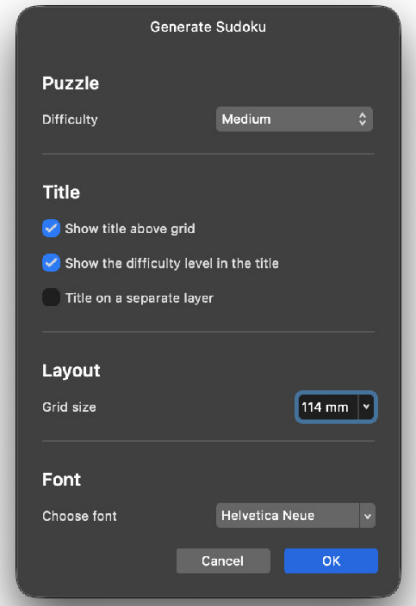
sudoku-generator

Generates valid Sudoku puzzles with unique solutions.



Sudoku · Medium

		6	7		4	8		3
							5	
		7	6	1	5			
2	5	4			7			
			3		1		4	5
	3	1		5	2			
	2			7		3		4
	6		2	4	8			1
	4			3			7	2

Installation

1. Download the zip file and unzip it.
2. Run the script in one of two ways:

a) Run directly — Open the Script Editor panel (**Window > Scripting > Script Editor**), paste the contents of `sudoku-generator.js` into the editor, and press **Run**.

b) Save to Script Library first — Follow step (a) above, then click **Save As...** to save the script to the Script Library. Open the Script Library panel (**Window > Scripting > Script Library**) and click the saved script once to run it.

Purpose

Generates a Sudoku puzzle and draws it onto the current document. If no document is open, a new A4 portrait document at 300 dpi is created automatically.

How it works

The script uses a standard backtracking algorithm to generate a valid complete grid, then digs out cells one at a time while verifying that the puzzle still has exactly one solution. This guarantees every generated puzzle is solvable and has a unique answer — not just random numbers.

Features

- **4 difficulty levels** — Easy (36 empty), Medium (46 empty), Hard (52 empty), Expert (58 empty)
- **Adjustable grid size** — from 40mm up to whatever actually fits the current page (capped at 500mm), so the field's range and its popup slider's travel always match what the grid can really reach; opens at 60% of that maximum, leaving room to grow or shrink via the slider; type a value directly or drag the slider, live
- **Optional title** — Two independent checkboxes: "Show title above grid" (`Sudoku`) and "Show the difficulty level in the title" (`<Difficulty>`). Either can be used alone; with both on, they combine into `Sudoku · <Difficulty>`
- **Title on a separate layer** — A third checkbox, greyed out unless either title checkbox above is on, that keeps the title out of the puzzle's named container — named `Sudoku title` in the Layers panel, so it's never mistaken for the puzzle's own `sudoku-<Difficulty>` group — so it can be repositioned, restyled, or deleted independently of the grid
- **Auto-document** — Creates an A4 300 dpi document if none is open
- **Grouped output** — All shapes and text are placed inside a named container (`sudoku-<Difficulty>`, e.g. `sudoku-Easy`), so the whole puzzle can be selected, moved, or deleted as one object
- **Live preview** — The puzzle redraws on the canvas as soon as the dialog opens, and again as difficulty, title, grid size, or font are changed, before OK is clicked
- **Scaled strokes** — Line weight scales with grid size so borders stay proportional

Usage tips

- Clue numbers are individually editable art-text nodes — double-click any number to change it after placement.
- Grid size should match your page layout; larger grids need more page space, especially with the title enabled.
- The script places all puzzle elements inside a named container (`sudoku-<Difficulty>`) so the whole puzzle moves as one object by default.

Known limitations

Each cell, clue, and the title is still an individually editable Affinity object inside that container. Moving or resizing cells individually (after entering the group) can break the grid alignment. Generating a puzzle takes two undo steps (container creation, then content) rather than one, because the SDK only exposes a container's node reference after it has been committed on its own — or three, with "Title on a separate layer" checked, since the title is then added as its own, separately-committed node after the grouped content.

The puzzle itself only regenerates when difficulty changes — resizing, retitling, or re-fonting redraws the same cached puzzle rather than dealing a new one, since regenerating on every tick would scramble the clue numbers while the user is still adjusting other settings.

The font picker fires no live-preview change events at all on this SDK — it shares the same bare `DialogControl` base class (no specialized value-change subclass) as the colour picker, which `docs/sdk-issues.md` BUG-018 already proved is silent in the same way. As of 1.0.9 this is worked around with a `/timers`-based poll (checks the picker's value every 350ms while the dialog is open) rather than an event handler — confirmed working in-app (see `docs/sdk-issues.md` BUG-025), which also confirms `/timers` callbacks do fire while `Dialog.runModal()` is blocking on this SDK build. Either way, the font actually applied is always read fresh at OK-time, so the committed puzzle is never wrong even if a preview tick were ever missed.

The "Grid size" field's maximum is fixed once when the dialog is built, using the page-fit size as if the title were shown (the worse-case budget), because the SDK doesn't let a dialog control's bounds change afterward. If both title checkboxes end up off, the grid could actually have fit a little larger (up to the title's own vertical allowance, 16mm) than the field will let you drag to — a minor, one-directional gap rather than a layout bug.

"Title on a separate layer" being greyed out correctly relies on a workaround: setting a dialog control's `isEnabled` live, from another control's change handler, updates the model but doesn't repaint the widget on this SDK build — confirmed by direct testing, it stayed visually enabled regardless (see `docs/sdk-issues.md` BUG-026). A hide/show cycle on the same control immediately after forces the repaint, so the dimmed/undimmed state now shows correctly.

The font poll (above) is guarded against firing after the dialog closes: `/timers`' `setInterval` wrapper calls its callback one more time even after being cancelled (cancelling a pending wait still delivers a final callback, with an error the wrapper otherwise ignores), so without a guard, cancelling the dialog right after changing the font could draw one last preview with nothing left to clear it. As of 1.0.11 the poll checks the dialog's own `isRunningModal` state before doing anything, so that trailing tick is always a no-op once the dialog has closed.

Changelog

2.0.0 - Major version bump marking a substantial redesign since 1.0.5: the dialog is now fully interactive (live on-canvas preview for every control, including grid size, title, and font, before OK is clicked), the output is grouped into a named container per difficulty, the title's content and placement are independently configurable (two checkboxes for wording, a third to put it on its own layer), and the grid-size field is bounded to what the page can actually fit with a built-in slider. See the 1.0.6-1.0.20 entries below for the full detail behind each change.

1.0.20 - Fixed "Title on a separate layer" not actually greying out: setting a checkbox's `isEnabled` live, from another checkbox's change handler, updates the model but doesn't repaint the widget on this SDK — confirmed by direct testing (both-off and each-off-separately all stayed visually enabled). Fixed with a hide/show cycle on the control immediately after, which forces the repaint — see `docs/sdk-issues.md` BUG-026.

1.0.19 - The separated title layer is now explicitly named `Sudoku title` instead of relying on Affinity's content-based auto-naming — which would otherwise name it just `Sudoku`, `Easy`, or similar, easy to confuse with the puzzle's own `sudoku-<Difficulty>` group.

1.0.18 - "Title on a separate layer" is now greyed out unless either "Show title above grid" or "Show the difficulty level in the title" is checked — it had no effect with neither on, so there was nothing to leave it enabled for.

1.0.17 - Added "Title on a separate layer" to the Title section — a third checkbox (only relevant when a title is actually shown) that, when checked, keeps the title out of the puzzle's named container on OK, leaving it as its own ungrouped node instead. Costs one more undo step (the title is committed separately, after the grouped content) when checked; unchecked (the default), behavior is unchanged from before.

1.0.16 - The grid now opens at 60% of the page-fit maximum instead of ~95% (previously nudged up to a fixed 5mm over that, both now removed). The old default left almost no visible room for the slider added in 1.0.13 to do anything meaningful in either direction; 60% leaves clear headroom to grow or shrink. (The page's shorter side was already correctly the limiting dimension for that maximum regardless of orientation — `Math.min` across both the width- and height-derived budgets already picks whichever is smaller.)

1.0.15 - Fixed the "Grid size" field (and its popup slider) allowing values far beyond what the page can actually fit — it was bounded by a constant 40-500mm regardless of the open document's size, so dragging the slider past the page-fit limit did nothing once the grid hit that ceiling, with no feedback that the rest of the slider's travel was dead. The field's own maximum is now capped to what the current page can fit (computed once at dialog-build time, since the control's bounds can't be changed afterward), so the slider's

range always matches what the grid can really reach.

1.0.14 - Fixed the title-to-grid gap not shrinking with the grid: it was clamped to a fixed 10mm minimum, which dominated the proportional scaling (`sizeMm * 0.07`) for every grid size below ~143mm — effectively locking the gap at 10mm for most practical sizes instead of shrinking with the grid. The floor is now 3mm, only relevant right at the smallest grid sizes (as a guard against the title touching the grid), so the gap scales proportionally across the whole normal range.

1.0.13 - The "Grid size" field now shows a built-in popup slider (click its icon and drag) alongside typing a value directly. Same control either way, so it stays live-previewed and in sync automatically — no separate widget to keep in sync.

1.0.12 - Fixed "Show the difficulty level in the title": it was wrongly gated behind "Show title above grid" (so it silently did nothing with the first checkbox off). The two are now independent — "Show the difficulty level" alone shows just `<Difficulty>`, "Show title above grid" alone shows just `Sudoku`, and both on combine into `Sudoku · <Difficulty>` (the bullet only ever appears when there's something on both sides of it).

1.0.11 - Fixed a leftover-preview bug: cancelling the dialog right after changing the font could leave the previewed puzzle on the canvas instead of clearing it, because `/timers`' `setInterval` wrapper still calls its callback once more after being cancelled. The font poll now checks the dialog's own `isRunningModal` state before acting, so that trailing tick is always a no-op once the dialog has closed.

1.0.10 - Added a "Title" section to the dialog with two checkboxes: "Show title above grid" (unchanged) and a new "Show the difficulty level in the title", so the difficulty can be left out of the title independently of showing the title itself. Both checkboxes are full-width, so they start flush at the dialog's left edge rather than aligned under the combo-box/ unit-editor values above. Also live-previewed.

1.0.9 - Font changes now preview live too (the font picker fires no change events on this SDK, same as the colour picker's documented BUG-018), via a `/timers` poll since `Dialog.runModal()` blocks and no event-based path exists. Confirmed working in-app — see `docs/sdk-issues.md` BUG-025.

1.0.8 - The dialog now previews the puzzle live on the canvas as difficulty, title, grid size, or font are changed — before clicking OK. The puzzle itself only regenerates when difficulty changes (not on every size/title/font tick), so the preview doesn't scramble the clue numbers while resizing.

1.0.7 - The dialog's "Grid size" field now pre-fills 5mm larger than the previous auto-calculated default.

1.0.6 - Generated puzzles are now grouped into a named container (`sudoku-<Difficulty>`, e.g. `sudoku-Easy`) so the whole puzzle can be selected, moved, or deleted as one object. Costs a second undo step (container creation, then content) — the SDK only exposes a new container's node reference after it has been

committed on its own, so this can't be collapsed into the previous single-undo-step commit.

1.0.5 - The font picker now opens pre-selected with the auto-detected default (bold/semibold Helvetica Neue or similar) instead of appearing blank, via the picker's `setFont()`. Still fully overridable before clicking OK.

1.0.4 - Added an optional font picker to the dialog. Leave it blank to keep auto-detecting a font as before; auto-detect and a family-only pick (no specific style chosen) now bias toward bold/semibold instead of regular, since regular rendered too light for clue numbers. Picking an exact style in the picker (e.g. "Helvetica Neue Bold") still honors that exact choice.

1.0.3 - Fixed "Font creation failed: INVALID_ARGS" — `Font.create()` always throws this error in Affinity's SDK, on every system, regardless of which fonts are installed; it was never a missing-font problem. Clue numbers and the title are now resolved via `FontFamily` lookup instead, so the preferred font is actually applied rather than silently falling back to the default. Also fixed a second, independent bug directly behind it that fully blocked the script: positioning the clue numbers/title relied on reading back newly-added nodes before the add command had actually been executed, which this SDK doesn't support. Clue numbers and the title are now measured via a throwaway preview pass and created already centred in the single final command — puzzle generation still commits as one undo step.

1.0.2 - CONFIG object; single undo; try/catch error handling; font error logged; toReversed; modern JS

1.0.1 - Stroke weight now scales with grid size

1.0.0 - Initial version