

This document applies to search (CMD+SHIFT+F/CTRL+SHIFT+F) and quickopen (CMD+P/CTRL+P). By default, VS Code uses the ripgrep tool to drive search. Learn more about how to use search in the documentation.

Missing search results

By far, the most common reason that expected search results don't appear is because of **exclude settings and ignore files**. Search and quickopen ignore files using patterns specified in the `search.exclude` and `files.exclude` settings, or covered by a pattern in a `.gitignore` file. So the very first thing to do is to carefully check these settings at the user and workspace levels, and your `.gitignore` file.

Besides `.gitignore`, we also look at the `.ignore`, `.rgignore`, and `.git/info/exclude` files. You can read more about ripgrep's ignore file logic [here](#).

Tip: You can set `"search.useIgnoreFiles": false` to disable using the `.gitignore` file for search

An easy way to validate whether exclude settings or ignore files are affecting your search is to turn off the “Use Exclude Settings and Ignore Files” button in the search viewlet. This is the gear button in the lower right corner:

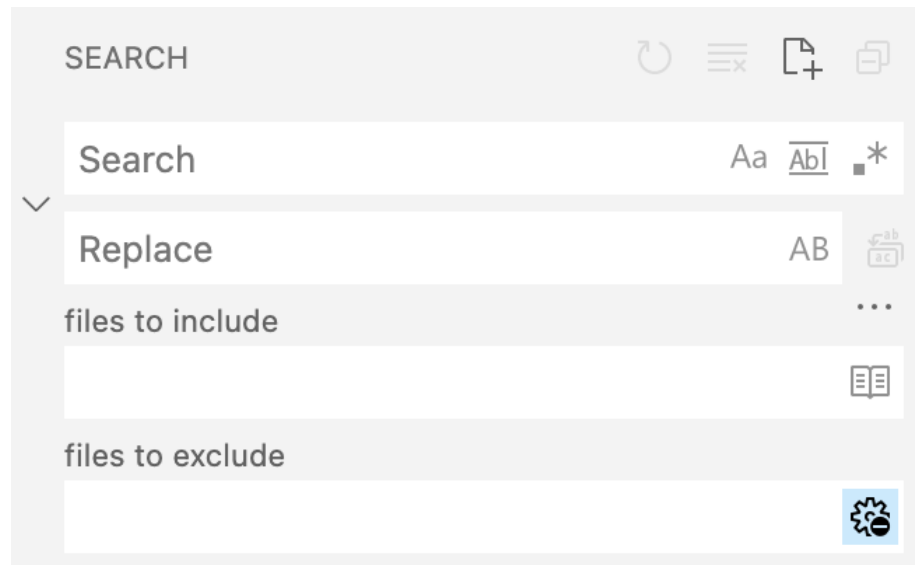


Figure 1: Use Exclude Settings and Ignore Files button

Another thing to watch for is that **it's possible to add a file covered by a `.gitignore` file to git**, and some tools like `git grep` will still search these files. Ripgrep is looking at the `.gitignore` file but doesn't know whether a file

has been added to git. So if a file is covered by the `.gitignore` file, it won't be searched, whether or not it's in git.

VS Code also supports searching only in the files that are currently opened as tabs, this mode is controlled by the book icon in the "files to include" input, and a message is shown when this is enabled:

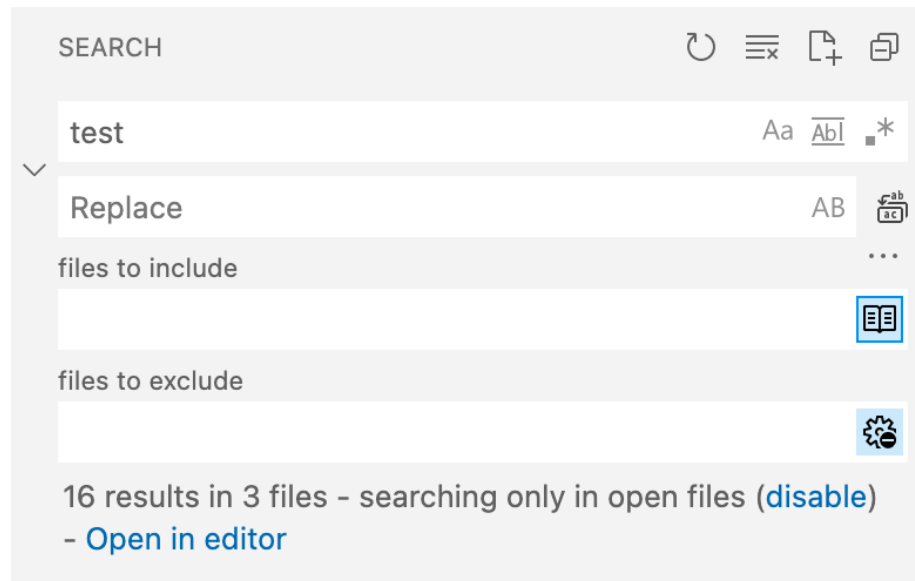


Figure 2: The book icon controlling whether only open editors are searched

If your workspace has files with a **non-UTF-8 encoding**, you will need to set the `files.encoding` setting to the correct encoding to search in those files. Note that the `files.autoGuessEncoding` works for the editor, but isn't supported for search.

This is less common, but VS Code doesn't handle files with **CR-only line endings** well. Files that originated on older OS's may use these, and may have missing or inconsistent search results. There is an open issue for this.

When looking at issues with missing results, remember that **search in open files** is implemented by the editor. Search in all other files is implemented by ripgrep. Some issues that look like search is working only in a few random files can be explained by a problem that only affects the ripgrep side of the search.

ENAMETOOLONG error

If you see this error appear and are missing search results, you may have so many patterns configured in `files.exclude` or `search.exclude` that we hit the OS limit for the length of a command line command. You can work around this by

getting rid of excluded patterns, or moving those patterns into a file in the root of your workspace named `.ignore`, same syntax as `.gitignore`, which ripgrep will also pick up.

Search including too many results

If your search includes results that you expect to be excluded by `search.exclude` and `files.exclude` settings or a `.gitignore` file, the first thing to do is to **check those very carefully** to ensure they cover what you expect them to cover. Your workspace may have settings that override your user settings. And setting `"search.useIgnoreFiles": false` will disable checking the `.gitignore` file.

The search viewlet has a **toggle button**, “Use Exclude Settings and Ignore Files”, which will disable checking both of these things. Make sure that it’s toggled on - it’s not uncommon that users turn it off and forget about it.

Another thing that can cause excluded files to be shown is that open files are still searched even when they are excluded. This is a vscode bug.

Slow search, rg running for a long time, or consuming lots of CPU/Memory

Does your workspace have lots of **symlinks**? Search can take a long time if it has to follow lots of symlinks. Setting `"search.followSymlinks": false` will disable this.

Does your search return results on **very long lines**? This will cause some slowdown and can even cause VS Code to hang. There is an open bug for this.

If your workspace is **very large**, you might want to use the exclude settings, or the “files to include/exclude” text boxes to narrow down your search to the parts you care about. Make sure the “Use Exclude Settings and Ignore Files” button is toggled on so your exclude settings are respected.

On Windows, the **Windows Defender** antivirus tool can sometimes kick in during a search and slow down searching. Look for a process named using high CPU named `MsMpEng.exe` in the Task Manager to tell whether Defender is active. Other antivirus programs can cause strange behavior as well.

You can start VS Code from the command line with the `--status` flag, or use the Process Explorer (Help > Open Process Explorer) to see running processes owned by VS Code. **The ripgrep process** will show up as a command that invokes the `rg` binary inside the VS Code install directory. The arguments passed to `rg` tell you why it was started and are useful to include in an issue report. You can even copy the command and run it yourself to check for the correct output.

The steps on the [\[\[Performance Issues\]\]](#) wiki page may also be useful.

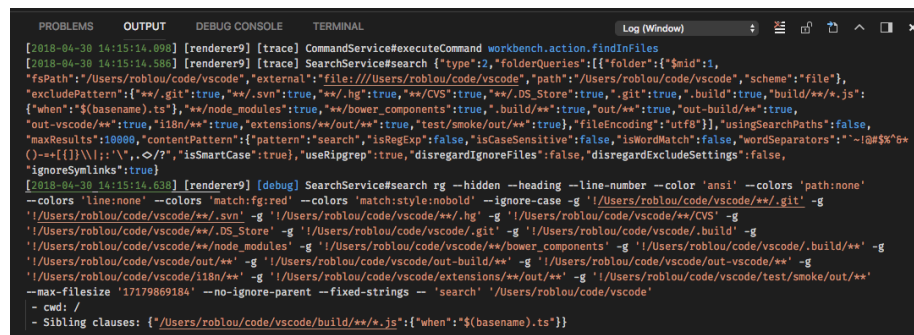
Filing a search issue

When filing a search-related issue on the VS Code repo, please try the steps above and include as many details as possible to help us diagnose the issue.

- If there's an issue with search, do you see the same issue with quickopen, (or vice versa)?
- If you see an issue with search when you have folder A open, do you see a similar issue with other folders on your computer?

Collecting search logs

Some details are logged for each search. To see these logs, run the command "Developer: Set Log Level...", select "Trace", and run the search again. Then in the output pane, see the logs in the channel named "Log (Window)". The logs show VS Code's internal query object, the arguments with which ripgrep was invoked, and any errors produced by ripgrep.



```
[2018-04-30 14:15:14.698] [renderer9] [trace] CommandService#executeCommand workbench.action.findInFiles
[2018-04-30 14:15:14.586] [renderer9] [trace] SearchService#search {type:'i',folderQueries:[{"folder":{"$mid":1,
"fsPath":"/Users/roblou/code/vscode","external":{"file:///Users/roblou/code/vscode","path":"/Users/roblou/code/vscode","scheme":"file"},
"excludePattern":{"**/.git":true,**/.svn":true,**/.hg":true,**/CVS":true,**/.DS_Store":true,**/.git":true,"build":true,"build/**/*":true,
"when":{"$(basename).ts"},"**/node_modules":true,**/bower_components":true,"build/**":true,"out/**":true,"out-build/**":true,
"out-vscode/**":true,"118n/**":true,"extensions/**/out/**":true,"test/smoke/out/**":true},"fileEncoding":"utf8"}],"usingSearchPaths":false,
"maxResults":10000,"contentPattern":{"pattern":"search","isRegExp":false,"isCaseSensitive":false,"isWordMatch":false,"wordSeparators":"~|@#%&*~
()-+{}[]\\|:;'\",./<?/","isSmartCase":true},"useRipgrep":true,"disregardIgnoreFiles":false,"disregardExcludeSettings":false,
"ignoreSymlinks":true}
[2018-04-30 14:15:14.638] [renderer9] [debug] SearchService#search rg --hidden --heading --line-number --color 'ansi' --colors 'path:none'
--colors 'line:none' --colors 'match:fg:red' --colors 'match:style:nobold' --ignore-case -g '/Users/roblou/code/vscode/**/.git' -g
'/Users/roblou/code/vscode/**/.svn' -g '/Users/roblou/code/vscode/**/.hg' -g '/Users/roblou/code/vscode/**/CVS' -g
'/Users/roblou/code/vscode/**/.DS_Store' -g '/Users/roblou/code/vscode/.git' -g '/Users/roblou/code/vscode/.build' -g
'/Users/roblou/code/vscode/**/node_modules' -g '/Users/roblou/code/vscode/**/bower_components' -g '/Users/roblou/code/vscode/.build/**' -g
'/Users/roblou/code/vscode/out/**' -g '/Users/roblou/code/vscode/out-build/**' -g '/Users/roblou/code/vscode/out-vscode/**' -g
'/Users/roblou/code/vscode/118n/**' -g '/Users/roblou/code/vscode/extensions/**/out/**' -g '/Users/roblou/code/vscode/test/smoke/out/**'
--max-filesize '17179869184' --no-ignore-parent --fixed-strings - 'search' '/Users/roblou/code/vscode'
- cwd: /
- Sibling clauses: {'/Users/roblou/code/vscode/build/**/*':{"when":{"$(basename).ts"}}
```

Figure 3: screen shot 2018-04-30 at 2 15 35 pm

There may also be errors that only show up in the developer tools. Open the developer tools (Help > Toggle Developer Tools) and check the Console for errors.

Notes on regular expression support

Text search uses two different sets of regular expression engines. The workspace is searched using ripgrep, which will use the Rust regex engine, and will fallback to PCRE2 if the regex fails to parse in the Rust regex engine. The Rust regex engine doesn't support some features like backreferences and look-around, so if you use those features, PCRE2 will be used. Open files are searched using a JS regex in the editor itself. Most of the time, you don't need to worry about this, but you may see an inconsistency in how some complex regexes are interpreted, and this can be an explanation. Especially when you see a regex interpreted one way when a file is open, and another way when it is not. During a Replace operation, each file will be opened in turn, and the search query will be run as a JS regex.

Another potential issue is how newlines are handled between ripgrep and the editor. The editor normalizes newlines, so that you can match both CRLF and LF line endings just with `\n`. It's actually not possible to match `\r` explicitly in the editor because it is normalized away. When searching in the workspace, VS Code tries to rewrite a regex so that `\n` will match CRLF. But `\r\n` or `\s\n` will also match CRLF in closed files, but not in open files.