESUME` in the middle of a command.
ALIASEDIT	—	Command alias editor: create, edit and delete user aliases (in the profile, they apply immediately); import and export `.pgp`.
APPLOAD	AP	Load/Unload Applications: loads LISP routines (`.lsp`) and keeps the Startup Suite (in the profile). ARX, .NET, VBA and `.fas`/`.vlx` are not supported.
SCRIPTEDITOR	SED	BadBit Scripts: the LISP (`.lsp`) and Python (`.py`) editor, with Load, Run, the Output with the error line and the console. BadBit's own (in AutoCAD, VLIDE).
(expression)	—	A line that starts with `(` is evaluated by LISP (the value goes to the history); `!name` shows a variable; `(defun c:NAME ...)` creates the command NAME.
MLINE	ML	Multiline with the current style, justification and scale (Justification, Scale, Style; Undo, Close).
MLSTYLE	—	Multiline Style dialog: set current, new, modify (elements, caps, fill, joints), rename, delete, load and save (`.mln`).
BURST	—	Express Tools: explodes inserts with attributes into texts with the value (layer 0 and ByBlock take those of the insert; nested ones remain inserts).
BLOCKSPALETTE	—	Blocks palette: Current Drawing, Recent, Favorites and Libraries, with thumbnails

Command	Aliases	What it does
		and the insertion options (point, scale, rotation, Repeat, Explode).
BLOCKSPALETTECLOSE	—	Closes the Blocks palette.
WIPEOUT	—	Wipeout: polygonal area in the background color that hides what comes before it in the draw order (points with Undo and Close; Polyline, with Erase polyline?; Frames: ON, OFF, Display but not plot).
TFRAMES	—	Express Tools: turns the frames of images (IMAGEFRAME) and wipeouts (WIPEOUTFRAME) on or off at once.
SCRIPTCALL	—	Runs a nested `.scr` script: at its end, the calling script continues on the next line (several levels, up to 16).

Keyboard shortcuts

Key	Effect
`F1`	Opens the manual (Help).
`F2`	Expands the command line history.
`F3`	Turns object snaps on/off.
`F7`	Turns the grid on/off.
`F8`	Turns Ortho on/off.
`F9`	Turns grid snap on/off.
`F10`	Turns Polar on/off.
`F11`	Turns object snap tracking on/off.
`F12`	Turns dynamic input off/on (it starts on).
`Tab`	In dynamic input, locks the typed value and moves to the other field (X/Y or length/angle).
`Ctrl` + `N`	New drawing.
`Ctrl` + `O`	Open.
`Ctrl` + `S`	Save.
`Ctrl` + `Shift` + `S`	Save as.
`Ctrl` + `1`	Opens and closes the properties palette.
`Ctrl` + `P`	Plot (PLOT).
`Ctrl` + `Z`	Undo (U).
`Ctrl` + `Y`	Redo.
`Enter` / space / right-click	Confirms; repeats the last command.
`Esc`	Cancels.
`Delete`	Erases the selection.
`Ctrl` + `2`	Opens and closes DesignCenter (ADCENTER).
`Ctrl` + `3`	Opens and closes the tool palettes (TOOLPALETTES).
`Ctrl` + `8`	Opens and closes QuickCalc (QUICKCALC).
`Ctrl` + `X`	Cuts the selection to the clipboard (CUTCLIP).
`Ctrl` + `Shift` + `X`	Cuts with a base point (CUTBASE).
`Ctrl` + `Shift` + `A`	Turns group selection on and off (PICKSTYLE).