DetentShell — technical reference

Version 1.0.0

Complete feature, menu, and control documentation for DetentShell, the PowerShell 7 scripting workbench from Detent Point.

---

1. Window orientation

DetentShell opens the way Windows PowerShell ISE does. You see a menu bar, an icon toolbar, a Functions navigator strip, a tab area, and a console with a status line at the bottom. The script pane stays hidden at launch, which is the ISE 5.x default. It appears when you create or open a script: File ▸ New Tab, File ▸ Open, or View ▸ Show Script Pane.

The window has two levels of tabs. Session tabs are the outer level — "PowerShell 1", then "PowerShell 2 (Windows PowerShell 5.1)" — and each one owns a PowerShell runspace and a console. Script tabs are the inner level — "Untitled-1.ps1" and friends — and they're editors that share their session's runspace. The *Sessions and tabs* entry in the A–Z covers what sharing means in practice.

2. UI map

File menu

Item	Gesture	Action
New Tab	Ctrl+N	New script tab in the current session (first use reveals the script pane)
New PowerShell Session	Ctrl+T	New isolated PS7 session tab (own runspace + console)
New Remote PowerShell Tab…	Ctrl+Shift+R	Enter-PSSession-style tab over WinRM (credential dialog)
New Windows PowerShell 5.1 Session	—	Live out-of-process 5.1 session tab; greyed if 5.1 absent
Start PowerShell 7	Ctrl+Shift+P	Launches the discovered external pwsh 7 console
Start Windows PowerShell 5.1	—	Launches external Windows PowerShell; honest error if absent
Open…	Ctrl+O	Open a script file into a tab
Save / Save As…	Ctrl+S / —	Save the active script tab
Close Tab / Close All Tabs	Ctrl+W / —	Close script tabs, save-prompting when dirty
Recent Files ▸	—	MRU list (count set in Settings); hidden while empty

Exit	—	Close the app (save prompts per dirty tab)

Edit menu

Undo (Ctrl+Z) · Redo (Ctrl+Y) · Cut/Copy/Paste · Select All (Ctrl+A) · Find (Ctrl+F) · Find Next (F3) · Find Previous (Shift+F3) · Replace (Ctrl+H) · Go To Line (Ctrl+G) · Go to Match (Ctrl+]) · Comment/Uncomment Selection (Ctrl+/) · Format Document (Shift+Alt+F) · Start IntelliSense (Ctrl+Space) · Start Snippets (Ctrl+J) · Toggle Outlining Expansion (Ctrl+M) · Quick Fix…

View menu

Show Script Pane · Show ToolBar · Show Outlining (Regions) · Show Line Numbers · Go to Script Pane (Ctrl+I) · Go To Console (Ctrl+D) · Session Variables · Call Stack · Watch / Locals · Pester · Show Command Add-on · Clear Console — the five panel items are checkable toggles.

Tools menu

Generate Pester Tests · Run as Administrator · Edit Profile · Settings

Debug menu

Start Debugging · Continue (F5) · Step Over (F10) · Step Into (F11) · Step Out (Shift+F11) · Break All (Ctrl+B) · Stop Debugging (Shift+F5) · Toggle Breakpoint (F9) · Remove All Breakpoints (Ctrl+Shift+F9) · Enable All Breakpoints · Disable All Breakpoints · List Breakpoints (Ctrl+Shift+L) · Display Call Stack (Ctrl+Shift+D). Items grey out when they can't act.

Help menu

User Guide (Help Viewer) · User Guide (PDF) · Online Documentation · Help on Symbol (F1) · Keyboard Shortcuts · About

Editor context menu (right-click)

Comment/Uncomment Selection · Toggle Breakpoint · Generate Pester Tests · Quick Fix… · Help on Symbol. The right-click moves the caret to the click point first, so every item acts on the line under the mouse.

Toolbar (left to right)

New Tab · Open Script · Save Script │ Cut · Copy · Paste │ Clear Console (squeegee) │ Undo · Redo │ Run Script · Run Selection · Stop · Continue │ Comment or Uncomment Selection · Format Document │ Show Outlining (toggle) · Settings (toggle) │ "Run in:" engine picker │ engine badge. The order mirrors the Windows PowerShell ISE strip. Every control announces its name to assistive technology, and hovering shows the tooltip with the gesture, even while a button is disabled.

Panels

Session Variables · Call Stack · Watch/Locals · Pester (Generate → Run → Save, coverage) · Settings · Help viewer · Show Command Add-on. The status line shows status text, an elevation

badge when you're running as Administrator, and a Cancel button while a test generation is in flight.

---

3. A–Z reference

About — Open it from Help ▸ About. It shows the product, the publisher (Detent Point), and the PowerShell runtimes the app discovered: the PS7 path and version, and Windows PowerShell 5.1 when it's present. When 5.1 isn't installed, the dialog omits that line.

Accessibility — Every interactive control exposes a UI Automation name, so a screen reader announces "Run Script" instead of "button". The editor publishes its caret line and column, selection, fold counts, and zoom through its automation status. The breakpoint gutter reports its full set, including which breakpoints are disabled, and exposes each dot as a toggleable element. The console's entire text is readable through the Value pattern. An automated UI gate walks this same tree on every release, the whole app is keyboard-operable, and every theme is contrast-tested against WCAG AA, High Contrast included.

ANSI color — The console renders color codes as colored text. This covers the color PowerShell's formatter produces, like `Get-ChildItem`'s file listings, and any escape sequences you write in your own strings. When you copy from the console, the app strips the codes and your clipboard gets plain text. Inside your scripts, `Out-String`, `Out-File`, and `>` behave the way they do in stock pwsh: captured and redirected text comes back clean. See *Parity*.

AST analysis — The app statically analyzes the active buffer as you type, using PowerShell AST visitors, and feeds the results to the diagnostics margin. You can turn it off with Settings ▸ "Enable AST analysis."

Auto-save — Set the interval with Settings ▸ "Auto-save interval (minutes, 0 = disabled)".

Break All — Debug ▸ Break All (Ctrl+B) stops a running debugged pipeline at its next statement by putting the debugger into step mode. The item is available only while you're debugging and the script isn't already paused.

Breakpoints — Breakpoints are file-and-line pairs on saved scripts. Set or clear one by clicking the gutter, pressing F9 at the caret, using Debug ▸ Toggle Breakpoint, or the editor context menu. The Debug menu manages the set: Remove All (Ctrl+Shift+F9), Enable All, Disable All, and List Breakpoints (Ctrl+Shift+L), which prints the set to the console. An unsaved buffer can't hold a breakpoint, because there's no file line to bind to — starting a debug run on one prompts you to save first. The gutter reports its set to UI Automation, including which lines are disabled.

Call Stack — Open it from View ▸ Call Stack, or Debug ▸ Display Call Stack (Ctrl+Shift+D). At a stop it lists the frames innermost first, with script locations and line numbers. Selecting a frame moves the execution pointer to it and re-scopes the Watch panel's evaluator to that frame.

Clear Console — Use the View menu, the toolbar squeegee button, or type `Clear-Host` or `cls` in the console. Each session clears only its own pane.

Close Tab / Close All Tabs — File menu, or Ctrl+W. A dirty tab prompts with Save, Don't Save, or Cancel. Cancel aborts the close, and during Close All it stops the sweep at that tab.

Comment/Uncomment Selection — Press Ctrl+/, or use the Edit menu, the context menu, or the `#` toolbar button. It toggles `#` on the selected lines, or on the caret's line when nothing is selected.

Compatibility badge — The colored badge at the toolbar's right end. Its color grades the active script's PowerShell 7 readiness: green runs on PS7, red needs Windows PowerShell 5.1, and yellow sits between the two. Its text names the engine the next run will actually use — "PS 7.x" or "WinPS 5.1" — following the engine picker and, under Auto, the classifier's routing. Inside a 5.1 session it reads WinPS 5.1, because every run there is 5.1. The badge is display-only; the *engine picker* decides what runs.

Console — Each session's interactive PowerShell surface. It shares the session's runspace with script runs, so variables persist between an F5 and the prompt. Enter submits, Shift+Enter inserts a newline, Up and Down recall history, Esc clears the prompt line, and Tab completes. You can select and copy text, and the copy is stripped of color codes. Output streams render distinctly — errors, warnings, verbose. While a run is active or the debugger is stopped, a submission is refused with a message rather than queued.

Continue — Press F5 at a breakpoint, use Debug ▸ Continue, or click the toolbar Continue button. The paused run resumes to completion or to the next breakpoint. F5 is run-or-continue: it starts a run when the session is idle and continues when it's paused.

Cut / Copy / Paste — Standard editing, routed to the focused editor. Use the Edit menu, the toolbar, or Ctrl+X, Ctrl+C, and Ctrl+V.

Duplicate-file warning — Settings ▸ "Warn me when I edit duplicate files" warns you when the same file is open in more than one tab.

Edit Profile — Tools menu. It opens `$PROFILE` (CurrentUserCurrentHost) in a tab, creating the file first when it doesn't exist.

Elevation — Tools ▸ Run as Administrator relaunches the app elevated through UAC and carries your open documents to the new instance. When you're already elevated, it does nothing except say so in the status line. An "Administrator" badge sits on the status line the whole time you're elevated.

Engine picker ("Run in:") — The toolbar dropdown that targets each tab's next run: Auto lets the classifier choose per the compatibility bucket, and the two fixed choices pin PowerShell 7 or Windows PowerShell 5.1. The pin is per tab. The picker is disabled inside a live 5.1 session, where every run is 5.1 by definition. Under Auto, a non-Green script prompts once before it runs on 5.1 — see *Migration gate*.

Exit — File menu. Exiting follows the same save-prompt discipline as closing tabs.

Find / Find Next / Find Previous — Ctrl+F opens the in-editor search panel with incremental highlighting. Enter, F3, and Shift+F3 step through the matches, and the Edit-menu items mirror the gestures.

Folding (Outlining) — The tokenizer computes region and brace folds. Triangles in the fold margin collapse and expand them, and Ctrl+M toggles the expansion state. View ▸ Show Outlining (Regions) and the toolbar toggle turn the feature on and off, and Settings holds the default. The editor's automation status reports the fold totals.

Format Document — Press Shift+Alt+F, or use the Edit menu or the toolbar button. Formatting runs PSScriptAnalyzer's `Invoke-Formatter` on a dedicated background runspace, never on a tab's session. Settings offers two modes: "Indentation only (keep my brace style)" and "Full format (braces + aligned assignments)". If PSScriptAnalyzer is missing, the command degrades to a message with the install hint, and a failed format returns your buffer untouched.

Function navigator — The "Functions:" dropdown above the tabs. It's parsed from the active buffer's AST, and selecting an entry jumps the caret to that function.

Generate Pester Tests — Tools menu or the editor context menu. Generation targets the function at the caret; a single-function file targets automatically, and an ambiguous file raises a picker. The result is a Pester v5 test file in a new tab, with a banner reporting the testability score and the count of `FILL IN` placeholders. The generator writes assertions only for what it can derive from your source, and marks everything else as a skipped stub for you to fill — the banner's "Next placeholder" button walks you through them. See *Pester panel* for running, saving, and coverage.

Go To Line — Ctrl+G. The dialog clamps to the document length.

Go to Match — Ctrl+]. The caret jumps between matching braces.

Help on Symbol (F1) — Renders help for the command under the caret in the Help panel. PowerShell 7 ships thin auto-generated help until `Update-Help` runs, so DetentShell detects that stub and offers a one-time "Run Update-Help" nudge alongside the online-documentation link. After the download, the panel re-renders with the full content.

IntelliSense — Press Ctrl+Space, or let `.` and `-` trigger it automatically. The lookups are debounced, run per tab, and are powered by the runspace's own `CommandCompletion`, so the suggestions know your loaded modules and variables. Settings covers the per-pane "Enter selects" behavior, the provider timeout, and enable/disable.

Keyboard Shortcuts — Help ▸ Keyboard Shortcuts lists every gesture. See §4.

Line numbers — Toggle them with View ▸ Show Line Numbers or in Settings. The menu toggle applies through the same validated Apply pipeline as the panel.

Migration gate — Under the Auto engine, a Yellow- or Red-classified script prompts before running. Yes routes this run to Windows PowerShell 5.1, No pins PowerShell 7, and Cancel runs nothing. The answer is remembered while the script stays unchanged; editing it re-arms the gate.

New Remote PowerShell Tab — Ctrl+Shift+R opens a WinRM session tab. You supply a computer name and credentials in a dialog, and the password stays a SecureString from the first keystroke on. The tab title carries "user@computer" so the remoteness stays visible.

New Tab / New PowerShell Session / New Windows PowerShell 5.1 Session — See *Sessions and tabs*. The 5.1 session is a persistent, streaming, out-of-process Windows

PowerShell, and its menu item greys out when 5.1 isn't installed.

Open / Recent Files — Ctrl+O opens a file; Recent Files keeps the number of entries you configure in Settings. Opening positions the caret at the top of the file.

Parity — The engine-behavior promise: scripts behave the way they do in stock pwsh. Captured formatting (`Format-List | Out-String`) returns clean text, `>` redirects write clean files, `$PSScriptRoot` resolves when you F5 a saved file (the file runs dot-sourced by path), and script variables persist to the console afterward. A suite of more than ninety engine-behavior checks runs against both the app and stock pwsh on every release, and the two tallies have to match.

Pester panel — View ▸ Pester hosts the Generate → Run → Save workflow for generated tests. "Run these tests" executes the generated buffer with your installed Pester (v5 or later) and reports the tally with pending stubs counted as pending, never as passes or failures. "Save test" writes to the conventional `Tests\Unit\<Function>.Tests.ps1` location. "Refresh coverage" audits which functions in the project root have tests, and "Generate for all missing" batches generation for the uncovered ones. If Pester is missing, the panel says so and names the install command.

Progress — `Write-Progress` renders as one updating "Activity: Status (pct%)" line per activity, and the line resolves when the activity completes. You never get a flood of raw progress records.

Prompts (Read-Host and friends) — `Read-Host` (plain and `-AsSecureString`), `Get-Credential`, mandatory-parameter prompts, and confirm/choice prompts all surface as real dialogs. The pipeline waits, the window stays responsive, and canceling the dialog cleanly cancels the wait. Interactive flows like `Install-Module` work instead of hanging.

Quick Fix — Edit menu or the context menu. Quick Fix dry-runs five hardening transforms from ModuleWorkshop.Transforms — try/catch wrapping, strict mode, parameter validation, dangerous-command guards, and secure error messages — against the caret's enclosing function, with strict mode considering the whole buffer. The list shows each fix as applicable or declined, and every refusal states its reason in a sentence, like "This script explicitly turns strict mode off; that looks deliberate, so this won't override it." Preview shows before and after side by side, and your buffer changes only when you click Apply. Refusals that can be overridden carry a "do it anyway" row. The feature needs the ModuleWorkshop.Transforms module and reports itself unavailable without it.

Run as Administrator — See *Elevation*.

Run Script (F5) / Run Selection (F8) — F5 runs the whole script. A saved, clean file runs by path (see *Parity*), and a dirty buffer runs the on-screen text. F8 runs just the selection. Output streams into the session's console, and the session's variables are live at the prompt afterward. Settings has an optional "prompt to save before running."

Save / Save As — Ctrl+S, or the File menu. The tab title tracks the file name, and a `*` marks unsaved changes.

Sessions and tabs — The ISE model. Script tabs inside one session share that session's runspace, so a variable set in tab A is visible in tab B and at the prompt. That sharing is by design. Sessions are isolated from each other, each with its own runspace, console, and script

tabs. Ctrl+Tab and Ctrl+Shift+Tab cycle the script tabs. Settings caps the tab count and the concurrent runspaces, and hitting a cap refuses with an actionable message instead of degrading.

Settings — Tools ▸ Settings or the toolbar gear. See §5. Apply validates before it commits, so invalid values block with a message. Changes take effect live and persist.

Show Command Add-on — View menu. The visual cmdlet parameter builder: pick a command, fill in its parameters in a form, and insert or run the composed line. The panel is non-modal and bound to the active session.

Snippets — Ctrl+J or Edit ▸ Start Snippets opens a picker of PowerShell constructs and inserts the chosen body at the caret. Settings ▸ "Use default snippets" gates the built-in set.

Start Debugging — Debug menu. The saved file runs under the debugger: with breakpoints set, it runs to the first one; with none, it breaks on entry so you can step. F5 with an enabled breakpoint does the same thing (see *Continue*). Starting a debug run on a dirty buffer prompts you to save, because breakpoints bind to file lines and the file on disk must match what you see.

Start-Job — The in-process PS7 engine can't spawn `Start-Job`'s worker process. The bundled `Start-ThreadJob` is the supported path, and calling `Start-Job` fails with exactly that guidance instead of a raw error. Live 5.1 sessions run `Start-Job` natively.

Status line — The window's bottom edge. It carries operational messages ("Applied: …", "Run cancelled."), the elevation badge, and the Cancel button while a Pester generation runs.

Step Into / Step Out / Step Over — F11, Shift+F11, and F10 at a stop. The Call Stack's innermost frame tracks the current statement as you step.

Stop (run) — The toolbar Stop button ends the active run promptly, stopping the pipeline without blocking the window. The session returns to idle and accepts the next command, with its variables intact.

Stop Debugging — Shift+F5 ends the debug session. The session's runspace persists.

Syntax highlighting — The PowerShell 7 grammar with theme-matched colors per token class: comments, strings, keywords, variables, cmdlets, parameters, numbers, operators, and types, plus dedicated classes for dbatools commands and fleet parameters. Every color is contrast-validated per theme.

Themes — Settings ▸ Theme. The catalog covers light themes (Sage is the default), dark themes, and High Contrast. Switching recolors the whole chrome live — editor, console, menus, toolbar, tabs, panels, status bar — and every text-and-background pair is WCAG AA tested. Menu popups stay light with pinned dark text in every theme.

Toolbar — View ▸ Show ToolBar toggles it. §2 lists the buttons in order. Every button is keyboard-reachable and named for assistive technology.

Undo / Redo — Ctrl+Z and Ctrl+Y, or the toolbar buttons. The editor keeps its full history.

Session Variables panel — View ▸ Session Variables shows the active session's variables, refreshed after each run.

Watch / Locals — View ▸ Watch / Locals. At a stop, the panel evaluates watch expressions in the stopped frame through the debugger channel, never through the busy prompt, and re-evaluates on each step. Selecting a frame in the Call Stack re-scopes it.

Window title — `$Host.UI.RawUI.WindowTitle = '…'` retitles the DetentShell window.

Zoom — Ctrl+mouse wheel, or Ctrl+= and Ctrl+-, with Ctrl+0 to reset. The size clamps between 6 and 48 points, and the current zoom is exposed to UI Automation with the caret status.

---

4. Keyboard shortcuts

Gesture	Action
Ctrl+N / Ctrl+T	New script tab / new session
Ctrl+Shift+R	New remote PowerShell tab
Ctrl+Shift+P	Start external PowerShell 7
Ctrl+O / Ctrl+S / Ctrl+W	Open / Save / Close tab
Ctrl+Z / Ctrl+Y	Undo / Redo
Ctrl+X / Ctrl+C / Ctrl+V / Ctrl+A	Cut / Copy / Paste / Select All
Ctrl+F / F3 / Shift+F3 / Ctrl+H	Find / next / previous / Replace
Ctrl+G / Ctrl+]	Go To Line / Go to matching brace
Ctrl+/	Comment/Uncomment selection
Shift+Alt+F	Format document
Ctrl+Space	IntelliSense
Ctrl+J	Snippets
Ctrl+M	Toggle outlining expansion
Ctrl+I / Ctrl+D	Focus script pane / console
Ctrl+Tab / Ctrl+Shift+Tab	Next / previous script tab
F5	Run script — or continue when paused at a breakpoint
F8	Run selection
F9 / Ctrl+Shift+F9	Toggle breakpoint / remove all
F10 / F11 / Shift+F11	Step over / into / out
Ctrl+B	Break all
Shift+F5	Stop debugging
Ctrl+Shift+L / Ctrl+Shift+D	List breakpoints / display call stack
F1	Help on the symbol at the caret
Ctrl+wheel, Ctrl+=, Ctrl+-, Ctrl+0	Zoom in/out/reset

Esc (console)	Clear the prompt line
Enter / Shift+Enter (console)	Submit / newline

5. Settings reference

Setting	What it does
Theme	Picks the color theme for the whole window
Default PowerShell version	Seeds each new tab's engine picker: Auto, PS7, or 5.1
Execution timeout (seconds)	Ends runs that exceed the limit; 0 means no limit
Max concurrent runspaces	Caps how many sessions can exist at once
Max open script tabs	Caps the script tabs per session
Show line numbers	Shows or hides the editor's line-number margin
Enable AST analysis	Turns the live diagnostics margin on or off
Show outlining in the script pane	Default for the fold margin
Warn me when I edit duplicate files	Warns when one file is open in two tabs
Prompt to save scripts before running them	Asks before an F5 on a dirty buffer
Format / Indent button applies	Indentation-only, or full format with brace placement
Enter selects in console pane / script pane	Whether Enter accepts the IntelliSense selection, per pane
IntelliSense timeout (seconds)	How long a completion lookup may take before it's dropped
Use default snippets	Gates the built-in snippet set
Auto-save interval (minutes)	Saves dirty tabs on the interval; 0 disables
Number of recent files to show	The Recent Files list length

Apply validates before committing, and Cancel reverts the form.

6. Index

Administrator → *Elevation* · Analysis → *AST analysis* · Braces → *Go to Match*, *Folding* · Break → *Break All*, *Breakpoints* · Clear screen → *Clear Console* · Color → *ANSI color*, *Syntax highlighting*, *Themes* · Completion → *IntelliSense* · Credentials → *Prompts*, *New Remote PowerShell Tab* · Dark mode → *Themes* · Debugger → *Start Debugging*, *Breakpoints*, *Continue*, *Step…*, *Call Stack*, *Watch / Locals* · Dollar- profile → *Edit Profile* · Elevated → *Elevation* · Engine → *Engine picker*, *Compatibility badge*, *Migration gate* · Errors (stream) → *Console* · Find → *Find* · Fonts → *Zoom* · Get-Credential → *Prompts* · Help → *Help on Symbol* · High contrast → *Themes*, *Accessibility* · History → *Console* · Indent → *Format Document* · Jobs → *Start-Job* · Narrator → *Accessibility* · Outlining → *Folding* · PSScriptAnalyzer → *Format Document* · Read-Host → *Prompts* · Regions → *Folding* · REPL → *Console* · Remote → *New Remote PowerShell Tab* · Runspace → *Sessions and tabs* · Screen reader → *Accessibility* · Search → *Find* · Stubs → *Generate Pester Tests* · Tabs → *Sessions and tabs* · Testing → *Generate Pester Tests*, *Pester panel* ·

Update-Help → *Help on Symbol* · Variables → *Session Variables panel*, *Watch / Locals* · WinRM → *New Remote PowerShell Tab* · 5.1 → *New Windows PowerShell 5.1 Session*, *Engine picker*