# C/C++ search

You can conduct a fully or partially qualified name search. Further qualifying a search increases the accuracy and relevance of search results. The sections below provide guidance on how to control the scope of your search through the use of search delimiters, correct syntax, and wildcards.

You can search for:

* language constructs within:
  + projects in your workspace
  + selected resources from various views
  + working sets
* a working set for references to particular elements
* declarations of particular elements
* definitions of particular elements
* references of particular elements

For information on working sets, see **Workbench User Guide > Concepts > Workbench > Working sets**

## What you can search for

The table below lists the element types that you can search for and special considerations to note when searching for a given element type. You can search for some or all of the element types matching a search string that you specify. If you choose to search for matching elements, all types, macros, and typdefs are included in the search.

| **Element** | **Note** |
| --- | --- |
| Class/Struct | Searches for classes and structs.  You can further qualify the search by specifying "class" or "struct" in front of the name that you are searching for. Specifying "class" or "struct" also allows you to search for anonymous classes and structures. |
| Function | Searches for global functions or functions in a namespace (functions that are not members of a class, struct, or union).  You can specify parameters to further qualify your search. When specifying a parameter list, everything between the parentheses should be valid C/C++ syntax.  Do not specify the return type of the function. |
| Variable | Searches for variables that are not members of a class, struct, or union. |
| Union | Searches for unions.  Anonymous unions can be searched for by specifying "union" as the search pattern. |
| Method | Searches for methods that are members of a class, struct, or union.  Searching for methods also finds constructors and destructors. See above note for functions. |
| Field | Searches for fields that are members of a class, struct, or union. |
| Enumeration | Searches for enumerations. |
| Enumerator | Searches for enumerators. |
| Namespace | Searches for namespaces. |

## How you can limit your search

You can limit your search to one or all of the following:

* Declarations
* References
* Definitions (for functions, methods, variables and fields)

You can control the scope of the search by specifying which of the following is to be searched:

* Workspace
* Working Set
* Selected Resources

## Wildcard characters

You can use wildcard characters to further refine your search.

| **Use this wildcard character** | **To search for this** |
| --- | --- |
| \* | Any string  **Tip:**  Use the character \* to search for operators that begin with \*. See syntax examples in the table below. |
| ? | A single character |
| :: | Nested elements |

**Tip:** Do not use wild cards between the brackets of a function or method pattern. For example, the search string f( \* ) is an invalid search that results in a search for any function f because the asterisk is interpreted as a pointer rather than a wild card.

## Syntax examples

The table below provides syntax examples and an explanation for each example to help you conduct an effective search.

| **Syntax** | **Searches for this** |
| --- | --- |
| ::\*::\*::A | A nested element two levels deep |
| ::\*::\*::A? | Any two-letter name that begins with A and is two levels deep |
| ::A | Searches for A not nested in anything |
| \*() | Any function taking no parameters |
| \*( A \* ) | Any function taking 1 parameter that is a pointer to type A |
| f( int \* ) | Will search for function f taking 1 parameter that is an int \* |
| f( const char [ ], A & ) | Will search for a function f, taking 2 parameters; one is a const char array, the other is a reference to type A |
| operator \\* | Finds only operator \* |
| operator \\*= | Finds only operator \*= |
| operator \* | Finds all operators |
| class | Searches for anonymous classes |
| struct | Searches for anonymous structs |
| union | Searches for anonymous unions |

## Search results

Search results are displayed in the Search view. You can sort your search by Name, Parent Name and Path. You can also repeat your last search.

## Search Concepts

### Declarations

According to the ANSI C++ Spec, a declaration is a statement that “introduces a name into a translation unit or re-declares a name that has been previously introduced by a previous declaration.

All C/C++ search elements can be searched for declarations.

### Definitions

Most declarations are also definitions; in other words, they also define the entity for they declare the name for. However there are some elements that can have separate definitions from their declarations.

For C/C++ search the following elements can be searched for definitions:

* Functions/Methods – the definition is where the code implementation resides
* Variable:
  1. Extern – the definition is where the variable is initialized
  2. Non extern - the definition of a variable is where it is declared
* Field:
  1. Static fields - the definition of a static field is where it gets initialized
  2. Non static fields - the definition corresponds to the fields declaration
* Namespace – the definition of a namespace is the same as its declaration

### References

By selecting references, C/C++ search will return all of the places the selected element is used.

### All Occurrences

Selecting ‘All Occurrences’ in the Limit To section will result in a search for declarations, definitions (if applicable) and references for whatever element or elements have been selected.

### Any Element

Selecting ‘Any Element’ in the Search For section will result in a search for all of the listed elements plus macros and typedefs.

For more information, see:

* **Workbench User Guide > Concepts > Views > Search view**
* **Workbench User Guide > Tasks > Navigating and finding resources**

****

[C/C++ Indexer](http://docs.google.com/concepts/cdt_c_indexer.htm)

[CDT Projects](http://docs.google.com/cdt_c_projects.htm)

[Open Declaration](http://docs.google.com/cdt_c_open_declarations.htm)



[Searching for C/C++ elements](http://docs.google.com/tasks/cdt_t_search.htm)

[Navigating to C/C++ declarations](http://docs.google.com/tasks/cdt_t_open_declarations.htm)



[C/C++ search page, Search dialog box](http://docs.google.com/reference/cdt_u_search.htm)

[C/C++ perspective icons](http://docs.google.com/reference/cdt_u_icons.htm)

