

Sidebar

Programming · VS Code Tasks Cheat Sheet

Running builds, scripts and checks from inside VS Code: the commands, the structure of `tasks.json`, and the `task.*` settings. Exact ID, what it does, and its default. Sheet 10 in the VS Code series.

RUNNING TASKS

<code>Ctrl+Shift+B</code>	Run the default build task
<code>Tasks: Run Task</code>	Command Palette: pick any task to run
<code>Tasks: Run Test Task</code>	Run the default test task
<code>Tasks: Configure Default Build Task</code>	Choose which task <code>Ctrl+Shift+B</code> runs
<code>Tasks: Configure Task</code>	Create or edit an entry in <code>tasks.json</code>
<code>Tasks: Terminate Task</code>	Stop a running task
<code>Tasks: Rerun Last Task</code>	Run the last task again
<code>Tasks: Restart Running Task</code>	Stop and start a running task
<code>Tasks: Show Running Tasks</code>	List the tasks that are running now
<code>Tasks: Open User Tasks</code>	Edit tasks that apply to every project

TASKS.JSON BASICS

<code>.vscode/tasks.json</code>	The file in your project that holds the tasks
<code>"version": "2.0.0"</code>	The version field at the top
<code>label</code>	The task's name in the list (also how other tasks refer to it)
<code>type</code>	<code>shell</code> (runs in your shell) or <code>process</code> (runs the program directly)
<code>command / args</code>	The program to run and its arguments (an array)
<code>detail</code>	Extra text shown under the label in the task list
<code>group</code>	<code>{ "kind": "build", "isDefault": true }</code> ; kind is <code>build</code> or <code>test</code>
<code>options</code>	<code>cwd</code> (working folder), <code>env</code> (variables), <code>shell</code> (which shell)
<code>windows / linux / osx</code>	Platform blocks that override <code>command</code> , <code>args</code> and so on

PRESENTATION (HOW THE TERMINAL LOOKS)

<code>presentation.reveal</code>	<code>always</code> , <code>silent</code> , <code>never</code> : show the terminal panel
<code>presentation.focus</code>	Move the cursor into the terminal (default <code>false</code>)
<code>presentation.echo</code>	Echo the command being run (default <code>true</code>)
<code>presentation.panel</code>	<code>shared</code> , <code>dedicated</code> , <code>new</code> : reuse a terminal or open another
<code>presentation.clear</code>	Clear the terminal before the task runs
<code>presentation.close</code>	Close the terminal when the task ends
<code>presentation.showReuseMessage</code>	Show the 'Terminal will be reused' message
<code>presentation.revealProblems</code>	<code>never</code> , <code>onProblem</code> , <code>always</code> : open the Problems panel
<code>presentation.group</code>	Name a group to show tasks as split terminals

PROBLEM MATCHERS & WATCH TASKS

<code>problemMatcher</code>	Turns output lines into entries in the Problems panel; <code>[]</code> means none
<code>\$tsc / \$tsc-watch</code>	TypeScript compiler, normal / watch mode
<code>\$eslint-stylish / \$eslint-compact</code>	ESLint output formats
<code>\$jshint / \$go / \$mscompile</code>	JSHint, Go and Microsoft compiler output
<code>\$lessc / \$node-sass</code>	Less and Sass compilers
<code>isBackground</code>	<code>true</code> for tasks that never end, such as watch or dev servers
<code>background.activeOnStart</code>	Treat the task as active from the start
<code>background.beginsPattern / endsPattern</code>	Regular expressions that mark when a watch run starts and fi

DEPENDENCIES & RUN OPTIONS

<code>dependsOn</code>	Labels of tasks to run first, e.g. <code>["Build", "Lint"]</code>
<code>dependsOrder</code>	<code>parallel</code> (default) or <code>sequence</code>
<code>runOptions.runOn</code>	<code>default</code> or <code>folderOpen</code> : start automatically when the folder opens
<code>runOptions.reevaluateOnRerun</code>	Re-resolve variables when the task is rerun
<code>runOptions.instanceLimit</code>	How many copies may run at once
<code>runOptions.instancePolicy</code>	<code>prompt</code> , <code>silent</code> , <code>terminateNewest</code> , <code>terminateOldest</code> , <code>warn</code>

TASK SETTINGS (TASK.*)

<code>task.saveBeforeRun</code>	Save files first: <code>always</code> (default), <code>never</code> , <code>prompt</code>
<code>task.allowAutomaticTasks</code>	Allow <code>folderOpen</code> tasks to start by themselves: <code>off</code> (default) or <code>on</code>
<code>task.autoDetect</code>	Find tasks from npm scripts, Gulp and so on: <code>on</code> (default) or <code>off</code>
<code>task.quickOpen.history</code>	Recently run tasks kept at the top of the list (default 30)
<code>task.quickOpen.detail</code>	Show the detail text in the list (default <code>true</code>)
<code>task.quickOpen.skip</code>	Skip the list when only one task exists (default <code>false</code>)
<code>task.quickOpen.showAll</code>	Show every task instead of only recent ones (default <code>false</code>)
<code>task.reconnection</code>	Reconnect to running tasks after a reload (default <code>true</code>)
<code>task.problemMatchers.neverPrompt</code>	Never ask to pick a problem matcher (default <code>false</code>)
<code>task.slowProviderWarning</code>	Warn when finding tasks is slow (default <code>true</code>)
<code>task.verboseLogging</code>	Extra logging for diagnosing task problems (default <code>false</code>)

VARIABLES IN TASKS

<code>\${workspaceFolder}</code>	The folder you opened in VS Code
<code>\${file}</code>	The file in the active editor
<code>\${fileBasename}</code>	That file's name with extension
<code>\${env:NAME}</code>	An environment variable
<code>\${config:setting.id}</code>	The value of a VS Code setting
<code>\${command:id}</code>	The result of a command, e.g. <code>python.interpreterPath</code>
<code>\${input:id}</code>	A value you type or pick when the task starts (defined in an <code>inputs</code> array)

TASKS TO COPY (EXAMPLES)

<pre>{ "label": "Run file", "type": "shell", "command": "python", "args": ["\${file}"] }</pre>	Runs t
<pre>{ "label": "Astro build", "type": "shell", "command": "npm run build", "options": { "cwd"</pre>	
<pre>{ "label": "All", "dependsOn": ["Lint", "Test"], "dependsOrder": "sequence" }</pre>	A task that c
<pre>"runOptions": { "runOn": "folderOpen" }</pre>	Start a task when the folder opens (needs <code>task.allowAut</code>
<pre>"problemMatcher": []</pre>	Add this if VS Code keeps asking which problem matcher to use

MY AUTOMATIC TASK DIDN'T START

A task with `"runOn": "folderOpen"` only runs if automatic tasks are allowed. The setting `task.allowAutomaticTasks` is off by default, and it only takes effect in a trusted workspace. Use *Tasks: Manage Automatic Tasks in Folder* to allow or disallow them for the current folder.

QUOTING PROBLEMS ON WINDOWS: TRY TYPE PROCESS

With `"type": "shell"` the command goes through your default shell (PowerShell on Windows unless you changed it), so spaces and quotes in arguments can behave unexpectedly. With `"type": "process"` the program is started directly and each entry in `args` is passed as it is.