

Sidebar

Programming · VS Code PHP (DEVSENSE) Cheat Sheet

The DEVSENSE PHP extensions: PHP Tools, Composer, IntelliPHP and PHP Profiler. Settings, commands and the Xdebug launch configuration, with the Premium features marked. Checked against PHP Tools 1.74.19252, Composer 1.74.19252, IntelliPHP 0.12.17700 and PHP Profiler 1.74.19252 installed here. Sheet 23 in the VS Code series, and the last of the extensions installed on this machine.

THE DEVSENSE BUNDLE & ITS LICENCE

PHP Tools	<code>devsense.phptools-vscode</code> (1.74.19252 here): IntelliSense, formatter, analysis, debugger, tests
Composer	<code>devsense.composer-php-vscode</code> : <code>composer.json</code> support and Composer commands
IntelliPHP	<code>devsense.intelli-php-vscode</code> (0.12.17700): whole-line suggestions from an offline AI model
PHP Profiler	<code>devsense.profiler-php-vscode</code> : open and inspect Xdebug profiling files
Premium features	The README marks these as Premium: starred suggestions, call and type hierarchy, code actions, inline refactorings, code fixes, code lenses, inlay hints, IntelliPHP, custom format rules, debug adornments, problem quick fixes and profiling tests. Whether your copy has them unlocked depends on your licence, which I cannot see
Activate License / Begin Free Trial...	Command Palette commands for the Premium features
<code>phpTools.suppressPremiumFeatures</code>	<code>true</code> : hide most Premium features and the messages about them (default <code>false</code>)
Quick Settings...	Command Palette: a page of the main options
Supported PHP versions	5.4 to 8.5, according to the extension's README badge

SETUP: PHP VERSION, PATHS & INDEXING

<code>php.executablePath</code>	Path to the <code>php</code> program (default none)
<code>php.executables</code>	Named PHP builds, e.g. <code>{ "8.3": "C:/p</code>
<code>php.version</code>	A PHP version number or one of the names abo
<code>php.stubs</code>	Extension stubs used for IntelliSense (default <code>["P</code>
<code>php.workspace.includePath</code>	Extra folders holding PHP dec
<code>php.workspace.shortOpenTag</code>	<code>auto</code> (default), <code>true</code> , <code>f</code>
<code>php.workspace.editorConfig</code>	<code>force</code> (default), <code>on</code> , <code>of</code> <code>.editorconfig</code>
<code>php.files.exclude</code>	Globs left out of indexing (default none)
<code>php.cache.location</code>	Folder for the cache of Composer pack
Clear /vendor/ cache.	Command Palette: rebuild the Inte
<code>phpTools.parallelismLimit</code>	Most threads used for indexin
<code>phpTools.language</code>	Language of the built-in PHP manual: e
Shipped defaults	<code>php.suggest.basic</code> and <code>php.valid</code> PHP features do not duplicate this extens

COMPLETION & HOVER

<code>php.completion.parameters</code>	After choosing a function: <code>parentheses</code> (default), <code>parameters</code> , <code>none</code>
<code>php.completion.autoimport</code>	Types outside the current namespace: <code>auto-import</code> (default), <code>auto-import-grouped</code> , <code>fqn</code> , <code>none</code> , <code>hide</code>
<code>php.completion.docblock</code>	Generate a DocBlock when you type <code>/**</code> (default <code>true</code> ; needs <code>editor.formatOnType</code>)
<code>php.completion.showNamespaceLabel</code> / <code>showParameters</code>	Show the namespace / the signature in the list (default <code>true</code> for both)
<code>php.completion.showDeprecated</code>	<code>showStrikethrough</code> (default), <code>showNormal</code> , <code>hide</code>
<code>php.completion.namingConvention</code>	<code>camelCase</code> (default), <code>PascalCase</code> , <code>snake_case</code> for generated names
<code>php.completion.starredSuggestions</code>	(Premium) Mark the likeliest completions with a star (default <code>true</code>)
<code>php.completion.intelliPHP.preSelect</code>	(Premium) Pre-select the AI's choice (default <code>false</code>)
<code>php.hover.parameters</code> / <code>hover.documentation</code>	Parameter descriptions and documentation link in tooltips (default <code>true</code>)
<code>php.hover.fullname</code> / <code>hover.parametersFullName</code>	Show fully qualified names (default <code>false</code>)
<code>php.hover.containingClass</code>	Show a member's class (default <code>true</code>)
<code>php.sortUses.caseSensitive</code>	Case-sensitive sorting of <code>use</code> statements (default <code>false</code>)
<code>php.linkedEditing.variables</code>	Rename variables together with <code>editor.linkedEditing</code> (default <code>false</code>)

INLAY HINTS & CODE LENS (PREMIUM)

<code>php.codeLens.enabled</code>	Reference counts and other lenses (d
<code>php.codeActions.enabled</code>	Quick fixes and code actions (de
<code>php.inlayHints.parameters.enabled</code>	Show parameter na
<code>php.inlayHints.parameters.suppressNameMatchingValue</code>	
<code>php.inlayHints.parameters.byReference</code>	Show <code>&</code> for t
<code>php.inlayHints.types.return</code>	Inferred return types: <code>true</code>
<code>php.inlayHints.types.variable</code>	Inferred variable types (d
<code>php.inlayHints.types.lambdaParameter</code>	Inferred lambda
<code>php.inlayHints.types.suppressVariableFromLiteral</code>	S
<code>php.inlayHints.insertOnDoubleClick</code>	Double-click a hin

PROBLEMS, NAVIGATION & TO-DOS

<code>php.problems.scope</code>	Which files are analysed: <code>opened</code> (default), <code>all</code> , <code>none</code> ; also the command <i>Background Code Analysis</i>
<code>php.problems.exclude</code>	Path patterns whose problems are not reported (default none)
<code>php.problems.excludeGitIgnore</code>	Do not report problems in files listed in <code>.gitignore</code> (default <code>false</code>)
<code>php.navigation.referencesScope</code>	<code>any</code> (default) or <code>strict</code> ; what counts as a reference
<code>php.highlight-todo.enable</code>	Highlight <code>TODO</code> and <code>@todo</code> comments (default <code>true</code>)
<code>php.highlight-todo.style</code>	Their style (default <code>{ "backgroundColor": "#eccc00" }</code>)
<code>Search todo...</code>	Command Palette: list every to-do in the workspace
<code>Import class... / FQN class...</code>	Command Palette: add a <code>use</code> statement / write the fully qualified name
<code>Implement class overrides...</code>	Command Palette: generate methods to override

FORMATTING

<code>php.format.codeStyle</code>	<code>PSR-12</code> (default), <code>PER</code> , <code>PSR-2 Joomla</code> , <code>PHP Tools</code> , <code>Off</code>
<code>php.format.exclude</code>	Glob patterns of files not to format (default none)
<code>php.format.rules.*</code>	(Premium) About a hundred individual rules, the code style decides
<code>php.format.rules.importSorting</code>	(Premium) <code>off</code> , <code>alpha</code> , <code>use</code>
<code>php.format.rules.openBraceOnNewLineForNamespaces</code>	(Premium) <code>off</code> , <code>on</code>
<code>php.format.rules.blankLinesAfterClasses, Interfaces, Traits</code>	(Premium) <code>off</code> , <code>on</code>
<code>php.format.rules.wrap</code>	(Premium) <code>off</code> , <code>onEveryItem</code>
<code>php.format.rules.space / align</code>	(Premium) Spaces in code, properties, match arrays
<code>Format Multiple Files</code>	Command Palette: batch-format PHP files
<code>editor.formatOnSave / formatOnType</code>	Format on save; format on type

DOCBLOCK SNIPPETS

<code>php.docblock.classSnippet</code>	Tags added above classes, interfaces, traits and enums, e.g. <code>{ "@author": "John", "@copyright": "(c) \$CURRENT_YEAR" }</code>
<code>php.docblock.packageSnippet</code>	Tags at the top of a file
<code>php.docblock.functionSnippet</code>	Tags above functions and methods
<code>php.docblock.propertySnippet / variableSnippet</code>	Tags above properties / variable assignments
<code>php.docblock.getterSetterSnippet</code>	Template for DocBlocks made by the getter/setter code action
<code>php.docblock.colorMode</code>	Colouring of DocBlocks
<code>php.completion.getterSnippet / setterSnippet</code>	Code placed inside generated getters and setters
<code>Variables</code>	Snippet values may use VS Code snippet variables such as <code>\$CURRENT_YEAR</code>

DEBUGGING WITH XDEBUG (LAUNCH.JSON TYPE: PHP)

<code>type / request</code>	<code>"type": "php"</code> , <code>"request": "launch"</code> , <code>"request": "attach"</code>
<code>php.debug.port</code>	Default Xdebug port(s) when <code>port</code> is not specified
<code>Snippets in launch.json</code>	<i>Launch built-in server, Launch & Listen for Xdebug</i>
<code>program / args / cwd</code>	The script to run, its arguments and the working directory
<code>runtimeExecutable / runtimeArgs</code>	A specific <code>php</code> program to use
<code>port / hostname</code>	Where the debugger listens (Xdebug 3 default <code>127.0.0.1</code>)
<code>pathMappings</code>	Map server folders to local ones, e.g. <code>{ "/var/www/html": "\${workspaceFolder}" }</code>
<code>stopOnEntry / noDebug / trigger</code>	Pause at the first line / no debug / trigger
<code>env / envFile / XdebugSettings</code>	Environment and Xdebug settings
<code>ignoreExceptions / skipEntryPaths / exclude</code>	Exceptions to ignore / skip entry paths / exclude
<code>dbgProxy / maxConnections / externalConsole</code>	Proxy settings / max connections / external console
<code>profile / openProfile</code>	Profile the run with Xdebug and open the profile
<code>Start Listening for Xdebug</code>	Command Palette: wait for a debugger to attach
<code>Debug PHP script / Run PHP Script</code>	Command Palette: debug / run PHP script
<code>Start Debugging and Stop on Entry</code>	Bound to F10 and F11

TESTING (PHPUNIT AND PEST)

<code>Test Explorer</code>	Lists PHPUnit, Paratest and Pest tests; run or debug all, a suite, one test or one dataset
<code>phpunit.phpunit</code>	Path to the PHPUnit binary (default none; found in <code>vendor</code> , <code>bin</code> or a <code>.phar</code>)
<code>phpunit.config</code>	Path or glob for <code>phpunit.xml</code> (default none)
<code>phpunit.command</code>	The command template; variables <code>\${php}</code> , <code>\${phpargs}</code> , <code>\${phpunit}</code> , <code>\${phpunitargs}</code> , <code>\${phpunitxml}</code>
<code>phpunit.enableAutomaticRefresh</code>	Refresh the test list when files change (default <code>true</code>)
<code>phpunit.preTask</code> / <code>phpunit.postTask</code>	Tasks from <code>tasks.json</code> to run before / after tests

COMPOSER, PROFILER & INTELLIPHP

<code>composer.workingPath</code>	Folder containing <code>composer.json</code>
<code>composer.bin</code>	Path to a specific <code>composer.phar</code> (default: <code>composer</code>)
<code>composer.inlayVersionEnabled</code>	Show the installed version (<code>true</code>)
<code>composer.diagnosticsEnabled</code>	Warn about abandoned packages (<code>true</code>)
Composer: <code>install</code> / <code>update</code> / <code>require package...</code> / <code>require dev package...</code>	

Composer: `outdated` / `show --tree` / `run-script...` / `clear-compiled`

`phpprofiler.hotpath.threshold` Percent of time (0 to 100)

Open Profile File (Xdebug Profiling Output) / Load Profile File

`intelliphp.inlineSuggestionsEnabled` All whole-line inline suggestions

FRAMEWORKS (FROM THE README)

<code>Laravel</code>	Blade highlighting, completion and formatting; Eloquent magic; routes, views, components, Livewire; command <i>Switch between Blade View and Class</i>
<code>WordPress</code>	WordPress stubs, hooks completion and WordPress code style
<code>CodeIgniter 3</code> , <code>Pest</code>	Dynamic services for CodeIgniter 3; Pest tests in the Test Explorer
<code>PHPStan</code> / <code>Psalalm</code>	Their annotations and generics are understood in IntelliSense

A REASONABLE STARTER SET (MY SUGGESTIONS, NOT THE README)

<code>"php.format.codeStyle": "PSR-12"</code>	The default; set it explicit
<code>"editor.formatOnSave": true</code>	Format PHP when you save
<code>"[php]": { "editor.defaultFormatter": "DEVSENSE.php" }</code>	Use DEVSENSE as the default formatter
<code>"php.problems.scope": "opened"</code>	Fast, quiet analysis (the default)
<code>"php.inlayHints.types.variable": true</code>	More hints, if you want
<code>"phpTools.suppressPremiumFeatures": true</code>	If you do not want to pay

GETTING XDEBUG WORKING

The debugger needs the Xdebug extension in your PHP installation. In `php.ini` that usually means a `zend_extension` line for Xdebug, `xdebug.mode=debug`, and `xdebug.start_with_request=yes` (or `trigger`); Xdebug 3 connects back to port **9003**, which is why that is first in `php.debug.port`. Run `php -v` in a terminal: its output names Xdebug when it is loaded. Then use *Launch built-in server* from the `launch.json` snippets to run and debug the site without Apache.

TWO PHP EXTENSIONS FIGHTING EACH OTHER?

This extension switches VS Code's own basic PHP suggestions and validation off by default. If you also install another PHP language extension (for example *Intelephense*), you may get double completions, duplicate errors or conflicting formatters: keep only one active for each job. Check *Extensions* > *@installed php* and use *Disable (Workspace)* on the one you do not want in a given project.