

Sidebar

Programming · VS Code Accessibility Cheat Sheet

Settings and shortcuts that make VS Code easier to see, hear and use by keyboard: screen reader support, sounds and announcements, zoom and contrast, and more. Useful whether or not you use assistive technology, for example audio cues or a calmer, larger display. Sheet 15 in the VS Code series.

SCREEN READERS & HELP

- `editor.accessibilitySupport` `auto` (default: detect a screen reader), `on` (always optimise for one), `off`
- `Alt+F1` Open Accessibility Help for whatever has focus: lists the shortcuts and settings that apply there
- `Alt+F2` Open the Accessible View: the current output as plain text you can read line by line
- `Ctrl+M` Toggle whether Tab moves focus out of the editor (Ctrl+Shift+M on macOS)
- `F8 / Shift+F8` Go to the next / previous problem
- `accessibility.accessibleView.closeOnKeyPress` Close the Accessible View when you press a key (default `true`)
- `accessibility.hideAccessibleView` Hide the Accessible View's own on-screen panel (default `false`)
- `accessibility.windowTitleOptimized` Shorter window titles that are quicker to read aloud (default `true`)
- `accessibility.debugWatchVariableAnnouncements` Announce changes to watched variables while debugging (default `true`)

VERBOSITY HINTS (ACCESSIBILITY.VERBOSITY.*)

- `accessibility.verbosity.area` Each is `true` by default: a screen reader says how to open Accessibility Help when you enter that area
- Areas `terminal` , `find` , `hover` , `notification` , `diffEditor` , `debug` , `comments` , `sourceControl` , `notebook` , `replEditor` , `keybindingsEditor` , `walkthrough` , `emptyEditorHint`
- Chat areas `panelChat` , `inlineChat` , `inlineCompletions` , `terminalChatOutput`
- Turn one off e.g. `"accessibility.verbosity.hover": false` once you know the shortcut

SOUNDS & ANNOUNCEMENTS (ACCESSIBILITY.SIGNALS.*)

<code>accessibility.signals.name</code>	An object with <code>sound</code> and <code>announcement</code> , e.g. <code>{ "sound": "on", "announcement": "off" }</code>
<code>sound</code> values	<code>auto</code> (sound only when a screen reader is attached; the default), <code>on</code> , <code>off</code>
<code>announcement</code> values	<code>auto</code> (default), <code>on</code> , <code>off</code> : a spoken message for screen readers
<code>lineHasError</code> / <code>lineHasWarning</code>	Signal when the cursor line has an error / warning (announcement off by default)
<code>positionHasError</code> / <code>positionHasWarning</code>	Signal when the cursor is exactly on one (announcement on by default)
<code>lineHasBreakpoint</code> , <code>lineHasFoldedArea</code> , <code>lineHasInlineSuggestion</code>	Signals for breakpoints, folded regions (sound off by default) and inline suggestions (sound off by default)
<code>taskCompleted</code> , <code>taskFailed</code> , <code>onDebugBreak</code>	Signals for task results and for stopping at a breakpoint
<code>terminalCommandSucceeded</code> / <code>terminalCommandFailed</code> / <code>terminalBell</code>	Signals from the terminal
<code>diffLineInserted</code> / <code>Deleted</code> / <code>Modified</code>	Sounds while moving through a diff
<code>save</code> , <code>format</code>	<code>sound</code> is <code>userGesture</code> , <code>always</code> or <code>never</code> (default <code>never</code>)
<code>accessibility.signalOptions.volume</code>	Volume of signal sounds, 0 to 100 (default 70)
<code>accessibility.signalOptions.debouncePositionChanges</code>	Wait before reporting position-based signals (default <code>false</code>)

CURSOR, MOTION & HOVER

<code>editor.cursorStyle</code> / <code>editor.cursorWidth</code>	A <code>block</code> or thicker cursor is easier to find
<code>editor.cursorBlinking</code>	<code>solid</code> stops the blinking
<code>editor.cursorSmoothCaretAnimation</code>	<code>off</code> (default); animated cursor movement
<code>workbench.reduceMotion</code>	<code>auto</code> (default, follows the OS), <code>on</code> , <code>off</code>
<code>workbench.reduceTransparency</code>	<code>off</code> (default), <code>on</code> , <code>auto</code>
<code>editor.hover.enabled</code>	<code>on</code> , <code>off</code> , <code>onKeyboardModifier</code> : pop-up information under the mouse
<code>editor.hover.delay</code> / <code>editor.hover.sticky</code>	How long to wait before a hover shows / whether you can move the mouse into it
<code>workbench.hover.delay</code>	Delay for interface tooltips in ms (default 500 on Windows and Linux, 1500 on macOS)

SEEING MORE EASILY

<code>window.zoomLevel</code>	Whole-window zoom, <code>0</code> normal (default); Ctrl+= and Ctrl+- change it
<code>editor.fontSize</code> / <code>editor.lineHeight</code>	Make the code text larger or more spaced
<code>editor.mouseWheelZoom</code>	Ctrl + mouse wheel zooms the editor text (default <code>false</code>)
<code>workbench.colorTheme</code>	Pick <code>"Dark High Contrast"</code> or <code>"Light High Contrast"</code> , or use Ctrl+K Ctrl+T
<code>window.autoDetectHighContrast</code>	Switch to a high-contrast theme when the OS is in high-contrast mode
<code>accessibility.underlineLinks</code>	Underline links in the interface (default <code>false</code>)
<code>accessibility.dimUnfocused.enabled</code>	Dim editors and views that do not have focus (default <code>false</code>)
<code>accessibility.dimUnfocused.opacity</code>	How dim: 0.2 to 1 (default 0.75)
<code>terminal.integrated.minimumContrastRatio</code>	Adjusts terminal text colours until they reach the contrast (default 4.5)

KEYBOARD NAVIGATION

<code>F6</code> / <code>Shift+F6</code>	Move focus to the next / previous part of the window (side bar, editor, panel)
<code>Ctrl+1</code> / <code>Ctrl+2</code> / <code>Ctrl+3</code>	Focus the first / second / third editor group
<code>Ctrl+Shift+P</code>	Command Palette: anything can be done from here without a mouse
<code>Ctrl+G</code> / <code>Ctrl+Shift+O</code>	Go to a line / a symbol in the file
<code>Ctrl+Shift+E</code> / <code>F</code> / <code>G</code> / <code>D</code> / <code>X</code>	Focus Explorer / Search / Source Control Run and Debug / Extensions
<code>Ctrl+`</code>	Focus the terminal
<code>Ctrl+K</code> <code>Ctrl+S</code>	Open the Keyboard Shortcuts editor to change any binding

TERMINAL ACCESSIBILITY

<code>terminal.integrated.accessibleViewPreserveCursorPosition</code>	<code>false</code> (default), <code>true</code> , <code>always</code> : keep your place when reopening the Accessible View
<code>terminal.integrated.accessibleViewFocusOnCommandExecution</code>	Move focus to the Accessible View when a command runs (default <code>false</code>)
<code>terminal.integrated.minimumContrastRatio</code>	Contrast adjustment for terminal text (default 4.5)
<code>Alt+F2 in the terminal</code>	Read the terminal output in the Accessible View

VOICE & CHAT ACCESSIBILITY

<code>accessibility.voice.speechLanguage</code>	Language for speech recognition (default <code>auto</code>); needs the VS Code Speech extension
<code>accessibility.voice.speechTimeout</code>	Milliseconds of silence before dictation stops
<code>accessibility.voice.autoSynthesize</code>	<code>off</code> (default) or <code>on</code> ; read chat answers aloud after you spoke
<code>accessibility.voice.ignoreCodeBlocks</code>	Skip code blocks when reading answers aloud (default <code>false</code>)
<code>accessibility.verboseChatProgressUpdates</code>	More detailed progress messages during chat requests (default <code>true</code>)
<code>accessibility.chat.showCheckmarks</code>	Show checkmarks in chat (default <code>false</code>)
<code>accessibility.openChatEditedFiles</code>	Open files edited by chat automatically (default <code>false</code>)
<code>accessibility.replEditor.autoFocusReplExecution</code>	After a REPL run, focus: <code>input</code> (default), <code>lastExecution</code> <code>none</code>

A REASONABLE STARTER SET (MY SUGGESTIONS, NOT DEFAULTS)

<code>"window.zoomLevel": 1</code>	A little larger everywhere; adjust to taste
<code>"editor.cursorBlinking": "solid"</code>	A steady cursor is easier to track
<code>"editor.cursorStyle": "block"</code>	A wider cursor stands out more
<code>"accessibility.underlineLinks": true</code>	Links do not depend on colour alone
<code>"accessibility.dimUnfocused.enabled": true</code>	Makes the focused area obvious
<code>"workbench.reduceMotion": "on"</code>	Fewer animations
<code>"accessibility.signals.taskCompleted": { "sound": "on" }</code>	Hear when a task finishes without a screen reader

WANT THE SOUNDS WITHOUT A SCREEN READER? SET SOUND TO ON

Most `accessibility.signals.*` settings default to `sound: auto`, which plays the sound only when a screen reader is attached. To hear them anyway, set the individual signal to `{ "sound": "on" }`, for example for task completion, terminal command failure or breakpoints. Use `accessibility.signalOptions.volume` to set the volume.

ALT+F1 SHOWS WHAT IS AVAILABLE RIGHT HERE

Accessibility Help is context sensitive: pressed in the editor it lists editor shortcuts and settings, in the terminal it lists terminal ones, and so on. It is the quickest way to discover features you did not know about, and the `accessibility.verbosity.*` settings control whether a screen reader reminds you of it.