GoFish: Searching in VFP Projects and Folders

Tamar E. Granor
Tomorrow's Solutions, LLC
Voice: 215-635-1958
Website: www.tomorrowssolutionsllc.com
Email: tamar@tomorrowssolutionsllc.com

GoFish is a community-grown search tool for VFP. It's faster and better than the Code References tool that comes with VFP. In this session, we'll look at how to use GoFish to find and replace in your VFP projects and folders.

Introduction

FoxPro didn't include projects until version 2.0. It was at that point that Fox Software realized that people needed a way to keep all the parts of an application together. The same version introduced Filer, a tool for searching in folders. But there was no tool for searching for text in a project.

Then in 2001, VFP developer Peter Diotte shared GoFish with the FoxPro community. Peter continued to fix and enhance GoFish for several years and quite a few developers adopted it. But after a while, Peter moved on to other projects, though many people continued to use GoFish.

VFP 8 introduced the Code References tool for searching in folders and projects. While Code References was effective for searches, it was slow and its replacement capability quickly got a reputation for mangling files. (I wrote about Code References soon after we got it: https://tinyurl.com/2wa46ruc.)

For a long time, nothing changed in VFP project searching. But in 2011, with Peter's permission, Matt Slay adopted GoFish, updated it, replacing the user interface entirely, and added it to the collection of VFP tools and components available through VFPX. The Thor team made it easy to get GoFish and keep it up to date through Thor. With those changes, a lot more developers (including me) started using GoFish.

Sadly, Matt died in 2021, but development of GoFish continues with Doug Hennig as the project manager.

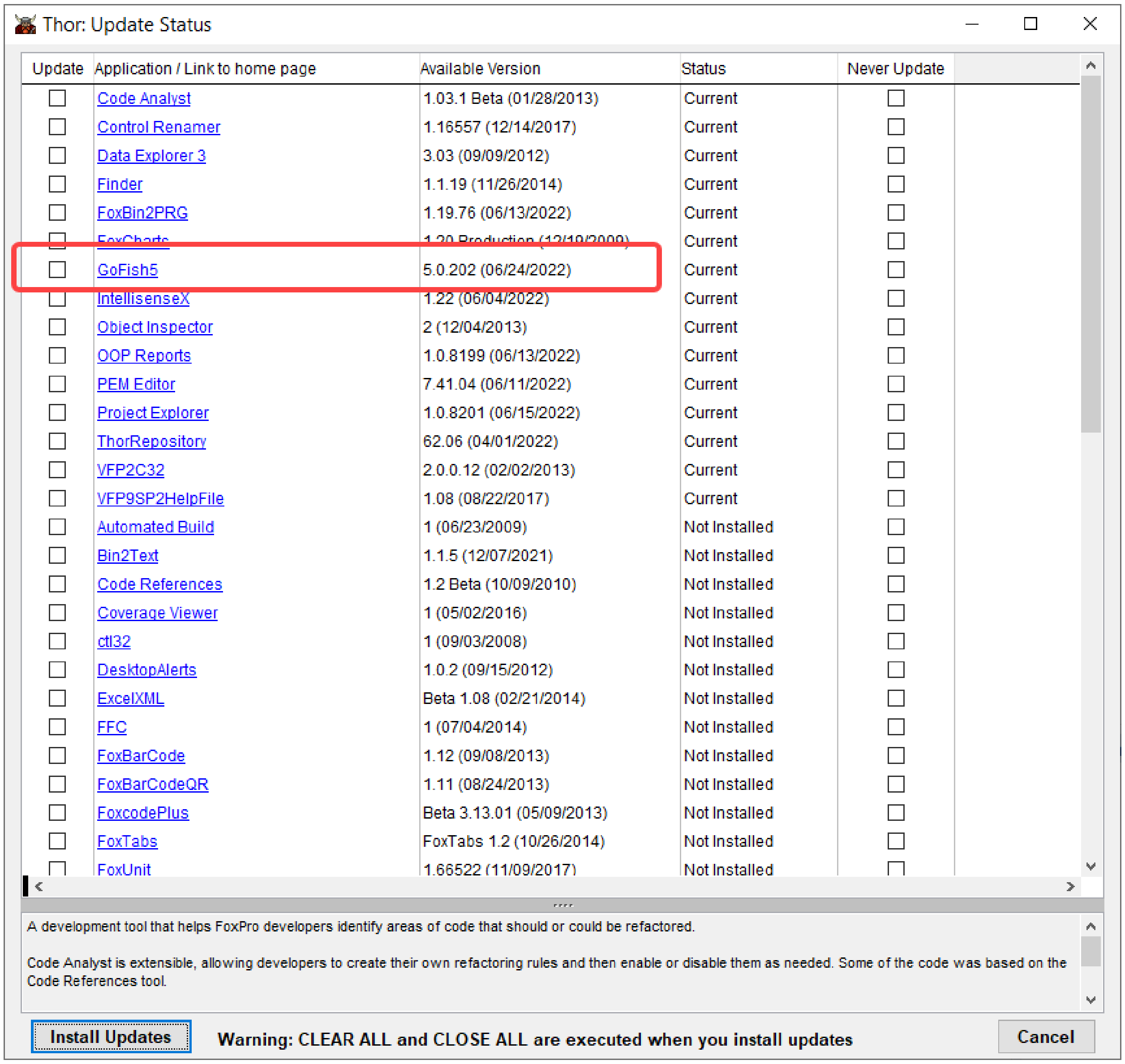
Setting up

You can get GoFish either directly from the VFPX Github site or through Thor. It's far simpler through Thor, so we'll look at that option first.

Installing GoFish from Thor

If you have Thor installed, it couldn't be easier.[1] From the Thor menu, choose Check for Updates. Then scroll down to the "Not Installed" section in the list and find GoFish5. (It's circled in **Figure 1**. Because I already have it installed, it's in the "Current" section rather than the "Not installed" section.) Check it and the click Install Updates.

[1] If you don't have Thor installed, why not? Learn why you should from my paper "Try Thor's Terrific Tools" at https://tinyurl.com/2p82ssbh.



Thor: Update Status

Update	Application / Link to home page	Available Version	Status	Never Update
☐	Code Analyst	1.03.1 Beta (01/28/2013)	Current	☐
☐	Control Renamer	1.16557 (12/14/2017)	Current	☐
☐	Data Explorer 3	3.03 (09/09/2012)	Current	☐
☐	Finder	1.1.19 (11/26/2014)	Current	☐
☐	FoxBin2PRG	1.19.76 (06/13/2022)	Current	☐
☐	FoxCharts	1.20 Production (12/10/2009)	Current	☐
☐	GoFish5	5.0.202 (06/24/2022)	Current	☐
☐	IntellisenseX	1.22 (06/04/2022)	Current	☐
☐	Object Inspector	2 (12/04/2013)	Current	☐
☐	OOP Reports	1.0.8199 (06/13/2022)	Current	☐
☐	PEM Editor	7.41.04 (06/11/2022)	Current	☐
☐	Project Explorer	1.0.8201 (06/15/2022)	Current	☐
☐	ThorRepository	62.06 (04/01/2022)	Current	☐
☐	VFP2C32	2.0.0.12 (02/02/2013)	Current	☐
☐	VFP9SP2HelpFile	1.08 (08/22/2017)	Current	☐
☐	Automated Build	1 (06/23/2009)	Not Installed	☐
☐	Bin2Text	1.1.5 (12/07/2021)	Not Installed	☐
☐	Code References	1.2 Beta (10/09/2010)	Not Installed	☐
☐	Coverage Viewer	1 (05/02/2016)	Not Installed	☐
☐	ctl32	1 (09/03/2008)	Not Installed	☐
☐	DesktopAlerts	1.0.2 (09/15/2012)	Not Installed	☐
☐	ExcelXML	Beta 1.08 (02/21/2014)	Not Installed	☐
☐	FFC	1 (07/04/2014)	Not Installed	☐
☐	FoxBarCode	1.12 (09/08/2013)	Not Installed	☐
☐	FoxBarCodeQR	1.11 (08/24/2013)	Not Installed	☐
☐	FoxcodePlus	Beta 3.13.01 (05/09/2013)	Not Installed	☐
☐	FoxTabs	FoxTabs 1.2 (10/26/2014)	Not Installed	☐
☐	FoxUnit	1.66522 (11/09/2017)	Not Installed	☐

A development tool that helps FoxPro developers identify areas of code that should or could be refactored.

Code Analyst is extensible, allowing developers to create their own refactoring rules and then enable or disable them as needed. Some of the code was based on the Code References tool.

Install Updates
Warning: CLEAR ALL and CLOSE ALL are executed when you install updates
Cancel

Figure 1. To install GoFish from Thor, find it in the Thor: Update Status dialog, check it and click Install Updates.

That's it. Thor handles the rest, downloading and installing the tool, and setting it up as a Thor tool, so you don't have to go hunting for it. I strongly recommend creating a shortcut in Thor for GoFish, so you can call it up with a single key combination. As **Figure 2** shows, mine is Ctrl+Alt+G.

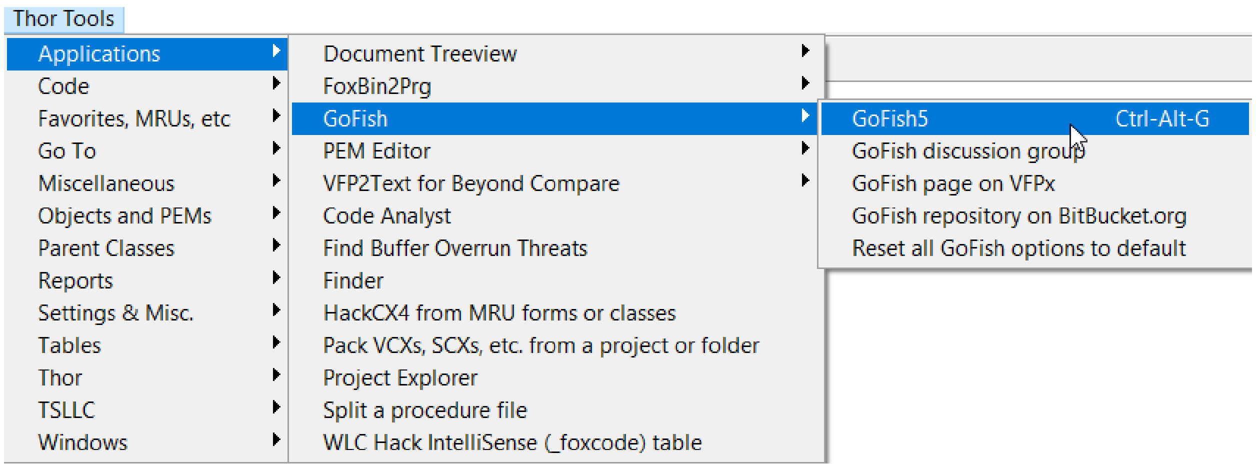
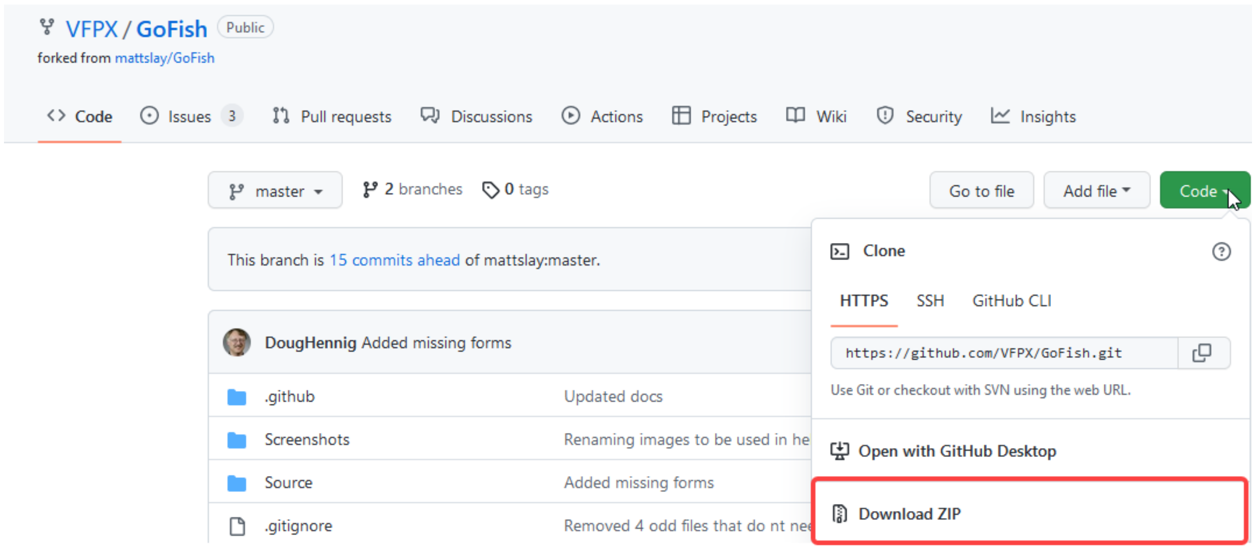


Figure 2. If you install GoFish via Thor, it's added to the Thor Tools menu and you don't have to do anything else to get it running.

Installing GoFish manually

To install GoFish without Thor, go to https://github.com/VFPX/GoFish. Click the Code dropdown (where the mouse arrow is in **Figure 3**), and choose Download Zip.



This branch is 15 commits ahead of mattslay:master.

DougHennig Added missing forms

.github	Updated docs
Screenshots	Renaming images to be used in he
Source	Added missing forms
.gitignore	Removed 4 odd files that do nt nee

Figure 3. Download GoFish from its Github site.

Once the zip has finished downloading, extract the file GoFish5.App from it and put it wherever you want. (It's a good idea to keep the path short, since you'll need it to run the tool.) To run GoFish, from the VFP Command Window, issue:

```
DO <path>\GoFish5.app
```

where <path> is the complete path to the file. If you have a custom menu of tools, you can add it there.

Getting help with GoFish

GoFish doesn't have built-in help. However, there is a Google group where you can ask questions, suggest enhancements, or report bugs. If you're using Thor, you can get to the group from the Thor menu, as in **Figure 4**.

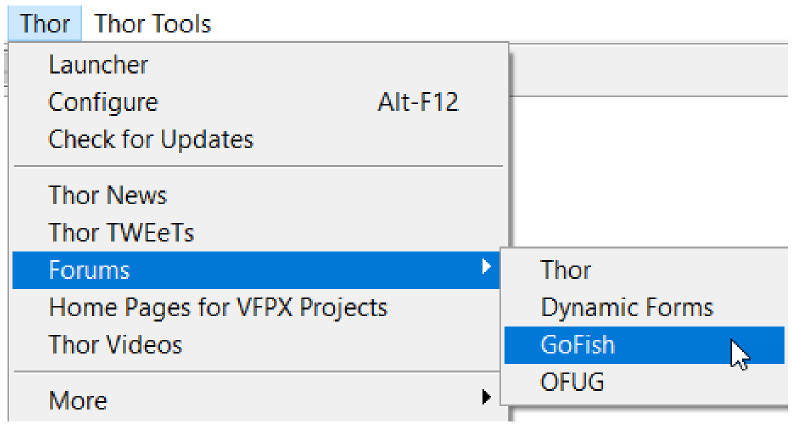


Figure 4. The Thor menu gives you an easy way to get to the Google group for GoFish.

If you're not using Thor, go to https://groups.google.com/g/FoxProGoFish. You can also report bugs or make suggestions in the Issues section of the GoFish Github site.

The VFPX home page for GoFish includes a lot of information about what GoFish can do and an introductory video. In addition, there is a chapter on GoFish in "VFPX: Open Source Treasure for the VFP Developer."

Finally, enough people in the VFP community are using GoFish that you can also ask questions in pretty much any of the online forums for VFP and find help.

Search

Using GoFish to do simple text searches is straightforward. Open GoFish and you'll see something like[2] **Figure 5**. To do a search, type a search string into the Search textbox and choose where to search using the Scope dropdown (see **Figure 6**). You can search in a project, in a single folder, or in a folder and all of its subfolders. GoFish remembers where you've searched before and offers those places. When you first open it, it offers all the folders in your "most recently used" list of folders, that is, the ones that would come up if you typed MODIFY PROJECT in the Command Window.

[2] I say "something like" because GoFish remembers a lot as you use it. While I tried to restore it to its initial appearance for this figure, I'm not sure I got everything.

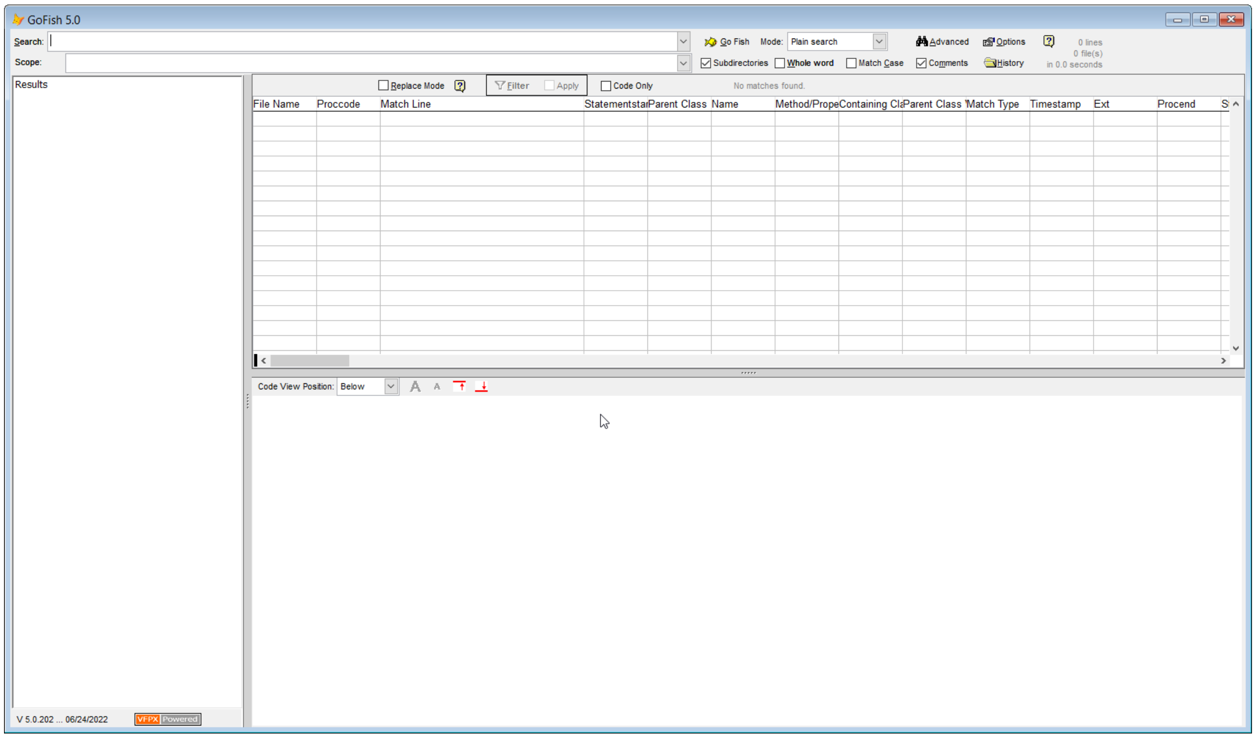
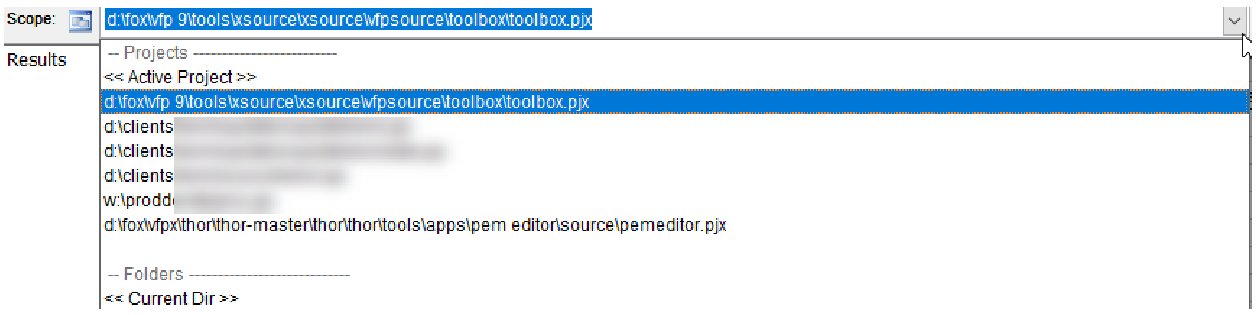


Figure 5. GoFish looks complicated but starting to use it is easy. Just type in a search string and select a project or folder to search. Then click the GoFish button.


Scope: d:\fox\vfp 9\tools\xsource\xsource\vfpsource\toolbox\toolbox.pjx

Results

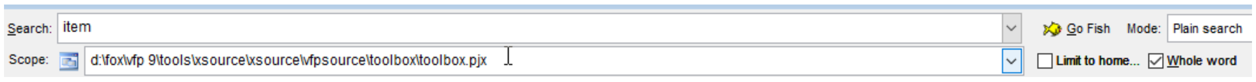
-- Projects -----------------------
<< Active Project >>
d:\fox\vfp 9\tools\xsource\xsource\vfpsource\toolbox\toolbox.pjx
d:\clients[redacted]
d:\clients[redacted]
d:\clients[redacted]
w:\prodd[redacted]
d:\fox\vfpx\thor\thor-master\thor\thor\tools\apps\pem editor\source\pemeditor.pjx

-- Folders -------------------------
<< Current Dir >>

Figure 6. The Scope dropdown shows projects you've searched, as well as those in VFP's most recently used list. It also remembers any folders you've searched. Here, there's no search history, so only the MRU projects are included. The icon between "Scope:" and the dropdown indicates whether the current selection is a project or a folder.

For our initial search, let's look for the word "item" in the VFP Toolbox project as in **Figure 7**. (We'll use the Toolbox project and folders for most of the examples in this session. Since that's code you get with VFP[3], it allows you to try the same searches.)

[3] If you don't already have the Toolbox source code available in a folder similar to the one shown in **Figure 6**, unzip the xsource.zip file in the HOME(1) + "Tools\xsource" folder.



Search: item

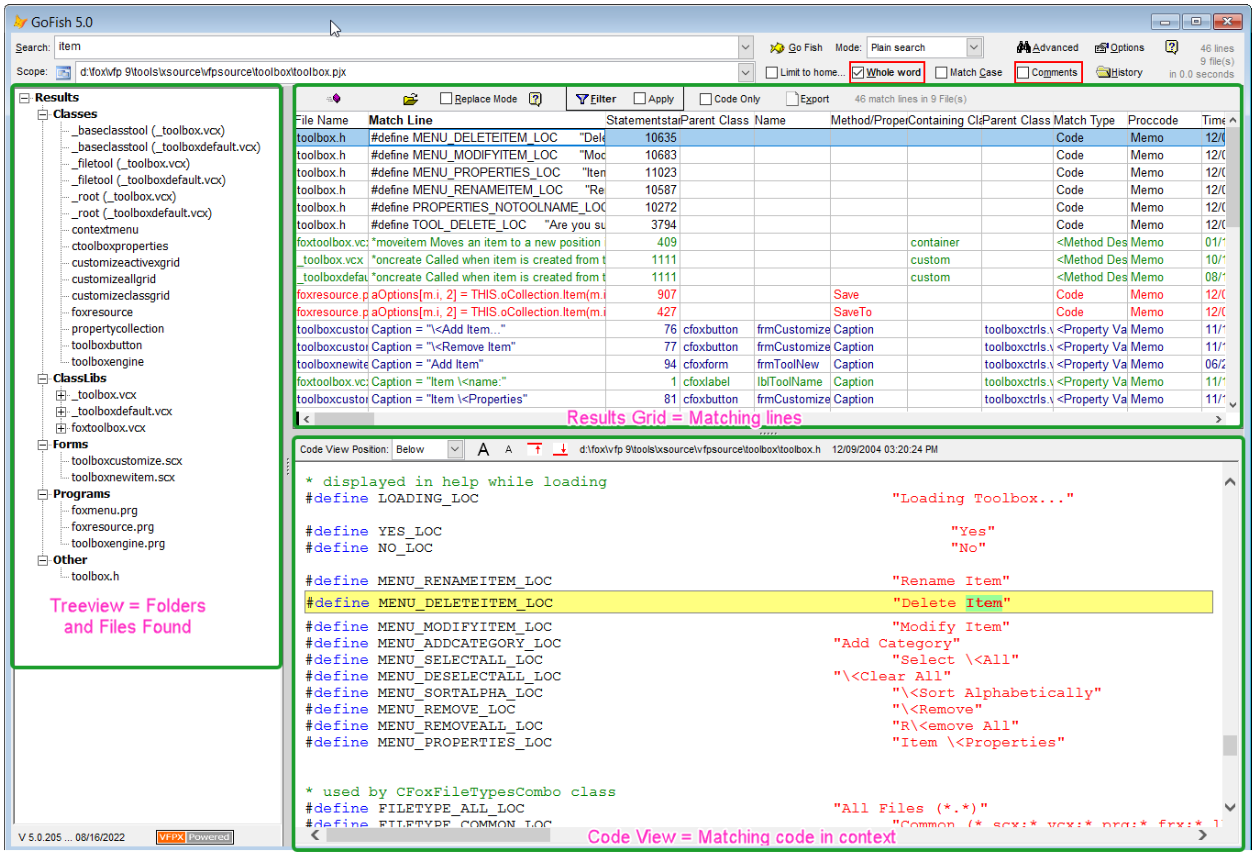
Scope: d:\fox\vfp 9\tools\xsource\xsource\vfpsource\toolbox\toolbox.pjx

Mode: Plain search

[ ] Limit to home... [✓ Whole word]

Figure 7. This search looks for the word “item” in the Toolbox project. Because the “Whole word” checkbox is checked, it looks only for “item” by itself, not as part of a longer word.

Click the GoFish button (or hit Enter) to begin the search. When it’s done, the rest of the form populates, as in **Figure 8**. There are three panes[4] (each highlighted and labelled in the Figure[5]) to show the search results. On the left, a treeview (the *Treeview*) shows a list of files and classes that contain matches. It’s organized by file type and each type can be collapsed and expanded.


GoFish 5.0
Go Fish
Advanced
Options
History
Treeview = Folders and Files Found
Results Grid = Matching lines
Code View = Matching code in context
V 5.0.205 ... 08/16/2022
VFPX Powered

Search: item

Scope: d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx

Mode: Plain search

[ ] Limit to home... [✓ Whole word] [ ] Match Case [ ] Comments

46 lines
9 file(s)
in 0.0 seconds

- Results
  - Classes
    - _baseclasstool (_toolbox.vcx)
    - _baseclasstool (_toolboxdefault.vcx)
    - _filetool (_toolbox.vcx)
    - _filetool (_toolboxdefault.vcx)
    - _root (_toolbox.vcx)
    - _root (_toolboxdefault.vcx)
    - contextmenu
    - ctoolboxproperties
    - customizeactivexgrid
    - customizeallgrid
    - customizeclassgrid
    - foxresource
    - propertycollection
    - toolboxbutton
    - toolboxengine
  - ClassLibs
    - _toolbox.vcx
    - _toolboxdefault.vcx
    - foxtoolbox.vcx
  - Forms
    - toolboxcustomize.scx
    - toolboxnewitem.scx
  - Programs
    - foxmenu.prg
    - foxresource.prg
    - toolboxengine.prg
  - Other
    - toolbox.h

[ ] Replace Mode | Filter [ ] Apply | [ ] Code Only | Export | 46 match lines in 9 File(s)

File Name	Match Line	Statementstar	Parent Class	Name	Method/Proper	Containing Cla	Parent Class	Match Type	Proccode	Time
toolbox.h	#define MENU_DELETEITEM_LOC "Dele	10635						Code	Memo	12/(
toolbox.h	#define MENU_MODIFYITEM_LOC "Moc	10683						Code	Memo	12/(
toolbox.h	#define MENU_PROPERTIES_LOC "Iten	11023						Code	Memo	12/(
toolbox.h	#define MENU_RENAMEITEM_LOC "Re	10587						Code	Memo	12/(
toolbox.h	#define PROPERTIES_NOTOOLNAME_LOC	10272						Code	Memo	12/(
toolbox.h	#define TOOL_DELETE_LOC "Are you su	3794						Code	Memo	12/(
foxtoolbox.vc:	*moveitem Moves an item to a new position	409				container		<Method Des	Memo	01/1
_toolbox.vcx	*oncreate Called when item is created from t	1111				custom		<Method Des	Memo	10/1
_toolboxdefau	*oncreate Called when item is created from t	1111				custom		<Method Des	Memo	08/1
foxresource.p	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	907			Save			Code	Memo	12/(
foxresource.p	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427			SaveTo			Code	Memo	12/(
toolboxcusto	Caption = "\<Add Item..."	76	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	Memo	11/1
toolboxcusto	Caption = "\<Remove Item"	77	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	Memo	11/1
toolboxnewite	Caption = "Add Item"	94	cfoxform	frmToolNew	Caption		toolboxctrls.\	<Property Va	Memo	06/2
foxtoolbox.vc:	Caption = "Item \<name:"	1	cfoxlabel	lblToolName	Caption		toolboxctrls.\	<Property Va	Memo	11/1
toolboxcusto	Caption = "Item \<Properties"	81	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	Memo	11/1

Code View Position: Below | d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.h 12/09/2004 03:20:24 PM

```
* displayed in help while loading
#define LOADING_LOC                                     "Loading Toolbox..."

#define YES_LOC                                                 "Yes"
#define NO_LOC                                                  "No"

#define MENU_RENAMEITEM_LOC                             "Rename Item"
#define MENU_DELETEITEM_LOC                             "Delete Item"
#define MENU_MODIFYITEM_LOC                             "Modify Item"
#define MENU_ADDCATEGORY_LOC                    "Add Category"
#define MENU_SELECTALL_LOC                              "Select \<All"
#define MENU_DESELECTALL_LOC                    "\<Clear All"
#define MENU_SORTALPHA_LOC                              "\<Sort Alphabetically"
#define MENU_REMOVE_LOC                                 "\<Remove"
#define MENU_REMOVEALL_LOC                              "R\<emove All"
#define MENU_PROPERTIES_LOC                             "Item \<Properties"


* used by CFoxFileTypesCombo class
#define FILETYPE_ALL_LOC                        "All Files (*.*)"
#define FILETYPE_COMMON_LOC                             "Common (*.scx;*.vcx;*.prg;*.frx;*.l
```

Figure 8. After a search, you can see the files that contain matches on the left, the specific lines containing matches in the middle on the right, and at the bottom right, the matching line in context for the highlighted item in the middle right.

There are some limits on GoFish’s ability to open the result items. For items in green, which are class definitions and items set in the Property Sheet, in general, the right editor is opened, but it’s not positioned on the specified item. In some cases, the editor doesn’t come

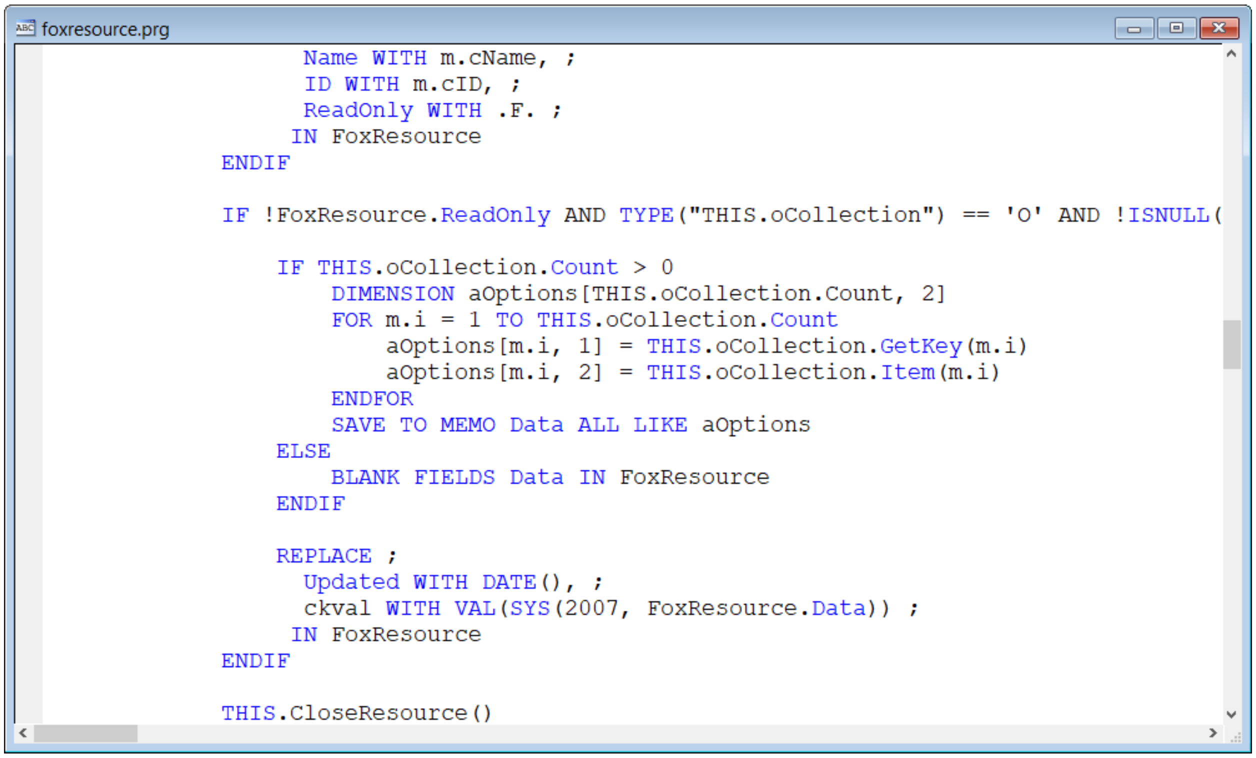
[4] Unfortunately, there’s no generally agreed on terminology for the panes of GoFish. The terms I use here are my own, with some influence from others in the VFP community.

[5] GoFish uses red for highlighting items that are in use. Therefore, I’ll use green to highlight items in GoFish in this paper.

to the front, so you have to notice that it's behind GoFish. In addition, if GoFish cannot open the item (because some error occurs while doing so), it doesn't tell you that.

The middle pane on the right (the *Results Grid*) contains a grid showing the individual lines that contain matches. You can click on the column headers to determine the sort order of the lines. As the figure shows, lines from different file types show in different colors.

Double-click any item in the Results Grid and the source item opens in its native editor, most often right at the line where the match was found. For example, if you double-click the first red line in **Figure 8**, FoxResource.PRG opens with the keyboard cursor on the matched line, as in **Figure 9**[6].



```
                 Name WITH m.cName, ;
                 ID WITH m.cID, ;
                 ReadOnly WITH .F. ;
                IN FoxResource
          ENDIF

          IF !FoxResource.ReadOnly AND TYPE("THIS.oCollection") == 'O' AND !ISNULL(

              IF THIS.oCollection.Count > 0
                  DIMENSION aOptions[THIS.oCollection.Count, 2]
                  FOR m.i = 1 TO THIS.oCollection.Count
                      aOptions[m.i, 1] = THIS.oCollection.GetKey(m.i)
                      aOptions[m.i, 2] = THIS.oCollection.Item(m.i)
                  ENDFOR
                  SAVE TO MEMO Data ALL LIKE aOptions
              ELSE
                  BLANK FIELDS Data IN FoxResource
              ENDIF

              REPLACE ;
                Updated WITH DATE(), ;
                ckval WITH VAL(SYS(2007, FoxResource.Data)) ;
               IN FoxResource
          ENDIF

          THIS.CloseResource()
```

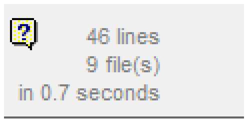
Figure 9. Double-click a result to open the source file, positioned appropriately.

The bottom pane on the right (the *Code View*) shows the code from whichever item is selected in the grid above[7] it. The line highlighted in the Results Grid is also highlighted in the Code View pane (in yellow) and the search string is highlighted (in green) wherever it appears[8].

[6] For some reason, the tool I use to capture images isn't picking up the keyboard cursor. In VFP, it's on the second line inside the For loop, which contains the search string "item."
[7] The positions of these panes can be changed; see "Customizing the GoFish User Interface" later in this paper.
[8] When I first wrote this section, there was a bug in the highlighting code for the bottom pane; it didn't respect the Whole word checkbox. That bug has been fixed, so if you see it, you're not on the latest version of GoFish.

Click on a different line in the Results Grid and the Code View shows that line in context. Click on a file type in the treeview and the Results Grid is limited to files of that type. Click on a specific file or class in the treeview and the Results Grid shows only results from that file.

On the right-hand side of the top pane (the *Selection Criteria*), you get a summary of how many matches were found in how many files. (The summary also says how long it took, no doubt because GoFish is far faster than the Code References tool that comes with VFP.) **Figure 10** shows the summary for our first search.



46 lines
9 file(s)
in 0.7 seconds

Figure 10. After each search, you get a summary of the results.

Once you've done a search, the search term is remembered. The Search entry is actually a dropdown combo rather than a textbox, so you can choose a recent search term and search it again (or modify it and search it again). **Figure 11** shows the dropdown after our initial search.

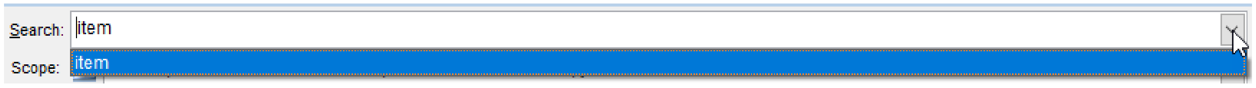
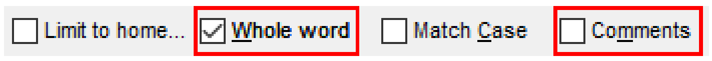


Figure 11. The Search dropdown remembers your most recently used search terms, making it easy to search for the same thing or something very similar.

Fine-tuning what to search for

As the mention earlier of the Whole Word checkbox indicates, there are several ways you can modify the search directly in the Selection Criteria pane. Those options are shown in **Figure 12**. The caption of the first checkbox changes depending whether you're searching in a project or a folder. For a project, it lets you limit the search to the home folder of the project and its subfolders (that is, omitting folders not in the project's directory tree, like those from a framework or shared library). For a folder, it lets you indicate whether to search subdirectories or just the project's home folder.



☐ Limit to home... ☑ Whole word ☐ Match Case ☐ Comments

Figure 12. You can make some simple modifications to the search right in the Selection Criteria pane.

Match Case, as you'd expect, indicates whether to find only matches that are exactly the same case as the search string. If you check it and repeat our initial search, you'll get only 9 lines in 7 files.

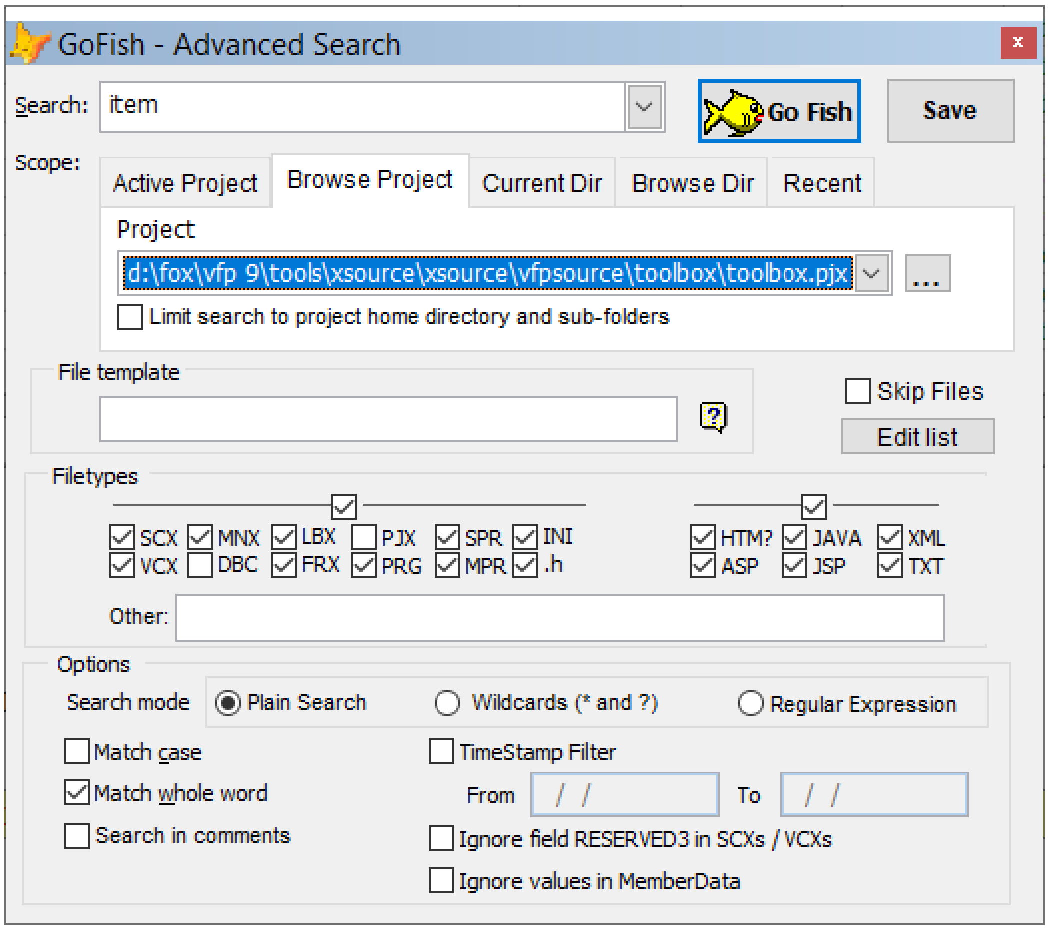
Comments indicates whether comments are included in the search; the default is to include them. (I wish there were an option to search comments only.)

As the figure shows, sometimes these checkboxes are surrounded by red outlines. The intention was to highlight any checked item would be highlighted, but there are some bugs in the version of GoFish I'm currently using. (I've reported the bugs.)

Fine-tuning where to search

You may have noticed that the Folders section of the Scope dropdown initially offers only the current folder. Once you've searched in some folders, they appear as well, but how do you search in a folder other than the current folder that you haven't search in before? The answer is to click the Advanced button, which opens the Advanced Search dialog shown in **Figure 13**. This form replicates many of the items in the Selection Criteria pane and offers several additional options.

The Scope section provides tabs to choose where to search. The Recent tab has a dropdown that echoes the one on the main GoFish form, while the Active Project and Current Dir tabs do what their names say and set the search in the active project or current folder, respectively. The real gains here are the Browse Project and Browse Dir tabs that let you go searching for the project or folder you want to search in. As **Figure 13** shows, the Browse Project tab includes the checkbox to limit the search to the project's folders. Similarly, the Browse Dir tab has a checkbox to indicate whether to search in subfolders. Most importantly, on each of these two, you can click the ellipsis (three dots) button and navigate to the desired search locale. Search locations you select here are added to the list of recent scopes.


GoFish - Advanced Search

Search: item

Scope: Active Project | [✓ Browse Project] | Current Dir | Browse Dir | Recent

Project: d:\fox\vfp 9\tools\xsource\xsource\vfpsource\toolbox\toolbox.pjx

☐ Limit search to project home directory and sub-folders

File template:

☐ Skip Files

Filetypes: ☑ SCX ☑ MNX ☑ LBX ☐ PJX ☑ SPR ☑ INI ☑ VCX ☐ DBC ☑ FRX ☑ PRG ☑ MPR ☑ .h ☑ HTM? ☑ JAVA ☑ XML ☑ ASP ☑ JSP ☑ TXT

Other:

Options

Search mode: [✓ Plain Search] Wildcards (* and ?) Regular Expression

☐ Match case

☑ Match whole word

☐ Search in comments

☐ TimeStamp Filter From / / To / /

☐ Ignore field RESERVED3 in SCXs / VCXs

☐ Ignore values in MemberData

Figure 13. The Advanced Search dialog makes it easier to choose projects and folders, and lets you fine-tune your search in other says.

The Advanced Search form also lets you limit the search in several ways. First, you can specify that only files whose names match a certain template should be searched. The template uses the asterisk wildcard and can have it at the beginning and/or end of the filestem (as in **Figure 14**) and the beginning and/or end of the extension. Click on the ? icon next to the File template textbox for more information.

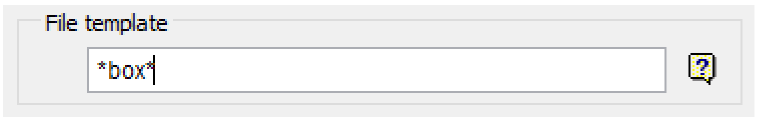

File template: *box*

Figure 14. The File template textbox lets you limit your search to a subset of files by name.

To the right of the File template section, a checkbox and button (shown in **Figure 15**) let you specify a list of files and file templates that should always be omitted from the search. (If you've worked with Mercurial or Git, this idea should be familiar, as it's similar to the ignore file those version control products use.)

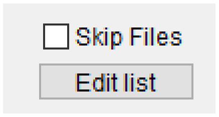
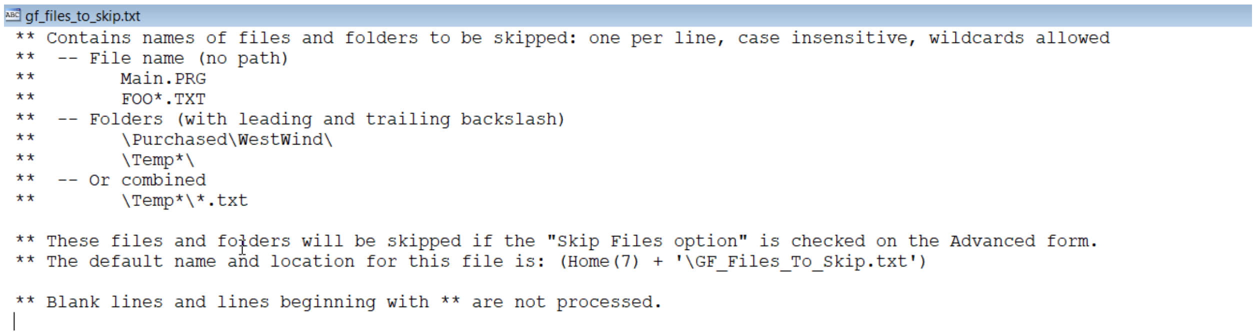


Figure 15. You can set up a master list of files and file templates that should always be skipped in search.

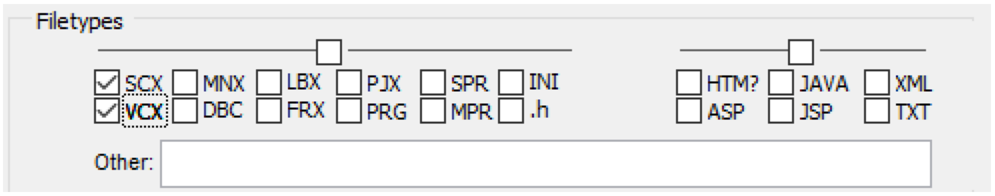
The text file that opens when you click Edit list gives you instructions on what to put in the file. It's shown in **Figure 16**. Keep in mind that the list is global, not project, folder or search specific.



```
** Contains names of files and folders to be skipped: one per line, case insensitive, wildcards allowed
**   -- File name (no path)
**        Main.PRG
**        FOO*.TXT
**   -- Folders (with leading and trailing backslash)
**        \Purchased\WestWind\
**        \Temp*\
**   -- Or combined
**        \Temp*\*.txt

** These files and folders will be skipped if the "Skip Files option" is checked on the Advanced form.
** The default name and location for this file is: (Home(7) + '\GF_Files_To_Skip.txt')

** Blank lines and lines beginning with ** are not processed.
```

Figure 16. Populate this file with files or file templates you always want to exclude from a search.

The checkboxes in the Filetypes section let you determine what kind of files are searched. For example, if you're only interested in searching in forms and classes, you can uncheck the checkboxes above the two main sections and then check the SCX and VCX checkboxes, as in **Figure 17**. (Be aware that if you specify a file template that includes an extension, the Filetypes section is disabled.)



Filetypes

☑ SCX ☐ MNX ☐ LBX ☐ PJX ☐ SPR ☐ INI ☐ HTM? ☐ JAVA ☐ XML
☑ VCX ☐ DBC ☐ FRX ☐ PRG ☐ MPR ☐ .h ☐ ASP ☐ JSP ☐ TXT

Other:

Figure 17. You can indicate what types of files to search.

Note the ability to specify file types for which there are no checkboxes. As the textbox's tooltip indicates, enter the desired extensions separated by spaces. GoFish seems to be able to do a text search in pretty much any file type and to open files in their native editors. However, it may or may not be able to display the file contents in the Code View pane, or to open the file to the found position in the native editor. (I tested with Word documents, and GoFish could not show the contents in the Code View pane nor position Word to the found string.)

You can also limit search in forms and class libraries to records whose timestamps are in a specified range, as in **Figure 18**. As you'd expect, you can specify both dates or just one. In the figure, we're limiting the search to those records changed since the start of 2022. Note that this filter doesn't find only changes made in that timeframe. Rather it searches for matches only in those records that were changed within the specified timeframe. (Remember that forms and class libraries are VFP tables with special extensions.)

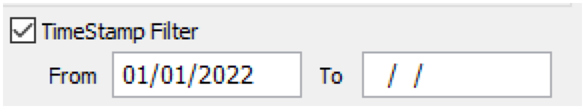


Figure 18. With the timestamp filter, you can search only those records in forms and classes that have been changed within the specified timeframe.

The last two options for limiting searches (shown in **Figure 19**) also address the contents of forms and class libraries. The Reserved3 field in SCX and VCX files contains information about the PEMs that have been added to the form or class, including any descriptions provided. Often, you're interested in the use of a PEM, not in its definition or text description, so you can shorten search results by omitting those. When I checked this checkbox and re-ran our initial search for "item" in the Toolbox project, the number of matches dropped from 46 to 43; three method descriptions include the word "item."

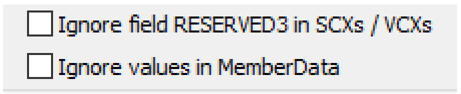

☐ Ignore field RESERVED3 in SCXs / VCXs
☐ Ignore values in MemberData

Figure 19. These two options let you eliminate information that's often extraneous from a search.

MemberData was introduced in VFP 9 to improve handling of PEMs. It allows you to specify capitalization of PEMs, put PEMs on the Favorites tab of the Property Sheet, and to create property editors (mini-builders) for PEMs. As with the information in Reserved3, most of the time, you're likely to want not to search within MemberData itself. (In our example, it makes no difference because the Toolbox code doesn't use MemberData.)

Fine-tuning how to search

By default, GoFish does a plain text search, looking for the exact string provided in the project or folder specified (subject, of course, the various conditions we've added to the search). But GoFish has two additional ways of searching.

You can use wildcards or regular expressions to specify the search string. In the main GoFish form, you choose among the three using the Mode dropdown, found next to the GoFish button in the Selection Criteria pane and shown in **Figure 20**. The Advanced Search dialog uses option buttons (**Figure 21**) to specify search mode.

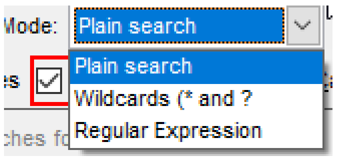


Figure 20. GoFish has three search modes: plain text, wildcard, and Regular Expression. This dropdown in the main GoFish form lets you choose.

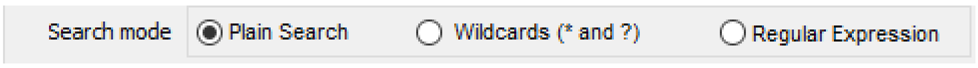
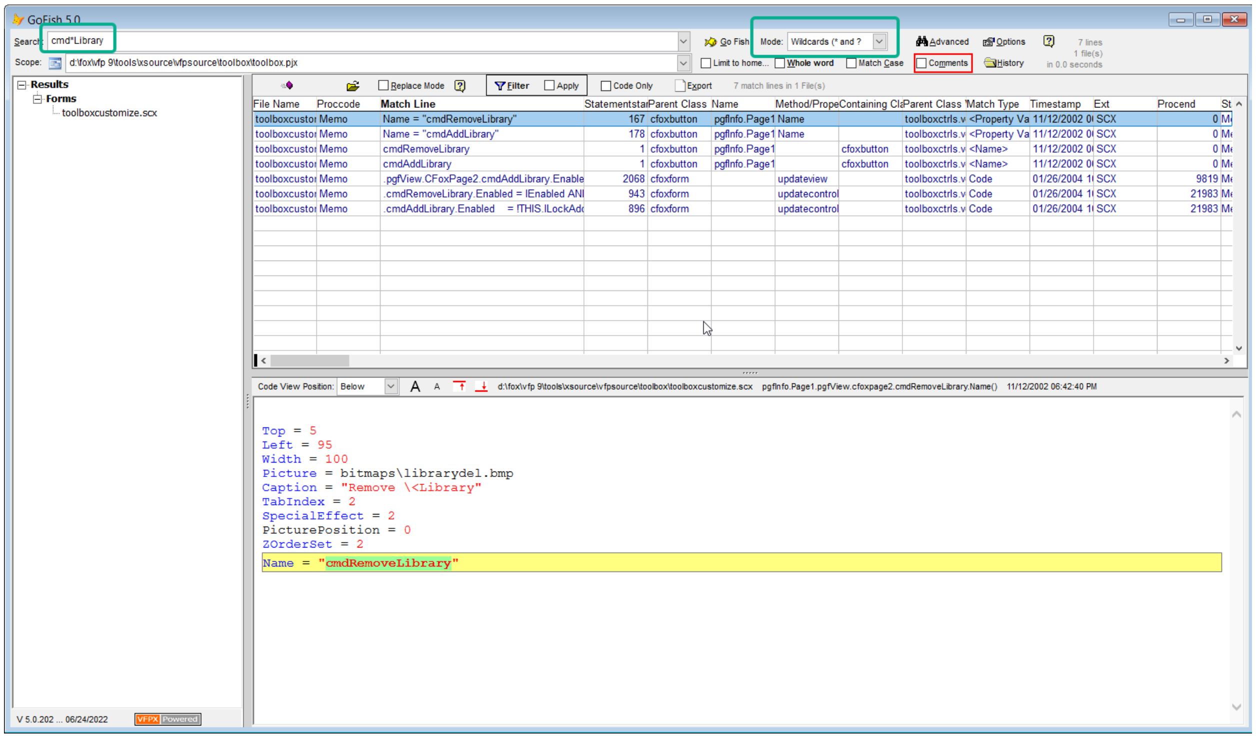

Search mode [✓ Plain Search] Wildcards (* and ?) Regular Expression

Figure 21. In the Advanced Search dialog, you choose search mode with these option buttons.

Wildcard search allows you to include the two common wildcard characters ("?" to specify a single character, "*" to specify zero or more consecutive characters) in the search string. In **Figure 22**, with Mode set to Wildcards and a search string of "cmd*Library", the results include uses of "cmdAddLibrary" and "cmdRemoveLibrary."



Search: cmd*Library

Scope: d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx

Mode: Wildcards (* and ?

Limit to home... Whole word Match Case Comments

7 lines
1 file(s)
in 0.0 seconds

7 match lines in 1 File(s)

File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Prope	Containing Cl	Parent Class '	Match Type	Timestamp	Ext	Procend	St
toolboxcusto	Memo	Name = "cmdRemoveLibrary"	167	cfoxbutton	pgfInfo.Page1	Name		toolboxctrls.v	<Property Va	11/12/2002 0	SCX	0	M
toolboxcusto	Memo	Name = "cmdAddLibrary"	178	cfoxbutton	pgfInfo.Page1	Name		toolboxctrls.v	<Property Va	11/12/2002 0	SCX	0	M
toolboxcusto	Memo	cmdRemoveLibrary	1	cfoxbutton	pgfInfo.Page1		cfoxbutton	toolboxctrls.v	<Name>	11/12/2002 0	SCX	0	M
toolboxcusto	Memo	cmdAddLibrary	1	cfoxbutton	pgfInfo.Page1		cfoxbutton	toolboxctrls.v	<Name>	11/12/2002 0	SCX	0	M
toolboxcusto	Memo	.pgfView.CFoxPage2.cmdAddLibrary.Enable	2068	cfoxform		updateview		toolboxctrls.v	Code	01/26/2004 1	SCX	9819	M
toolboxcusto	Memo	.cmdRemoveLibrary.Enabled = lEnabled ANI	943	cfoxform		updatecontrol		toolboxctrls.v	Code	01/26/2004 1	SCX	21983	M
toolboxcusto	Memo	.cmdAddLibrary.Enabled = !THIS.lLockAd	896	cfoxform		updatecontrol		toolboxctrls.v	Code	01/26/2004 1	SCX	21983	M

Code View Position: Below

d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolboxcustomize.scx pgfInfo.Page1.pgfView.cfoxpage2.cmdRemoveLibrary.Name() 11/12/2002 06:42:40 PM

```
Top = 5
Left = 95
Width = 100
Picture = bitmaps\librarydel.bmp
Caption = "Remove \<Library"
TabIndex = 2
SpecialEffect = 2
PicturePosition = 0
ZOrderSet = 2
Name = "cmdRemoveLibrary"
```

Figure 22. This wildcard search finds strings that begin with "cmd" and end with "Library".

Regular expressions are a big topic, too big to fully explain in this paper. Briefly, they are a way to specify a pattern for the strings you want to find. You might think of them as wildcards on steroids. (For a good introduction to regular expressions, check out https://www.regular-expressions.info/.)

When you choose Regular Expression from the Mode dropdown on the main GoFish form, an icon appears next to the dropdown. Click on it for not only a bit of cheat sheet for regular expressions, but help entering them in the search string. Choose an item in the pop-up menu that appears (shown in **Figure 23**) and it's added to the Search textbox. Note that choosing Regular Expression doesn't clear the Search textbox, so unless you do so manually, you're adding to whatever is already there.

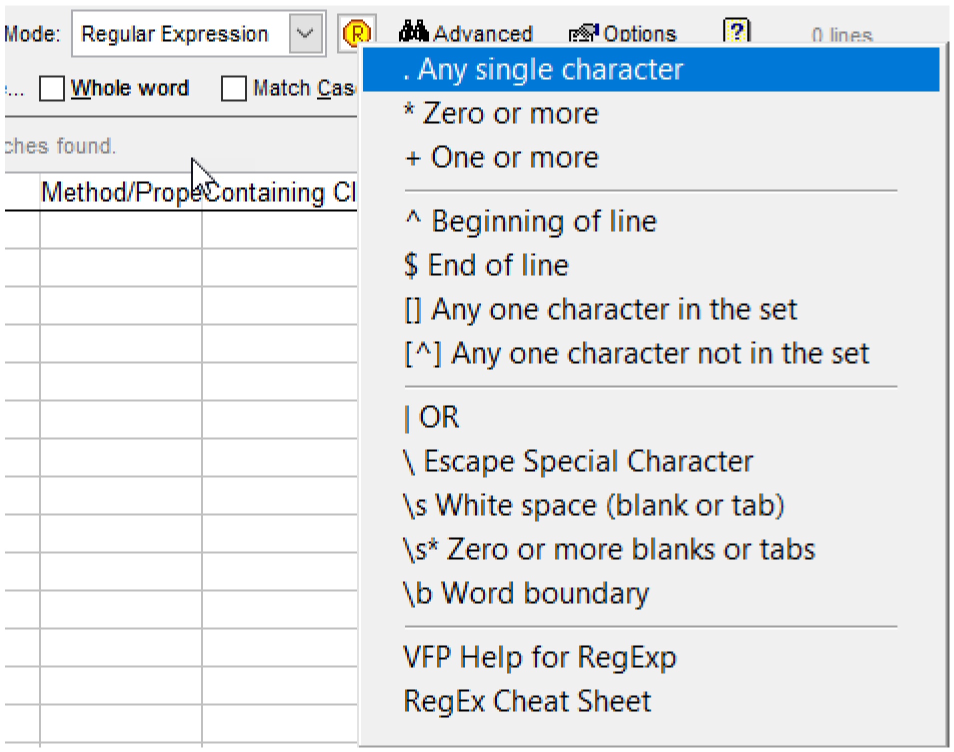


Figure 23. When you choose regular expressions, the main GoFish form offers guidance.

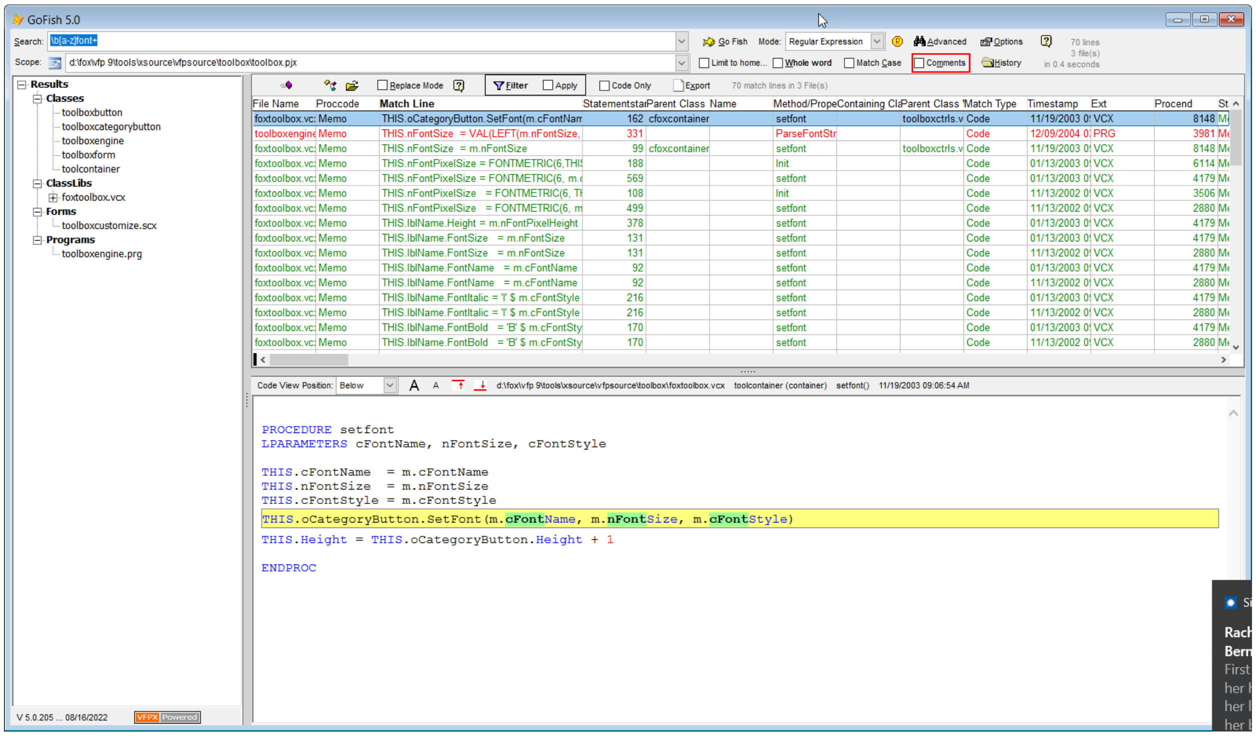
For example, the search in **Figure 24** finds references to font-related properties and variables. The search string is:

\b[a-z]font+

Breaking it down there are four components, shown in **Table 1**. Put together, the search string says find words that begin with a single letter, then have the word "font" and then have at least one more character. Thus, this search finds things like "nFontSize" and "cFontName", but doesn't find method names like "ParseFontString" or function names like FontMetric" or control names like "txtFontString". (I'm not that familiar with regular expressions, so suspect that this search is imperfect and someone with more experience could probably create a better search string for this purpose.)

Table 1. The search string in the example has four elements.

Element	**Meaning**
\b	Look only at the beginning of words
[a-z]	Look for a single alphabetic character. The square brackets mean to look for exactly one of the listed items. A-Z (the case doesn't matter) indicates that the letters of the alphabet are the list of items.
font	Look for the exact string "font".
+	Look for one or more characters



Search: \b[a-z]font+

Scope: d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx

Mode: Regular Expression

Limit to home... Whole word Match Case Comments

70 lines
3 file(s)
in 0.4 seconds

70 match lines in 3 File(s)

File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Prope	Containing Cla	Parent Class	Match Type	Timestamp	Ext	Procend	St
foxtoolbox.vc:	Memo	THIS.oCategoryButton.SetFont(m.cFontNam	162	cfoxcontainer		setfont		toolboxctrls.v	Code	11/19/2003 0!	VCX	8148	Me
toolboxengine	Memo	THIS.nFontSize = VAL(LEFT(m.nFontSize,	331			ParseFontStr			Code	12/09/2004 0:	PRG	3981	Me
foxtoolbox.vc:	Memo	THIS.nFontSize = m.nFontSize	99	cfoxcontainer		setfont		toolboxctrls.v	Code	11/19/2003 0!	VCX	8148	Me
foxtoolbox.vc:	Memo	THIS.nFontPixelSize = FONTMETRIC(6,THIS	188			Init			Code	01/13/2003 0!	VCX	6114	Me
foxtoolbox.vc:	Memo	THIS.nFontPixelSize = FONTMETRIC(6, m.c	569			setfont			Code	01/13/2003 0!	VCX	4179	Me
foxtoolbox.vc:	Memo	THIS.nFontPixelSize = FONTMETRIC(6, TH	108			Init			Code	11/13/2002 0!	VCX	3506	Me
foxtoolbox.vc:	Memo	THIS.nFontPixelSize = FONTMETRIC(6, m	499			setfont			Code	11/13/2002 0!	VCX	2880	Me
foxtoolbox.vc:	Memo	THIS.lblName.Height = m.nFontPixelHeight	378			setfont			Code	01/13/2003 0!	VCX	4179	Me
foxtoolbox.vc:	Memo	THIS.lblName.FontSize = m.nFontSize	131			setfont			Code	01/13/2003 0!	VCX	4179	Me
foxtoolbox.vc:	Memo	THIS.lblName.FontSize = m.nFontSize	131			setfont			Code	11/13/2002 0!	VCX	2880	Me
foxtoolbox.vc:	Memo	THIS.lblName.FontName = m.cFontName	92			setfont			Code	01/13/2003 0!	VCX	4179	Me
foxtoolbox.vc:	Memo	THIS.lblName.FontName = m.cFontName	92			setfont			Code	11/13/2002 0!	VCX	2880	Me
foxtoolbox.vc:	Memo	THIS.lblName.FontItalic = 'I' $ m.cFontStyle	216			setfont			Code	01/13/2003 0!	VCX	4179	Me
foxtoolbox.vc:	Memo	THIS.lblName.FontItalic = 'I' $ m.cFontStyle	216			setfont			Code	11/13/2002 0!	VCX	2880	Me
foxtoolbox.vc:	Memo	THIS.lblName.FontBold = 'B' $ m.cFontSty	170			setfont			Code	01/13/2003 0!	VCX	4179	Me
foxtoolbox.vc:	Memo	THIS.lblName.FontBold = 'B' $ m.cFontSty	170			setfont			Code	11/13/2002 0!	VCX	2880	Me

Code View Position: Below   d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\foxtoolbox.vcx   toolcontainer (container)   setfont()   11/19/2003 09:06:54 AM

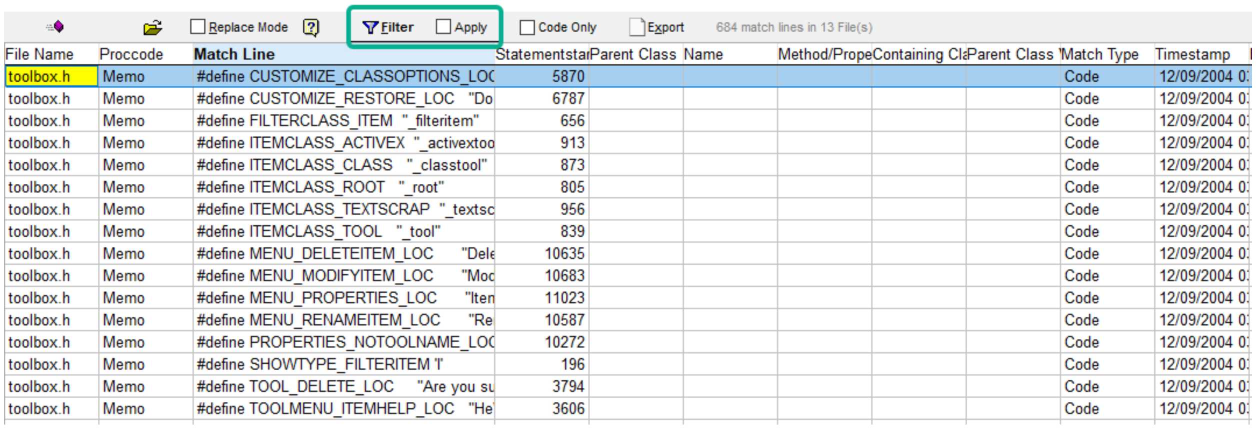
```
PROCEDURE setfont
LPARAMETERS cFontName, nFontSize, cFontStyle

THIS.cFontName  = m.cFontName
THIS.nFontSize  = m.nFontSize
THIS.cFontStyle = m.cFontStyle
THIS.oCategoryButton.SetFont(m.cFontName, m.nFontSize, m.cFontStyle)
THIS.Height = THIS.oCategoryButton.Height + 1

ENDPROC
```

Figure 24. Regular expressions let you do complex searches.

Filtering search results

Once you've done a search, GoFish lets you narrow down the results even more via the Filter button above the Results grid, shown in **Figure 25**. (Note that I've returned here to our original search for the word "item" in the Toolbox project.) When you click the button, the GoFish Filter Builder (**Figure 26**) appears. It has six tabs that roughly break down into three groups: nearby code (the Code tab), where it occurs (the Classes, Names, and Files tabs), and the result type (the File Types and Match Types) tabs.

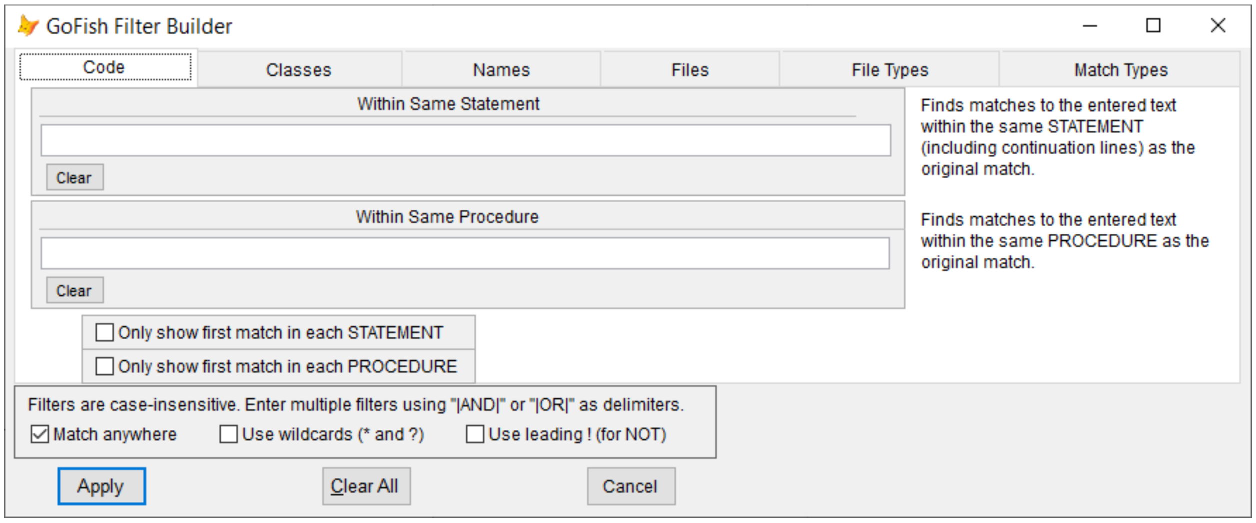
Be aware that filters are additive across tabs, so if you specify information in more than one tab, the results will be filtered to only those that match all the conditions you specify. The label of any tab that has conditions specified is shown in red bold type, so it's easier to get it right.



684 match lines in 13 File(s)

File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Prope	Containing Cla	Parent Class	Match Type	Timestamp
toolbox.h	Memo	#define CUSTOMIZE_CLASSOPTIONS_LOC	5870						Code	12/09/2004 0
toolbox.h	Memo	#define CUSTOMIZE_RESTORE_LOC "Do	6787						Code	12/09/2004 0
toolbox.h	Memo	#define FILTERCLASS_ITEM "_filteritem"	656						Code	12/09/2004 0
toolbox.h	Memo	#define ITEMCLASS_ACTIVEX "_activextoo	913						Code	12/09/2004 0
toolbox.h	Memo	#define ITEMCLASS_CLASS "_classtool"	873						Code	12/09/2004 0
toolbox.h	Memo	#define ITEMCLASS_ROOT "_root"	805						Code	12/09/2004 0
toolbox.h	Memo	#define ITEMCLASS_TEXTSCRAP "_textsc	956						Code	12/09/2004 0
toolbox.h	Memo	#define ITEMCLASS_TOOL "_tool"	839						Code	12/09/2004 0
toolbox.h	Memo	#define MENU_DELETEITEM_LOC "Dele	10635						Code	12/09/2004 0
toolbox.h	Memo	#define MENU_MODIFYITEM_LOC "Moc	10683						Code	12/09/2004 0
toolbox.h	Memo	#define MENU_PROPERTIES_LOC "Iten	11023						Code	12/09/2004 0
toolbox.h	Memo	#define MENU_RENAMEITEM_LOC "Re	10587						Code	12/09/2004 0
toolbox.h	Memo	#define PROPERTIES_NOTOOLNAME_LOC	10272						Code	12/09/2004 0
toolbox.h	Memo	#define SHOWTYPE_FILTERITEM 'I'	196						Code	12/09/2004 0
toolbox.h	Memo	#define TOOL_DELETE_LOC "Are you su	3794						Code	12/09/2004 0
toolbox.h	Memo	#define TOOLMENU_ITEMHELP_LOC "He	3606						Code	12/09/2004 0

Figure 25. The Filter button lets you narrow down results.



Finds matches to the entered text within the same STATEMENT (including continuation lines) as the original match.

Finds matches to the entered text within the same PROCEDURE as the original match.

☐ Only show first match in each STATEMENT

☐ Only show first match in each PROCEDURE

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

☑ Match anywhere ☐ Use wildcards (* and ?) ☐ Use leading ! (for NOT)

Figure 26. The GoFish Filter Builder lets you narrow down search results using a variety of criteria.

Filtering on result type

The last two tabs (File Types and Match Types) are the easiest to understand. File Types lets you filter in or out different file types. Types that don't appear in your existing results are disabled. **Figure 27** shows the File Types tab for the search for "item"; all results are in SCX, VCX, or PRG files. For example, we can filter down to just those in PRG files by checking the PRG checkbox and clicking Apply. In this case, shown in **Figure 28**, we reduce the result set to 9 lines in 3 files. That information appears on the row above the Results grid and is highlighted in the figure. (While we could have done the same filtering before searching, we may not have known which file types mattered until we saw the initial search results.)

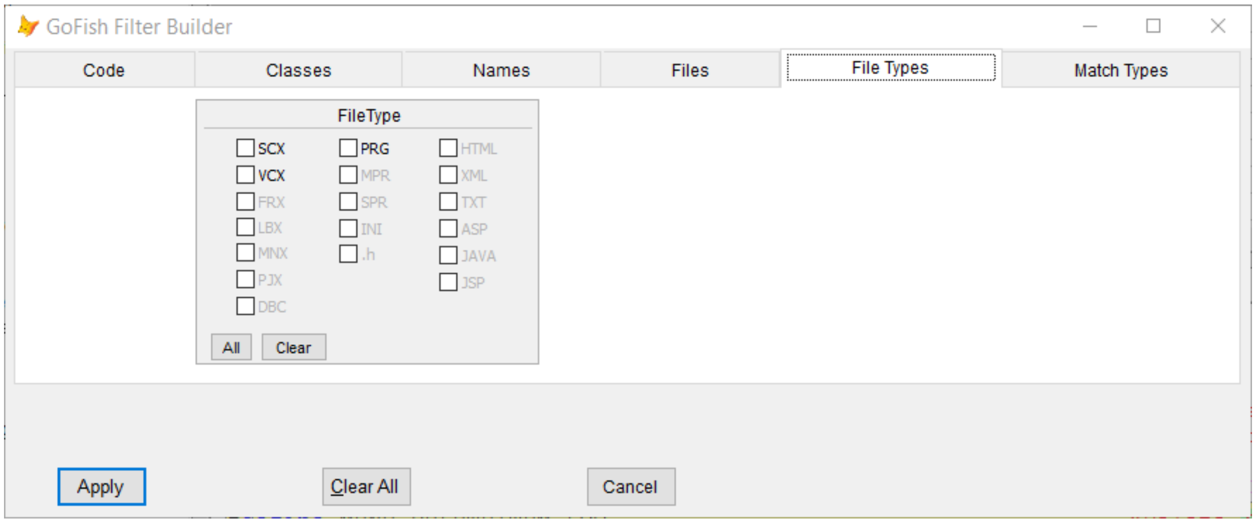
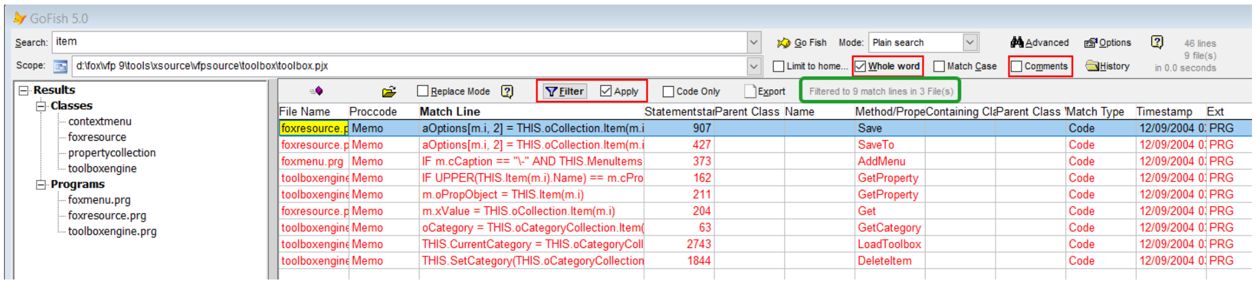


Figure 27. The File Types tab lets you limit search results based on the type of file they are in.



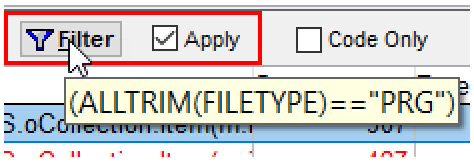
Search: item
Scope: d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx
Mode: Plain search
☐ Limit to home... ☑ Whole word ☐ Match Case ☐ Comments
46 lines
9 file(s)
in 0.0 seconds
☑ Apply ☐ Code Only
Filtered to 9 match lines in 3 File(s)

File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Prope	Containing Cl	Parent Class '	Match Type	Timestamp	Ext
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	907			Save			Code	12/09/2004 0:	PRG
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427			SaveTo			Code	12/09/2004 0:	PRG
foxmenu.prg	Memo	IF m.cCaption == "\-" AND THIS.MenuItems	373			AddMenu			Code	12/09/2004 0:	PRG
toolboxengine	Memo	IF UPPER(THIS.Item(m.i).Name) == m.cPro	162			GetProperty			Code	12/09/2004 0:	PRG
toolboxengine	Memo	m.oPropObject = THIS.Item(m.i)	211			GetProperty			Code	12/09/2004 0:	PRG
foxresource.p	Memo	m.xValue = THIS.oCollection.Item(m.i)	204			Get			Code	12/09/2004 0:	PRG
toolboxengine	Memo	oCategory = THIS.oCategoryCollection.Item(	63			GetCategory			Code	12/09/2004 0:	PRG
toolboxengine	Memo	THIS.CurrentCategory = THIS.oCategoryColl	2743			LoadToolbox			Code	12/09/2004 0:	PRG
toolboxengine	Memo	THIS.SetCategory(THIS.oCategoryCollection	1844			DeleteItem			Code	12/09/2004 0:	PRG

Figure 28. Filtering on file type can cut the list of results way down.

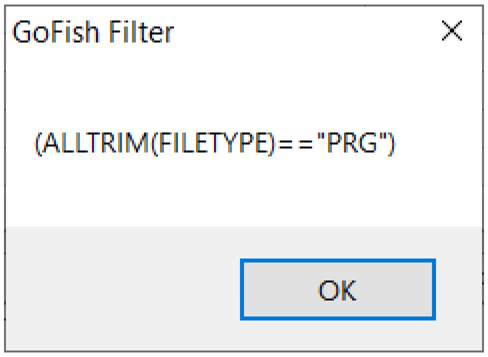
Note also in **Figure 28** that the Apply checkbox next to the Filter button is now set. You can use that checkbox to easily switch between the original result set and the filtered results.

Once you've set a filter, you can see the expression GoFish is filtering on by hovering over the Filter button, as in **Figure 29**. Right-clicking the filter button opens a messagebox showing the filter expression. As with any VFP messagebox, you can use CTRL+C to put the messagebox contents onto the clipboard and then pare it down to just the filter expression. **Figure 31** shows what lands on the clipboard in this example.



```
(ALLTRIM(FILETYPE)=="PRG")
```

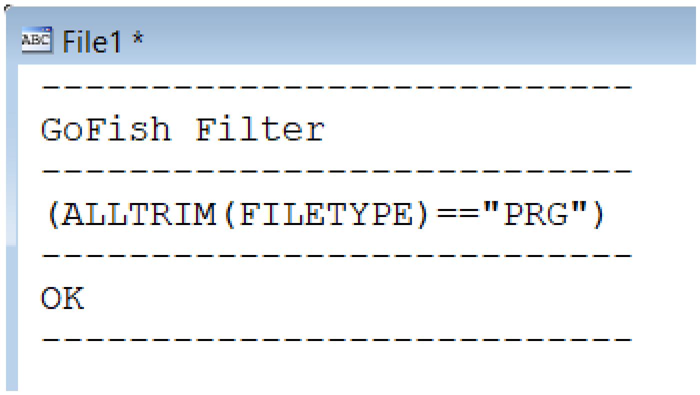
Figure 29. Hovering over the Filter button shows you the current filter expression.



GoFish Filter

(ALLTRIM(FILETYPE)=="PRG")

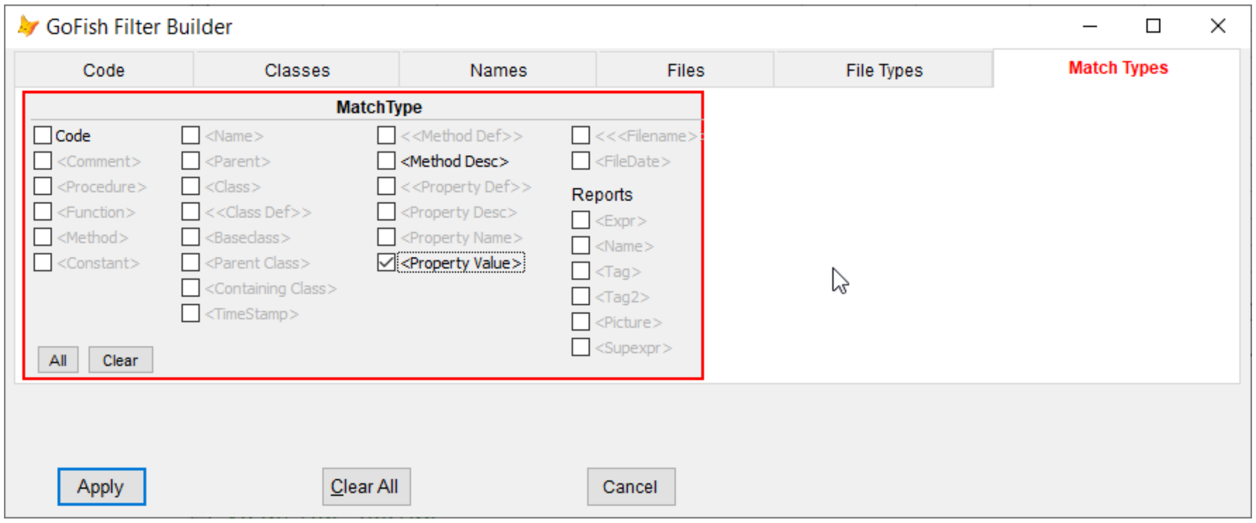
Figure 30. Right-clicking the Filter button displays a messagebox showing the current filter expression.



```
---------------------------
GoFish Filter
---------------------------
(ALLTRIM(FILETYPE)=="PRG")
---------------------------
OK
---------------------------
```

Figure 31. When a VFP messagebox is displayed, Ctrl+C puts the messagebox contents (including the title and buttons) onto the clipboard, from which you can paste it into a window.

The Match Types tab (shown in **Figure 32** with "<Property Value>" checked) filters based on the location of the string within the matched file. Most of the choices here relate to fields in SCX and VCX files. The "Code" item takes in code wherever it appears: in a PRG, in a method, in a stored procedure, and so on. As on the File Types tab, choices irrelevant to the current search results are disabled. **Figure 33** shows the results grid after applying the filter in **Figure 32**. The previous File Types filter was removed before applying this filter. (You can tell because only the Match Types caption is red bold.)

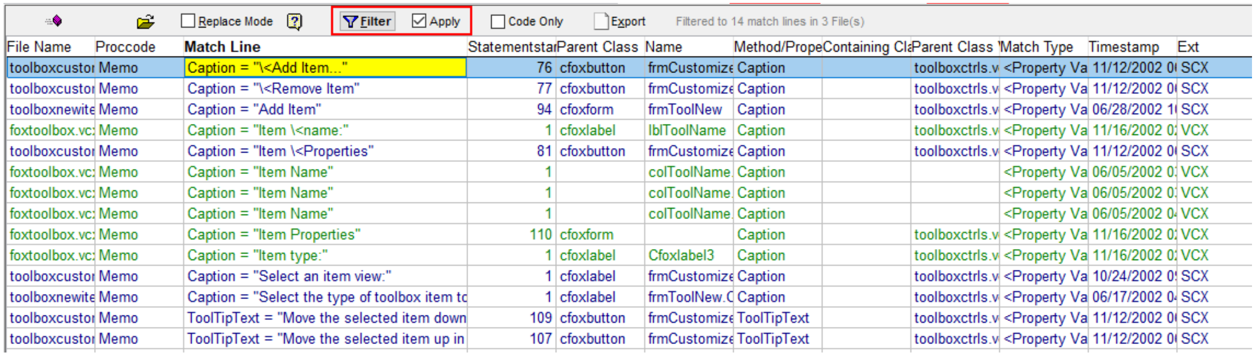


- [ ] Code
- [ ] <Comment>
- [ ] <Procedure>
- [ ] <Function>
- [ ] <Method>
- [ ] <Constant>
- [ ] <Name>
- [ ] <Parent>
- [ ] <Class>
- [ ] <<Class Def>>
- [ ] <Baseclass>
- [ ] <Parent Class>
- [ ] <Containing Class>
- [ ] <TimeStamp>
- [ ] <<Method Def>>
- [ ] <Method Desc>
- [ ] <<Property Def>>
- [ ] <Property Desc>
- [ ] <Property Name>
- [✓] <Property Value>
- [ ] <<<Filename>
- [ ] <FileDate>

Reports

- [ ] <Expr>
- [ ] <Name>
- [ ] <Tag>
- [ ] <Tag2>
- [ ] <Picture>
- [ ] <Supexpr>

Figure 32. On the Match Types tab, you can filter based on where the string appears within a file.



[ ] Replace Mode   Filter   [✓] Apply   [ ] Code Only   Filtered to 14 match lines in 3 File(s)

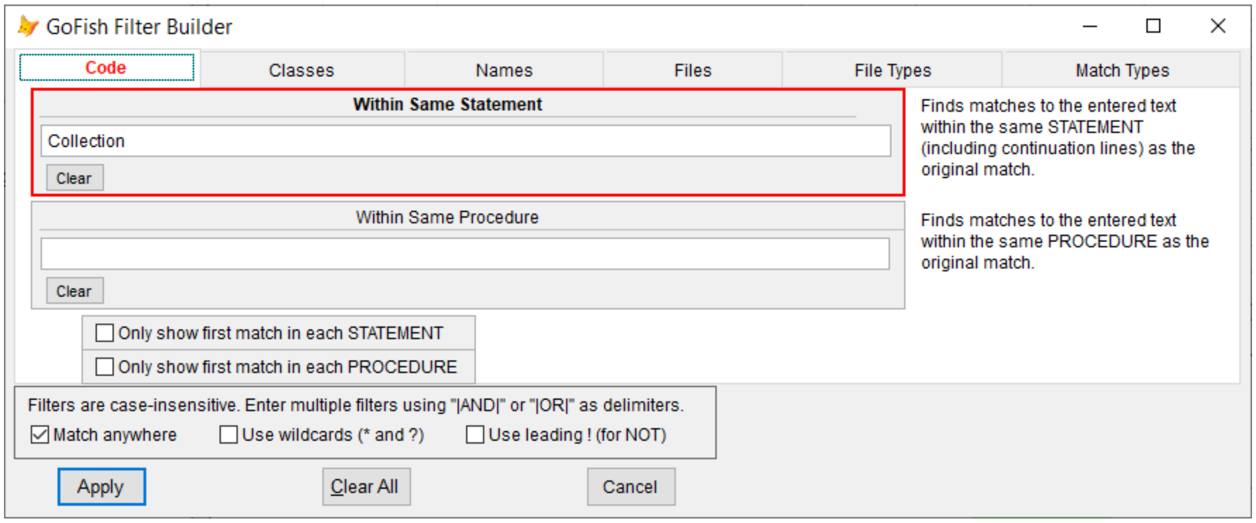
File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Prope	Containing Cla	Parent Class '	Match Type	Timestamp	Ext
toolboxcustor	Memo	Caption = "\<Add Item..."	76	cfoxbutton	frmCustomize	Caption		toolboxctrls.v	<Property Va	11/12/2002 0	SCX
toolboxcustor	Memo	Caption = "\<Remove Item"	77	cfoxbutton	frmCustomize	Caption		toolboxctrls.v	<Property Va	11/12/2002 0	SCX
toolboxnewite	Memo	Caption = "Add Item"	94	cfoxform	frmToolNew	Caption		toolboxctrls.v	<Property Va	06/28/2002 1	SCX
foxtoolbox.vc	Memo	Caption = "Item \<name:"	1	cfoxlabel	lblToolName	Caption		toolboxctrls.v	<Property Va	11/16/2002 0	VCX
toolboxcustor	Memo	Caption = "Item \<Properties"	81	cfoxbutton	frmCustomize	Caption		toolboxctrls.v	<Property Va	11/12/2002 0	SCX
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/05/2002 0	VCX
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/05/2002 0	VCX
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/05/2002 0	VCX
foxtoolbox.vc	Memo	Caption = "Item Properties"	110	cfoxform		Caption		toolboxctrls.v	<Property Va	11/16/2002 0	VCX
foxtoolbox.vc	Memo	Caption = "Item type:"	1	cfoxlabel	Cfoxlabel3	Caption		toolboxctrls.v	<Property Va	11/16/2002 0	VCX
toolboxcustor	Memo	Caption = "Select an item view:"	1	cfoxlabel	frmCustomize	Caption		toolboxctrls.v	<Property Va	10/24/2002 0	SCX
toolboxnewite	Memo	Caption = "Select the type of toolbox item tc	1	cfoxlabel	frmToolNew.C	Caption		toolboxctrls.v	<Property Va	06/17/2002 0	SCX
toolboxcustor	Memo	ToolTipText = "Move the selected item down	109	cfoxbutton	frmCustomize	ToolTipText		toolboxctrls.v	<Property Va	11/12/2002 0	SCX
toolboxcustor	Memo	ToolTipText = "Move the selected item up in	107	cfoxbutton	frmCustomize	ToolTipText		toolboxctrls.v	<Property Va	11/12/2002 0	SCX

Figure 33. Here, we limited results to only those specified as part of property values.

Filtering on nearby code

The first tab in the GoFish Filter Builder gives you four ways to filter results based on the code itself. **Figure 34** shows the first; you can request only matches that also contain another specified string. In the figure, we ask for only those matches that have the string “collection” in the same statement. As elsewhere in GoFish, things you’re using get red highlights. In this case, the whole section of the page is highlighted with red. With this filter (and previous filters removed), we’re down 20 matches in 4 files, shown in **Figure 35**.

Note that the filter is based on the statement, not on the line. So, if you have a statement spread across many lines with continuation characters, the filter looks at the whole statement, not just the part that’s on the same line as the original match.


GoFish Filter Builder

Code | Classes | Names | Files | File Types | Match Types

Within Same Statement

Collection

Clear

Finds matches to the entered text within the same STATEMENT (including continuation lines) as the original match.

Within Same Procedure

Clear

Finds matches to the entered text within the same PROCEDURE as the original match.

☐ Only show first match in each STATEMENT

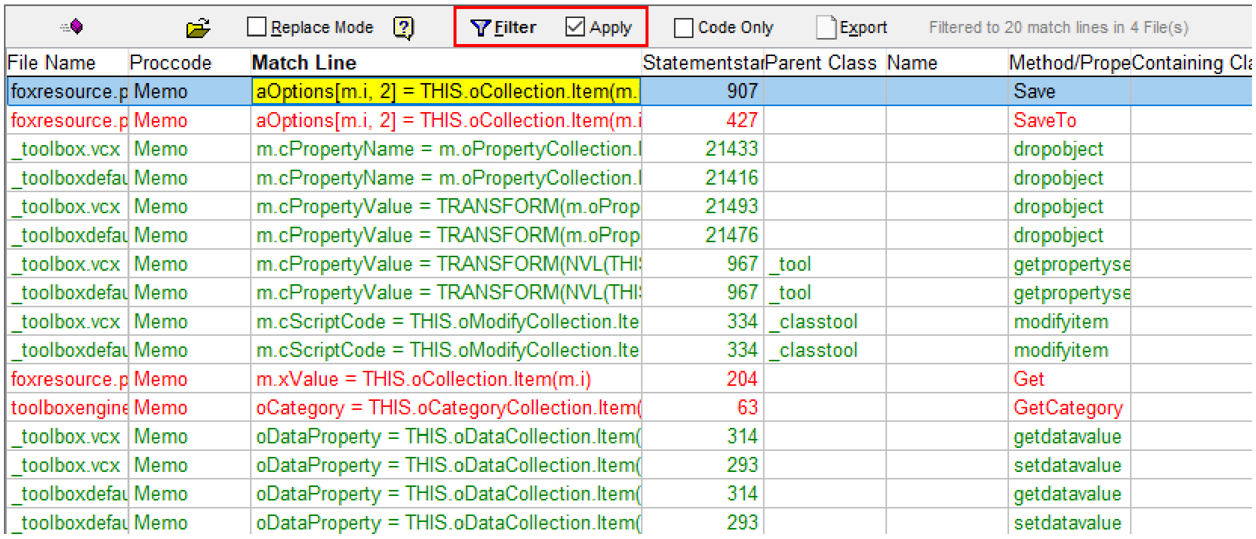
☐ Only show first match in each PROCEDURE

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

☑ Match anywhere ☐ Use wildcards (* and ?) ☐ Use leading ! (for NOT)

Apply | Clear All | Cancel

Figure 34. The Code tab lets you filter results based on code itself.

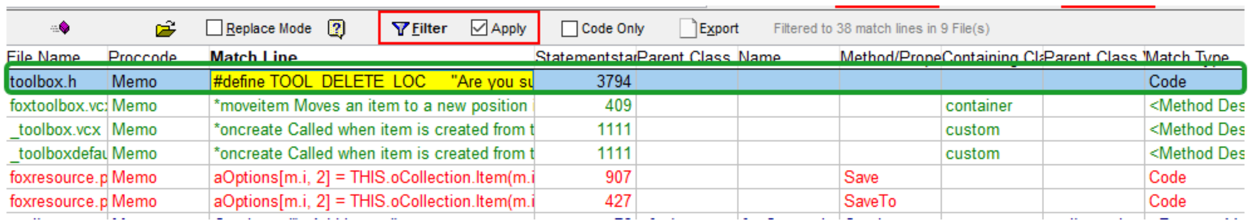

☐ Replace Mode | Filter | ☑ Apply | ☐ Code Only | Export | Filtered to 20 match lines in 4 File(s)

File Name	Proccode	**Match Line**	Statementstai	Parent Class	Name	Method/Prope	Containing Cla
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.	907			Save	
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427			SaveTo	
_toolbox.vcx	Memo	m.cPropertyName = m.oPropertyCollection.I	21433			dropobject	
_toolboxdefau	Memo	m.cPropertyName = m.oPropertyCollection.I	21416			dropobject	
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21493			dropobject	
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21476			dropobject	
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse	
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse	
_toolbox.vcx	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool		modifyitem	
_toolboxdefau	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool		modifyitem	
foxresource.p	Memo	m.xValue = THIS.oCollection.Item(m.i)	204			Get	
toolboxengine	Memo	oCategory = THIS.oCategoryCollection.Item(	63			GetCategory	
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	314			getdatavalue	
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	293			setdatavalue	
_toolboxdefau	Memo	oDataProperty = THIS.oDataCollection.Item(	314			getdatavalue	
_toolboxdefau	Memo	oDataProperty = THIS.oDataCollection.Item(	293			setdatavalue	

Figure 35. One way to filter based on code is to look for statements that also contain another string.

The second section of this tab filters based on matching a string anywhere in the procedure as the original match. “Procedure” here means PRG, method, function or procedure within a PRG, stored procedure, or any block of code that VFP programmers would see as a routine.

The pair of checkboxes in the middle of the Code tab let you limit repeated results. Use these if it’s enough to just find the unique locations where your search string appears. For example, with our (otherwise unfiltered) “item” example, checking the second checkbox (“Only show first match in each PROCEDURE”) reduces the list of results from Toolbox.h to a single item, as in **Figure 36**.



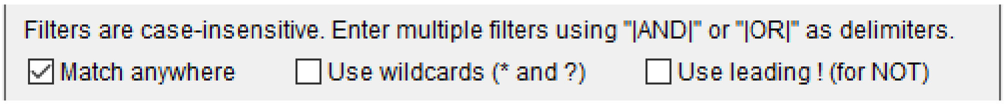
Filtered to 38 match lines in 9 File(s)

File Name	Proccode	Match Line	Statementsta	Parent Class	Name	Method/Prope	Containing Cl	Parent Class	Match Type
toolbox.h	Memo	#define TOOL_DELETE_LOC "Are you su	3794						Code
foxtoolbox.vc:	Memo	*moveitem Moves an item to a new position	409				container		<Method Des
_toolbox.vcx	Memo	*oncreate Called when item is created from t	1111				custom		<Method Des
_toolboxdefau	Memo	*oncreate Called when item is created from t	1111				custom		<Method Des
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	907			Save			Code
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427			SaveTo			Code

Figure 36. You can filter out repeated uses of the search string in a statement or, as in this example, in a routine.

Fine-tuning filters

The three checkboxes at the bottom of Filter Builder (shown in **Figure 37**) indicate how string-based filters are to be applied, while the text above them specifies how to combine multiple conditions. If you uncheck the first ("Match anywhere"), you include only matches that occur at the beginning of a line for "Within Same Statement" or at the beginning of the routine for "Within Same Procedure." That latter behavior makes it hard for me to think of many times when I'd use "Match anywhere" for "Within Same Procedure". However, unless you also check the Use Wildcards checkbox, "Match anywhere" doesn't work; the filter it generates can never find any records.[9]



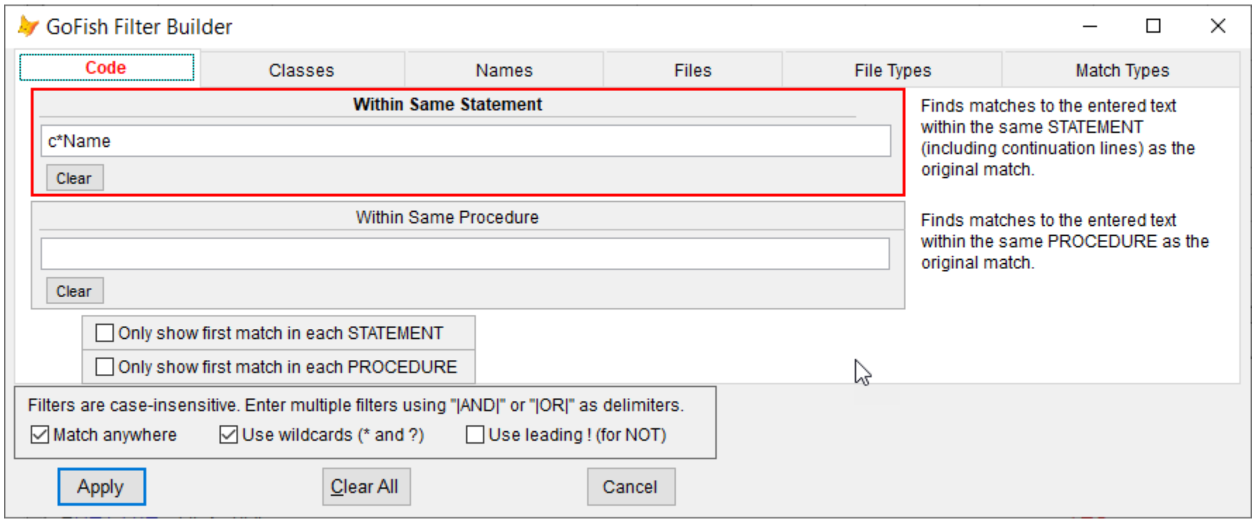
Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

[✓] Match anywhere   [ ] Use wildcards (* and ?)   [ ] Use leading ! (for NOT)

Figure 37. You can use wildcards and other special strings to specify more complex filters.

As with Wildcard search, the "Use wildcards" checkbox lets you include the "*"and "?" wildcards in filter strings. For example, in **Figure 38**, we want to find only results where the same line contains a string beginning with "c" and ending with "name". The results shown in **Figure 39**, demonstrate a challenge of using wildcards; in the highlighted code, the filter on "c*name" matches the "c" at the end of "MENU_RENAMEITEM_LOC" and "name" in the string "Rename", likely not what we were looking for.

[9] I've reported this bug.



Within Same Statement

c*Name

Finds matches to the entered text within the same STATEMENT (including continuation lines) as the original match.

Within Same Procedure

Finds matches to the entered text within the same PROCEDURE as the original match.

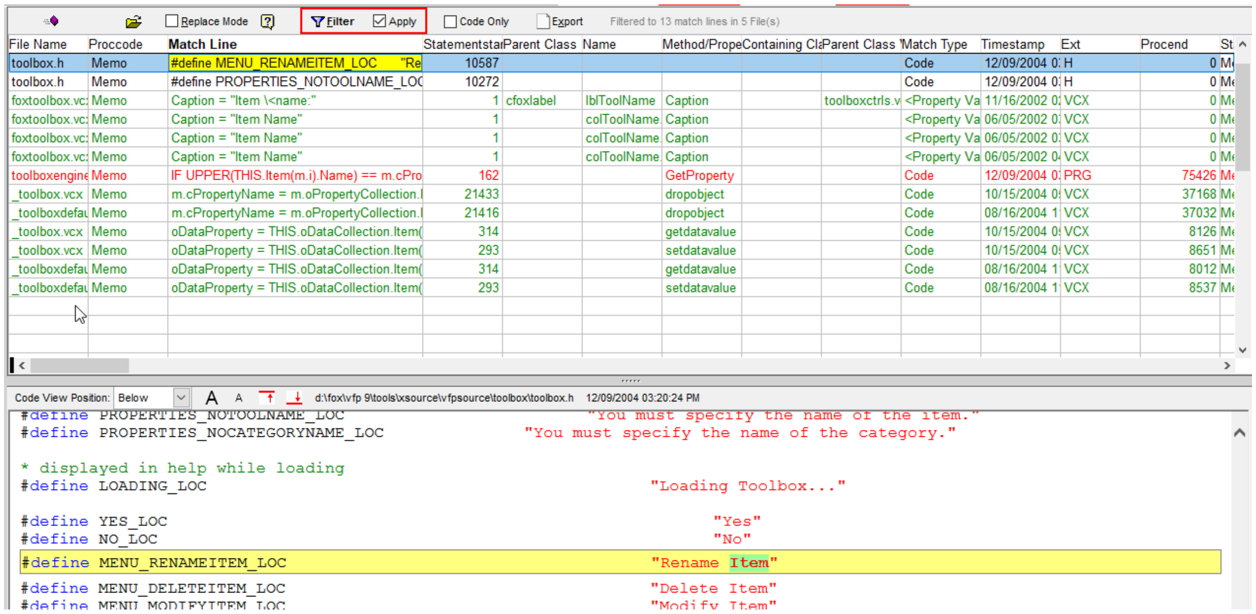
☐ Only show first match in each STATEMENT

☐ Only show first match in each PROCEDURE

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

☑ Match anywhere ☑ Use wildcards (* and ?) ☐ Use leading ! (for NOT)

Figure 38. The Use wildcards checkbox in the Filter Builder lets you include the * and ? wildcards in your filter expression.



☐ Replace Mode ☑ Apply ☐ Code Only Filtered to 13 match lines in 5 File(s)

File Name	Proccode	Match Line	Statementsta	Parent Class	Name	Method/Prope	Containing Cl	Parent Class '	Match Type	Timestamp	Ext	Procend	St
toolbox.h	Memo	#define MENU_RENAMEITEM_LOC "Re	10587						Code	12/09/2004 0:	H	0	Me
toolbox.h	Memo	#define PROPERTIES_NOTOOLNAME_LOC	10272						Code	12/09/2004 0:	H	0	Me
foxtoolbox.vc:	Memo	Caption = "Item \<name:"	1	cfoxlabel	lblToolName	Caption		toolboxctrls.v	<Property Va	11/16/2002 0:	VCX	0	Me
foxtoolbox.vc:	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/05/2002 0:	VCX	0	Me
foxtoolbox.vc:	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/05/2002 0:	VCX	0	Me
foxtoolbox.vc:	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/05/2002 0:	VCX	0	Me
toolboxengine	Memo	IF UPPER(THIS.Item(m.i).Name) == m.cPro	162			GetProperty			Code	12/09/2004 0:	PRG	75426	Me
_toolbox.vcx	Memo	m.cPropertyName = m.oPropertyCollection.I	21433			dropobject			Code	10/15/2004 0:	VCX	37168	Me
_toolboxdefau	Memo	m.cPropertyName = m.oPropertyCollection.I	21416			dropobject			Code	08/16/2004 1	VCX	37032	Me
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	314			getdatavalue			Code	10/15/2004 0:	VCX	8126	Me
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	293			setdatavalue			Code	10/15/2004 0:	VCX	8651	Me
_toolboxdefau	Memo	oDataProperty = THIS.oDataCollection.Item(	314			getdatavalue			Code	08/16/2004 1	VCX	8012	Me
_toolboxdefau	Memo	oDataProperty = THIS.oDataCollection.Item(	293			setdatavalue			Code	08/16/2004 1	VCX	8537	Me

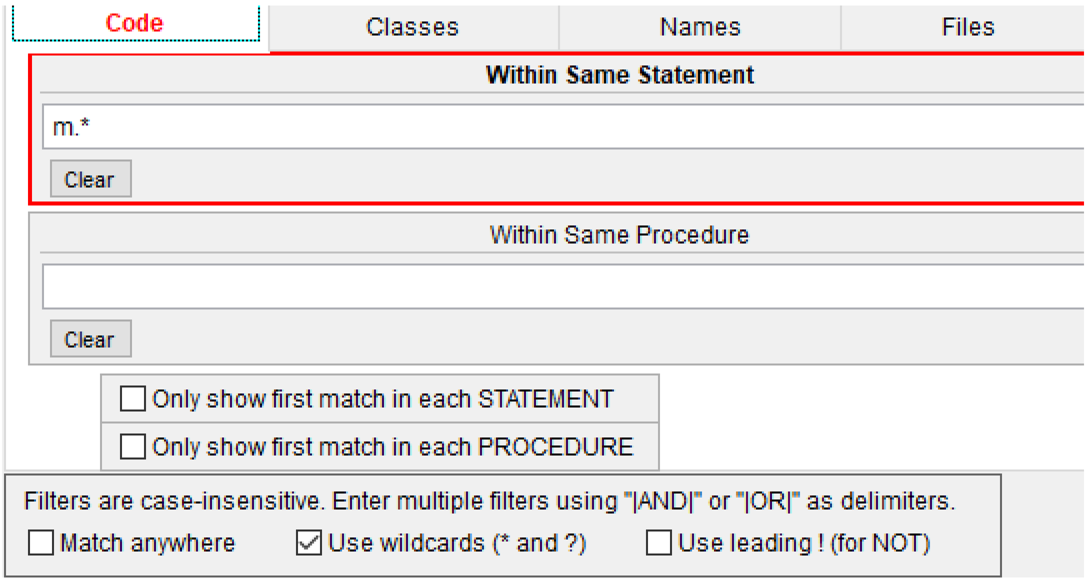
Code View Position: Below   d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.h   12/09/2004 03:20:24 PM

```
#define PROPERTIES_NOTOOLNAME_LOC                           "You must specify the name of the item."
#define PROPERTIES_NOCATEGORYNAME_LOC                       "You must specify the name of the category."

* displayed in help while loading
#define LOADING_LOC                                                      "Loading Toolbox..."

#define YES_LOC                                                                "Yes"
#define NO_LOC                                                                 "No"
#define MENU_RENAMEITEM_LOC                                              "Rename Item"

#define MENU_DELETEITEM_LOC                                              "Delete Item"
#define MENU_MODIFYITEM_LOC                                              "Modify Item"
```

Figure 39. Wildcards in filters can produce some unexpected results.

As mentioned earlier, checking Use wildcards and unchecking "Match everywhere" lets you filter down to lines that begin with a specific string. For example, the combination of settings in **Figure 40** filters the result set down to only those lines that begin with "m."[10] The results are shown in **Figure 41**.

[10] The Toolbox code uses mdot on the left-hand side of assignment states, so this filter finds some results.



Within Same Statement

m.*

Within Same Procedure

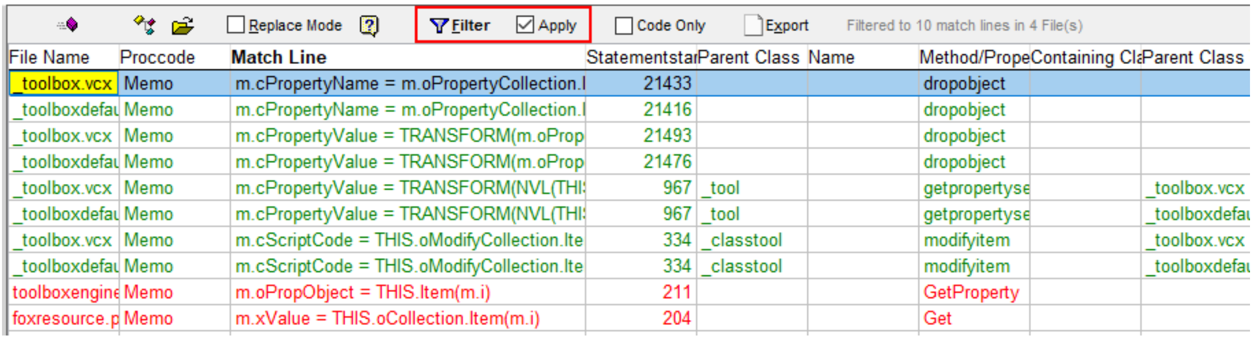
☐ Only show first match in each STATEMENT

☐ Only show first match in each PROCEDURE

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

☐ Match anywhere ☑ Use wildcards (* and ?) ☐ Use leading ! (for NOT)

Figure 40. You can combine "Use wildcards" with no "Match anywhere" to find only lines in the result set that begin with a specified string.

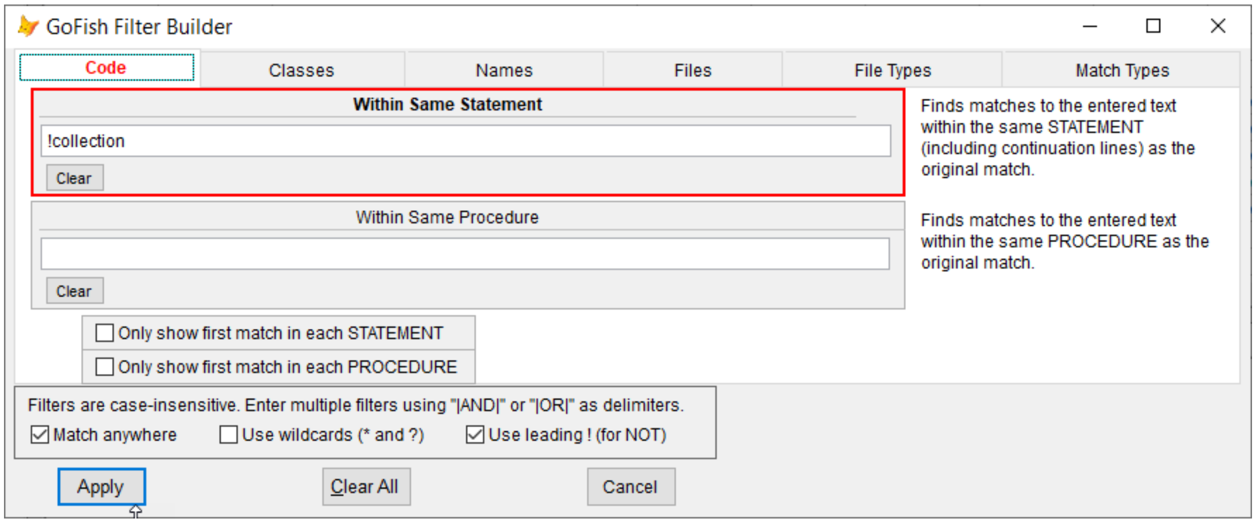


☐ Replace Mode ☑ Apply ☐ Code Only Filtered to 10 match lines in 4 File(s)

File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Prope	Containing Cla	Parent Class
_toolbox.vcx	Memo	m.cPropertyName = m.oPropertyCollection.I	21433			dropobject		
_toolboxdefau	Memo	m.cPropertyName = m.oPropertyCollection.I	21416			dropobject		
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21493			dropobject		
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21476			dropobject		
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse		_toolbox.vcx
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse		_toolboxdefau
_toolbox.vcx	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool		modifyitem		_toolbox.vcx
_toolboxdefau	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool		modifyitem		_toolboxdefau
toolboxengine	Memo	m.oPropObject = THIS.Item(m.i)	211			GetProperty		
foxresource.p	Memo	m.xValue = THIS.oCollection.Item(m.i)	204			Get		

Figure 41. The filter on "m." at the beginning of the line reduces the result set to just 10 lines.

The third checkbox determines whether you can use an exclamation point at the beginning of your filter string to indicate that you want to find only matches that do NOT include the filter string. For example, in **Figure 42**, the filter string is "!collection" and the checkbox is set, so we find those matches that do not contain the string "collection" on the same line.


GoFish Filter Builder

Code | Classes | Names | Files | File Types | Match Types

Within Same Statement

!collection

Clear

Finds matches to the entered text within the same STATEMENT (including continuation lines) as the original match.

Within Same Procedure

Clear

Finds matches to the entered text within the same PROCEDURE as the original match.

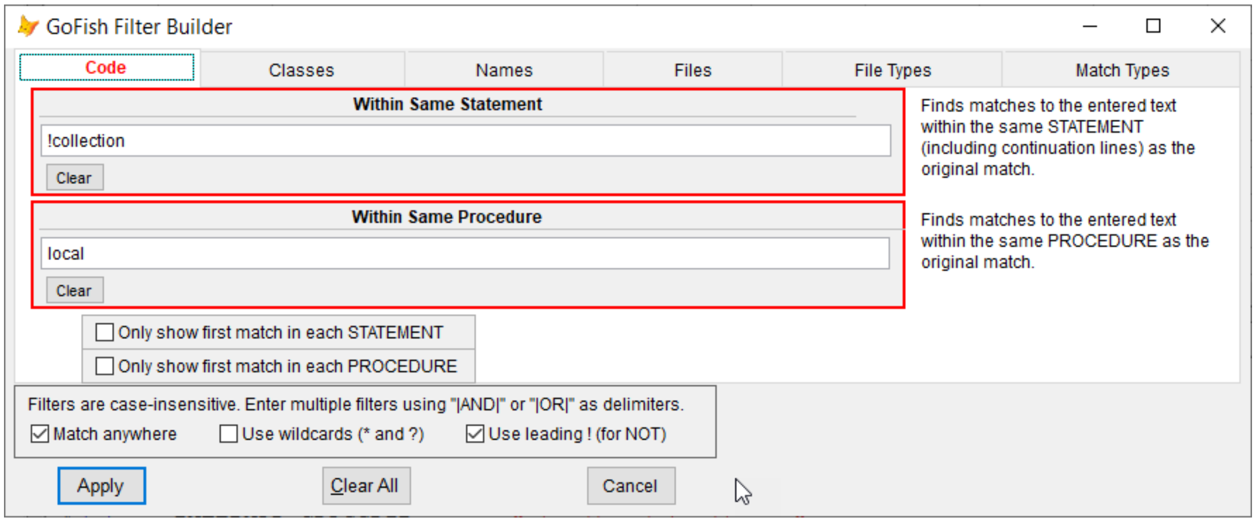
☐ Only show first match in each STATEMENT
☐ Only show first match in each PROCEDURE

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.
☑ Match anywhere ☐ Use wildcards (* and ?) ☑ Use leading ! (for NOT)

Apply | Clear All | Cancel

Figure 42. Checking the "Use leading ! (for NOT)" checkbox alone doesn't invert your filter. You must also put an exclamation point at the front of the filter expression.

Be aware that you cannot use the exclamation point multiple times in the line to negate some portions of the filter and leave others positive. As the checkbox caption indicates, only a leading exclamation point is interpreted as "NOT." This is actually a good thing as it means you can, for example, look for lines that do not contain a particular string, but only within routines that contain a different filter string, as in **Figure 43**.


GoFish Filter Builder

Code | Classes | Names | Files | File Types | Match Types

Within Same Statement

!collection

Clear

Finds matches to the entered text within the same STATEMENT (including continuation lines) as the original match.

Within Same Procedure

local

Clear

Finds matches to the entered text within the same PROCEDURE as the original match.

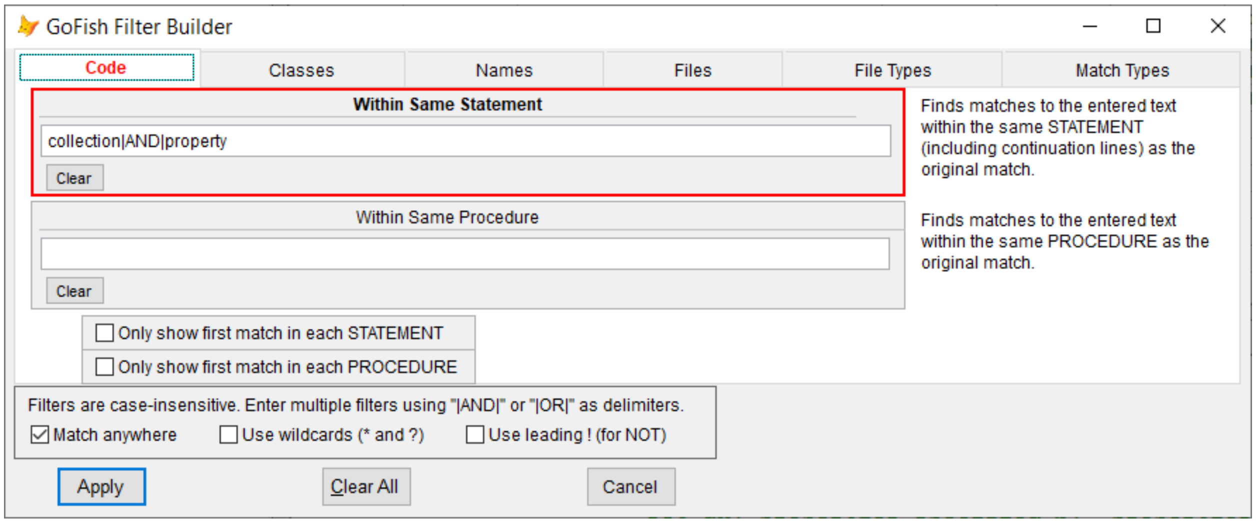
☐ Only show first match in each STATEMENT
☐ Only show first match in each PROCEDURE

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.
☑ Match anywhere ☐ Use wildcards (* and ?) ☑ Use leading ! (for NOT)

Apply | Clear All | Cancel

Figure 43. You can combine the different filters and have some use NOT and others stay positive.

As the comment explains, you can combine filter expressions of a particular type using "|AND|" and "|OR|". For example, using the filter shown in **Figure 44**, you get only those results where the line contains both "collection" and "property". As **Figure 45** shows, this cuts down to 10 matches. "|AND|" says to find results that include both expressions in the same statement or same procedure. "|OR|" finds those results that contain either of the expressions in the same statement or same procedure.


GoFish Filter Builder

Within Same Statement

collection|AND|property

Finds matches to the entered text within the same STATEMENT (including continuation lines) as the original match.

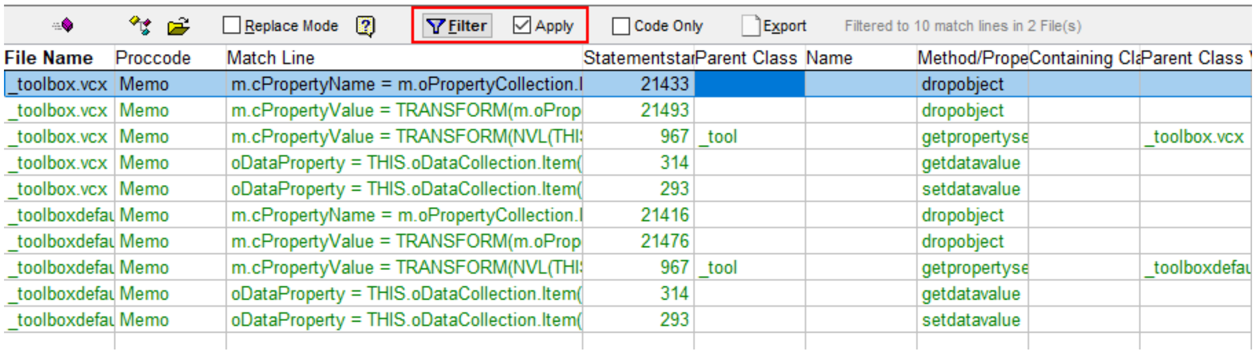
Within Same Procedure

Finds matches to the entered text within the same PROCEDURE as the original match.

☐ Only show first match in each STATEMENT
☐ Only show first match in each PROCEDURE

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.
☑ Match anywhere ☐ Use wildcards (* and ?) ☐ Use leading ! (for NOT)

Figure 44. You can combine filter strings with "|AND|" and "|OR|" to narrow or expand your result set.


☐ Replace Mode | Filter | ☑ Apply | ☐ Code Only | Export | Filtered to 10 match lines in 2 File(s)

File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Prope	Containing Cla	Parent Class
_toolbox.vcx	Memo	m.cPropertyName = m.oPropertyCollection.I	21433			dropobject		
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21493			dropobject		
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse		_toolbox.vcx
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	314			getdatavalue		
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	293			setdatavalue		
_toolboxdefau	Memo	m.cPropertyName = m.oPropertyCollection.I	21416			dropobject		
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21476			dropobject		
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse		_toolboxdefau
_toolboxdefau	Memo	oDataProperty = THIS.oDataCollection.Item(	314			getdatavalue		
_toolboxdefau	Memo	oDataProperty = THIS.oDataCollection.Item(	293			setdatavalue		

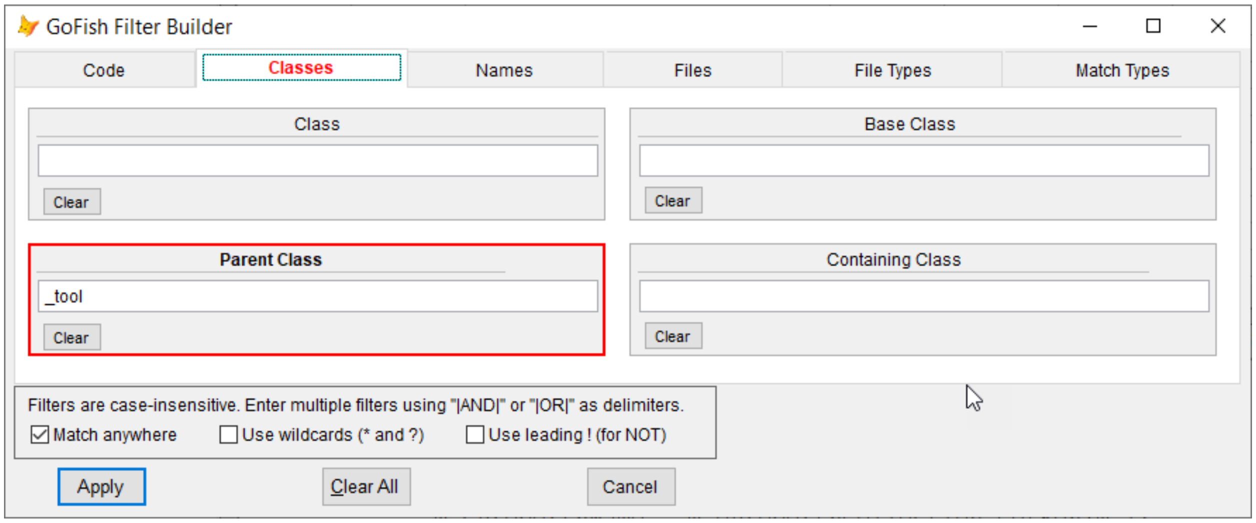
Figure 45. When you combine filter expressions with "|AND|", you make the result set smaller.

You can use more than one of the "|AND|" and "|OR|" operators. In that case, the list of expressions is evaluated from left to right; there's no way to include parentheses to change the order of evaluation. That means you can't set up filter expressions in the form "(a and b) or (c and d)".

Filtering on where the match occurs

The middle three tabs of the GoFish Filter Builder let you limit results based on where the match occurs, such as the class name, the property or method name, or the file name. Each of the three tabs contains multiple ways to specify which matches should be retained. These tabs also are affected by the three checkboxes described in "Fine-tuning filters," earlier in this document.

The Classes tab gives you four ways to identify the classes of interest: class name, base class name, parent class name, and containing class name. For example, the filter specified in **Figure 46** keeps only matches that occur in classes subclassed from _tool; the result is shown in **Figure 47**.

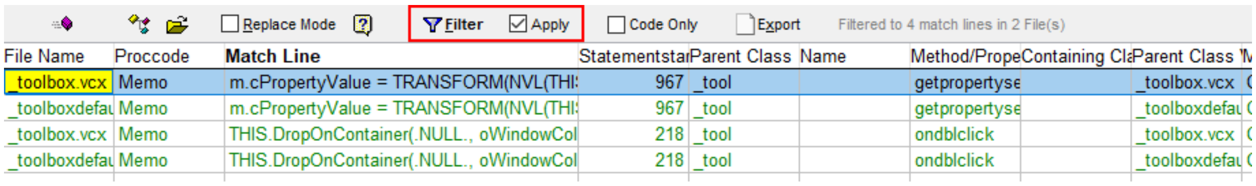


Parent Class: _tool

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

☑ Match anywhere ☐ Use wildcards (* and ?) ☐ Use leading ! (for NOT)

Figure 46. On the Classes tab, you can limit results based on information about the class where the match occurred.



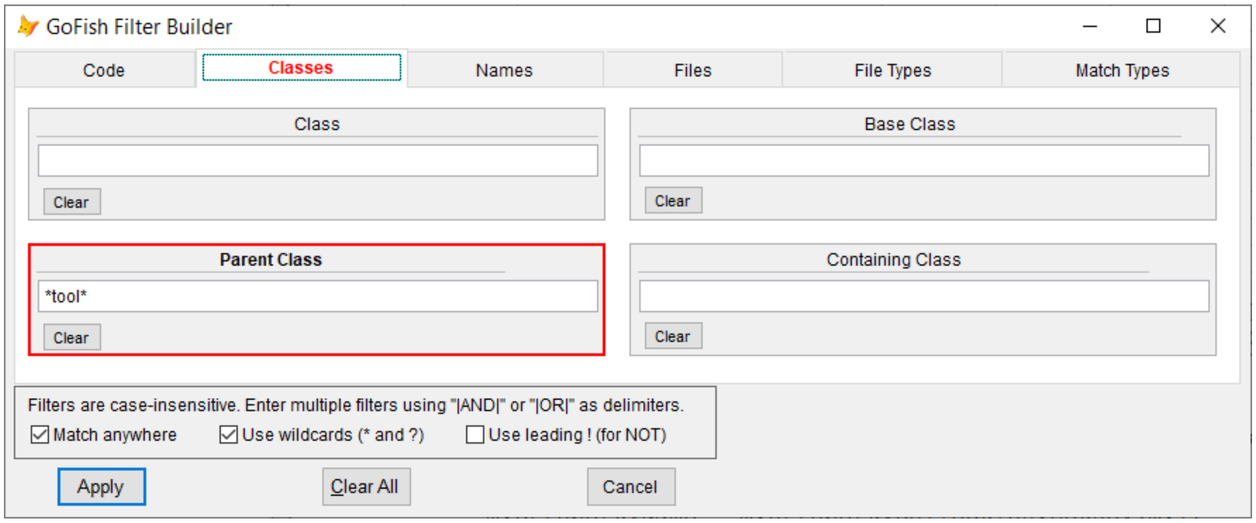
☐ Replace Mode ☑ Apply ☐ Code Only Filtered to 4 match lines in 2 File(s)

File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Prope	Containing Cla	Parent Class 'M
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse		_toolbox.vcx (
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse		_toolboxdefau (
_toolbox.vcx	Memo	THIS.DropOnContainer(.NULL., oWindowCol	218	_tool		ondblclick		_toolbox.vcx (
_toolboxdefau	Memo	THIS.DropOnContainer(.NULL., oWindowCol	218	_tool		ondblclick		_toolboxdefau (

Figure 47. Only four matches are left when applying the filter in **Figure 46.**

Be aware that the Parent Class and Containing Class filters look up only one level. You can't filter to find results in any class derived from a particular class, no matter how many subclasses occur in between, or any class contained in a particular container class, no matter how other containers are involved in between.

As noted earlier, you can use wildcards here to look for the search string anywhere in the name. In **Figure 48**, a filter string of "*tool*" for Parent Class and having "Use wildcards" checked finds results in any parent class name containing "tool". **Figure 49** shows the results.


GoFish Filter Builder

Code | Classes | Names | Files | File Types | Match Types

Class:

Base Class:

Parent Class: *tool*

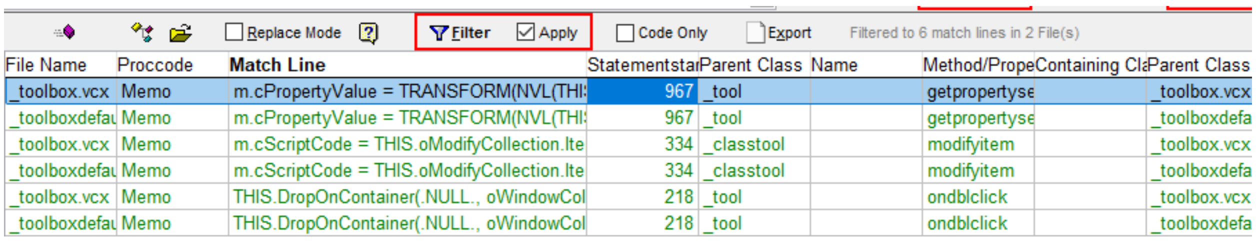
Containing Class:

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

[✓] Match anywhere [✓] Use wildcards (* and ?) [ ] Use leading ! (for NOT)

Apply | Clear All | Cancel

Figure 48. With wildcards, you can search for a wide range of classes.

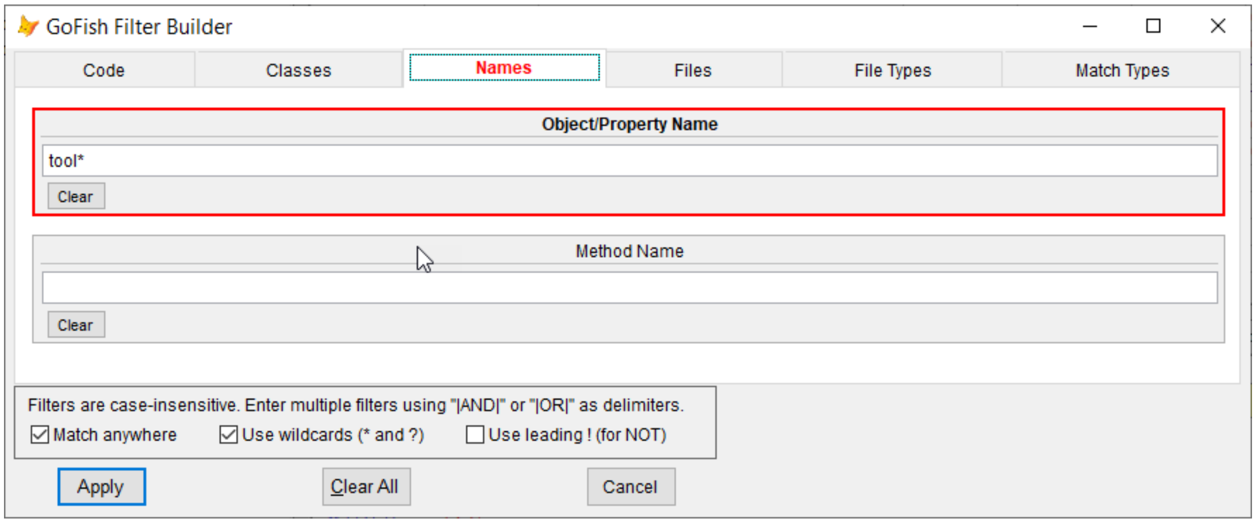

[ ] Replace Mode | Filter | [✓] Apply | [ ] Code Only | Export | Filtered to 6 match lines in 2 File(s)

File Name	Proccode	Match Line	Statementsta	Parent Class	Name	Method/Prope	Containing Cla	Parent Class
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(NVL(THI:	967	_tool		getpropertyse		_toolbox.vcx
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(NVL(THI:	967	_tool		getpropertyse		_toolboxdefa
_toolbox.vcx	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool		modifyitem		_toolbox.vcx
_toolboxdefau	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool		modifyitem		_toolboxdefa
_toolbox.vcx	Memo	THIS.DropOnContainer(.NULL., oWindowCol	218	_tool		ondblclick		_toolbox.vcx
_toolboxdefau	Memo	THIS.DropOnContainer(.NULL., oWindowCol	218	_tool		ondblclick		_toolboxdefa

Figure 49. The results selected by the filter shown in **Figure 48** include those derived from the "_tool" and "_classtool" classes.

You can use the "|AND|" and "|OR|" operators on these tabs, too, so you can look for results in several different classes, for example. Remember that in such cases, you want "|OR|", not "|AND|", since no item will belong to more than one class.

The Names tab lets you filter on the object name, property name, or method name. In **Figure 50**, we filter down to only results that are in objects or properties that contain the string "tool"; **Figure 51** shows the result, with the name column (expanded and) highlighted so you can see that each remaining result includes "tool" in the object name. As the example shows, you have the same set of options for modifying the filter expression here as on the Code and Classes tabs. That's true for the Files tab, as well.



Object/Property Name

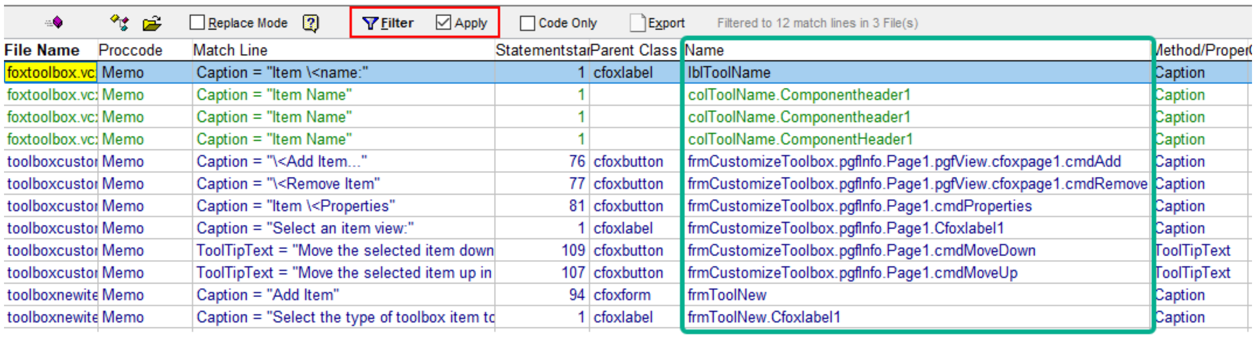
tool*

Method Name

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

☑ Match anywhere ☑ Use wildcards (* and ?) ☐ Use leading ! (for NOT)

Figure 50. The Names tab of the Filter Builder lets you narrow down to results only in specific objects, properties or methods.

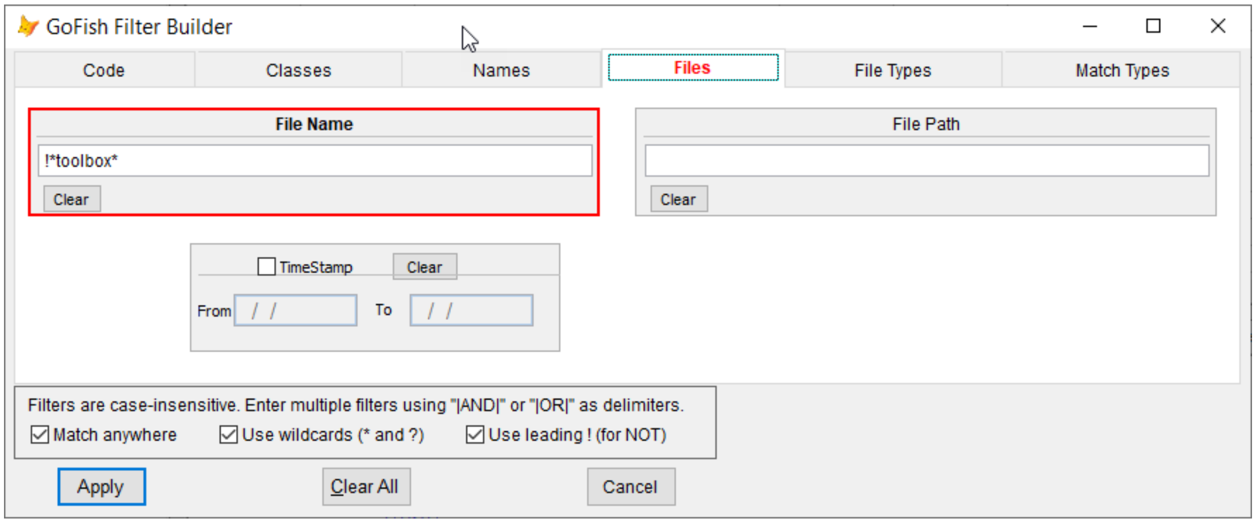


Filtered to 12 match lines in 3 File(s)

File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Propert
foxtoolbox.vc	Memo	Caption = "Item \<name:"	1	cfoxlabel	lblToolName	Caption
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.Componentheader1	Caption
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.Componentheader1	Caption
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.ComponentHeader1	Caption
toolboxcusto	Memo	Caption = "\<Add Item..."	76	cfoxbutton	frmCustomizeToolbox.pgfInfo.Page1.pgfView.cfoxpage1.cmdAdd	Caption
toolboxcusto	Memo	Caption = "\<Remove Item"	77	cfoxbutton	frmCustomizeToolbox.pgfInfo.Page1.pgfView.cfoxpage1.cmdRemove	Caption
toolboxcusto	Memo	Caption = "Item \<Properties"	81	cfoxbutton	frmCustomizeToolbox.pgfInfo.Page1.cmdProperties	Caption
toolboxcusto	Memo	Caption = "Select an item view:"	1	cfoxlabel	frmCustomizeToolbox.pgfInfo.Page1.Cfoxlabel1	Caption
toolboxcusto	Memo	ToolTipText = "Move the selected item down	109	cfoxbutton	frmCustomizeToolbox.pgfInfo.Page1.cmdMoveDown	ToolTipText
toolboxcusto	Memo	ToolTipText = "Move the selected item up in	107	cfoxbutton	frmCustomizeToolbox.pgfInfo.Page1.cmdMoveUp	ToolTipText
toolboxnewite	Memo	Caption = "Add Item"	94	cfoxform	frmToolNew	Caption
toolboxnewite	Memo	Caption = "Select the type of toolbox item to	1	cfoxlabel	frmToolNew.Cfoxlabel1	Caption

Figure 51. The highlighted column shows the name of the object or property containing the match; here, they're filtered to only those with the string "tool" in the name.

The Files tab lets you limit results to only those in particular files or on a particular file path or with a timestamp in a specified range. The filter in **Figure 52** keeps results only in files whose names do not include the string "toolbox". The results, shown in **Figure 53**, contain only 4 of the original matches.

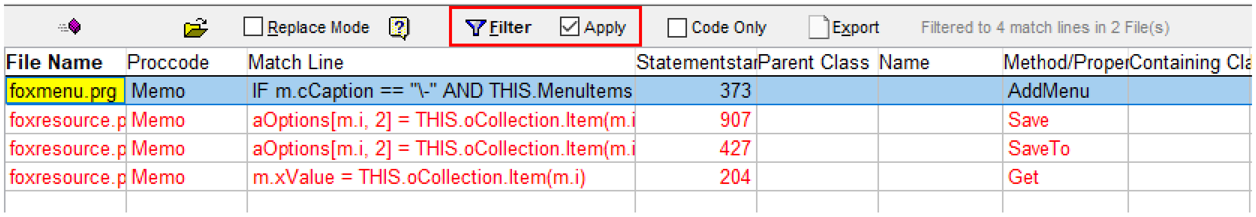


File Name: !*toolbox*

Filters are case-insensitive. Enter multiple filters using "|AND|" or "|OR|" as delimiters.

☑ Match anywhere ☑ Use wildcards (* and ?) ☑ Use leading ! (for NOT)

Figure 52. This file limits results to those in files whose names do NOT include “toolbox”.



Filtered to 4 match lines in 2 File(s)

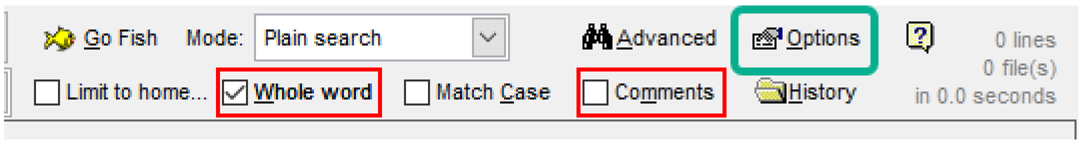
File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Proper	Containing Cla
foxmenu.prg	Memo	IF m.cCaption == "\-" AND THIS.MenuItems	373			AddMenu	
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	907			Save	
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427			SaveTo	
foxresource.p	Memo	m.xValue = THIS.oCollection.Item(m.i)	204			Get	

Figure 53. When file names containing “toolbox” are filtered out, only a few matches are left.

Overall, GoFish’s filter mechanism gives us many, many ways to reduce result sets to something manageable.

Replacing strings

In the computing world, we generally talk about “search and replace.” So far, this paper has looked only at search, but GoFish has a strong replace game, too. In order to use replace, however, you have to turn it on and decide how much risk tolerance you have. (The risk is that performing replace will damage your code, but in years of using GoFish’s replace, I’ve never had that happen.) To enable replacement, you use the Replace tab of the GoFish Options dialog, which is opened by clicking the Options button on the main GoFish form (highlighted in **Figure 54**) .



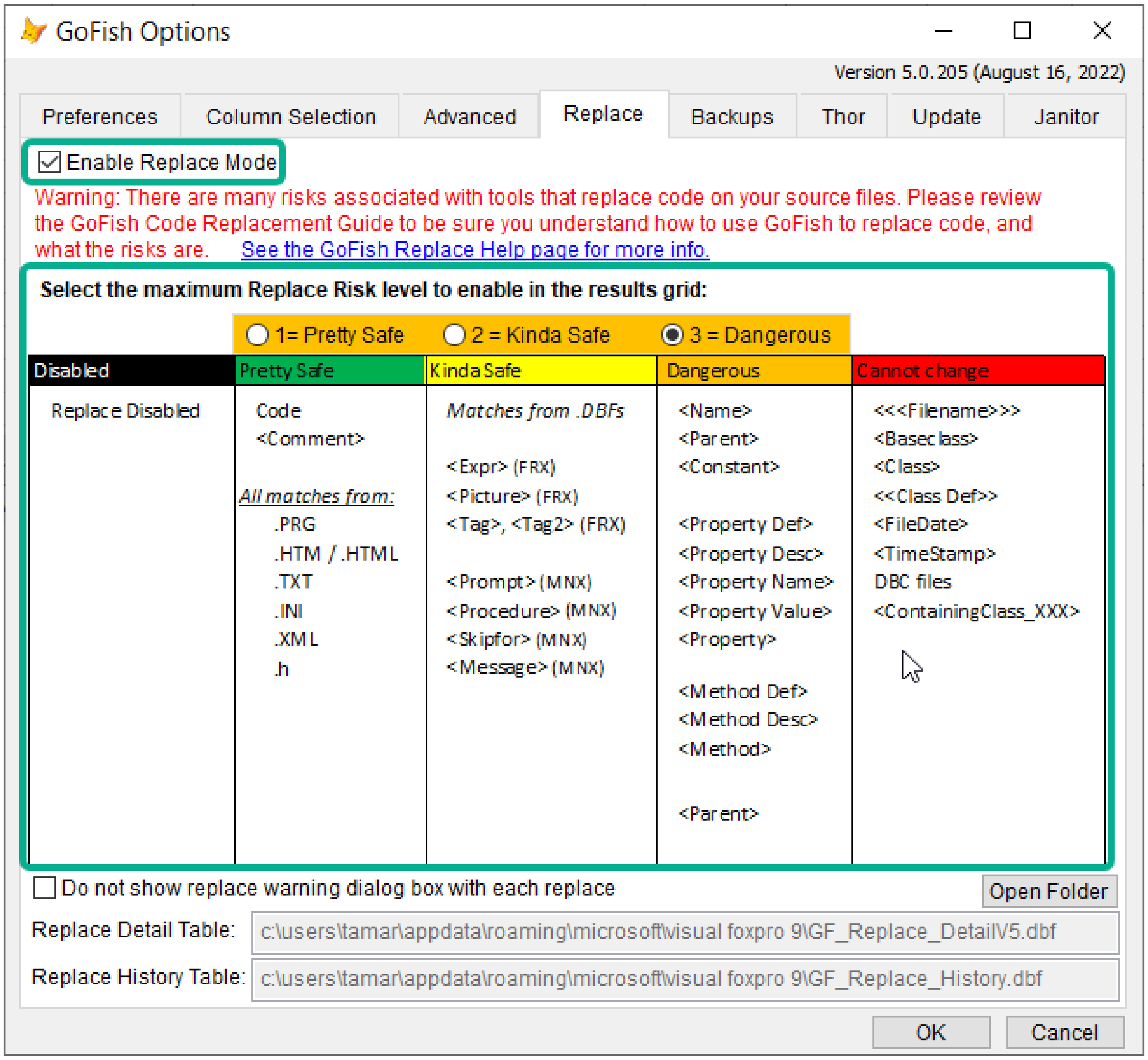
Mode: Plain search

☐ Limit to home... ☑ Whole word ☐ Match Case ☐ Comments

0 lines
0 file(s)
in 0.0 seconds

Figure 54. Click Options in the main GoFish form to open the GoFish Options dialog.

The first step (highlighted near the top in Error! Reference source not found.) is to check Enable Replace Mode. Once you’ve done that, you can which risk level you want. A table (also highlighted in the figure) shows exactly what kind of items can be replaced at each level. I’ve selected the highest risk level because my experience tells me GoFish does replacements carefully. In addition, by default, files are backed up automatically before replacing anything.



[✓] Enable Replace Mode

Warning: There are many risks associated with tools that replace code on your source files. Please review the GoFish Code Replacement Guide to be sure you understand how to use GoFish to replace code, and what the risks are. See the GoFish Replace Help page for more info.

Select the maximum Replace Risk level to enable in the results grid:

1= Pretty Safe 2 = Kinda Safe [● 3 = Dangerous]

Disabled	Pretty Safe	Kinda Safe	Dangerous	Cannot change
Replace Disabled	Code <Comment> *All matches from:* .PRG .HTM / .HTML .TXT .INI .XML .h	*Matches from .DBFs* <Expr> (FRX) <Picture> (FRX) <Tag>, <Tag2> (FRX) <Prompt> (MNX) <Procedure> (MNX) <Skipfor> (MNX) <Message> (MNX)	<Name> <Parent> <Constant> <Property Def> <Property Desc> <Property Name> <Property Value> <Property> <Method Def> <Method Desc> <Method> <Parent>	<<<Filename>>> <Baseclass> <Class> <<Class Def>> <FileDate> <TimeStamp> DBC files <ContainingClass_XXX>

☐ Do not show replace warning dialog box with each replace

Replace Detail Table: c:\users\tamar\appdata\roaming\microsoft\visual foxpro 9\GF_Replace_DetailV5.dbf

Replace History Table: c:\users\tamar\appdata\roaming\microsoft\visual foxpro 9\GF_Replace_History.dbf

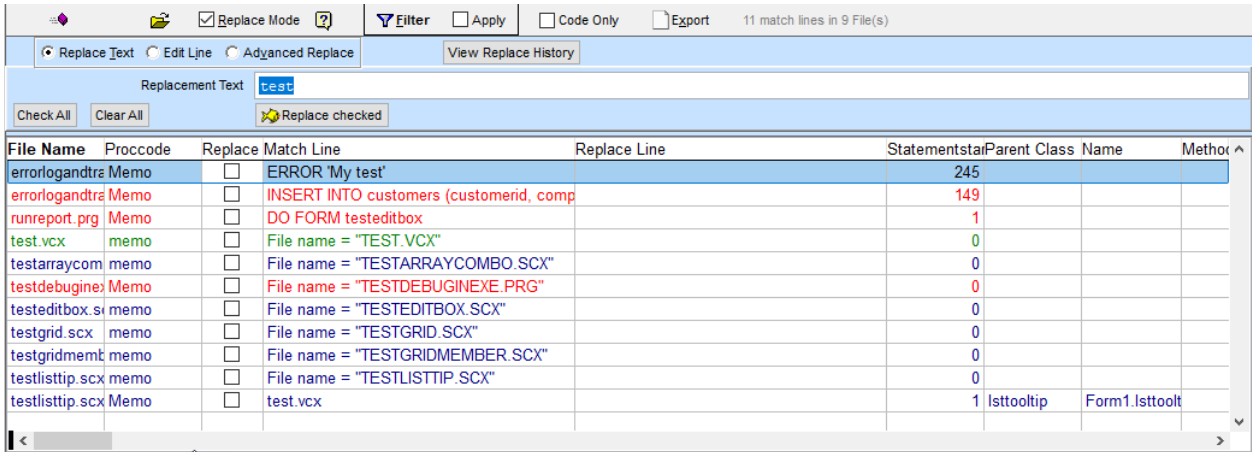
Figure 55. You can decide how much risk you're willing to take when using GoFish's replacement feature.

Once you've set a risk level, click OK to return to the main GoFish form and replacement is available.

For the replace examples, I'm moving from the Toolbox code to a folder where I throw quick and dirty code to test VFP features. That way, I can perform replacements without damaging code I may need in the future. I'll search for the string "test" and then do various replacements.

To begin a replacement, check the Replace Mode checkbox after a search (and any filtering to get to the items of interest). The header area of the results grid expands to show a small

Replace pane and the grid itself adds a Replace column filled with checkboxes to allow you to replace only in the rows you want. **Figure 56** shows the new version of the Results pane.



File Name	Proccode	Replace	Match Line	Replace Line	Statementstar	Parent Class	Name	Methoc
errorlogandtra	Memo	☐	ERROR 'My test'		245			
errorlogandtra	Memo	☐	INSERT INTO customers (customerid, comp		149			
runreport.prg	Memo	☐	DO FORM testeditbox		1			
test.vcx	memo	☐	File name = "TEST.VCX"		0			
testarraycom	memo	☐	File name = "TESTARRAYCOMBO.SCX"		0			
testdebugine:	Memo	☐	File name = "TESTDEBUGINEXE.PRG"		0			
testeditbox.s	memo	☐	File name = "TESTEDITBOX.SCX"		0			
testgrid.scx	memo	☐	File name = "TESTGRID.SCX"		0			
testgridmemb	memo	☐	File name = "TESTGRIDMEMBER.SCX"		0			
testlisttip.scx	memo	☐	File name = "TESTLISTTIP.SCX"		0			
testlisttip.scx	Memo	☐	test.vcx		1	lsttooltip	Form1.lsttoolt	

Figure 56. When you check Replace Mode, the results grid turns into a Replacement pane, including a new column in the grid to control whether a given result should be replaced.

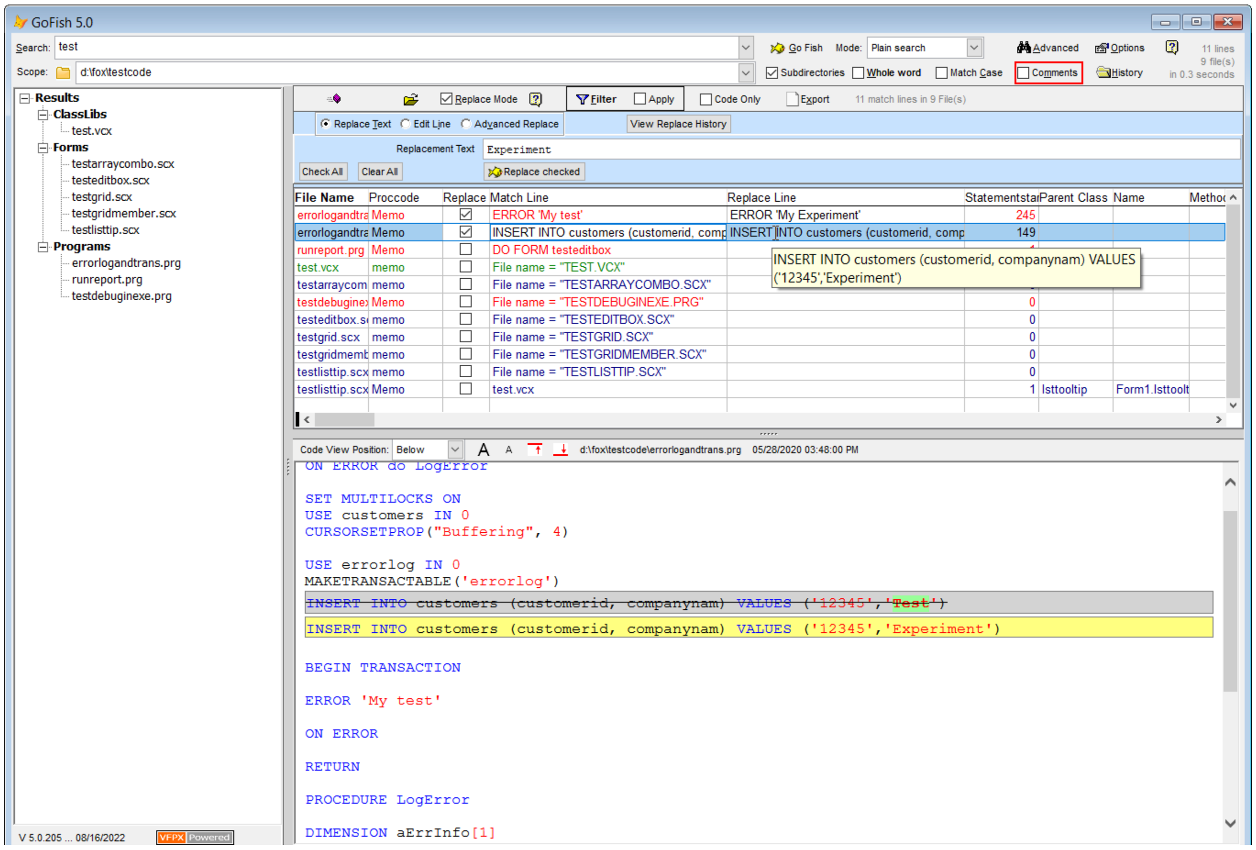
The first line in the new Replace pane lets you choose one of three replace types: Replace Text, Edit Line, or Advanced Replace. Each is described in detail in a section later in this document. The appearance of the Replace pane changes depending which type you've chosen; **Figure 56** shows it for Replace Text.

Replacing text

The simplest version of replacement is Replace Text; it's like the replacements in most applications. The target text (that is, the text to be replaced) is replaced in full by the replacement text, which you specify in the Replacement Text textbox (the big textbox on the second line of the Replace pane shown in **Figure 56**).

You control which instances of the result are replaced. You can use the Check All and Clear All buttons to select or deselect them all at once or you can check and uncheck lines individually. (Obviously, it's best if you've already filtered your results to get as close as you can to the exact set of lines in which you want to replace the search string.)

Once you've selected a line, the Replace Line column shows what the result line will look like after the replacement and the Code View windows shows the code with the existing line stricken through and the new line beneath it, as in **Figure 57**. Here, the replacement string for "test" is set to "Experiment" and two lines are checked. The tooltip (a standard feature of VFP 9 grids) shows the replacement line for the second checked line.



File Name	Proccode	Replace	Match Line	Replace Line	Statementstar	Parent Class	Name	Methoc
errorlogandtra	Memo	✓	ERROR 'My test'	ERROR 'My Experiment'	245			
errorlogandtra	Memo	✓	INSERT INTO customers (customerid, comp	INSERT INTO customers (customerid, comp	149			
runreport.prg	Memo		DO FORM testeditbox		[illegible]			
test.vcx	memo		File name = "TEST.VCX"		[illegible]			
testarraycom	memo		File name = "TESTARRAYCOMBO.SCX"		[illegible]			
testdebuginex	Memo		File name = "TESTDEBUGINEXE.PRG"		0			
testeditbox.s	memo		File name = "TESTEDITBOX.SCX"		0			
testgrid.scx	memo		File name = "TESTGRID.SCX"		0			
testgridmemb	memo		File name = "TESTGRIDMEMBER.SCX"		0			
testlisttip.scx	memo		File name = "TESTLISTTIP.SCX"		0			
testlisttip.scx	Memo		test.vcx		1	lsttooltip	Form1.lsttoolt	

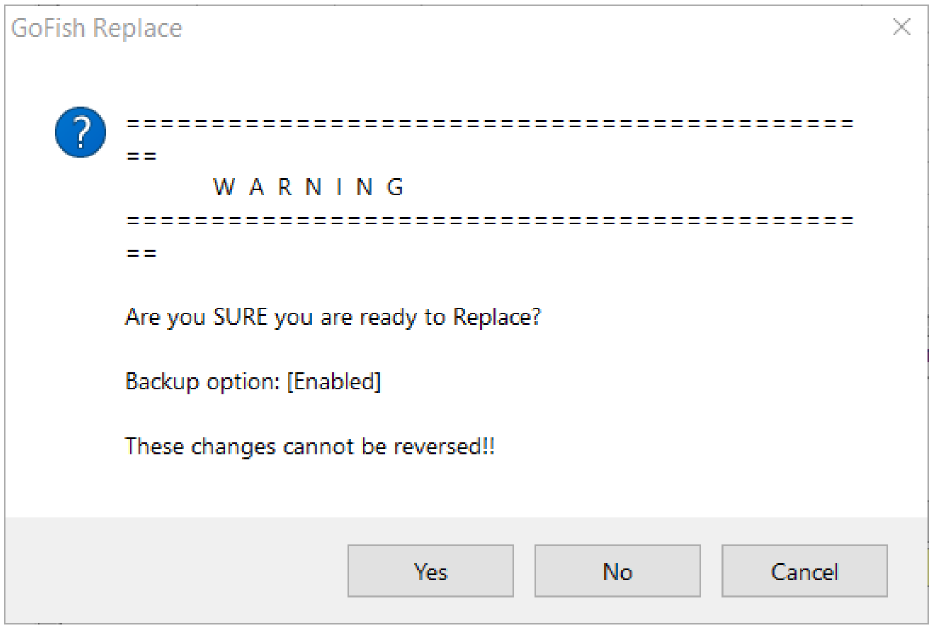
INSERT INTO customers (customerid, companynam) VALUES ('12345','Experiment')

```
ON ERROR do LogError

SET MULTILOCKS ON
USE customers IN 0
CURSORSETPROP("Buffering", 4)

USE errorlog IN 0
MAKETRANSACTABLE('errorlog')
~~INSERT INTO customers (customerid, companynam) VALUES ('12345','Test')~~
INSERT INTO customers (customerid, companynam) VALUES ('12345','Experiment')

BEGIN TRANSACTION

ERROR 'My test'

ON ERROR

RETURN

PROCEDURE LogError

DIMENSION aErrInfo[1]
```

Figure 57.When you check a line in Replace Text mode, the Replace Line column shows what it will look like if you do the replacement and the Code View pane shows the change in context. (Here, the VFP 9 grid feature that gives you tooltips for long columns is used to see the result for the second row.)

Note that the replacement is semi-case sensitive. That is, as always, the search is case-insensitive, but the replacement string is inserted with case as you typed it in the Replacement Text textbox.

When you're sure you've chosen the right lines and specified the right replacement string, click Replace checked and the actual replacement happens. When you click the button, you get a scary warning, shown in **Figure 58**. As noted earlier, by default, all files to be replaced are backed up before the replacement begins.


GoFish Replace

==
==
W A R N I N G
==
==

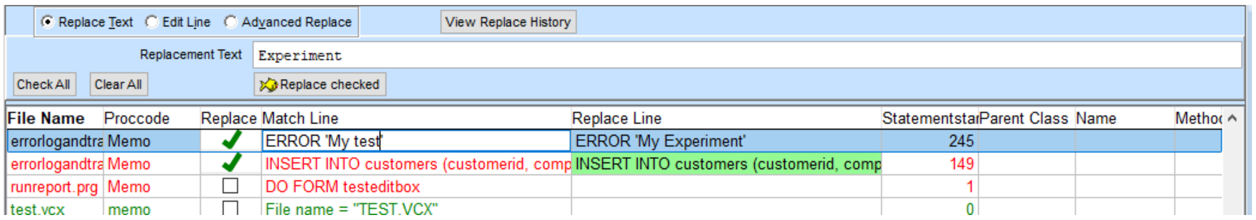
Are you SURE you are ready to Replace?

Backup option: [Enabled]

These changes cannot be reversed!!

Figure 58. This warning appears when you click the Replace checked button.

After the replacement is completed, the checkbox in the Replace column turns into a big green checkmark and the Replace Line column has a green background (when the row is not selected), as in **Figure 59**.



Replacement Text: Experiment

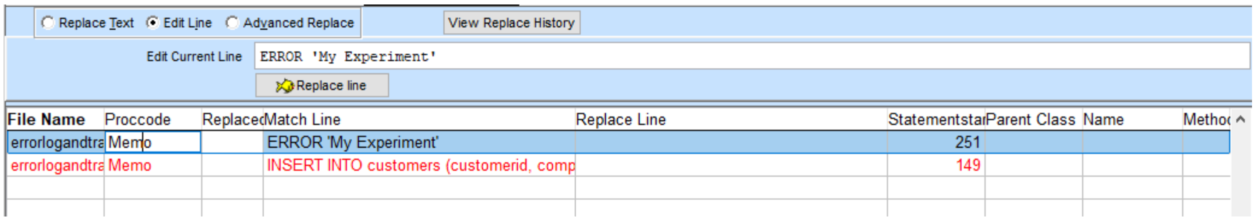
File Name	Proccode	Replace	Match Line	Replace Line	Statementstar	Parent Class	Name	Methoc
errorlogandtra	Memo	✓	ERROR 'My test'	ERROR 'My Experiment'	245			
errorlogandtra	Memo	✓	INSERT INTO customers (customerid, comp	INSERT INTO customers (customerid, comp	149			
runreport.prg	Memo	☐	DO FORM testeditbox		1			
test.vcx	memo	☐	File name = "TEST.VCX"		0			

Figure 59. After the replacement is complete, you can identify the replaced rows in the result grid.

Note that GoFish does not immediately do the original search again, which might exclude the replaced lines; you have to do that yourself, if that's what you want. Also, one you've replaced in a line in a particular search, you cannot select that line for another replacement unless you execute another search that includes that line.

Replacing by direct edit

The second way to replace data is called Edit Line and it's exactly what it sounds like. You can make changes directly to the result line. In this mode (shown in **Figure 60**, where we've searched my test code folder for the string "experiment"), the content of the currently selected result line is shown in the textbox (now labelled "Edit Current Line"). You can edit that line directly and, when you have what you want, click the Replace Line button. As before, you'll get the scary warning shown in **Figure 58**. If you choose Yes, your change is saved to the original file. As with Replace Text, the result line gets a big green checkmark and the Replace Line field gets a green background; in addition, the Code View pane shows the change.

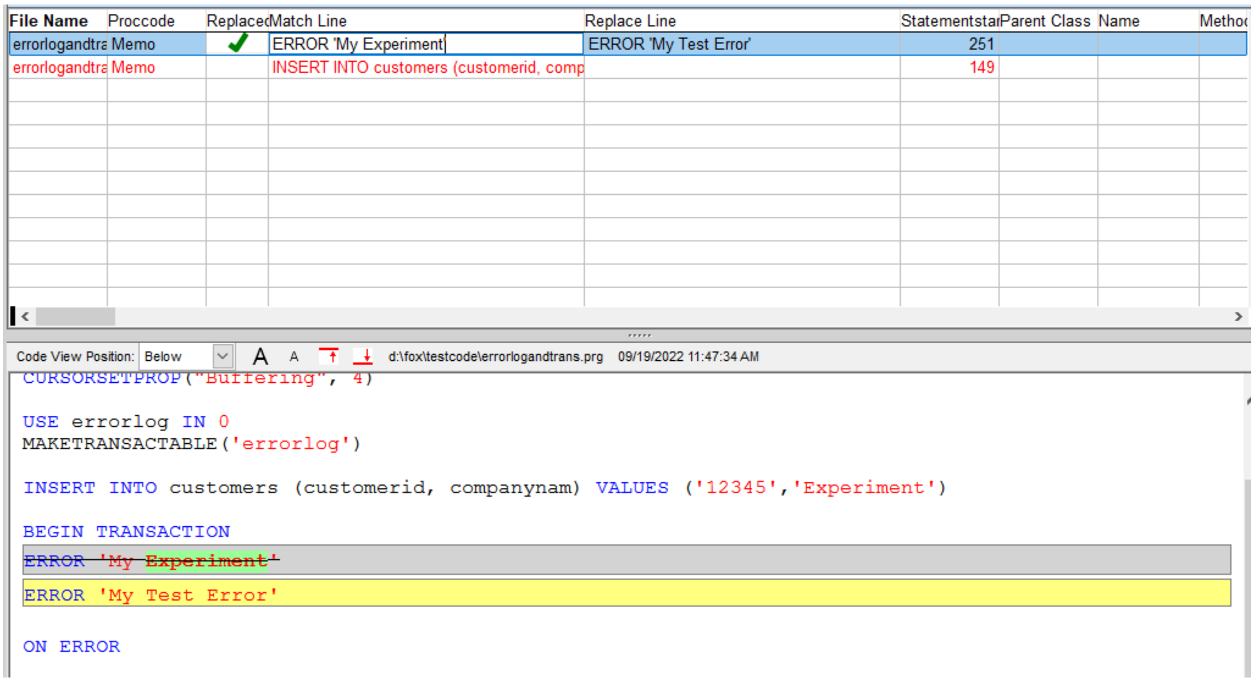

Replace Text [✓ Edit Line] Advanced Replace

Edit Current Line: `ERROR 'My Experiment'`

File Name	Proccode	Replace	Match Line	Replace Line	Statementsta	Parent Class	Name	Methoc
errorlogandtra	Memo		ERROR 'My Experiment'		251			
errorlogandtra	Memo		INSERT INTO customers (customerid, comp		149			

Figure 60. Edit Line lets you directly edit any result line.

Figure 61 shows the result of editing the string "Experiment" and changing it to "Test Error". It's important to note that, in fact, the whole line was eligible for editing, so I could have changed the ERROR command to be a call to a custom function, for example.


File Name	Proccode	Replace	Match Line	Replace Line	Statementsta	Parent Class	Name	Methoc
errorlogandtra	Memo	✓	ERROR 'My Experiment'	ERROR 'My Test Error'	251			
errorlogandtra	Memo		INSERT INTO customers (customerid, comp		149			

Code View Position: Below   d:\fox\testcode\errorlogandtrans.prg 09/19/2022 11:47:34 AM

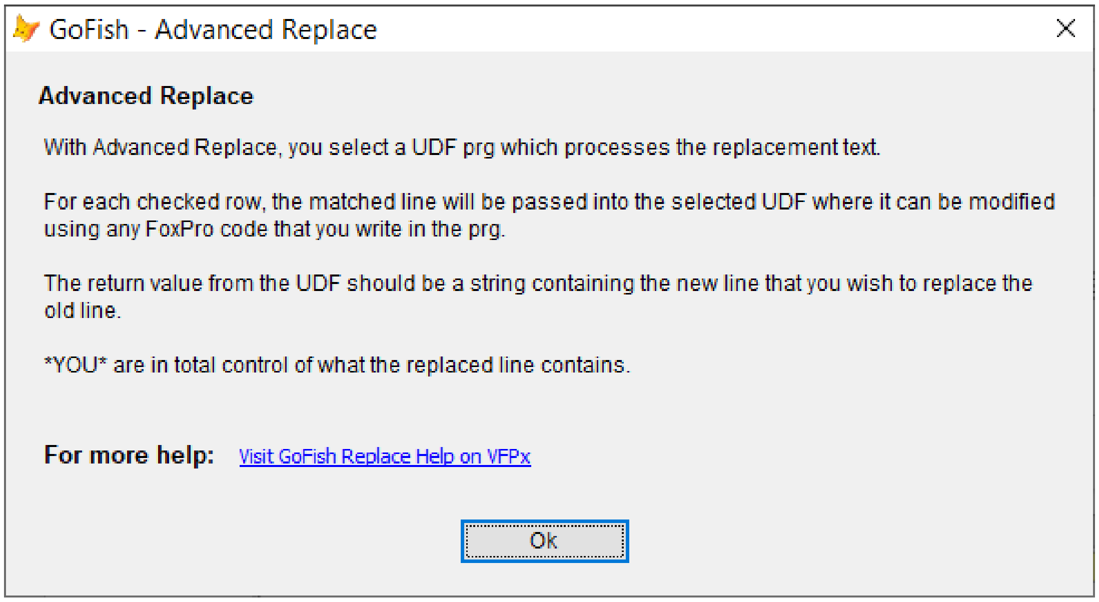
```
CURSORSETPROP("Buffering", 4)

USE errorlog IN 0
MAKETRANSACTABLE('errorlog')

INSERT INTO customers (customerid, companynam) VALUES ('12345','Experiment')

BEGIN TRANSACTION
~~ERROR 'My Experiment'~~
ERROR 'My Test Error'

ON ERROR
```

Figure 61. Here, "Experiment" has been changed to "Test Error"

In Edit Line mode, you can only work on one line at a time. If you modify a line and navigate away before clicking the Replace Line button, your changes are lost.

Replacing via code

The really exciting option for replace is Advanced Replace. In this mode, you select a function to call and it's applied to each checked line. Because you can do tremendous damage with this option, when you choose it, you get a message (shown in **Figure 62**) that's part warning and part instruction manual. Once you click OK, the replace pane changes to allow you to specify the function you want to use; see **Figure 63**.



GoFish - Advanced Replace

Advanced Replace

With Advanced Replace, you select a UDF prg which processes the replacement text.

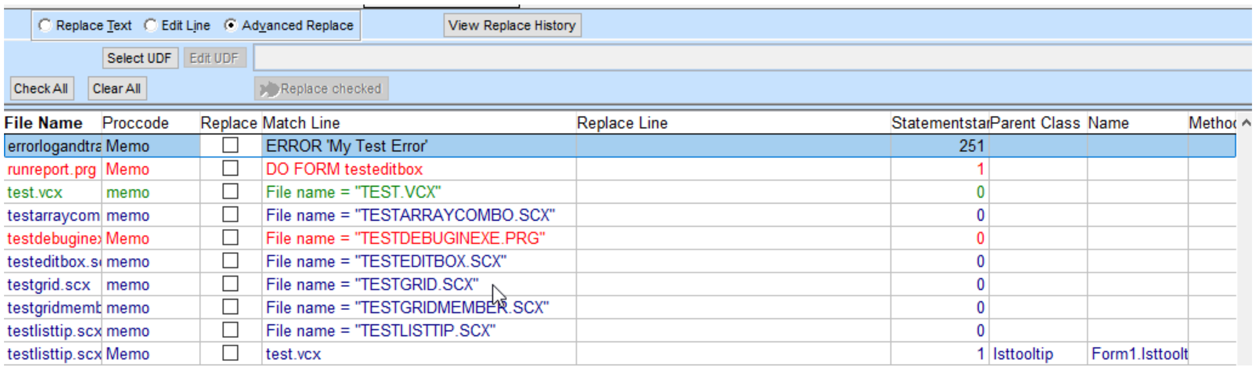
For each checked row, the matched line will be passed into the selected UDF where it can be modified using any FoxPro code that you write in the prg.

The return value from the UDF should be a string containing the new line that you wish to replace the old line.

YOU are in total control of what the replaced line contains.

For more help: Visit GoFish Replace Help on VFPx

Figure 62. This dialog appears when you choose Advanced Replace. It's both a warning and an instruction sheet.



File Name	Proccode	Replace	Match Line	Replace Line	Statementstar	Parent Class	Name	Methoc
errorlogandtra	Memo	☐	ERROR 'My Test Error'		251			
runreport.prg	Memo	☐	DO FORM testeditbox		1			
test.vcx	memo	☐	File name = "TEST.VCX"		0			
testarraycom	memo	☐	File name = "TESTARRAYCOMBO.SCX"		0			
testdebugine›	Memo	☐	File name = "TESTDEBUGINEXE.PRG"		0			
testeditbox.s	memo	☐	File name = "TESTEDITBOX.SCX"		0			
testgrid.scx	memo	☐	File name = "TESTGRID.SCX"		0			
testgridmemb	memo	☐	File name = "TESTGRIDMEMBER.SCX"		0			
testlisttip.scx	memo	☐	File name = "TESTLISTTIP.SCX"		0			
testlisttip.scx	Memo	☐	test.vcx		1	lsttooltip	Form1.lsttoolt	

Figure 63. In Advanced Replace mode, you specify a function to process each selected line.

As the warning dialog says, the function you call receives a result line as a parameter. The function is called once for each checked line. Whatever the function returns replaces the result line.

As with Replace Text, once you've specified the function, the Replace Line column of the results grid shows you what the result will be for each checked line, so you can confirm you're getting the desired results before you actually run the replace.

I've used Advanced Replace several times for actual development tasks. I wrote about the first instance; you'll find that at https://tinyurl.com/2fv3tw8h. In another case, we decided to replace (almost) all uses of the UNLOCK command in a project with calls to a wrapper function that would ensure, among other things, that changes to data were committed before unlocking.

Once we'd written the necessary function (called CmtUnLck), we were faced with finding and changing every instance of UNLOCK in a large project. (The PJX has over 2000 records.)

Some instances of UNLOCK included the IN clause while others didn't. Some that used the IN clause had indirect references to the file alias, while others used it directly. Enter Advanced Replace.

I wrote the function shown in **Listing 1** to handle the job. It first trims all spaces and tabs from the front of the line, while preserving the indentation (and converting tabs to spaces, the standard for this application).

Next, the line is divided into words. If the first word isn't "UNLOCK", then this is not a line of interest and we simply return the original line. Next, we go looking for lines that use IN; they must have at least three words ("UNLOCK", "IN", and the alias in some form). A case statement finds the alias string.

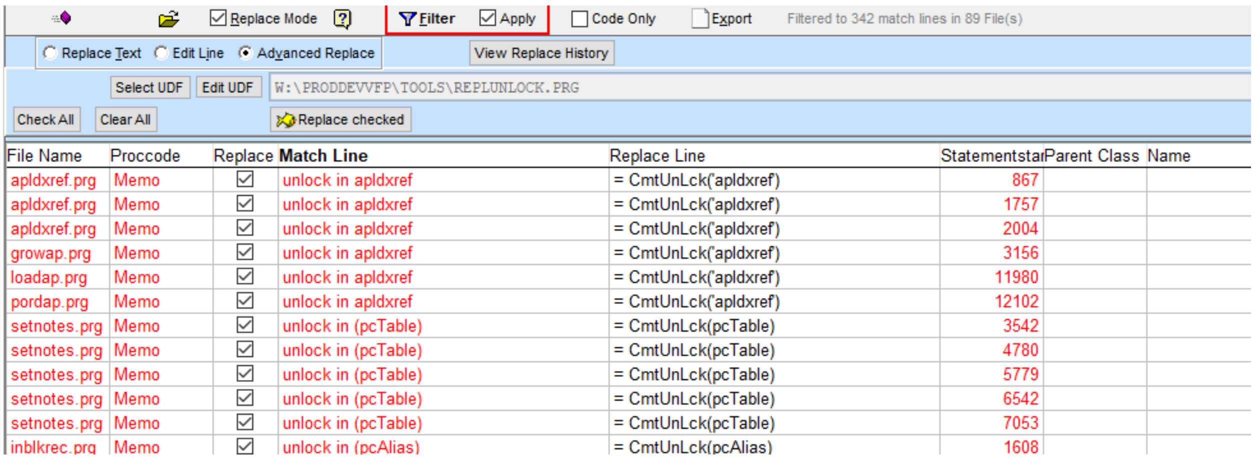
Finally, we assemble the new call which is either " = CmtUnLck()" or " = CmtUnLck('someAlias')". (We needed the equal sign in front because some of the code needed to run in FoxPro 2.6.)

Listing 1. This short function let us replace over 600 instances of UNLOCK with a function call in just a few minutes.

```
* GoFish replacement code
* Replace UNLOCK lines with call to CmtUnLck.prg.
* If UNLOCK has an IN clause, grab that alias.
LPARAMETERS tcOldLine

LOCAL lcTrimmed, lcPrefix, laWords[1], lnWords, lcAliasString
LOCAL lcReturn

lcTrimmed = LTRIM(m.tcOldLine, ' ', CHR(9))
lcPrefix = LEFT(m.tcOldLine, LEN(m.tcOldLine) - LEN(m.lcTrimmed))
lcPrefix = STRTRAN(m.lcPrefix, CHR(9), SPACE(4))

lnWords = ALINES(laWords, m.lcTrimmed, ' ')

IF ! (UPPER(laWords[1]) == "UNLOCK")

   lcReturn = m.tcOldLine

ELSE

   IF m.lnWords >= 3
      * Modified 6-March-2018 by TEG
      * Check whether filename has parens or is already quoted
      DO CASE
      CASE INLIST(LEFT(laWords[3], 1), ['], ["])
         lcAliasString = laWords[3]
      CASE m.lnWords > 3 AND laWords[3] = '('
         lcAliasString = laWords[4]
      CASE LEFT(laWords[3], 1) = '('
         lcAliasString = SUBSTR(laWords[3], 2, LEN(ALLTRIM(laWords[3])) - 2)
      OTHERWISE
         lcAliasString = ['] + laWords[3] + [']
      ENDCASE

      lcReturn = '= CmtUnLck(' + m.lcAliasString + ')'
   ELSE
      lcReturn = '= CmtUnLck()'
   ENDIF

   lcReturn = m.lcPrefix + m.lcReturn
ENDIF

RETURN m.lcReturn
```

To specify the function, click the Select UDF button and navigate to find and select the desired function. As with Replace Text, check the result lines where you want to apply the function.

Figure 64 shows the results grid after the function has been specified and all rows checked. You can review the contents and if the function isn't giving exactly the results you want, click the Edit UDF button to open the function in a normal code-editing window.



File Name	Proccode	Replace	Match Line	Replace Line	Statementstar	Parent Class	Name
apldxref.prg	Memo	☑	unlock in apldxref	= CmtUnLck('apldxref')	867		
apldxref.prg	Memo	☑	unlock in apldxref	= CmtUnLck('apldxref')	1757		
apldxref.prg	Memo	☑	unlock in apldxref	= CmtUnLck('apldxref')	2004		
growap.prg	Memo	☑	unlock in apldxref	= CmtUnLck('apldxref')	3156		
loadap.prg	Memo	☑	unlock in apldxref	= CmtUnLck('apldxref')	11980		
pordap.prg	Memo	☑	unlock in apldxref	= CmtUnLck('apldxref')	12102		
setnotes.prg	Memo	☑	unlock in (pcTable)	= CmtUnLck(pcTable)	3542		
setnotes.prg	Memo	☑	unlock in (pcTable)	= CmtUnLck(pcTable)	4780		
setnotes.prg	Memo	☑	unlock in (pcTable)	= CmtUnLck(pcTable)	5779		
setnotes.prg	Memo	☑	unlock in (pcTable)	= CmtUnLck(pcTable)	6542		
setnotes.prg	Memo	☑	unlock in (pcTable)	= CmtUnLck(pcTable)	7053		
inblkrec.prg	Memo	☑	unlock in (pcAlias)	= CmtUnLck(pcAlias)	1608		

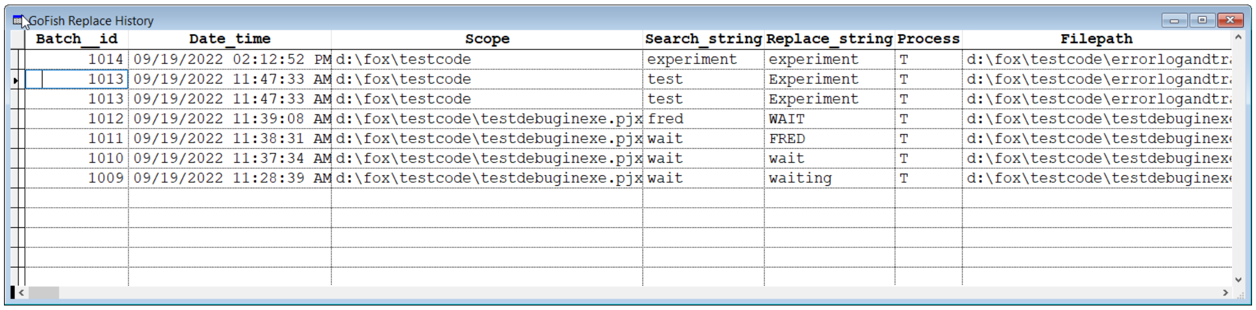
Figure 64. In Advanced Replace, as in Replace Text, you can see the result before you commit to the replace.

When you're satisfied, click the Replace Checked button to perform the replacement. As always, you'll get the scary warning and, if you haven't turned it off, your files will be backed up before making the changes.

Note that GoFish remembers your last replacement; when you check the Replace Mode checkbox, even for a different search, you'll see the previous replace type chosen and your replacement string or function will be filled in. If you last used Advanced Replace, the warning shown in **Figure 62** appears. You have to click OK before you can do anything, even choose a different replace type.

Replace history

You can see a list of the replacements you've made by clicking the View Replace History button.[11] The button opens a Browse showing each replacement you've done, as in **Figure 65**. The word "view" in the button caption seems to be quite literal. You can look through this list (and there's lots more data off to the right), but you can't change anything or change the state of the main GoFish form.



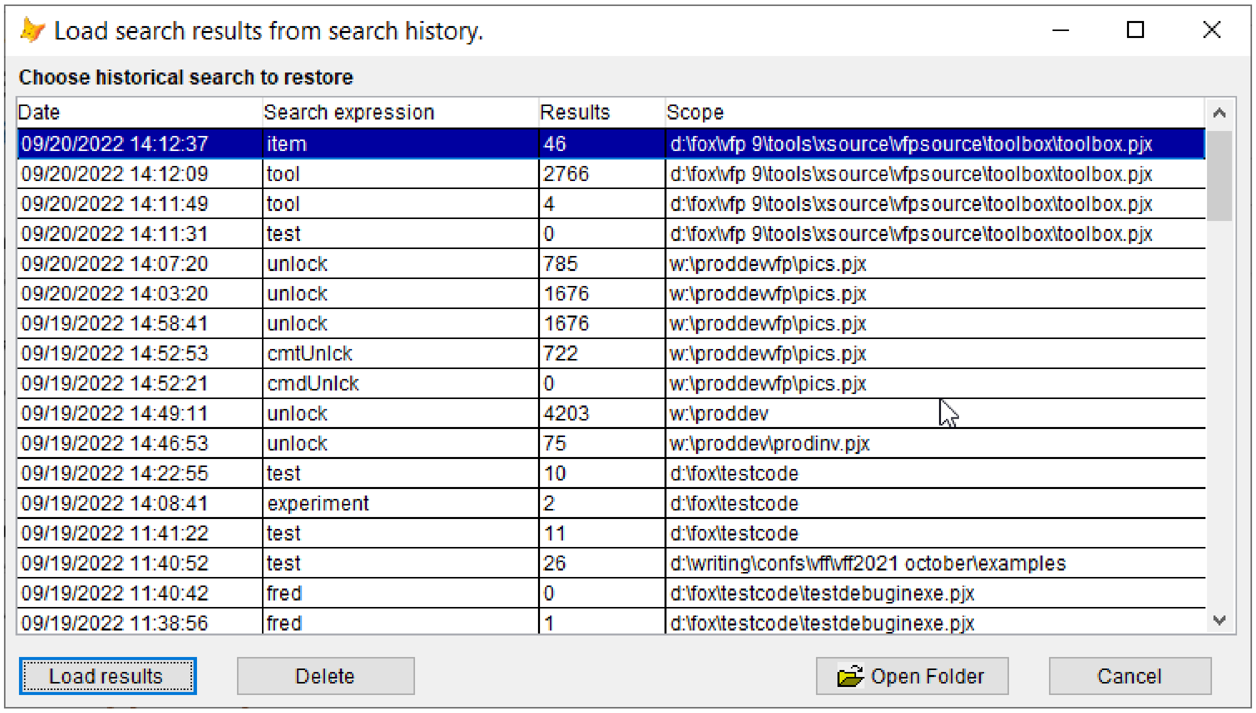
Batch__id	Date_time	Scope	Search_string	Replace_string	Process	Filepath
1014	09/19/2022 02:12:52 PM	d:\fox\testcode	experiment	experiment	T	d:\fox\testcode\errorlogandtr
1013	09/19/2022 11:47:33 AM	d:\fox\testcode	test	Experiment	T	d:\fox\testcode\errorlogandtr
1013	09/19/2022 11:47:33 AM	d:\fox\testcode	test	Experiment	T	d:\fox\testcode\errorlogandtr
1012	09/19/2022 11:39:08 AM	d:\fox\testcode\testdebuginexe.pjx	fred	WAIT	T	d:\fox\testcode\testdebuginex
1011	09/19/2022 11:38:31 AM	d:\fox\testcode\testdebuginexe.pjx	wait	FRED	T	d:\fox\testcode\testdebuginex
1010	09/19/2022 11:37:34 AM	d:\fox\testcode\testdebuginexe.pjx	wait	wait	T	d:\fox\testcode\testdebuginex
1009	09/19/2022 11:28:39 AM	d:\fox\testcode\testdebuginexe.pjx	wait	waiting	T	d:\fox\testcode\testdebuginex

Figure 65. View Replace History lets you see what replacements you've made.

Restoring old searches

GoFish remembers your previous searches and allows you to restore them. Click the History button in the Search panel and a form appears (**Figure 66**). Choose an item from the grid (either by double-clicking it or by selecting it and clicking Load Results) and GoFish loads the results from that search. Note that it doesn't run the search again, so if the underlying code has changed, the results may be out-of-date. Of course, once you have the search parameters loaded, it's easy to click GoFish to run the search again.

[11] When I initially tried to test this feature, it gave me an error. I was able to find and fix the bug. The version of GoFish now available on VFPX has this bug fixed.

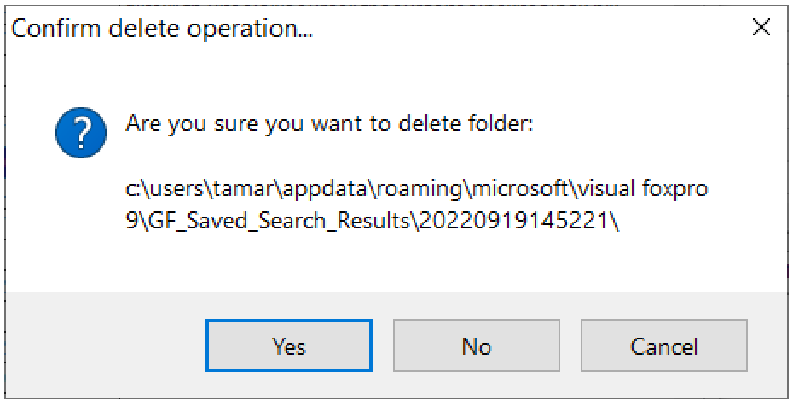

Load search results from search history.

Choose historical search to restore

Date	Search expression	Results	Scope
09/20/2022 14:12:37	item	46	d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx
09/20/2022 14:12:09	tool	2766	d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx
09/20/2022 14:11:49	tool	4	d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx
09/20/2022 14:11:31	test	0	d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx
09/20/2022 14:07:20	unlock	785	w:\proddevvfp\pics.pjx
09/20/2022 14:03:20	unlock	1676	w:\proddevvfp\pics.pjx
09/19/2022 14:58:41	unlock	1676	w:\proddevvfp\pics.pjx
09/19/2022 14:52:53	cmtUnlck	722	w:\proddevvfp\pics.pjx
09/19/2022 14:52:21	cmdUnlck	0	w:\proddevvfp\pics.pjx
09/19/2022 14:49:11	unlock	4203	w:\proddev
09/19/2022 14:46:53	unlock	75	w:\proddev\prodinv.pjx
09/19/2022 14:22:55	test	10	d:\fox\testcode
09/19/2022 14:08:41	experiment	2	d:\fox\testcode
09/19/2022 11:41:22	test	11	d:\fox\testcode
09/19/2022 11:40:52	test	26	d:\writing\confs\vff\vff2021 october\examples
09/19/2022 11:40:42	fred	0	d:\fox\testcode\testdebuginexe.pjx
09/19/2022 11:38:56	fred	1	d:\fox\testcode\testdebuginexe.pjx

Figure 66. GoFish can restore a previous search.

If you're done with any of the search results (perhaps you ran the wrong search as I did in a couple of the lines shown), you can remove that search from the list (which also removes the stored results from your drive) by clicking Delete. When you do so, you're prompted to confirm, as in **Figure 67**. That dialog gives us some clues about how the results are stored. It appears they're in the folder hierarchy that starts at HOME(7) and each result is put in a separate folder named with a timestamp indicating when it was executed.

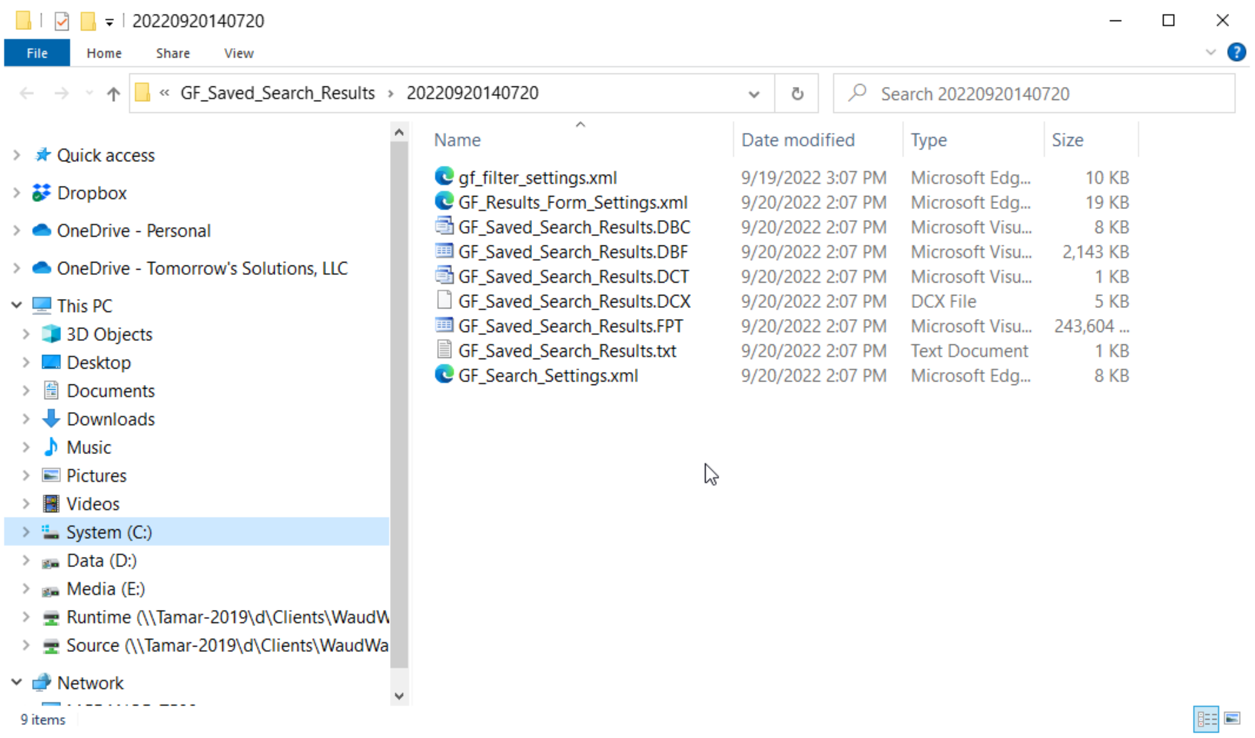

Confirm delete operation...

Are you sure you want to delete folder:

c:\users\tamar\appdata\roaming\microsoft\visual foxpro 9\GF_Saved_Search_Results\20220919145221\

Figure 67. Before deleting a search result, you're prompted to confirm.

You can take a look at the folder where a given result is stored. Highlight the relevant line and click Open Folder. Windows Explorer opens to the folder for that search result, as in **Figure 68**.



Name	Date modified	Type	Size
gf_filter_settings.xml	9/19/2022 3:07 PM	Microsoft Edg...	10 KB
GF_Results_Form_Settings.xml	9/20/2022 2:07 PM	Microsoft Edg...	19 KB
GF_Saved_Search_Results.DBC	9/20/2022 2:07 PM	Microsoft Visu...	8 KB
GF_Saved_Search_Results.DBF	9/20/2022 2:07 PM	Microsoft Visu...	2,143 KB
GF_Saved_Search_Results.DCT	9/20/2022 2:07 PM	Microsoft Visu...	1 KB
GF_Saved_Search_Results.DCX	9/20/2022 2:07 PM	DCX File	5 KB
GF_Saved_Search_Results.FPT	9/20/2022 2:07 PM	Microsoft Visu...	243,604 ...
GF_Saved_Search_Results.txt	9/20/2022 2:07 PM	Text Document	1 KB
GF_Search_Settings.xml	9/20/2022 2:07 PM	Microsoft Edg...	8 KB

Figure 68. The Open Folder button of the GoFish History dialog opens Explorer to the folder holding results from the specified search.

The history button has a context menu (shown in **Figure 69**) that lets you do some things very quickly. You can save or restore the most recent search or open the search history with the first three items, respectively. The two items at the bottom change GoFish settings.

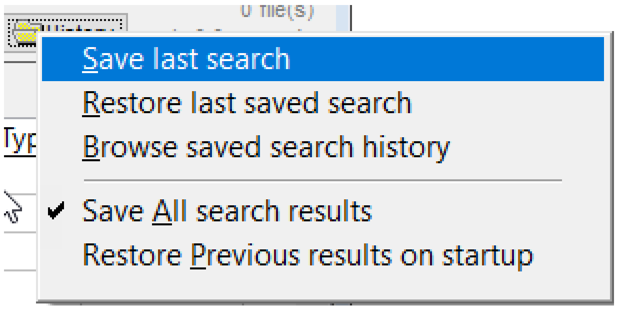
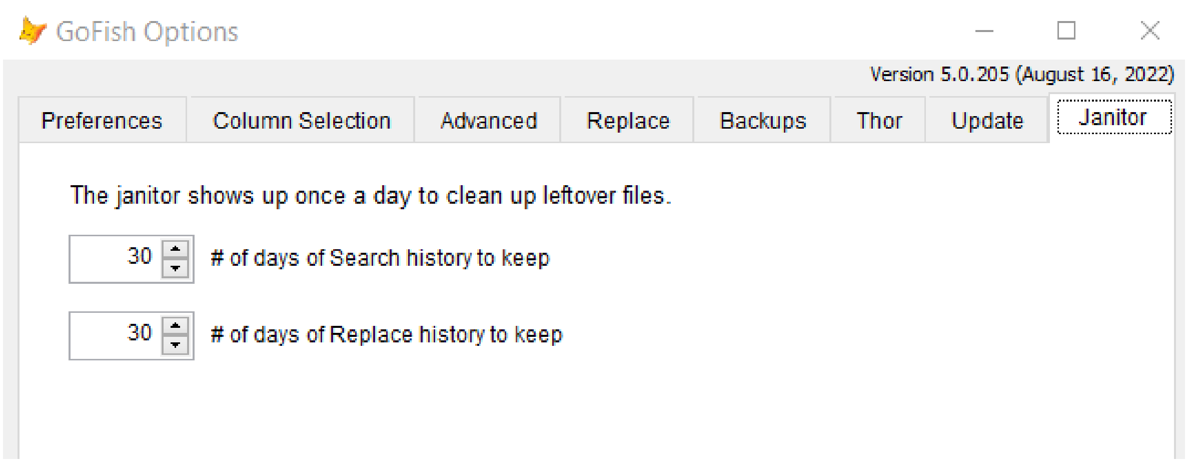

Save last search
Restore last saved search
Browse saved search history
✓ Save All search results
Restore Previous results on startup

Figure 69. The History button's context menu makes it easy to save and restore your most recent search, and to change GoFish's behavior.

Not surprisingly, search results can take up a lot of disk space. So, in addition to the ability to delete them individually, you have control over how long GoFish retains them. The Janitor tab of the GoFish Options dialog (see **Figure 70**) lets you specify how many days to keep the search history and replace history. By default, they're set for 10,000 days each, which means forever. When I changed mine to 30 days each, I freed up over 20GB of disk space.



The janitor shows up once a day to clean up leftover files.

30 # of days of Search history to keep

30 # of days of Replace history to keep

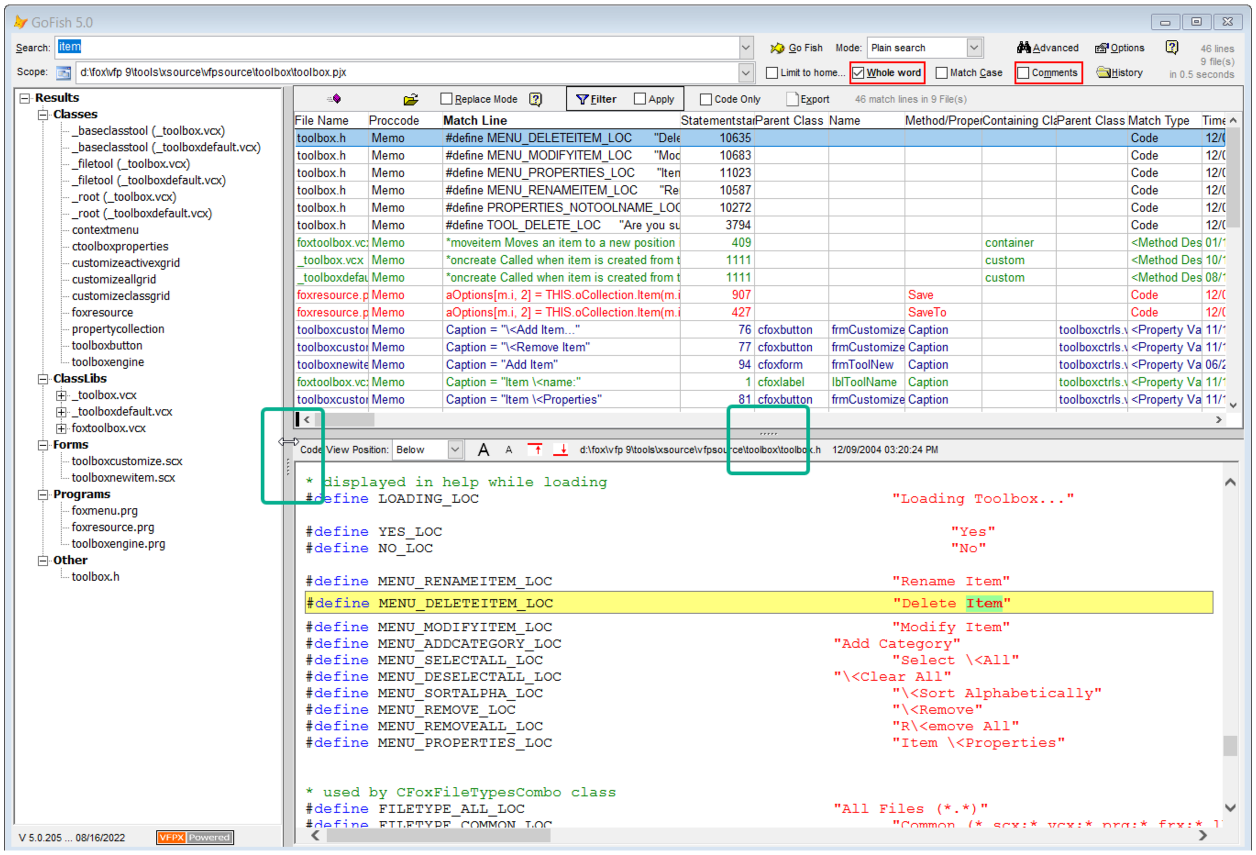
Figure 70. Use the Janitor tab of the GoFish Options dialog to decide how long to keep Search and Replace data.

Configuring GoFish

There are quite a few ways you can configure GoFish to work better for you. Some of them are right in the main interface while others appear in the Options dialog.

Customizing the GoFish User Interface

There are several ways you can configure the GoFish user interface. First, the treeview, the Results Grid and the Code View can be resized by dragging the splitters between them, as shown in **Figure 71**.



Results
- Classes
  - _baseclasstool (_toolbox.vcx)
  - _baseclasstool (_toolboxdefault.vcx)
  - _filetool (_toolbox.vcx)
  - _filetool (_toolboxdefault.vcx)
  - _root (_toolbox.vcx)
  - _root (_toolboxdefault.vcx)
  - contextmenu
  - ctoolboxproperties
  - customizeactivexgrid
  - customizeallgrid
  - customizeclassgrid
  - foxresource
  - propertycollection
  - toolboxbutton
  - toolboxengine
- ClassLibs
  - _toolbox.vcx
  - _toolboxdefault.vcx
  - foxtoolbox.vcx
- Forms
  - toolboxcustomize.scx
  - toolboxnewitem.scx
- Programs
  - foxmenu.prg
  - foxresource.prg
  - toolboxengine.prg
- Other
  - toolbox.h

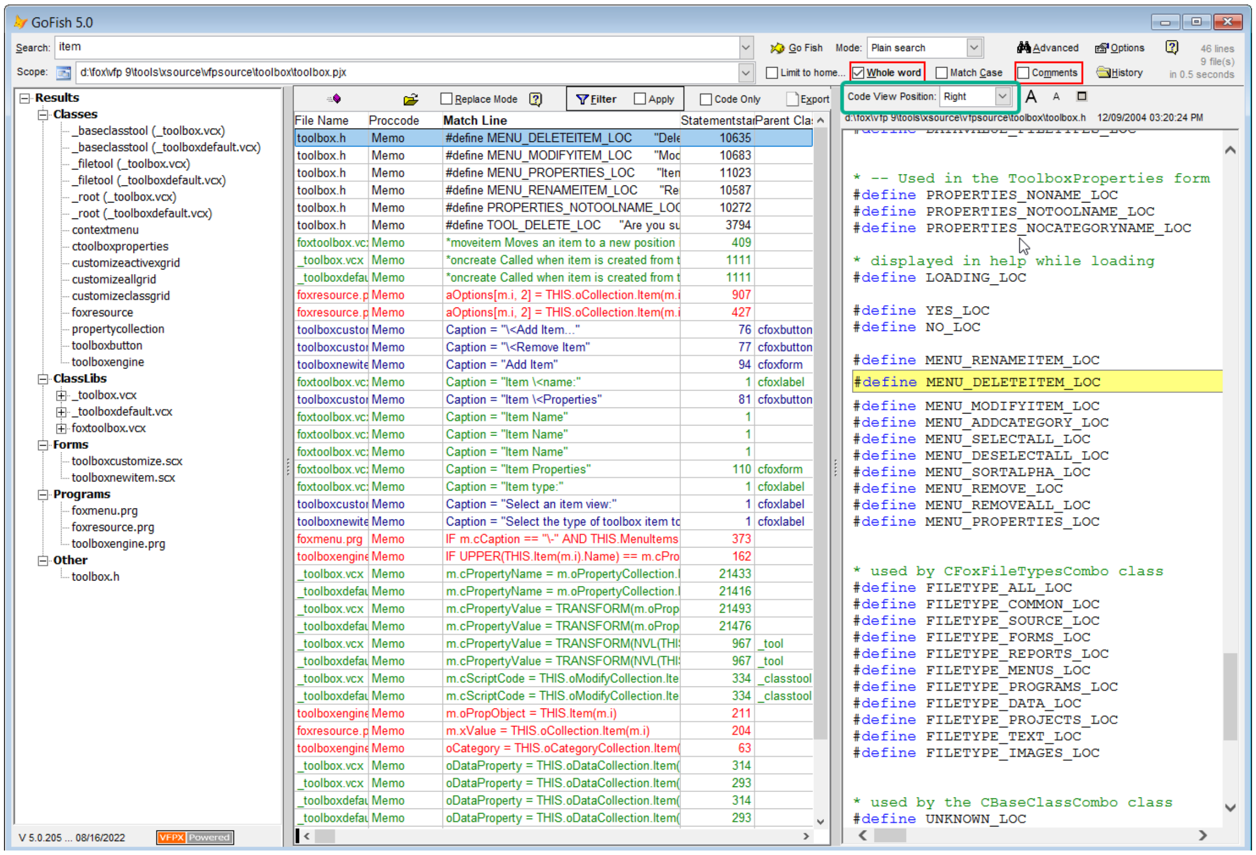
File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Proper	Containing Cla	Parent Class	Match Type	Time
toolbox.h	Memo	#define MENU_DELETEITEM_LOC "Dele	10635						Code	12/(
toolbox.h	Memo	#define MENU_MODIFYITEM_LOC "Moc	10683						Code	12/(
toolbox.h	Memo	#define MENU_PROPERTIES_LOC "Iten	11023						Code	12/(
toolbox.h	Memo	#define MENU_RENAMEITEM_LOC "Re	10587						Code	12/(
toolbox.h	Memo	#define PROPERTIES_NOTOOLNAME_LOC	10272						Code	12/(
toolbox.h	Memo	#define TOOL_DELETE_LOC "Are you su	3794						Code	12/(
foxtoolbox.vc:	Memo	*moveitem Moves an item to a new position	409				container		<Method Des	01/1
_toolbox.vcx	Memo	*oncreate Called when item is created from t	1111				custom		<Method Des	10/1
_toolboxdefau	Memo	*oncreate Called when item is created from t	1111				custom		<Method Des	08/1
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	907			Save			Code	12/(
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427			SaveTo			Code	12/(
toolboxcustor	Memo	Caption = "\<Add Item..."	76	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/1
toolboxcustor	Memo	Caption = "\<Remove Item"	77	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/1
toolboxnewite	Memo	Caption = "Add Item"	94	cfoxform	frmToolNew	Caption		toolboxctrls.\	<Property Va	06/2
foxtoolbox.vc:	Memo	Caption = "Item \<name:"	1	cfoxlabel	lblToolName	Caption		toolboxctrls.\	<Property Va	11/1
toolboxcustor	Memo	Caption = "Item \<Properties"	81	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/1

d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.h 12/09/2004 03:20:24 PM

```
* displayed in help while loading
#define LOADING_LOC                                    "Loading Toolbox..."

#define YES_LOC                                               "Yes"
#define NO_LOC                                                "No"

#define MENU_RENAMEITEM_LOC                            "Rename Item"
#define MENU_DELETEITEM_LOC                            "Delete Item"
#define MENU_MODIFYITEM_LOC                            "Modify Item"
#define MENU_ADDCATEGORY_LOC                     "Add Category"
#define MENU_SELECTALL_LOC                             "Select \<All"
#define MENU_DESELECTALL_LOC                     "\<Clear All"
#define MENU_SORTALPHA_LOC                             "\<Sort Alphabetically"
#define MENU_REMOVE_LOC                                "\<Remove"
#define MENU_REMOVEALL_LOC                             "R\<emove All"
#define MENU_PROPERTIES_LOC                            "Item \<Properties"


* used by CFoxFileTypesCombo class
#define FILETYPE_ALL_LOC                         "All Files (*.*)"
#define FILETYPE_COMMON_LOC                            "Common (*.scx;*.vcx;*.prg;*.frx;*.
```

Figure 71. The three panes that show results can be resized using the splitters, highlighted here.

You can move the Code View pane to be next to, rather than below, the Results Grid. Use the dropdown at the top of the Code View grid. In **Figure 72**, Code View has been moved to the right.



```
Results
  Classes
    _baseclasstool (_toolbox.vcx)
    _baseclasstool (_toolboxdefault.vcx)
    _filetool (_toolbox.vcx)
    _filetool (_toolboxdefault.vcx)
    _root (_toolbox.vcx)
    _root (_toolboxdefault.vcx)
    contextmenu
    ctoolboxproperties
    customizeactivexgrid
    customizeallgrid
    customizeclassgrid
    foxresource
    propertycollection
    toolboxbutton
    toolboxengine
  ClassLibs
    _toolbox.vcx
    _toolboxdefault.vcx
    foxtoolbox.vcx
  Forms
    toolboxcustomize.scx
    toolboxnewitem.scx
  Programs
    foxmenu.prg
    foxresource.prg
    toolboxengine.prg
  Other
    toolbox.h
```

File Name	Proccode	Match Line	Statementstar	Parent Cla:
toolbox.h	Memo	#define MENU_DELETEITEM_LOC "Dele	10635	
toolbox.h	Memo	#define MENU_MODIFYITEM_LOC "Moc	10683	
toolbox.h	Memo	#define MENU_PROPERTIES_LOC "Iten	11023	
toolbox.h	Memo	#define MENU_RENAMEITEM_LOC "Re	10587	
toolbox.h	Memo	#define PROPERTIES_NOTOOLNAME_LOC	10272	
toolbox.h	Memo	#define TOOL_DELETE_LOC "Are you su	3794	
foxtoolbox.vc:	Memo	*moveitem Moves an item to a new position	409	
_toolbox.vcx	Memo	*oncreate Called when item is created from t	1111	
_toolboxdefau	Memo	*oncreate Called when item is created from t	1111	
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	907	
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427	
toolboxcustor	Memo	Caption = "\<Add Item..."	76	cfoxbutton
toolboxcustor	Memo	Caption = "\<Remove Item"	77	cfoxbutton
toolboxnewite	Memo	Caption = "Add Item"	94	cfoxform
foxtoolbox.vc:	Memo	Caption = "Item \<name:"	1	cfoxlabel
toolboxcustor	Memo	Caption = "Item \<Properties"	81	cfoxbutton
foxtoolbox.vc:	Memo	Caption = "Item Name"	1	
foxtoolbox.vc:	Memo	Caption = "Item Name"	1	
foxtoolbox.vc:	Memo	Caption = "Item Name"	1	
foxtoolbox.vc:	Memo	Caption = "Item Properties"	110	cfoxform
foxtoolbox.vc:	Memo	Caption = "Item type:"	1	cfoxlabel
toolboxcustor	Memo	Caption = "Select an item view:"	1	cfoxlabel
toolboxnewite	Memo	Caption = "Select the type of toolbox item tc	1	cfoxlabel
foxmenu.prg	Memo	IF m.cCaption == "\-" AND THIS.MenuItems	373	
toolboxengine	Memo	IF UPPER(THIS.Item(m.i).Name) == m.cPro	162	
_toolbox.vcx	Memo	m.cPropertyName = m.oPropertyCollection.I	21433	
_toolboxdefau	Memo	m.cPropertyName = m.oPropertyCollection.I	21416	
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21493	
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21476	
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(NVL(THI:	967	_tool
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(NVL(THI:	967	_tool
_toolbox.vcx	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool
_toolboxdefau	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool
toolboxengine	Memo	m.oPropObject = THIS.Item(m.i)	211	
foxresource.p	Memo	m.xValue = THIS.oCollection.Item(m.i)	204	
toolboxengine	Memo	oCategory = THIS.oCategoryCollection.Item(	63	
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	314	
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	293	
_toolboxdefau	Memo	oDataProperty = THIS.oDataCollection.Item(	314	
_toolboxdefau	Memo	oDataProperty = THIS.oDataCollection.Item(	293	

```
* -- Used in the ToolboxProperties form
#define PROPERTIES_NONAME_LOC
#define PROPERTIES_NOTOOLNAME_LOC
#define PROPERTIES_NOCATEGORYNAME_LOC

* displayed in help while loading
#define LOADING_LOC

#define YES_LOC
#define NO_LOC

#define MENU_RENAMEITEM_LOC
#define MENU_DELETEITEM_LOC
#define MENU_MODIFYITEM_LOC
#define MENU_ADDCATEGORY_LOC
#define MENU_SELECTALL_LOC
#define MENU_DESELECTALL_LOC
#define MENU_SORTALPHA_LOC
#define MENU_REMOVE_LOC
#define MENU_REMOVEALL_LOC
#define MENU_PROPERTIES_LOC

* used by CFoxFileTypesCombo class
#define FILETYPE_ALL_LOC
#define FILETYPE_COMMON_LOC
#define FILETYPE_SOURCE_LOC
#define FILETYPE_FORMS_LOC
#define FILETYPE_REPORTS_LOC
#define FILETYPE_MENUS_LOC
#define FILETYPE_PROGRAMS_LOC
#define FILETYPE_DATA_LOC
#define FILETYPE_PROJECTS_LOC
#define FILETYPE_TEXT_LOC
#define FILETYPE_IMAGES_LOC

* used by the CBaseClassCombo class
#define UNKNOWN_LOC
```

Figure 72. The Code View pane can be to either side of the Results grid, not just below it. Here, the dropdown you use the specify the Code View position is highlighted.

The same row that contains the dropdown has buttons to increase and decrease the font size in the Code View page, and to shrink and grow the pane. Exactly what you can do (and what buttons you see) varies depending on the position of the pane. When it's below the Results Grid, you can enlarge it to fill the entire space occupied by both the Results Grid and the Code View or shrink it to a single row below the Results Grid. The buttons for shrinking and growing are shown in **Figure 73**. In **Figure 74**, the Code View has been shrunk and the shrink button has changed to a restore button. When the Code View pane is to the left or right of the Results Grid, the only option is to expand to fill the entire combined space.

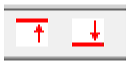
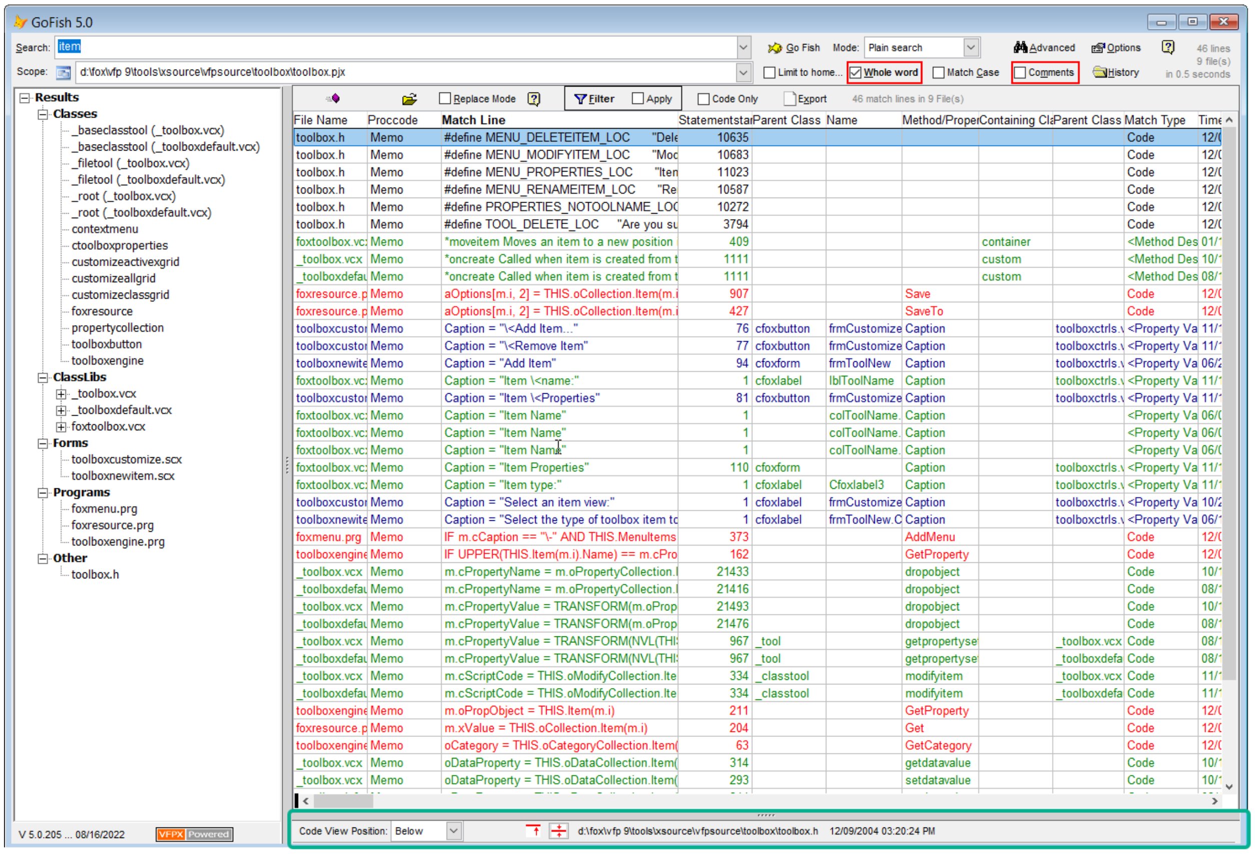


Figure 73. These buttons let you enlarge and shrink the Code View pane. Exactly what buttons you see depends on the current situation.



GoFish 5.0

Search: item

Scope: d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx

Mode: Plain search

☐ Limit to home... ☑ Whole word ☐ Match Case ☐ Comments

46 lines 9 file(s) in 0.5 seconds

☐ Replace Mode ☐ Apply ☐ Code Only

46 match lines in 9 File(s)

- Results
  - Classes
    - _baseclasstool (_toolbox.vcx)
    - _baseclasstool (_toolboxdefault.vcx)
    - _filetool (_toolbox.vcx)
    - _filetool (_toolboxdefault.vcx)
    - _root (_toolbox.vcx)
    - _root (_toolboxdefault.vcx)
    - contextmenu
    - ctoolboxproperties
    - customizeactivexgrid
    - customizeallgrid
    - customizeclassgrid
    - foxresource
    - propertycollection
    - toolboxbutton
    - toolboxengine
  - ClassLibs
    - _toolbox.vcx
    - _toolboxdefault.vcx
    - foxtoolbox.vcx
  - Forms
    - toolboxcustomize.scx
    - toolboxnewitem.scx
  - Programs
    - foxmenu.prg
    - foxresource.prg
    - toolboxengine.prg
  - Other
    - toolbox.h

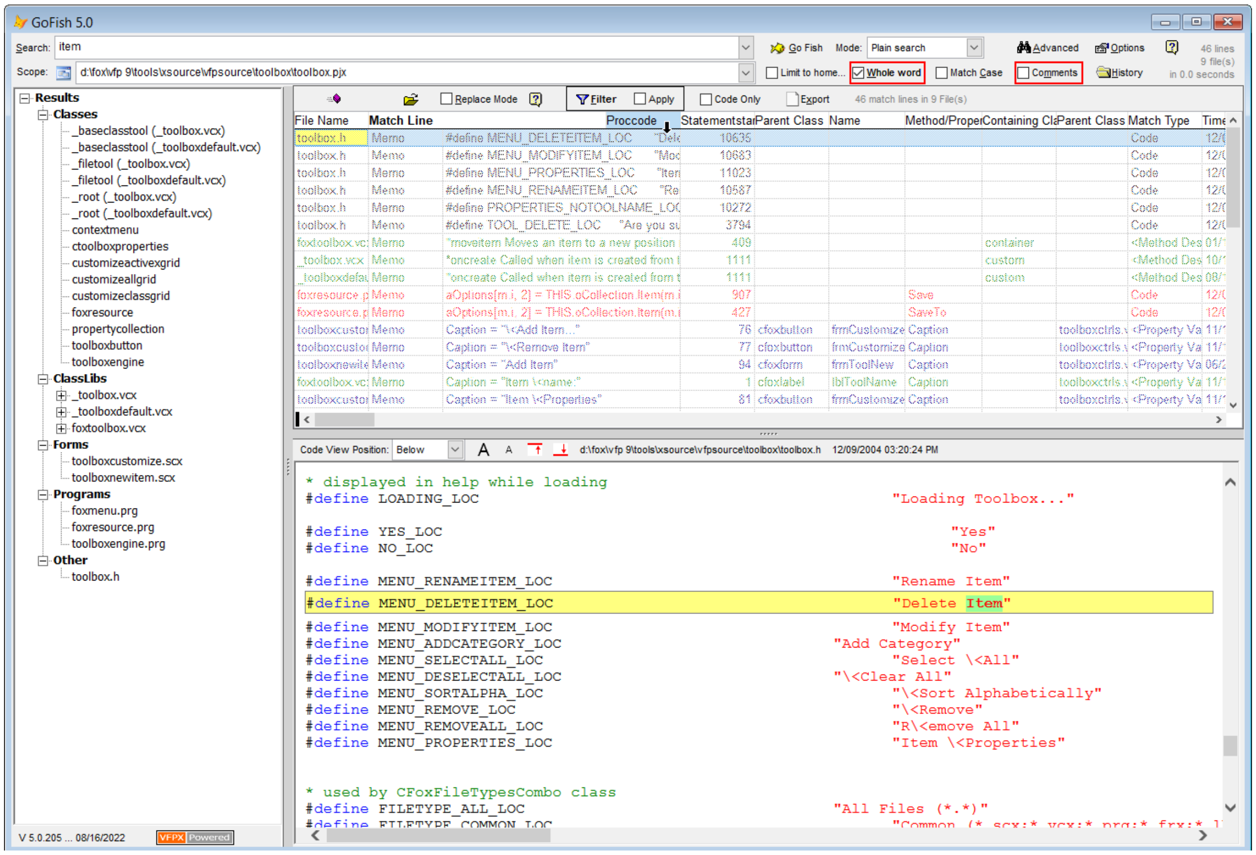
File Name	Proccode	Match Line	Statementstar	Parent Class	Name	Method/Proper	Containing Cla	Parent Class	Match Type	Time
toolbox.h	Memo	#define MENU_DELETEITEM_LOC "Dele	10635						Code	12/(
toolbox.h	Memo	#define MENU_MODIFYITEM_LOC "Moc	10683						Code	12/(
toolbox.h	Memo	#define MENU_PROPERTIES_LOC "Iten	11023						Code	12/(
toolbox.h	Memo	#define MENU_RENAMEITEM_LOC "Re	10587						Code	12/(
toolbox.h	Memo	#define PROPERTIES_NOTOOLNAME_LOC	10272						Code	12/(
toolbox.h	Memo	#define TOOL_DELETE_LOC "Are you su	3794						Code	12/(
foxtoolbox.vc	Memo	*moveitem Moves an item to a new position	409				container		<Method Des	01/1
_toolbox.vcx	Memo	*oncreate Called when item is created from t	1111				custom		<Method Des	10/1
_toolboxdefau	Memo	*oncreate Called when item is created from t	1111				custom		<Method Des	08/1
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	907			Save			Code	12/(
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427			SaveTo			Code	12/(
toolboxcusto	Memo	Caption = "\<Add Item..."	76	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/1
toolboxcusto	Memo	Caption = "\<Remove Item"	77	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/1
toolboxnewite	Memo	Caption = "Add Item"	94	cfoxform	frmToolNew	Caption		toolboxctrls.\	<Property Va	06/2
foxtoolbox.vc	Memo	Caption = "Item \<name:"	1	cfoxlabel	lblToolName	Caption		toolboxctrls.\	<Property Va	11/1
toolboxcusto	Memo	Caption = "Item \<Properties"	81	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/1
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/(
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/(
foxtoolbox.vc	Memo	Caption = "Item Name"	1		colToolName.	Caption			<Property Va	06/(
foxtoolbox.vc	Memo	Caption = "Item Properties"	110	cfoxform		Caption		toolboxctrls.\	<Property Va	11/1
foxtoolbox.vc	Memo	Caption = "Item type:"	1	cfoxlabel	Cfoxlabel3	Caption		toolboxctrls.\	<Property Va	11/1
toolboxcusto	Memo	Caption = "Select an item view:"	1	cfoxlabel	frmCustomize	Caption		toolboxctrls.\	<Property Va	10/2
toolboxnewite	Memo	Caption = "Select the type of toolbox item tc	1	cfoxlabel	frmToolNew.C	Caption		toolboxctrls.\	<Property Va	06/1
foxmenu.prg	Memo	IF m.cCaption == "\-" AND THIS.MenuItems	373			AddMenu			Code	12/(
toolboxengine	Memo	IF UPPER(THIS.Item(m.i).Name) == m.cPro	162			GetProperty			Code	12/(
_toolbox.vcx	Memo	m.cPropertyName = m.oPropertyCollection.I	21433			dropobject			Code	10/1
_toolboxdefau	Memo	m.cPropertyName = m.oPropertyCollection.I	21416			dropobject			Code	08/1
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21493			dropobject			Code	10/1
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(m.oProp	21476			dropobject			Code	08/1
_toolbox.vcx	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse		_toolbox.vcx	Code	08/1
_toolboxdefau	Memo	m.cPropertyValue = TRANSFORM(NVL(THI	967	_tool		getpropertyse		_toolboxdefa	Code	08/1
_toolbox.vcx	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool		modifyitem		_toolbox.vcx	Code	11/1
_toolboxdefau	Memo	m.cScriptCode = THIS.oModifyCollection.Ite	334	_classtool		modifyitem		_toolboxdefa	Code	11/1
toolboxengine	Memo	m.oPropObject = THIS.Item(m.i)	211			GetProperty			Code	12/(
foxresource.p	Memo	m.xValue = THIS.oCollection.Item(m.i)	204			Get			Code	12/(
toolboxengine	Memo	oCategory = THIS.oCategoryCollection.Item(	63			GetCategory			Code	12/(
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	314			getdatavalue			Code	10/1
_toolbox.vcx	Memo	oDataProperty = THIS.oDataCollection.Item(	293			setdatavalue			Code	10/1

Code View Position: Below d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.h 12/09/2004 03:20:24 PM

V 5.0.205 ... 08/16/2022

Figure 74. Here, the Code View pane has been shrunk to a single row below the Results grid (highlighted in green). Note that the shrink button has been replaced by a restore button.

Finally, you can drag columns in the Results grid to change their order. **Figure 75** shows the ProcCode column being dragged to the right. Once you move columns around, GoFish remembers the order you left them in and shows them that way the next time you open it.



Search: item
Scope: d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx
Mode: Plain search
Limit to home... [ ] Whole word [✓] Match Case [ ] Comments [ ]
46 lines
9 file(s)
in 0.0 seconds
Apply [ ] Code Only [ ]
46 match lines in 9 File(s)

- Results
  - Classes
    - _baseclasstool (_toolbox.vcx)
    - _baseclasstool (_toolboxdefault.vcx)
    - _filetool (_toolbox.vcx)
    - _filetool (_toolboxdefault.vcx)
    - _root (_toolbox.vcx)
    - _root (_toolboxdefault.vcx)
    - contextmenu
    - ctoolboxproperties
    - customizeactivexgrid
    - customizeallgrid
    - customizeclassgrid
    - foxresource
    - propertycollection
    - toolboxbutton
    - toolboxengine
  - ClassLibs
    - _toolbox.vcx
    - _toolboxdefault.vcx
    - foxtoolbox.vcx
  - Forms
    - toolboxcustomize.scx
    - toolboxnewitem.scx
  - Programs
    - foxmenu.prg
    - foxresource.prg
    - toolboxengine.prg
  - Other
    - toolbox.h

File Name	Match Line	Proccode	Statementstar	Parent Class	Name	Method/Proper	Containing Cl	Parent Class	Match Type	Time
toolbox.h	Memo	#define MENU_DELETEITEM_LOC "Dele	10635						Code	12/(
toolbox.h	Memo	#define MENU_MODIFYITEM_LOC "Moc	10683						Code	12/(
toolbox.h	Memo	#define MENU_PROPERTIES_LOC "Iter	11023						Code	12/(
toolbox.h	Memo	#define MENU_RENAMEITEM_LOC "Re	10587						Code	12/(
toolbox.h	Memo	#define PROPERTIES_NOTOOLNAME_LOC	10272						Code	12/(
toolbox.h	Memo	#define TOOL_DELETE_LOC "Are you su	3794						Code	12/(
foxtoolbox.vc:	Memo	*moveitem Moves an item to a new position	409				container		<Method Des	01/
_toolbox.vcx	Memo	*oncreate Called when item is created from t	1111				custom		<Method Des	10/
_toolboxdefau	Memo	*oncreate Called when item is created from t	1111				custom		<Method Des	08/
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	907			Save			Code	12/(
foxresource.p	Memo	aOptions[m.i, 2] = THIS.oCollection.Item(m.i	427			SaveTo			Code	12/(
toolboxcusto	Memo	Caption = "\<Add Item..."	76	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/
toolboxcusto	Memo	Caption = "\<Remove Item"	77	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/
toolboxnewite	Memo	Caption = "Add Item"	94	cfoxform	frmToolNew	Caption		toolboxctrls.\	<Property Va	06/2
foxtoolbox.vc:	Memo	Caption = "Item \<name:"	1	cfoxlabel	lblToolName	Caption		toolboxctrls.\	<Property Va	11/
toolboxcusto	Memo	Caption = "Item \<Properties"	81	cfoxbutton	frmCustomize	Caption		toolboxctrls.\	<Property Va	11/

Code View Position: Below d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.h 12/09/2004 03:20:24 PM

```
* displayed in help while loading
#define LOADING_LOC                                         "Loading Toolbox..."

#define YES_LOC                                                    "Yes"
#define NO_LOC                                                     "No"

#define MENU_RENAMEITEM_LOC                                 "Rename Item"
#define MENU_DELETEITEM_LOC                                 "Delete Item"
#define MENU_MODIFYITEM_LOC                                 "Modify Item"
#define MENU_ADDCATEGORY_LOC                         "Add Category"
#define MENU_SELECTALL_LOC                                  "Select \<All"
#define MENU_DESELECTALL_LOC                         "\<Clear All"
#define MENU_SORTALPHA_LOC                                  "\<Sort Alphabetically"
#define MENU_REMOVE_LOC                                     "\<Remove"
#define MENU_REMOVEALL_LOC                                  "R\<emove All"
#define MENU_PROPERTIES_LOC                                 "Item \<Properties"


* used by CFoxFileTypesCombo class
#define FILETYPE_ALL_LOC                             "All Files (*.*)"
#define FILETYPE_COMMON_LOC                                 "Common (*.scx;*.vcx;*.prg;*.frx;*.l
```

Figure 75. You can drag columns in the Results grid to rearrange them. GoFish remembers your changes.

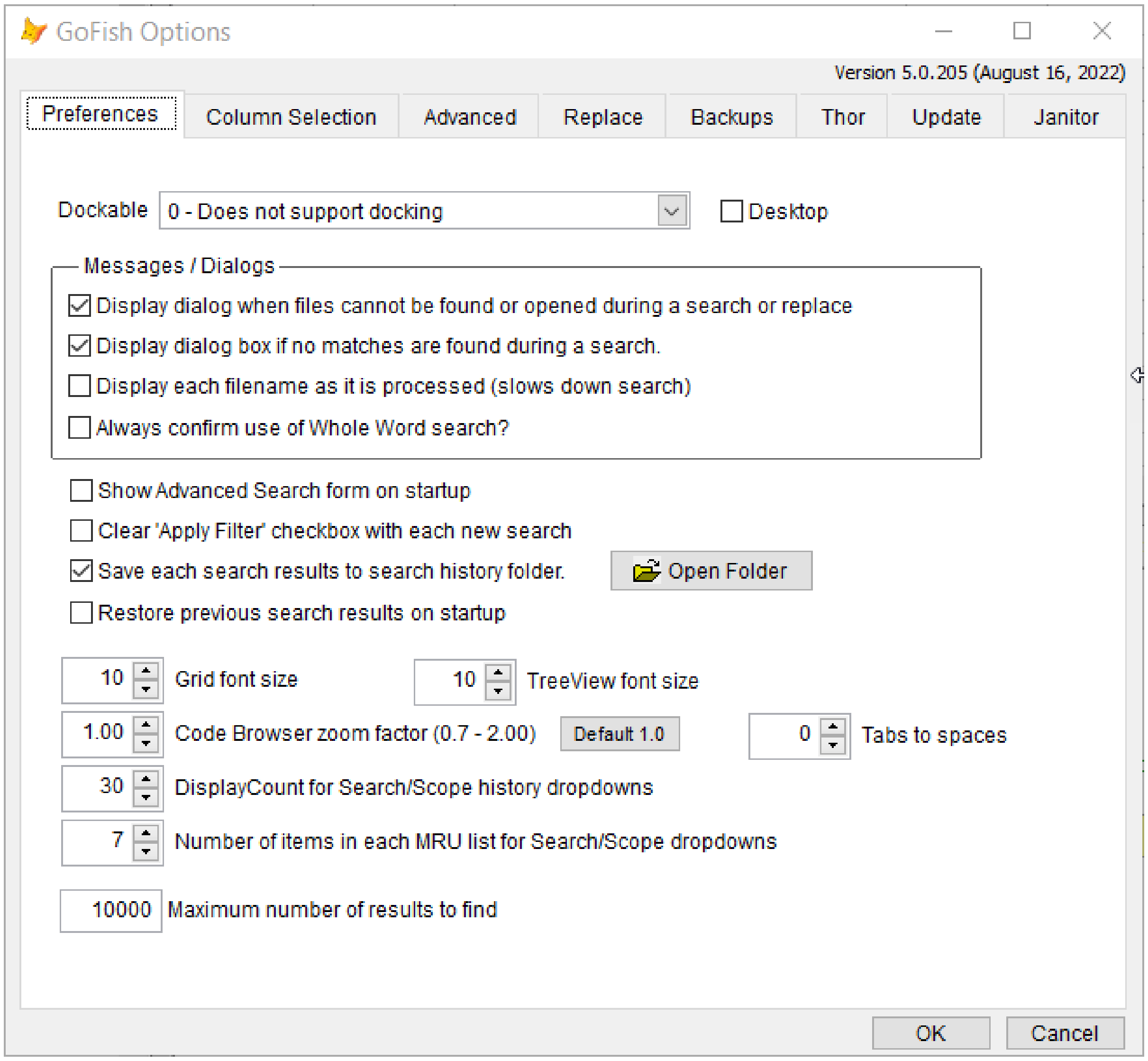
You can also change which columns are displayed; see "The Column Selection tab" in the next section.

The Options dialog

We've seen several features of the GoFish Options dialog already in this paper, but you can also control appearance and functionality, manage backups and much more.

The Preferences tab

The top section and bottom section of the Preferences page (**Figure 76**) control the appearance of the main GoFish form, including whether the form is dockable, whether it's confined to the VFP window (the Desktop checkbox), font sizes, and more. The Messages/Dialogs section lets you determine whether dialogs are shown in various situations. The middle set of checkboxes control various behaviors of GoFish—whether to always open the Advanced Search dialog on startup, whether to save search results, and more.



Dockable: 0 - Does not support docking
☐ Desktop

Messages / Dialogs
☑ Display dialog when files cannot be found or opened during a search or replace
☑ Display dialog box if no matches are found during a search.
☐ Display each filename as it is processed (slows down search)
☐ Always confirm use of Whole Word search?

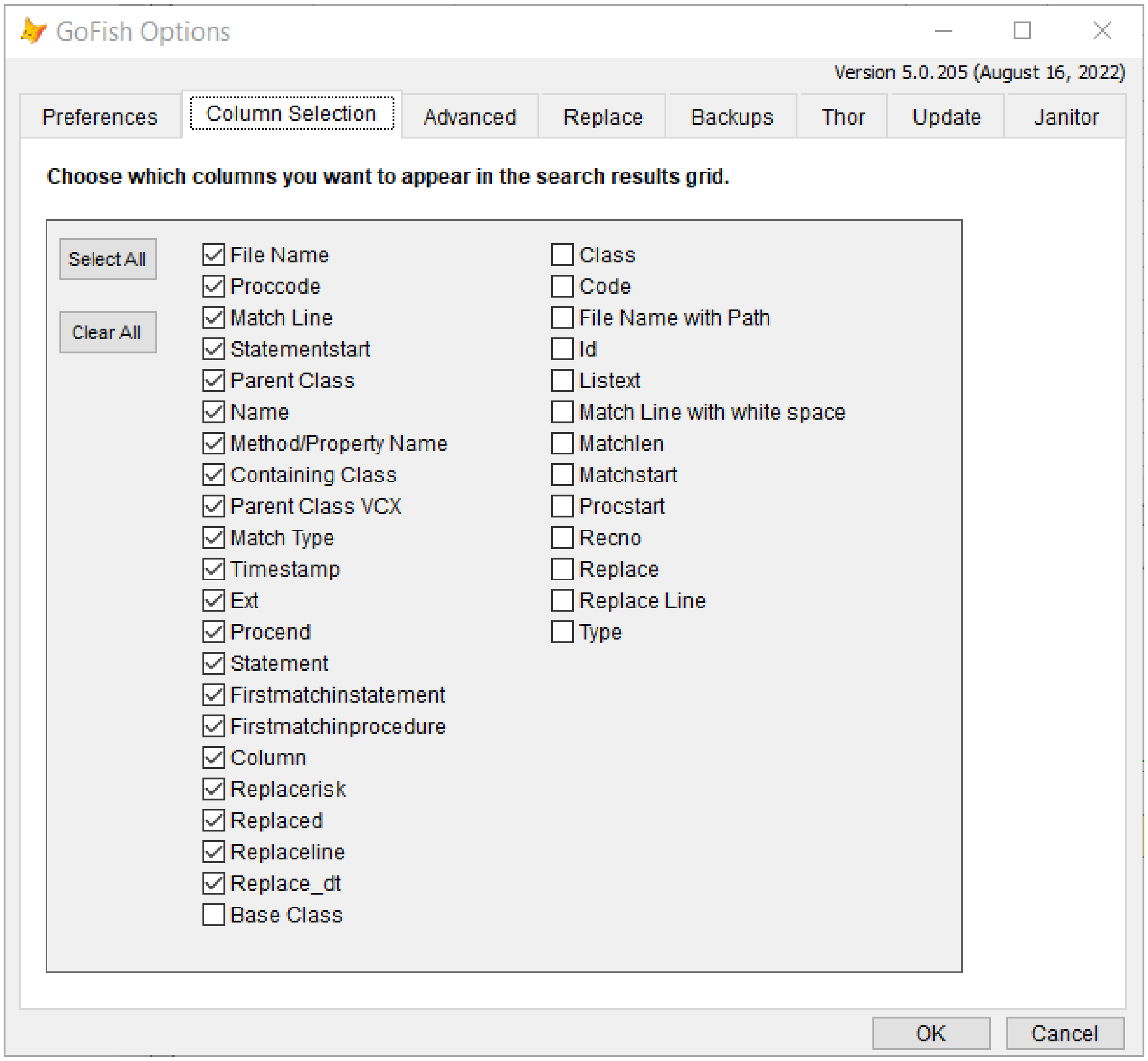
☐ Show Advanced Search form on startup
☐ Clear 'Apply Filter' checkbox with each new search
☑ Save each search results to search history folder.
☐ Restore previous search results on startup

10 Grid font size
10 TreeView font size
1.00 Code Browser zoom factor (0.7 - 2.00)
0 Tabs to spaces
30 DisplayCount for Search/Scope history dropdowns
7 Number of items in each MRU list for Search/Scope dropdowns
10000 Maximum number of results to find

Figure 76. The Preferences page of the GoFish Options dialog lets you set up many display features, as well as a few functionality items.

The Column Selection tab

The Column Selection tab lets you determine what information appears in the results grid. **Figure 77** shows the default settings. Note that some of the items here (such as the Replace checkbox, listed just as Replace here) normally appear only in certain circumstances. You can experiment here to see what works best for you.



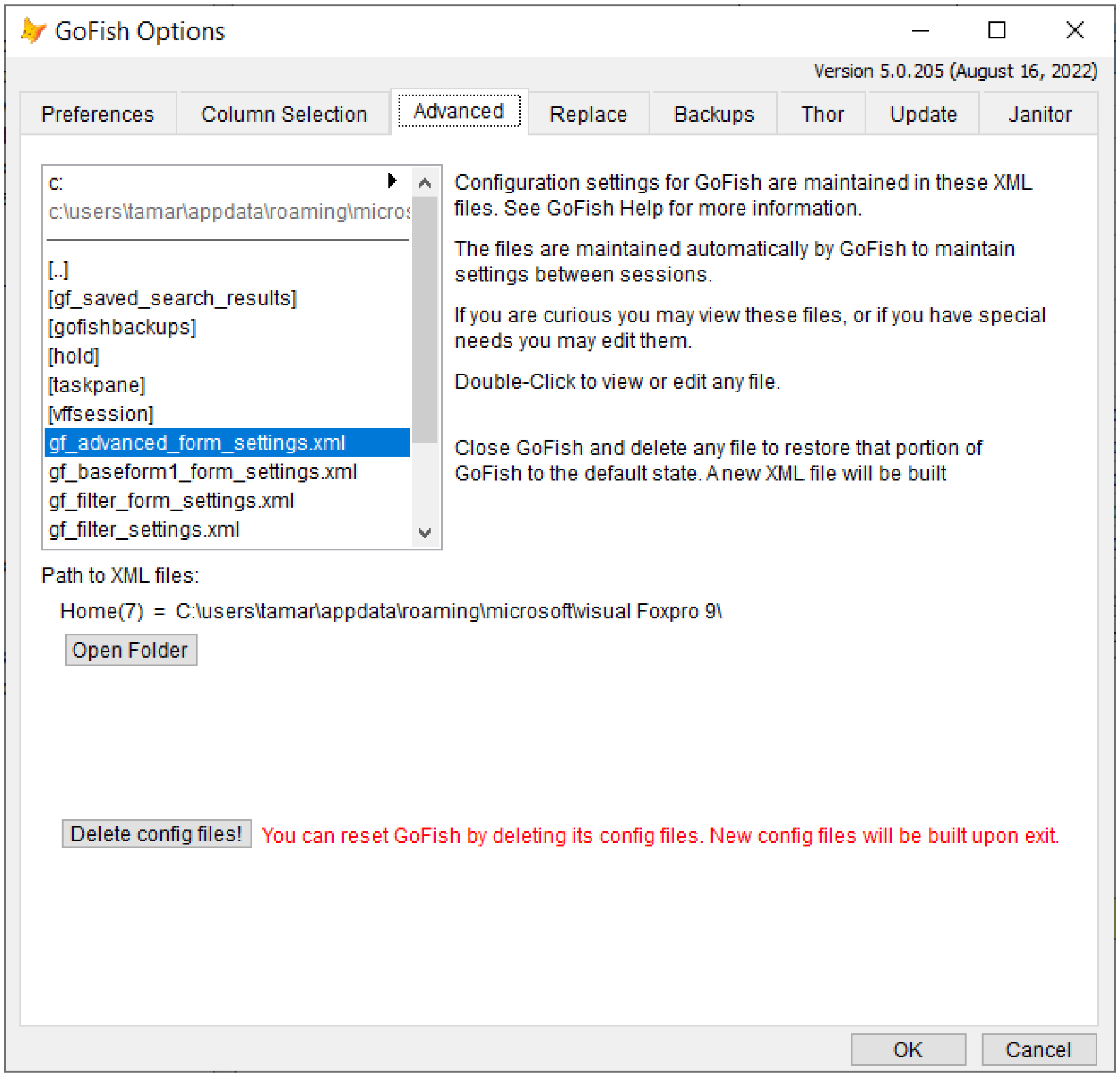
Choose which columns you want to appear in the search results grid.

- [✓] File Name
- [✓] Proccode
- [✓] Match Line
- [✓] Statementstart
- [✓] Parent Class
- [✓] Name
- [✓] Method/Property Name
- [✓] Containing Class
- [✓] Parent Class VCX
- [✓] Match Type
- [✓] Timestamp
- [✓] Ext
- [✓] Procend
- [✓] Statement
- [✓] Firstmatchinstatement
- [✓] Firstmatchinprocedure
- [✓] Column
- [✓] Replacerisk
- [✓] Replaced
- [✓] Replaceline
- [✓] Replace_dt
- [ ] Base Class
- [ ] Class
- [ ] Code
- [ ] File Name with Path
- [ ] Id
- [ ] Listext
- [ ] Match Line with white space
- [ ] Matchlen
- [ ] Matchstart
- [ ] Procstart
- [ ] Recno
- [ ] Replace
- [ ] Replace Line
- [ ] Type

Figure 77. The Column Selection page of the GoFish Options dialog determines

The Advanced tab

The Advanced tab (**Figure 78**) provides access to the XML files that contain GoFish's settings. Double-click any of the XML files in the list to open it. **Figure 79** shows part of the contents of gf_search_settings.xml; that file stores the current search settings when you close GoFish.



Configuration settings for GoFish are maintained in these XML files. See GoFish Help for more information.

The files are maintained automatically by GoFish to maintain settings between sessions.

If you are curious you may view these files, or if you have special needs you may edit them.

Double-Click to view or edit any file.

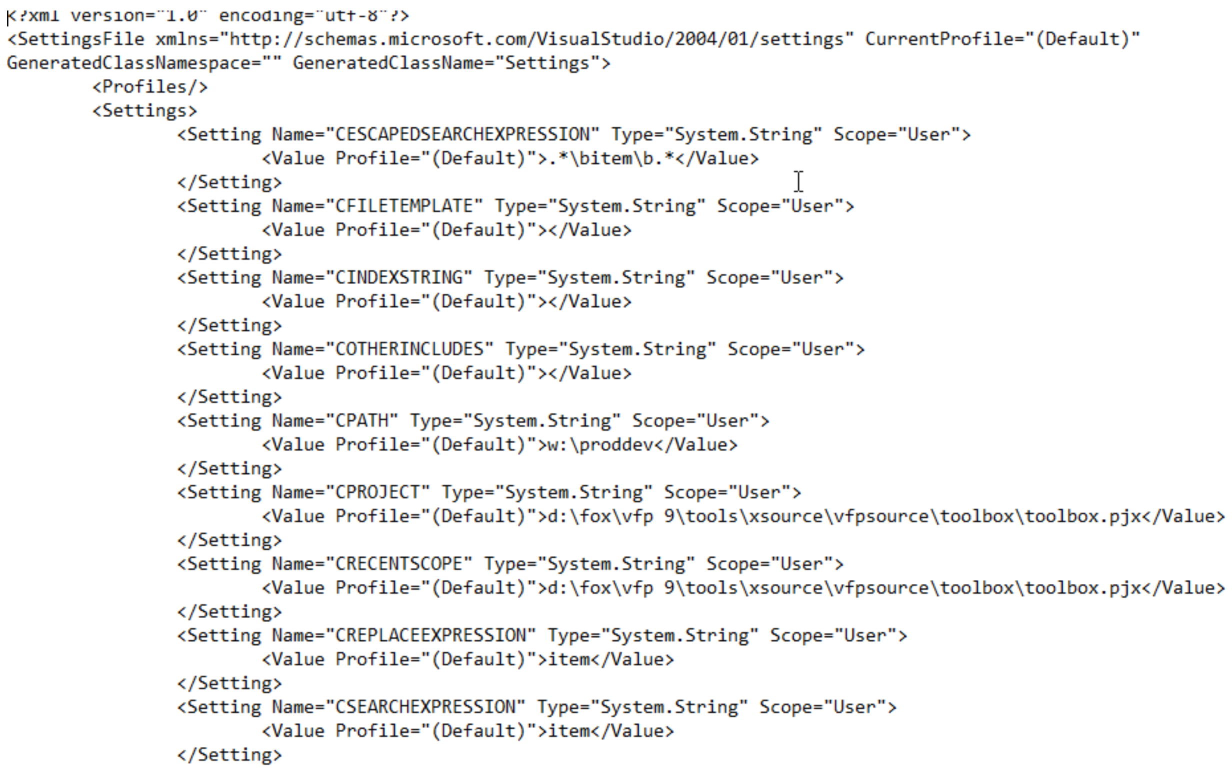
Close GoFish and delete any file to restore that portion of GoFish to the default state. A new XML file will be built

Path to XML files:

Home(7) = C:\users\tamar\appdata\roaming\microsoft\visual Foxpro 9\

You can reset GoFish by deleting its config files. New config files will be built upon exit.

Figure 78. Use the Advanced tab to explore the XML files that store the GoFish settings, as well as to discard them and start over.



```
<?xml version="1.0" encoding="utf-8"?>
<SettingsFile xmlns="http://schemas.microsoft.com/VisualStudio/2004/01/settings" CurrentProfile="(Default)"
GeneratedClassNamespace="" GeneratedClassName="Settings">
        <Profiles/>
        <Settings>
                <Setting Name="CESCAPEDSEARCHEXPRESSION" Type="System.String" Scope="User">
                        <Value Profile="(Default)">.*\bitem\b.*</Value>
                </Setting>
                <Setting Name="CFILETEMPLATE" Type="System.String" Scope="User">
                        <Value Profile="(Default)"></Value>
                </Setting>
                <Setting Name="CINDEXSTRING" Type="System.String" Scope="User">
                        <Value Profile="(Default)"></Value>
                </Setting>
                <Setting Name="COTHERINCLUDES" Type="System.String" Scope="User">
                        <Value Profile="(Default)"></Value>
                </Setting>
                <Setting Name="CPATH" Type="System.String" Scope="User">
                        <Value Profile="(Default)">w:\proddev</Value>
                </Setting>
                <Setting Name="CPROJECT" Type="System.String" Scope="User">
                        <Value Profile="(Default)">d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx</Value>
                </Setting>
                <Setting Name="CRECENTSCOPE" Type="System.String" Scope="User">
                        <Value Profile="(Default)">d:\fox\vfp 9\tools\xsource\vfpsource\toolbox\toolbox.pjx</Value>
                </Setting>
                <Setting Name="CREPLACEEXPRESSION" Type="System.String" Scope="User">
                        <Value Profile="(Default)">item</Value>
                </Setting>
                <Setting Name="CSEARCHEXPRESSION" Type="System.String" Scope="User">
                        <Value Profile="(Default)">item</Value>
                </Setting>
```

Figure 79. GoFish uses XML to store a variety of settings. Shown here is part of my search settings. Settings are saved when you close GoFish.

The Advanced tab also lets you open the folder containing the XML files; click Open Folder. You can also throw your current settings away and start from scratch by clicking Delete config files! Not surprisingly, there's a confirmation dialog for that action.

The Replace tab

The most important function of the Replace tab, setting your risk level for replacements, was discussed in "Replacing strings," earlier in this document; it's shown in **Figure 55.** However, this tab contains a few other items. First, as that section discussed, each time you actually begin a replacement, a dialog appears. A checkbox on the Replace tab lets you turn that warning dialog off.

This tab also lets you open the folder hosting the tables that contain information about replaces you've performed.

The Backup tab

As mentioned multiple times earlier in this document, by default, when you perform a replacement, the original files are backed up. The Backup tab lets you turn that feature off. (Though I've never had a GoFish replace go wrong, I don't recommend turning the feature off. Instead, shorten the period the replacement data is maintained, as described in "Restoring old searches," earlier in this document.) Be aware that the Backup tab is disabled unless Replace mode has been turned on.

The Backup tab also lets you open the folder that contains the backup files.

The Thor and Update tabs

Neither the Thor tab nor the Update tab actually let you configure anything. The Thor tab provides a little bit of information about how GoFish integrates with Thor and offers a link to the Thor website. The Update tab shows how integration with Thor provides an easy update mechanism for GoFish and provides a link to the GoFish website.

The Janitor tab

As described in "Restoring old searches," earlier in this document, the Janitor tab lets you decide how long GoFish retains search and replace histories.

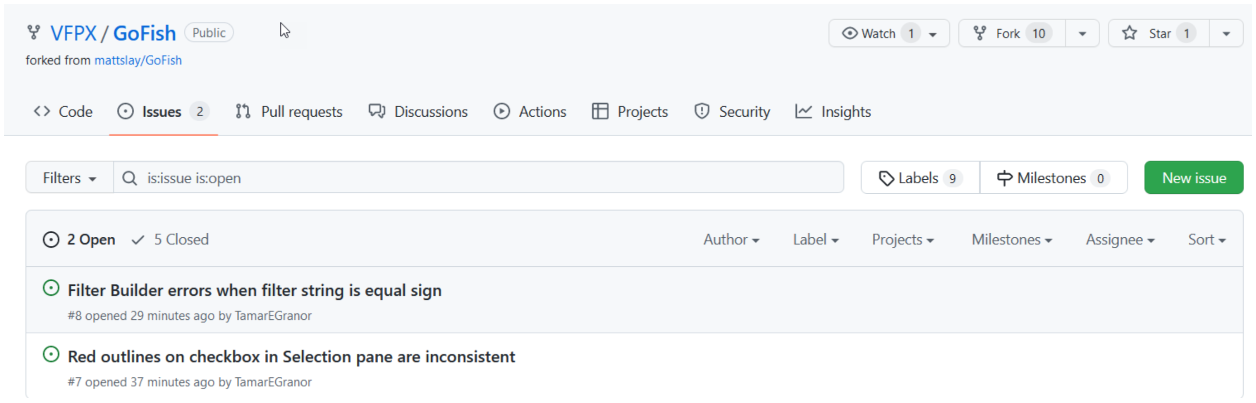
Going farther

Like so many VFP tools, GoFish is written in VFP. When you install it through Thor, you get the source code, too. Also, like many other VFP tools, GoFish stores some data in DBFs; other data is stored as XML. GoFish's data is stored in the folder returned by HOME(7), along with other VFP data.

Again like many other VFP tools, GoFish separates the search engine from the user interface. This means you can use the GoFish Search Engine in other tools and in your own applications. The engine is found in GoFishSearchEngine.PRG in the Lib folder of the GoFish source code folder. (If you have Thor installed, you can find the source code by choosing Thor | More | Open Folder | Apps from the menu. You should see GoFish5 in the folder that opens.) The class definition opens with a long list of properties, some of which are the ones you need to set to perform a search. There's not much documentation for GoFish's object model, but you'll find some in the GoFish chapter in the book "VFPX: Open Source Treasure for the VFP Developer."

Improvements to GoFish continue. If you have ideas for how to improve it, submit them on the Issues page of the GoFish site (**Figure 80**).[12] Use the same page to report bugs. If you're interested in helping improve GoFish, check the Issues page for things to work on. To learn the right way to get a copy of the code to work with and to submit your updates, check https://tinyurl.com/25ej87sw.

[12] Be aware that Matt Slay's original GitHub site for GoFish still exists. However, since Matt's death, there is no one who can make changes to that site, including marking bugs as resolved. Be sure to post bugs and enhancement requests to the VFPX GoFish GitHub site, so that the VFP community can respond to them.



is:issue is:open

2 Open ✓ 5 Closed

Filter Builder errors when filter string is equal sign
#8 opened 29 minutes ago by TamarEGranor

Red outlines on checkbox in Selection pane are inconsistent
#7 opened 37 minutes ago by TamarEGranor

Figure 80. You can submit bugs or enhancement requests on the Issues page of the GoFish site. You can also find issues for you to work on.

Summing up

GoFish is a very capable, VFP-specific search tool that makes it easy to find things inside both VFP projects and folders. Every VFP developer should have it in their toolbox.