

Adobe

Adobe ColdFusion Builder extension for Visual Studio Code

Pre-release Document

This is a pre-release documentation and does not constitute final documentation relating to the product it is distributed with.

If this guide is distributed with software that includes an end user agreement, this guide, as well as the software described in it, is furnished under license and may be used or copied only in accordance with the terms of such license. Except as permitted by any such license, no part of this guide may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of Adobe Systems Incorporated. Please note that the content in this guide is protected under copyright law even if it is not distributed with software that includes an end user license agreement.

The content of this guide is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies that may appear in the informational content contained in this guide.

Please remember that existing artwork or images that you may want to include in your project may be protected under copyright law. The unauthorized incorporation of such material into your new work could be a violation of the rights of the copyright owner. Please be sure to obtain any permission required from the copyright owner.

Any references to company names in sample templates are for demonstration purposes only and are not intended to refer to any actual organization.

Adobe, the Adobe logo, Adobe Content Server, Adobe Digital Editions, and Adobe PDF are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. Java is a trademark or registered trademark of Sun Microsystems, Inc. in the United States and other countries. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries. Microsoft, Windows and Windows Server are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Macintosh and MacOS are trademarks of Apple Inc., registered in the U.S. and other countries. All other trademarks are the property of their respective owners.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

Notice to U.S. Government End Users. The Software and Documentation are "Commercial Items," as that term is defined at 48 C.F.R. §2.101, consisting of "Commercial Computer Software" and "Commercial Computer Software Documentation," as such terms are used in 48 C.F.R. §12.212 or 48 C.F.R. §227.7202, as applicable. Consistent with 48

C.F.R. §12.212 or 48 C.F.R. §§227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to

all other end users pursuant to the terms and conditions herein. Unpublished rights reserved under the copyright laws of the United States.

For U.S. Government End Users, Adobe agrees to comply with all applicable equal opportunity laws including, if appropriate, the provisions of Executive Order 11246, as amended, Section 402 of the Vietnam Era Veterans Readjustment Assistance Act of 1974 (38 USC 4212), and Section 503 of the Rehabilitation Act of 1973, as amended, and the regulations at 41 CFR Parts 60-1 through 60-60, 60-250, and 60-741. The affirmative action clause and regulations contained in the preceding sentence shall be incorporated by reference.

Introduction	5
Why do I need this extension?	5
Code assist	5
Security code analyzer	5
Debugger.....	5
Workspace viewer- Project Manager	5
RDS support and file viewer	6
Code Profiler	6
Service browser	6
Code refactoring	6
Getting started.....	6
System requirements.....	6
Install the extension.....	6
Set JAVA_HOME.....	6
Windows	6
macOS	7
Add a ColdFusion server	8
Project Manager	10
Create a workspace	10
Add a ColdFusion project.....	11
Create a CFM file.....	11
Import project from ColdFusion Builder.....	11
Profile Preferences	12
Create a profile	12
Editor preferences	13
Code Assist preferences.....	13
Formatter preferences.....	13
Outline preferences	14
Syntax Checking preferences.....	14
Set file icon theme	14
Work with ColdFusion code.....	15
Code assist	16
Using Code Assist	16

Code Assist for CFM pages.....	16
Code navigation	17
Jump to next member	17
Jump to prev member	17
Jump to next attribute	17
Jump to previous attribute	18
Jump to end of tag/block.....	18
Select tag/block	18
Quick fix	19
Go to definition	20
Syntax check	20
Debug applications	21
Pre-requisites.....	21
Set and remove breakpoints.....	21
Set a Breakpoint.....	22
Remove a breakpoint.....	22
Disable a breakpoint	22
Run code line by line.....	22
Inspect Variables.....	23
Watch expressions.....	23
Run and debug in VS Code.....	24
RDS support	24
RDS Dataview	24
RDS Query Viewer	24
Security Code Analyzer	25
Accessing security analyzer in the extension	25
Using the Security Analyzer	26
Export Security Analyzer results	27
Services Browser.....	28
Services Browser filter	31
PMT Code Profiler Report Integration.....	31
Refactoring.....	32
Refactor CFC or CFM filenames	32

Reference search	34
Refactor function or variable	34
Known issues in this release	36

Introduction

Coding can feel tedious when dealing with complex codes, language overload, syntax errors, and so on.

The all-new Adobe ColdFusion Builder extension for VS Code helps developers to edit and validate code, manage files and projects, and debug and scan for security vulnerabilities.

Download now and get easy access to data sources on your server with tools that let you run queries without installing an external client.

Why do I need this extension?

Integrate the Adobe ColdFusion Builder Extension on your VS Code to:

- Automate repetitive tasks and navigate code for a smoother and faster process.
- Enjoy built-in support for IntelliSense code completion, better semantic code understanding, and code refactoring.
- Identify security vulnerabilities and maintain the integrity of your code.
- Manage your work with extensions, remote project support, integrated server management, a log viewer, and more!
- Customize every feature to your liking by creating shortcuts, easily formatting and reusing code, and using powerful extensions to better your best.

Code assist

Accelerate your application development with intelligent code assist for CFML, CFScript, HTML, JavaScript, and CSS.

Security code analyzer

Use the security code analyzer to scan existing application code to automatically detect vulnerabilities and potential security breaches. Identify the exact vulnerable code, type of vulnerability and severity level and mitigate the vulnerability with the suggestion provided.

Debugger

Reduce testing time and hard-to-fix bugs with integrated debugging to control the execution of the code and observe the proceedings. Launch and step through applications directly within Visual Studio for analysis of the code.

Workspace viewer- Project Manager

Projects contain resources such as ColdFusion components, interfaces, and CFML pages that you can use to develop ColdFusion applications.

You must create a workspace before creating a project if there is no workspace opened. If the workspace is already created and opened, the project will be tagged under that.

The workspace stores your projects and other metadata. The preferred workspace location is the ColdFusion document root.

RDS support and file viewer

Manage access to files and databases on a server hosting ColdFusion. The RDS File View displays the files and directories on both remote and local servers. The RDS Data View displays the data sources configured in a remote server.

When you add a ColdFusion server instance in ColdFusion Builder, it automatically becomes available in RDS File View and RDS Data View.

Code Profiler

Identify performance bottlenecks, memory issues, and so on, with the help of Performance Monitoring Toolset, and import the data in Visual Studio.

Service browser

Quickly access web services exposed by the ColdFusion server from Visual Studio.

Code refactoring

Code refactoring is the process of improving the source code of a program without changing the overall result. Generally, code refactoring improves code readability and maintainability.

The extension supports various refactoring techniques like renaming, searching, and previewing of CFCs, CFMs, and UDFs at the project and workspace levels.

Getting started

System requirements

- Windows: Windows 10, Windows 11.
- macOS: macOS 11, macOS 12.
- Java 11 as the recommended JDK. For download and install instructions, see [Download Java](#).
- Version 1.68 of VS Code or above. Download VS Code from [Microsoft Visual Studio Code](#).

Install the extension

1. On the Extensions search bar, search for the Adobe ColdFusion Builder extension for Visual Studio Code.

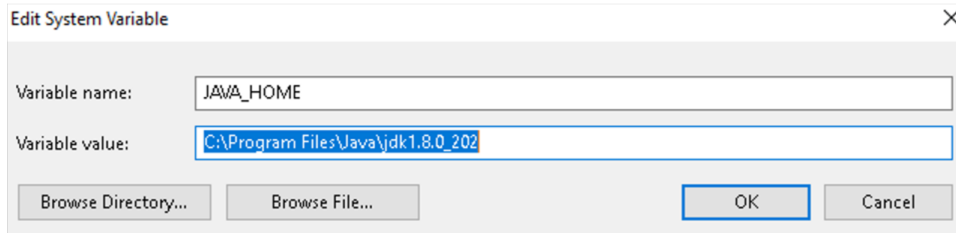
Alternatively, from the command palette, type CTRL+SHIFT+P (Windows) or CMD+SHIFT+P (macOS), search Install Extension, click the option, and then install the extension.

2. On the extension details page, click Install.

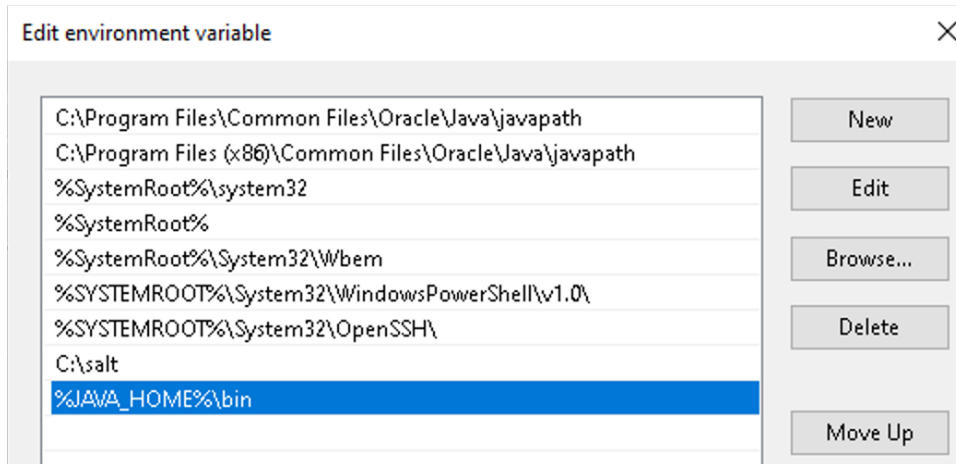
Set JAVA_HOME

Windows

1. Download and install Java.
2. Set JAVA_HOME to appropriate location. Click OK.



3. Add %JAVA_HOME%\bin to path.



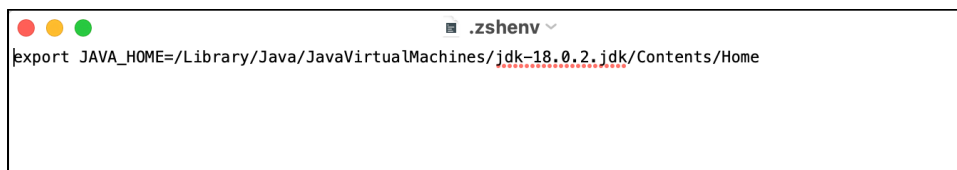
4. Restart VS Code.
5. Verify Java version in VS Code terminal - java -version.

macOS

Note: macOS 10.15 Catalina and above uses zsh shell. All older macs before macOS 10.15 use bash shell. Type `echo $SHELL` to find the shell.

zsh shell

1. Download and install [Java](#). Ensure that you download the correct version, which depends on whether your system supports M1 architecture.
2. Create zsh environment file, if not already created - `touch ~/.zshenv`
3. Locate the Java binaries - `/usr/libexec/java_home`. The following path appears, `/Library/Java/JavaVirtualMachines/jdk-18.0.2.jdk/Contents/Home`
4. Copy and paste the Java binary path to the zshenv file. Type `open ~/.zshenv`.
5. In the environment editor, type `export`
`JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk-18.0.2.jdk/Contents/Home`



6. Save and exit the editor.
7. Apply the change into the current shell- source ~/.zshenv
8. Check if the environment variable is set properly- echo \$JAVA_HOME

bash shell

1. Create bash profile- touch ~/.bash_profile
2. Open the bash script editor – open ~/.bash_profile
3. Add JAVA_HOME- export JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk-18.0.2.jdk/Contents/Home
4. Save and exit.
5. Apply the change into the current shell- source ~/.bash_profile
6. Check if the environment variable is set properly- echo \$JAVA_HOME

Add a ColdFusion server

To add a ColdFusion server, follow the steps below:

1. In the CF Servers pod, click + (Add Server).
2. Enter the following details. Some are mandatory.

General Settings tab	
Name	Description
Server Name (mandatory)	ColdFusion server name.
Host Name (mandatory)	Name of the ColdFusion server host. For example, localhost or 127.0.0.1.
Description	Description of the server.
Webserver Port (mandatory)	Specify the port number of the ColdFusion server instance you are configuring. The default port number of the ColdFusion server is 8500.
Application Server (mandatory)	Select the server on which ColdFusion is deployed.
RDS User Name	If you are using RDS, specify the RDS username.
RDS Password	Enter the RDS password. Click Test Connection. If you'd configured RDS as expected, you will see RDS Fileview and Dataview pods with the file paths and the built-in databases respectively.
Type	Select the appropriate option if the ColdFusion server is hosted on your local machine or on a remote machine.
Enable SSL	Select this option to enable SSL support in the extension. Servers registered in the Server Manager can communicate using SSL.
Context Root	Enter the context root. The JEE environment supports multiple, isolated web applications running in a server instance. Hence, JEE web applications running in a server are each rooted at a unique base URL, called a context root (or context path).
Application Server Name	Name of the JEE Server on which ColdFusion is deployed.

Local Server Settings tab	
Name	Description

Server Home (mandatory)	Browse and select the ColdFusion Server home directory. For example, <ColdFusion Home>/cfusion.
Document Root (mandatory)	Browse and select the web root location. If ColdFusion is configured with a web server, say IIS, then select the document root of the web server; for example, ColdFusion webroot.
Version	Select the ColdFusion server version from the Version drop-down list.
Windows Service	Applicable to Server configuration deployments running on Windows. The Windows Service option is available only for Server configuration deployments, and not JEE configuration deployments of ColdFusion. If you want to start and stop the ColdFusion server using the Windows Service, select Use Windows Service to Start/Stop Server.

URL prefix tab	
<p>A URL prefix maps a local file system resource with a URL. You can specify a URL prefix while creating a server, or by editing settings for an existing server. You can also specify a URL prefix to an existing project or folder. For example, there is a project called "Project1". Project1 is configured to server1, whose document root is at C:\server1\MyDocs and URL is http://www.example1.com. Within Project1, you have a linked folder called "xyz". The folder xyz points to the document root of server2, which is C:\server2\MyDocs and the URL to access it is http://www.example2.com. You want to preview all the files within the xyz linked folder in ColdFusion Builder. In this scenario, to preview files within the xyz linked folder, you create a URL prefix. You create a URL prefix by specifying the following details:</p> <ul style="list-style-type: none"> • Absolute path: C:\server2\MyDocs\ • URL to access the xyz folder: http://www.example2.com 	
Local path (mandatory)	Browse to or enter the path to the local file system resource.
URL prefix (mandatory)	Enter the URL prefix.

Virtual Host Settings tab	
<p>Virtual Host refers to the method of hosting multiple websites (domain names) on a single web server. The multiple websites are differentiated by their apparent host names. For example, you can run websites www.example1.com, www.example2.com, and www.example3.com, on a single IP address.</p>	
Name (mandatory)	Specify a name for the virtual host.
Host Name (mandatory)	The virtual host name that is mentioned in your IIS or Apache web server settings. For example, www.example1.com. When you create a virtual host in ColdFusion Builder extension, the virtual host uses a naming convention server name-virtual host name. For example, if you created a virtual host named "vh1" in your ColdFusion server (localhost), the naming convention that ColdFusion Builder extension uses to identify the virtual host is "localhost-vh1".
Port (mandatory)	The port assigned to the virtual host in the web server.
Type (mandatory)	Select HTTP or HTTPS from the drop-down list.

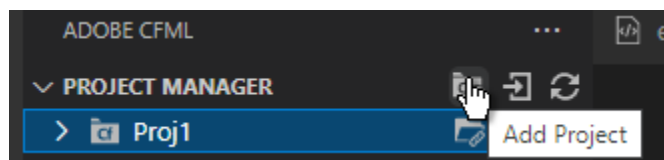
Document Root (mandatory)	Browse to or enter the home directory of the virtual host. For example, if your website www.example.com is mapped to the directory C:\abc on the Apache web server, enter C:\abc as the home directory.
Virtual Directory settings	
Alias	Specify an alias for the folder path.
Location	Browse to or enter the folder path to which you want to specify an alias. For example, if the document root of your website (www.example.com) is c:\abc, and you want to include images from a folder available on d:\xyz\images. Then, you can define an alias called "images" for the folder path "d:\xyz\images".

Project Manager

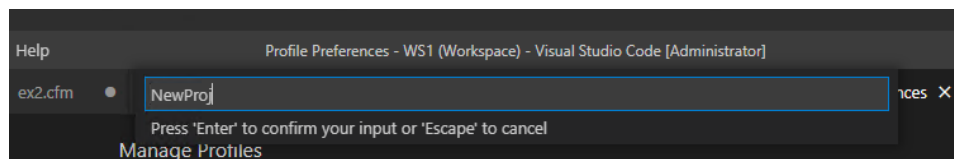
In the Project Manager section, you can create a workspace that will contain your project files. A workspace is like a file system. The workspace contains the resources (files and folders) that are a part of your application projects. A workspace may contain multiple projects. Each project is stored in a default workspace. The workspace stores your projects and other metadata.

Create a workspace

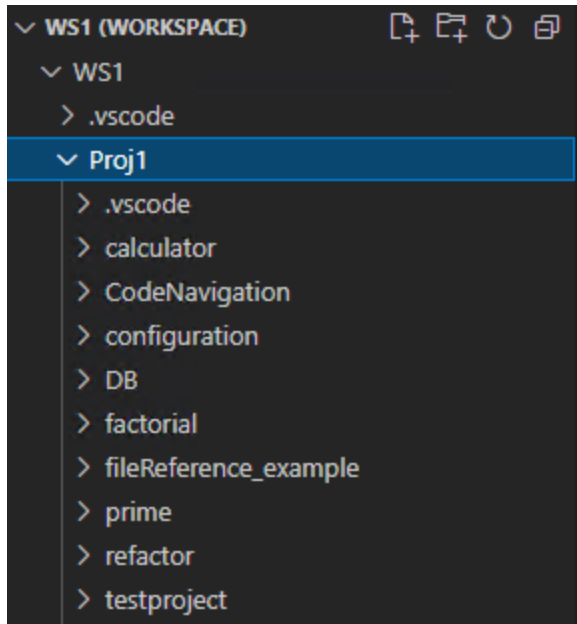
1. In Project Manager, click Add Project.



2. Add the name of the workspace.



3. Navigate to the webroot and create a folder. This is the new workspace.
4. The created workspace now appears in the explorer. Inside the workspace, create a project.



You can create a workspace inside/outside wwwroot and add a project inside or outside of that workspace.

If you want to run project files, the project must be inside the webroot.

Add a ColdFusion project

To add a ColdFusion project, follow the steps below:

1. In the Adobe CFML Extension Project Manager pod, click Add Project.
2. Enter the name of the project.
3. In the workspace, create a folder in which the project will reside.
4. Click Choose project folder to choose the project folder.
5. Choose the CF server, CFML dictionary and the linked folder. You may choose to skip the linked folder.

Create a CFM file

To create a CFM file, follow the steps below:

1. In the Project Manager pod, right-click the project folder, and click Create ColdFusion Page.
2. Name the file. You need not specify any extension. The file extension defaults to .cfm. Write the code.
3. To run the code, right-click anywhere on the code, and click Run as ColdFusion Application.
4. The default browser launches with the results of the code.

Import project from ColdFusion Builder

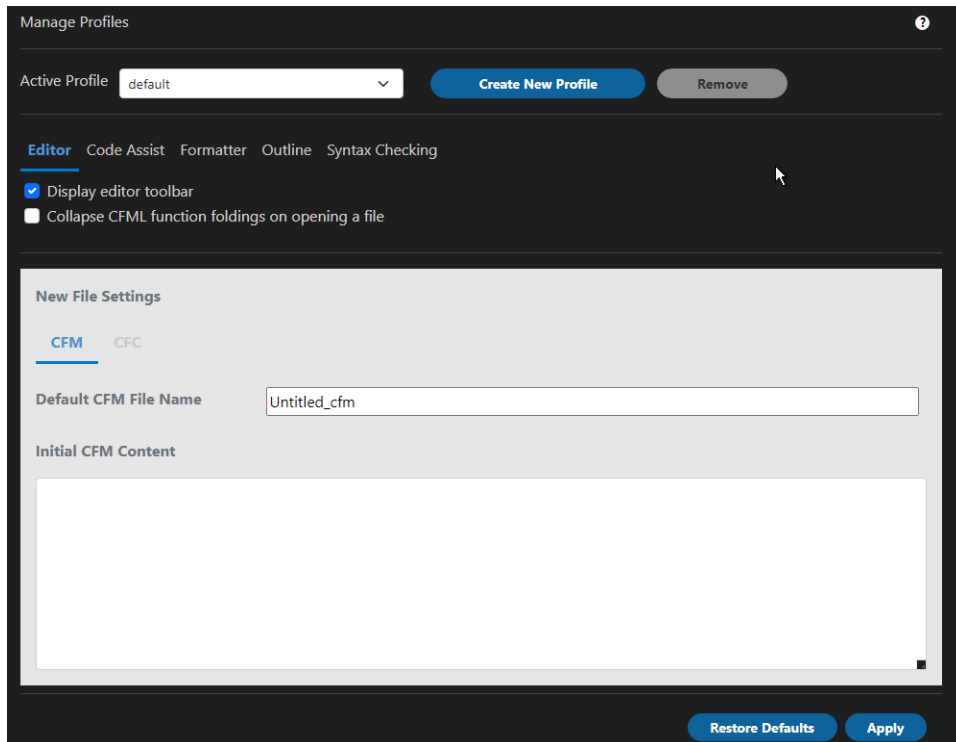
1. In the Project Manager, click Import Project.
2. Enter the name of the project.
3. Browse to the location of the ColdFusion Builder workspace.

4. Select the project. Click Import.
5. Project will be imported and will be listed in Project Manager.

Profile Preferences

Set and manage your preferences for various features in the extension using the Profile Preferences options.

To launch Profile Preferences, hit Ctrl+Shift+P to launch the command palette. Search for Profile Preferences and hit Enter.



An editor profile lets you group and save the following editor preferences under one profile.

Create a profile

Editor profiles are useful when you have different editor preferences for various development needs. For example, you can select a set of preferences like Code Assist, code colorization, and keyboard shortcut preferences and save them under a single profile.

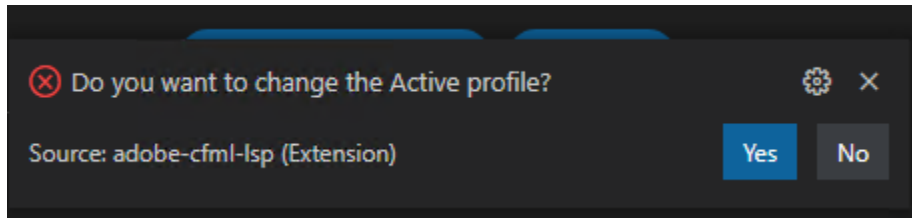
By default, ColdFusion Builder provides three active profiles:

- Default – Sets the extension’s default editor preferences.
- Dreamweaver – Sets editor preferences like Dreamweaver editor preferences.
- CFEclipse – Sets editor preferences like CFEclipse editor preferences.

To create a profile,

1. Click the button Click New Profile.
2. Enter the name of the profile.

3. When you select the profile from the Active Profile drop-down list, you have an option to set the new profile as the current active profile.



4. Make the required profile changes and click Apply to save the changes.

Editor preferences

- Display editor toolbar: Display or hide the toolbar in the editor. The editor toolbar contains buttons that are user interface shortcuts to frequently used commands.
- Collapse CFML function folding on opening a file: When enabled, all functions appear in a collapsed state when you open a CFM or CFC.

Code Assist preferences

- Code Assist Dictionary Version: Choose the required language dictionary while creating a project. Choose any version from the list.
- Filter Proposals Containing Text: Filter proposals based on the text you specify. All proposals that contain the filter text are listed with selection set to the proposal that starts with the specified text.

Formatter preferences

General:

- Specify if you want to maintain the case currently used for tags and attributes or change it to upper or lowercase.
- Append /> at the end of the tag, for example, modify <cfargument> as <cfargument/>. Specify the tag to which you want to append /> and then click Add.
- Place the closing tag for cfoutput (</cfoutput>) on a new line only if the content spans across multiple lines.

Indentation: Specify the indentation details and the name of the tags for which you do not want to apply indentation.

Whitespace:

- Add white space based on your selections.
- Add blank lines based on your selections.
- Specify the number of blank lines that you want to retain.

Wrapping:

- Specify the number of attributes in a line within a tag and set the column width.
- Choose the various constructs for wrapping.

Braces: Specify if the curly braces are placed in the same or new line for component and function declarations and switch, if, else, and try/catch blocks.

Outline preferences

The Outline view displays a hierarchy of elements in the file that is currently open in the editor. For example, it displays all tags or some chosen tags in the view.

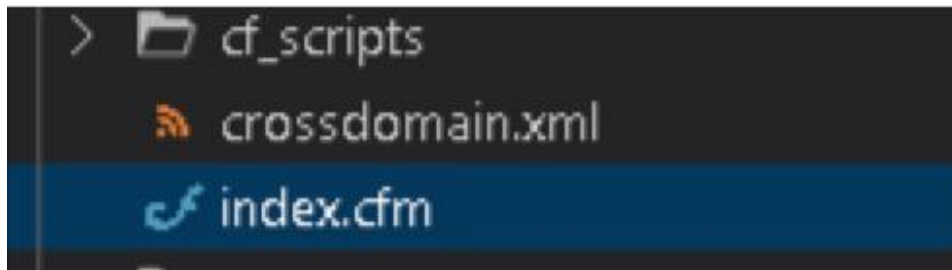
You use Outline view to inspect and navigate the structure of your CFML file and see a hierarchical view of the code structure of the page.

Syntax Checking preferences

- **Enable Syntax Checking:** Syntax checking is enabled by default. When this option is enabled, the extension displays all or any syntax errors while writing the code.
- **Enable Quick Fix:** Quick Fix recognizes the usage of methods, classes, and CFC/CFM files in the code and helps generate them. For example, if you type a UDF `myFunc()` that is not defined in the page or any other page, Quick Fix generates the function. The function call is inserted in the file.
- **Display Syntax Errors Only on File Save:** If enabled, you can only see the syntax errors after you save the file, not while writing the code.

Set file icon theme

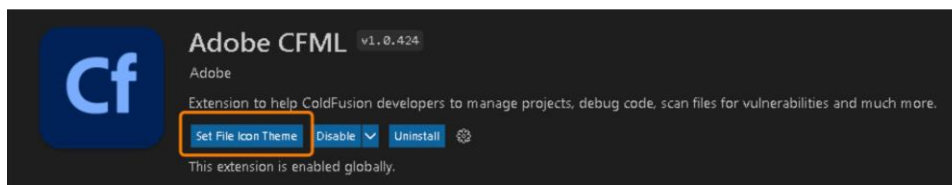
When you add a ColdFusion file, for example, `cfm`, in a project, the file icon is set to the default VS Code theme, which is called Seti (Visual Studio Code).



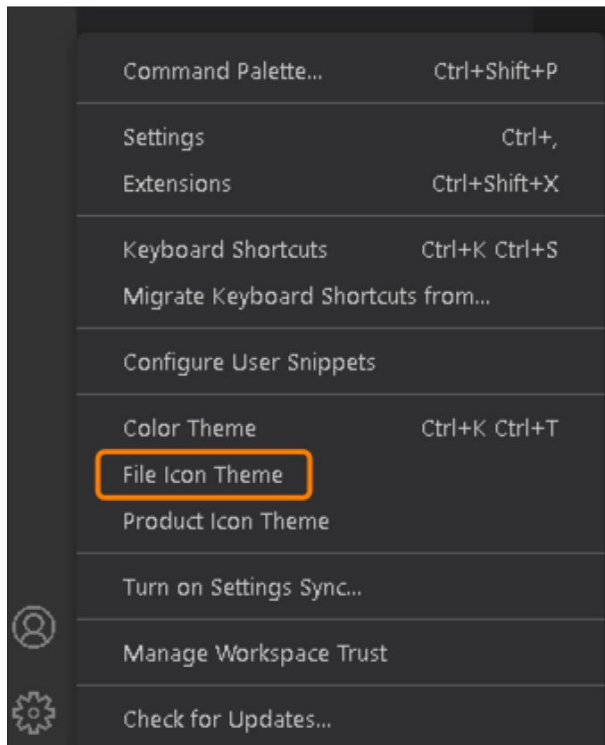
After you install the extension, you can change the icon theme to the one from the extension.

There are two ways to set the theme. Select either:

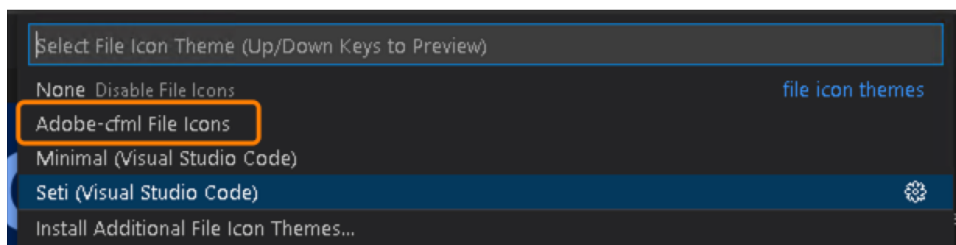
- a. On the Adobe CFML extension page, click Set File Icon Theme.



- b. Click Manage > File Icon Theme.



Then select Adobe CFML file icons from the list.



You can set any file icon from the available list, and this is not overridden.

Work with ColdFusion code

Adobe ColdFusion Builder extension for Visual Studio Code provides a CFML Editor that has feature-rich code editing capabilities for CFM, CFC, HTML, JavaScript, and CSS files. The CFML Editor assists you in writing code by including features like, code completion, and streamlined code navigation. The CFML Editor lets you use different colors and fonts to display your code in the workspace.

The other features available in the extension include:

- Code assist
- Dictionary support
- Code refactoring and formatting
- Code navigation and outline
- Code folding and unfolding
- Code coloring

- Context sensitive help

Code assist

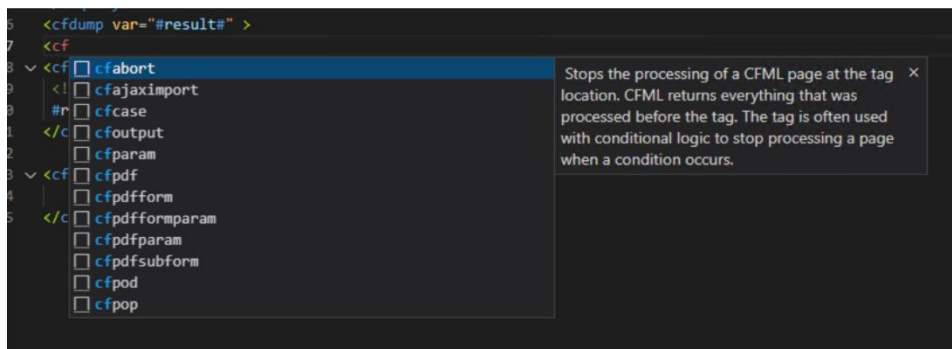
Code Assist is designed to assist you with code completion. Depending on the code that you enter, hints relevant to complete the code appear. As you type the code in the CFML editor, Code Assist prompts you with a list of valid CFML tags, parameters, and attributes. These suggestions appear in a pop-up menu. If you have HTML, JavaScript, or CSS content within the CFML code, Code Assist displays code completion hints for this code as well. Double-click or press Enter, to insert the code completion hint in the CFML Editor.

Using Code Assist

Code hints appear whenever the framework or language (CFML, HTML, JavaScript, and CSS) provides options for you to complete the current expression.

For example, if you type within a CFML tag, you are prompted with a list of all the attributes of that tag.

In a CFML page, begin entering a CFML tag by typing:<cf Relevant code hints are displayed as follows:



Code Assist for CFM pages

- Displays a drop-down list of predefined attributes and values when you press <cf>+Ctrl+<space>.
- Displays a list of components (CFC) that can be loaded using createobject() or the <cfobject> tag.
- Displays methods created in a CFC, which can be called using the component object created in a CFM. Also displays a list of methods of all extended CFC files.
- Displays a drop-down list of all built-in and user-defined functions.
- Provides a list of variables, like, struct, array, query. These variables are declared in a page. The variables also appear as Code Assist for attribute values.
- Displays all the queries created using <cfquery> or queryNew(). To view the recordset, type <cfoutput query="">, and press Enter. or use queryname . (that is, query name followed by dot).
- Includes the functions, variables, tags, and queries from another CFM page once it is included in the current CFM page using the {{<cfinclude> }}tag.

Code navigation

Navigate seamlessly within your code. If you have some functions or properties in a CFC, you can navigate through the function, property, if/else block, or even a try/catch block.

Ctrl+Shift+P launches the command palette, and you can search for the required navigation shortcut.

Jump to next member

If you press Ctrl+Shift+E, you can navigate to the next function definition.

```
<cffunction name="getFullName" output="false" access="public" returnType="string">
  <cfargument name="firstName" type="string" required="false" />
  <cfargument name="lastName" type="string" required="false" />
  <cfset var fullName = arguments.firstName & " " & arguments.lastname />
  <cfreturn fullName />
</cffunction>
```

After pressing Ctrl+Shift+E, the next function definition gets highlighted.

```
<cffunction name="getFullName" output="false" access="public" returnType="string">
  <cfargument name="firstName" type="string" required="false" />
  <cfargument name="lastName" type="string" required="false" />
  <cfset var fullName = arguments.firstName & " " & arguments.lastname />
  <cfreturn fullName />
</cffunction>

<cffunction name="getHandlerJSON"
  access="remote"
  httpmethod="GET"
  restpath="{customerID}"
  returnType="query"
  produces="application/json">
```

Jump to prev member

If you press Ctrl+Shift+Q, you can navigate to the previous function definition.

```
<cffunction name="getFullName" output="false" access="public" returnType="string">
  <cfargument name="firstName" type="string" required="false" />
  <cfargument name="lastName" type="string" required="false" />
  <cfset var fullName = arguments.firstName & " " & arguments.lastname />
  <cfreturn fullName />
</cffunction>

<cffunction name="getHandlerJSON"
  access="remote"
  httpmethod="GET"
  restpath="{customerID}"
  returnType="query"
  produces="application/json">
```

Jump to next attribute

Press Ctrl+] to move to the next attribute in the function. It also highlights the same attribute values.

```
<cffunction name="getFullName" output="false" access="public" returnType="string">
  <cfargument name="firstName" type="string" required="false" />
  <cfargument name="lastName" type="string" required="false" />
  <cfset var fullName = arguments.firstName & " " & arguments.lastname />
  <cfreturn fullName />
</cffunction>
```

Ctrl+] produces:

```
<cffunction name="getFullName" output="false" access="public" returnType="string">
  <cfargument name="firstName" type="string" required="false" />
  <cfargument name="lastName" type="string" required="false" />
  <cfset var fullName = arguments.firstName & " " & arguments.lastname />
  <cfreturn fullName />
</cffunction>
```

Jump to previous attribute

Press Ctrl+[to move to the previous attribute in the function.

```
<cffunction name="getFullName" output="false" access="public" returnType="string">
  <cfargument name="firstName" type="string" required="false" />
  <cfargument name="lastName" type="string" required="false" />
  <cfset var fullName = arguments.firstName & " " & arguments.lastname />
  <cfreturn fullName />
</cffunction>
```

Ctrl+[produces:

```
<cffunction name="getFullName" output="false" access="public" returnType="string">
  <cfargument name="firstName" type="string" required="false" />
  <cfargument name="lastName" type="string" required="false" />
  <cfset var fullName = arguments.firstName & " " & arguments.lastname />
  <cfreturn fullName />
</cffunction>
```

Jump to end of tag/block

Navigate across all functions using the shortcut Ctrl+Shift+E. Place your cursor on the function definition and then press the keyboard shortcut. The cursor moves to the next function definition.

Select tag/block

Select a code block for a tag or script by pressing Ctrl+Alt+F.

```

<cffunction name="getHandlerJSON"
    access="remote"
    httpmethod="GET"
    restpath="{customerID}"
    returntype="query"
    produces="application/json">

    <cfargument name="customerID"
        required="true"
        restargsource="Path"
        type="numeric"/>

    <cfset myQuery = queryNew("id,name",
        "Integer,varchar",
        [[1, "John"], [2, "Doe"]])>
    <cfquery dbtype="query"
        name="resultQuery">
        select * from myQuery where id = #arguments.customerID#
    </cfquery>
    <cfreturn resultQuery>
</cffunction>

```

Quick fix

Quick fix is about flagging mistakes in the code and providing suggestions. For example, if you are creating a CFC and if the CFC is not already present or you have specified an incorrect CFC, then the extension displays an error.

```

<cfscript>
    mycfc = new mycfc()
    mycfc = createObject("component", "mycfc1") // error due to mismatch of component
    mycfc.func()
    fullName = getFullName(firstName="John", lastName="Adams")
</cfscript>
<cfset fullName = getFullName(firstName="John", lastName="Adams") />

```

The Problems tab below the code lists the error messages.

A yellow bulb now appears near the line that has the problems along with the suggested fix.

```

<cfscript>
    mycfc = new mycfc()
    mycfc = createObject("component", "mycfc1")
    mycfc.func()
    fullName = getFullName(firstName="John", lastName="Adams")
</cfscript>
<cfset fullName = getFullName(firstName="John", lastName="Adams") />

```

Create new coldfusion component mycfc1

Click the suggested fix and in this case, the CFC mycfc1 gets created in the same folder as the CFM. The code also gets updated with the name of the new CFC.

Similarly, you can fix errors in any UDF locally or in an external CFC.

In all cases, the extension creates a CFC with pre-filled skeletal code.

```

<cfcomponent name = mycfc1>
    <cfscript>
        function func ()
        {
        }
    </cfscript>
</cfcomponent><cfcomponent name = "mycfc1">
</cfcomponent>

```

Quick Fix also helps you in the following scenarios:

- Call to local UDF: For example, cfset or cfscript assignment, function arguments, or any other expression. If you specify any arguments in the UDF, then Quick Fix also creates the arguments.
- Method call on a CFC: For example, if function1 is not defined in cfc1, then Quick Fix creates it.
- Create a CFC from createObject, new, cfobject, and cfinvoke.
- Create CFM page from cfinclude and cfmodule.
- Create CFCs and CFMs from extends and implement attributes.

Go to definition

You can go to a function or CFC directly by using Command+Click on Mac or Control+Click on Windows.

On a function call or a CFC call a CFC invocation, when you click Ctrl+Click, you go to the definition directly.

Whether it's a local UDF or an external CFC, Ctrl/Cmd+Click takes you to the definition of the function.

Syntax check

When you select a tag or code bracket in a CFM file, the matching pair of the tag or bracket is automatically highlighted.

If you type code that is not recognized as valid CFML code, you are notified accordingly.

```

WS1 > Proj1 > ex1.cfm > cfout
1 <cffunction name="add"
2 <cfargument name="
3 <cfargument name="x" type="numeric"
4 <cfreturn x+y>
5 </cffunction>
6
7 <cfoutput >
8 #add(5,3)#
9 </cfoutput>

```

Invalid Token at line 3, column 24. Encountered "\" (34), after: "" y

Missing token > or /> "

View Problem No quick fixes available

Click View Problem and you can view the error in the code.

```
WS1 > Proj1 > ex1.cfm > add(x) > cfargument
1  <cffunction name="add" returntype="numeric" >
2  |   <cfargument name="x" type="numeric" >
3  |   <cfargument name="y" type="numeric" >
4  |   <cfreturn x+y>
5  | </cffunction>
6
7  <cfoutput >
8  |   #add(5,3)#
9  </cfoutput>
```

Invalid Token at line 3, column 24. Encountered "\" (34), after: "" y

Use the previous/next arrows to view the other syntax errors.

Debug applications

Debugging lets you examine and troubleshoot your application. When you debug, you can control when the application must stop at specific points in the code. You can also monitor important variables and test your code. Debugging uses a configuration to control how applications are launched. When you debug your application, you run the debug version of the application file.

ColdFusion Builder extension for Visual Studio Code offers a powerful new developer tool in its debugger, an interactive step debugger that allows CFML developers to walk through the execution of their code, observing what lines of code it executes, what other CFML files (include files, custom tags, or CFCs) it executes, the value of all variables (optionally in all scopes) at a given point during execution, and much more.

Debugging lets you examine and troubleshoot your application. When you debug, you can control when the application must stop at specific points in the code. You can also monitor important variables and test your code. Debugging uses a configuration to control how applications are launched. When you debug your application, you run the debug version of the application file.

You can even change the value of variables on the fly within the debugger, such as to alter the runtime flow of execution of your code.

Pre-requisites

On the ColdFusion Administrator, enable the following:

- a. RDS
- b. Enable Request Debugging Output
- c. Allow Line Debugging

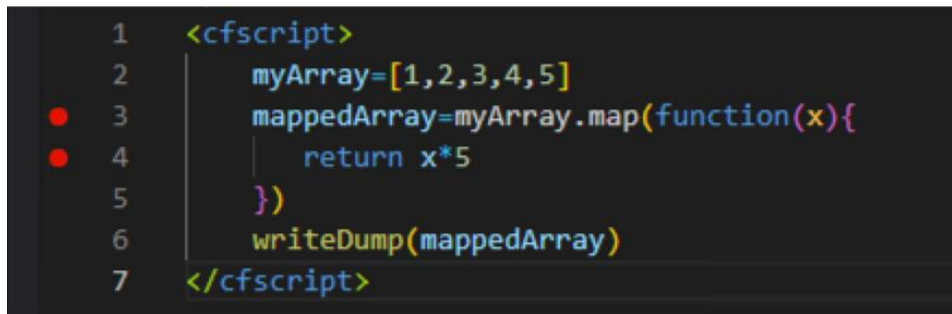
After making the changes, restart ColdFusion.

Set and remove breakpoints

You use breakpoints to control the running of your application so you can inspect your code and debug your application. You add breakpoints in the code editor and then manage them in the Breakpoints view. You can also set breakpoints as you write code or while you debug.

Set a Breakpoint

1. In the editor, locate the line of code where you want to set a breakpoint, and click in the marker bar along the left-edge of the editor.



2. Set the breakpoints in your CFML file to stop executing the page at points. When you set a breakpoint on a line, the CFML stops executing just before that line. For example, if you set a breakpoint on the third line in the following CFML page, execution stops before that line.

Remove a breakpoint

In the marker bar along the left-edge of the editor, click an existing breakpoint.

The breakpoint is removed from the marker bar and the Breakpoints view of the extension.

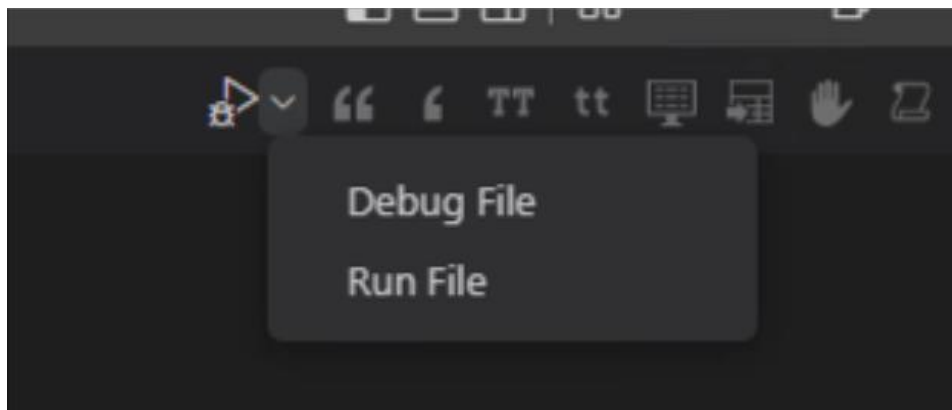
To remove all the breakpoints in the file, select Run > Remove all Breakpoints from the main toolbar menu.

Disable a breakpoint

To disable all the breakpoints in the file, select Run > Disable all Breakpoints from the main toolbar menu.

Run code line by line

1. Insert some breakpoints in the code.
2. In the editor, click Run or Debug and then click Debug File.



The Step menu appears

You can use the Step Into, Step Over, and Step Return buttons to proceed through your CFML code line by line.

STEP INTO

Use Step Into for UDFs, CFCs, custom tags, and included files. Avoid using Step Into on CFML tags such as the cfset tag. Step Into is more performance intensive than Step Over. When stepping into functions, tags, and files, the file must be displayed in one of the open projects. The file that you are stepping in must be in an open project.

STEP OVER

Use Step Over to proceed through your CFML application, bypassing included files, such as UDFs or CFCs.

STEP RETURN

Use Step Return to return to the original page from which you entered the included file, such as UDFs or CFCs.

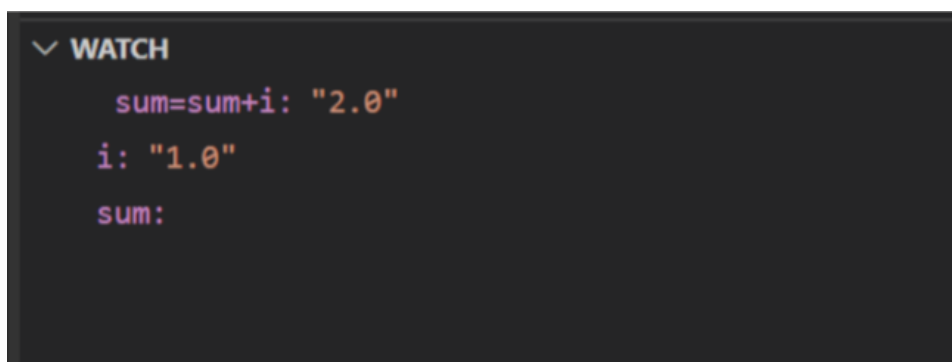
Inspect Variables

As you run CFML code, you can see the values and scope of variables in the Variables view.



Watch expressions

Watch expressions are useful to watch critical variables that sometimes go out of scope when you step into a different function. You can create your own expressions to watch and evaluate. You can modify the expressions during the debugging session.



You can do the following in the Expressions view:


- Create a watch expression by right-clicking and selecting Add Watch Expression. You can then enter the expression in the Add Watch Expression dialog box.

- Ignore a watch expression that you've added by right-clicking the expression and selecting Disable.
- Edit a watch expression by right-clicking the expression and selecting Edit Watch Expression. You can then modify the expression.

Run and debug in VS Code

If you have not configured running and debugging, then VS Code displays the Run start view. 

Run and Debug is a custom debug option provided by VS Code. If you click Create a JSON launch file and follow the prompts, you can associate your file with "CFML Debug" profile and then use it to run the debugger.

 CFML debug has overridden VS Code's default debug and run commands. So, debugger can be started without creating json file.

RDS support

RDS lets you access files and data sources registered in the ColdFusion Administrator on a ColdFusion server. To use RDS Support, you must enable RDS when you install ColdFusion. With RDS Support, you can access ColdFusion files remotely.

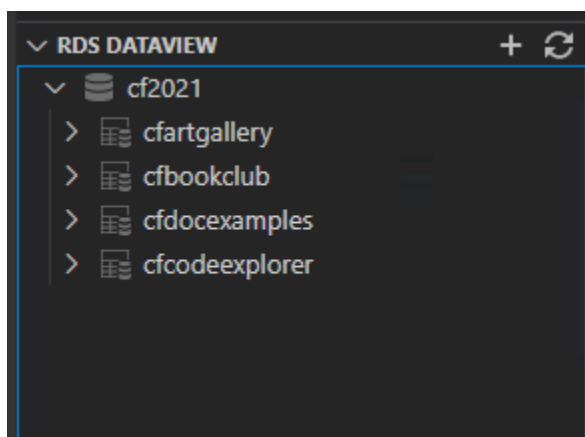
RDS allows developers to access key pieces of information about a remote server. RDS must be set up as part of your ColdFusion server install. We recommend that you do not use RDS on public facing or shared hosting machines.

RDS Dataview

To use RDS, enable RDS while installing the VS Code Plugin. The extension provides views to access files and data sources on a local and remote server.

The RDS Dataview displays the data sources configured in a remote server.

When you add a ColdFusion server instance in the extension, it automatically becomes available in RDS DataView.



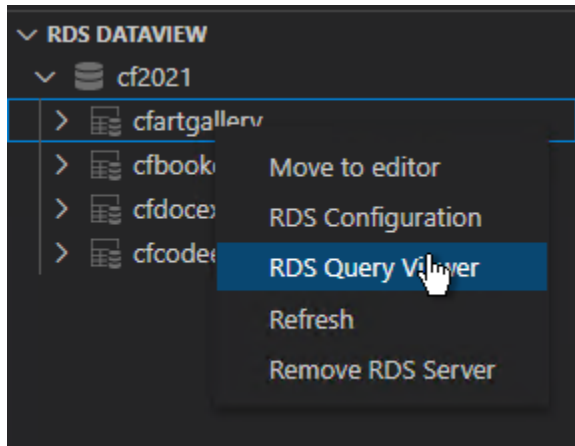
RDS Query Viewer

The RDS Query viewer lets you create and run queries on a selected data source.

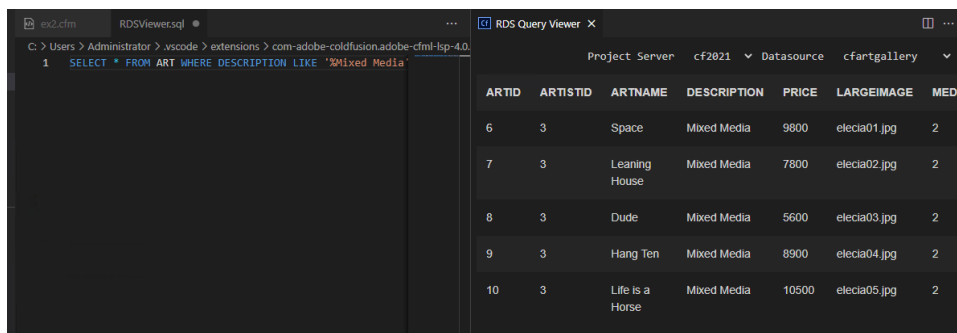
The RDS Query Viewer is available in the RDS Dataview toolbar.

To create and execute a query using the RDS Query viewer, do the following:

1. Right-click any datasource and click RDS Query Viewer.



2. In the RDSViewer.sql editor, build a query using any datasource and table.
3. Click the Execute Query button.
4. See the results in the Query Viewer.



Security Code Analyzer

For any web application, security plays a critical role. It is important to avoid security pitfalls while developing web applications.

Security Analyzer enables developers to avoid common security pitfalls and vulnerabilities while writing ColdFusion code.

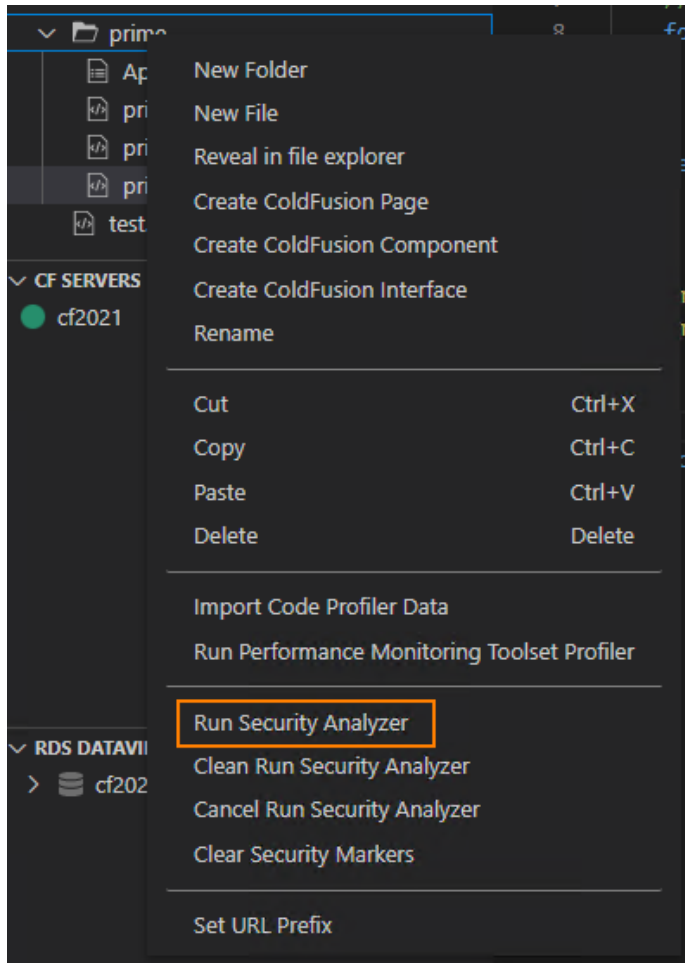
Use this feature to view:

- Vulnerable code in the editor
- Vulnerability or type of attack (Error and Warning)
- Severity level of vulnerability (High, Medium, and Low)
- Suggestion to avoid vulnerability.

Accessing security analyzer in the extension

Follow the steps below to access Security Analyzer in the extension:

1. Right-click the project folder or the project file in the Project Manager.
2. Click Run Security Analyzer.



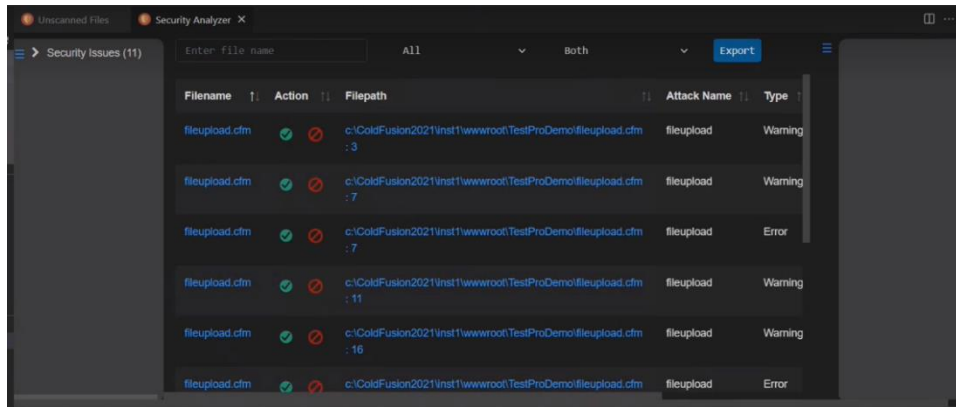
You have three options in Security Analyzer:

- **Run Security Analyzer:** Analyzes and displays vulnerabilities in the code.
- **Clean Run Security Analyzer:** Clears the history of all ignored messages and warnings. It clears the ignored vulnerabilities (which are marked as Ignore during the Run Security Analyzer) and displays all vulnerabilities for the project.
- **Cancel Run Security Analyzer:** Aborts the Security Analyzer.
- **Clear Security Markers:** Removes all security warnings and resources. Run the security analyzer again to view the vulnerabilities for your resource.

Using the Security Analyzer

Follow the steps below to use Security Analyzer for your project folder or file:

1. Create a ColdFusion project or use an existing project. Ensure that the project is configured to the preferred server.
2. Right-click the project folder or project file and click Run Security Analyzer. Security analyzer analyzes the code and displays a pop-up dialog when the task is completed.



You can view all the vulnerabilities in the bottom pane of the Editor as shown below.

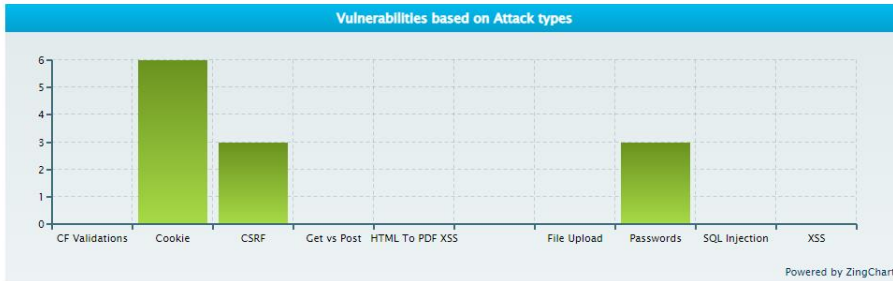
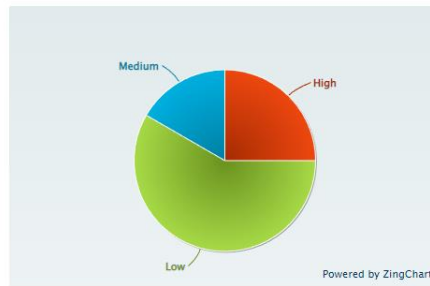
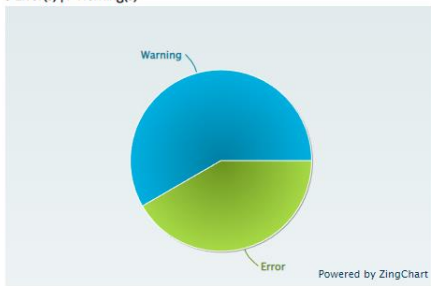
3. Click Security Issues on the left pane to view the list of vulnerabilities.
 - As shown in the left pane of the snapshot, click the vulnerability type (such as SQL Injection or XSS attack) to view the corresponding problem statement. You can also view the suggested solution in the right pane.
 - Alternatively, you can click any error on the middle pane to view the corresponding statement and solution at the right pane.
 - Double-click each error on the middle pane to view the corresponding line in the Editor.
 - Use filters for File Name, Attack Name, Severity Level, and Type in the middle pane. Start typing the file name in the search area to locate the files with vulnerabilities. You can narrow down your search based on severity level as high, medium and low by clicking All drop-down list.
4. After you fix the error in the code, right-click the corresponding error on the middle pane and choose the status as Fixed. Mark the status as Ignore if you ignore the error.

Export Security Analyzer results

Click Export on the upper-right corner of the Security Analyzer pane to export all the vulnerabilities to a report.html file.

You can view the graphical representation of all vulnerabilities for your resource in the exported file, as shown below:

5 Error(s) | 7 Warning(s)



Services Browser

The Services Browser lets you view all the component files that are associated with your projects. The Service Browser lists all the component files that are present inside those ColdFusion servers.

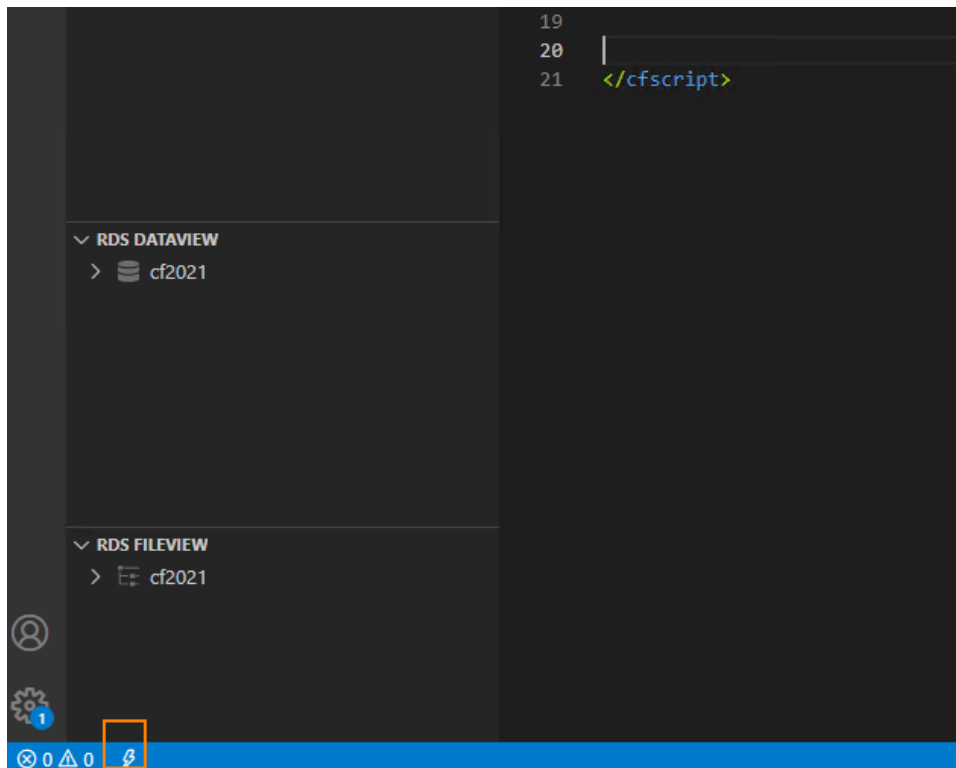
The Service Browser lists the following components:

- Components that the ColdFusion component browser listsThe ColdFusion component browser is located at `cf_root/wwwroot/CFIDE/componentutils/componentdoc.cfm`.
- Components that are in any directories specified in the ColdFusion Administrator Mappings page
- Components that are in any directories specified in the ColdFusion Administrator Custom Tag paths page

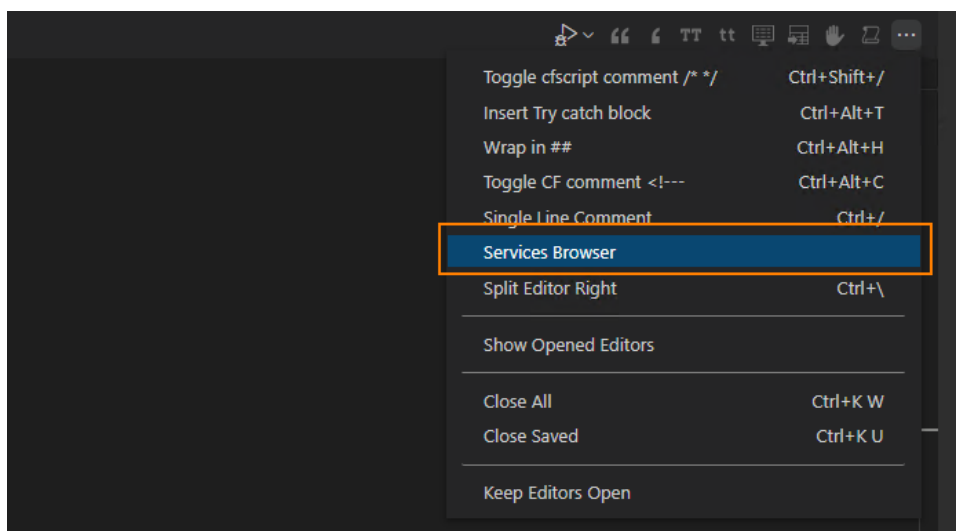
You can restrict the list of CFCs according to whether the functions in a CFC are remote, public, or private.

To launch the Services Browser, perform either of the following:

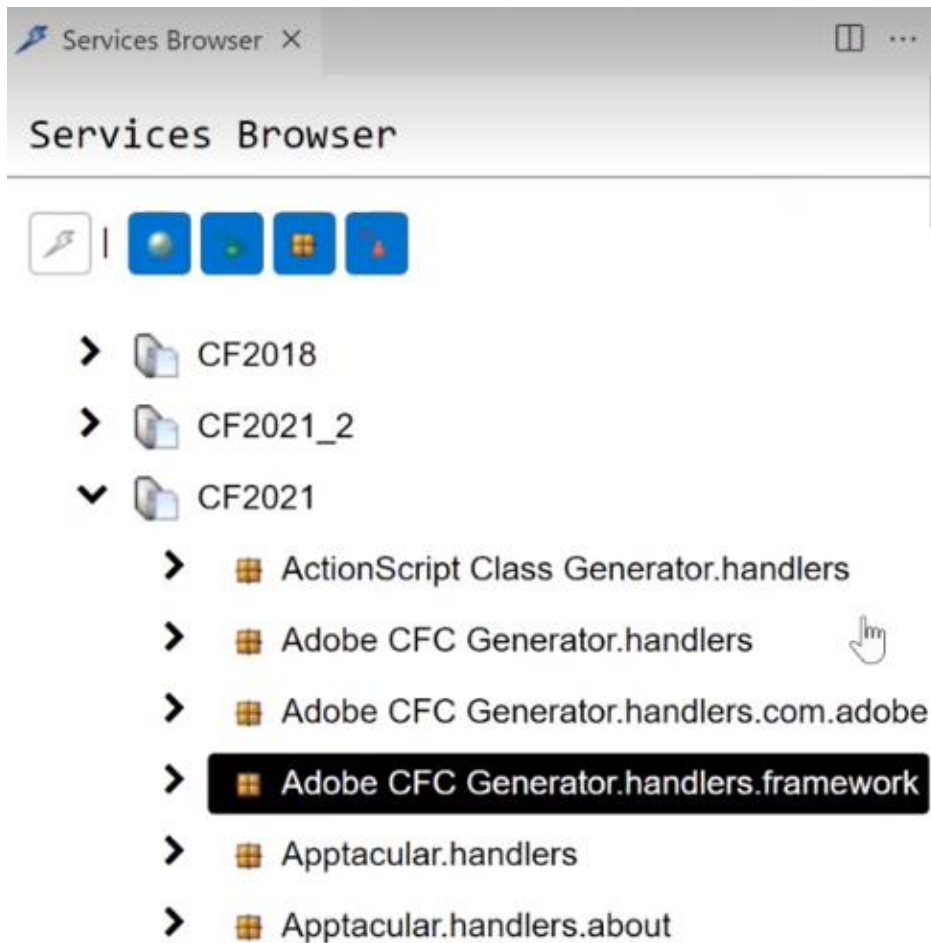
- Click the lightning icon on the lower left corner of the VS Code editor. The Services Browser tab launches on the right side of the editor.



- Click the options button on the right side of the editor and choose

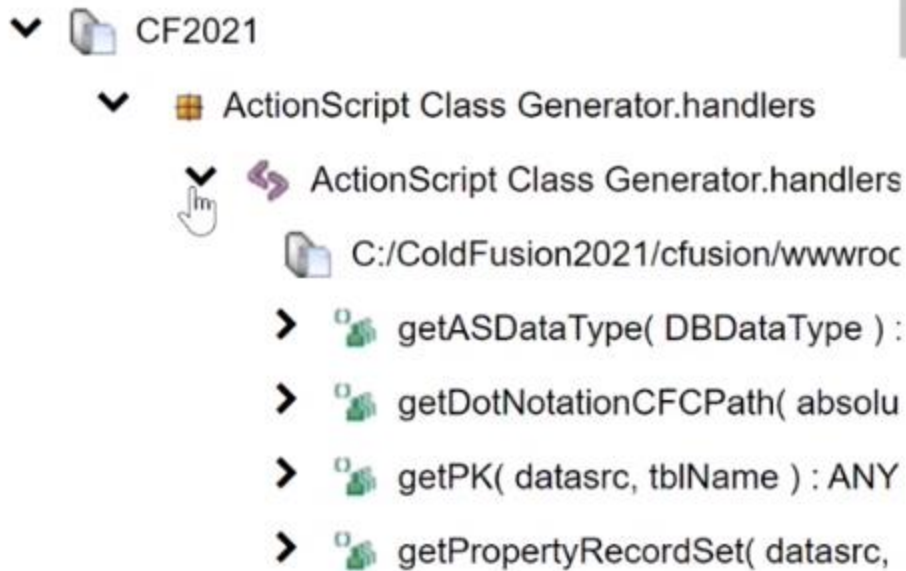


The Services Browser's primary role is to list all the CFCs defined for a server. It creates a package base listing of each CFC. You can expand a package to see individual CFCs, and then expand a CFC to see the physical file path.



If you click any class, you can see the handlers and the list of functions of all scopes, private, public, or remote.

The Services Browser presents a high-level view of the component file and the functions that are present in the component. You can also see the data type, arguments, access type, and the return type of that function.



Services Browser filter

On top of the Services Browser, there are the following filters that filter the CFC files based on their access types.

- Show Remote CFCs
- Show Public
- Show Package
- Show Private

PMT Code Profiler Report Integration

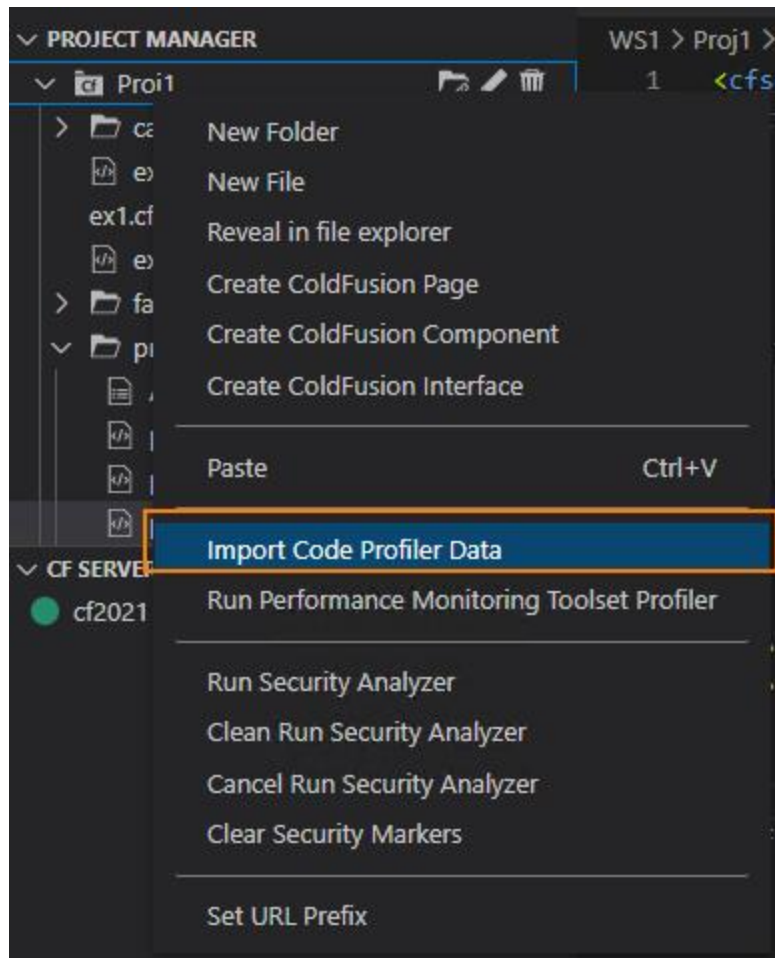
The Performance Monitoring Toolset monitors all transactions on a ColdFusion server and captures the response times along with other basic details of a transaction.

The procedure requires recording metrics for all components of a transaction, and thus you have the Code Profiler.

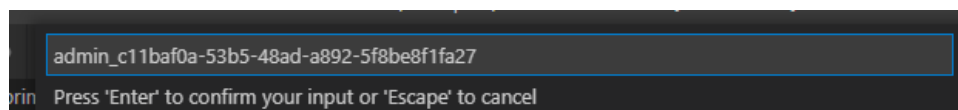
The first step is to run the Code Profiler in Performance Monitoring Toolset and then download the profiler data as a json file. For more information, see [Code Profiler in Performance Monitoring Toolset](#).

The next step is to import the profiler data in the VS Code extension. To do so, follow the steps below:

1. In VS Code, right-click the project and select Import Code Profiler Data.



2. Choose the json file and hit Enter.



3. After you import the profiler data, right-click the project and click Run Performance Monitoring Toolset Profiler.
4. In the profiler tab, expand each item and see the time taken by each function/UDF in a CFM or CFC. To view the line in the file, double-click any item.

Refactoring

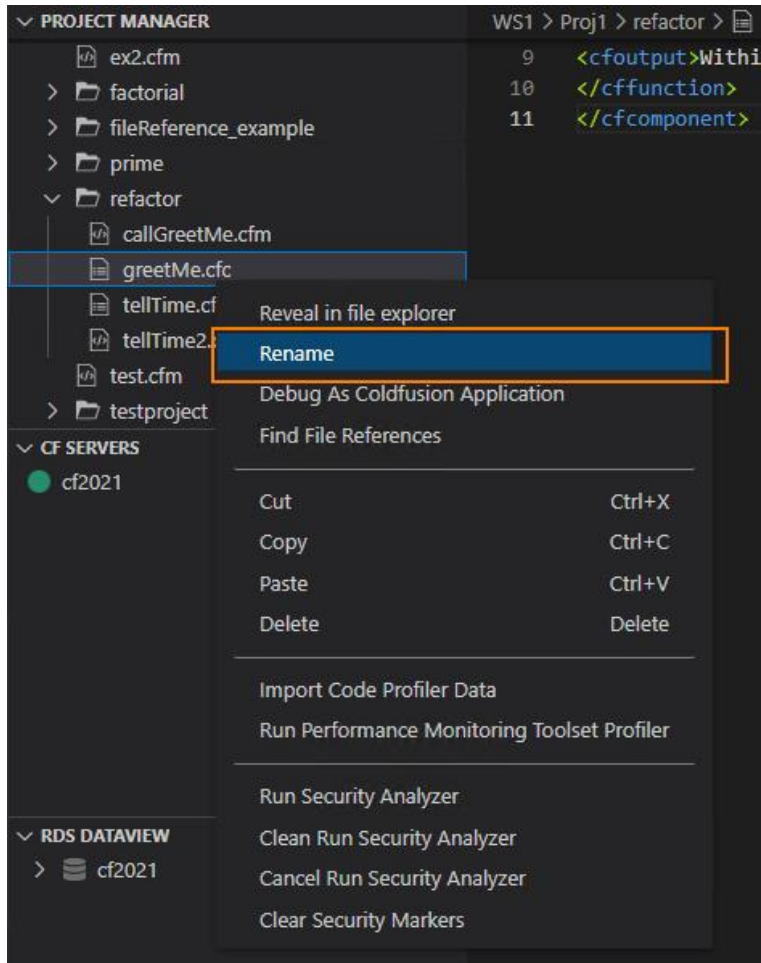
Code refactoring is the process of improving the source code of a program without changing the overall result. Generally, code refactoring improves code readability and maintainability.

The VS Code extension supports various refactoring techniques like renaming, searching, and previewing of CFCs, CFMs, and UDFs at the project and workspace levels.

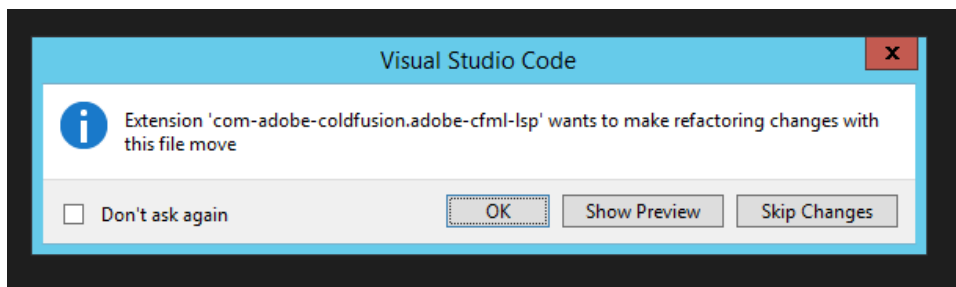
Refactor CFC or CFM filenames

When you rename a CFC or CFM, you can refactor all valid instances of the CFM or CFC, including all references to the CFM or CFC.

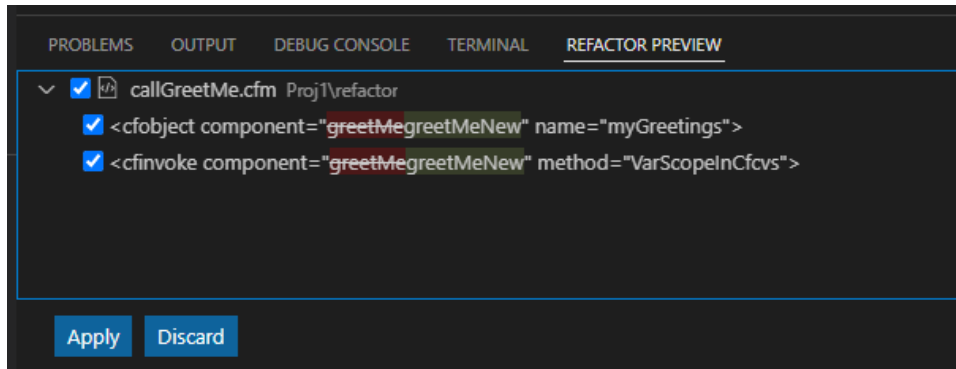
1. Right-click the file in the Project Manager.
2. Select Rename.



3. Enter the new name and preview the changes. Click Show Preview.



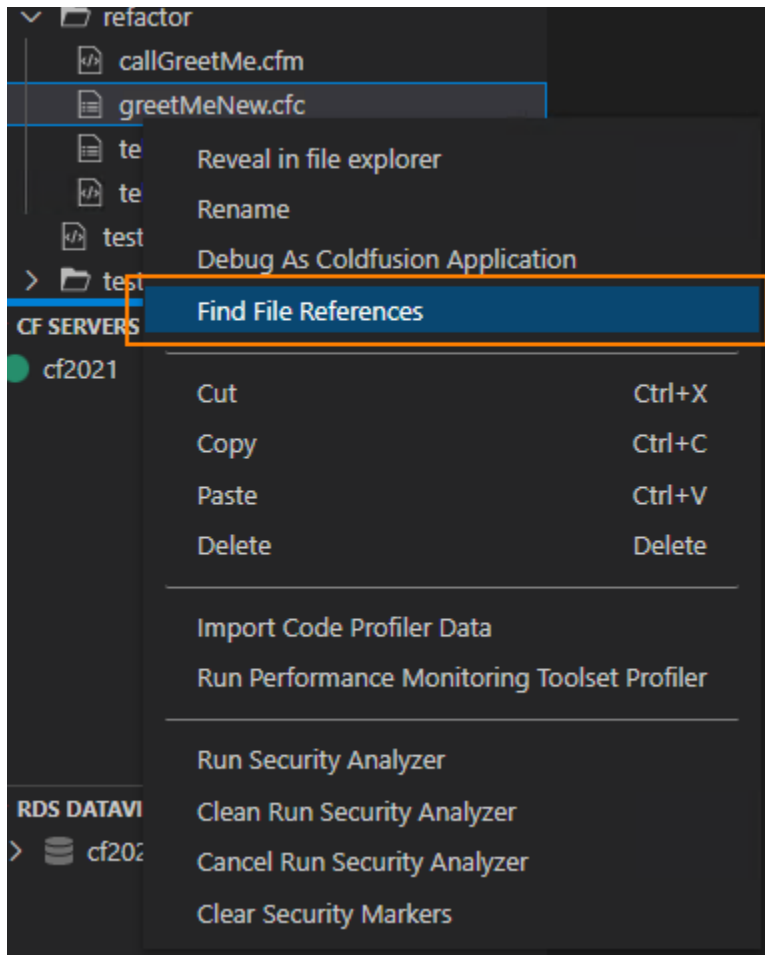
4. When you preview the changes, you can review the selected instances and de-select any instance that you do not want to rename.



Reference search

Check where the refactored UDF or CFC or CFM has been used in the project.

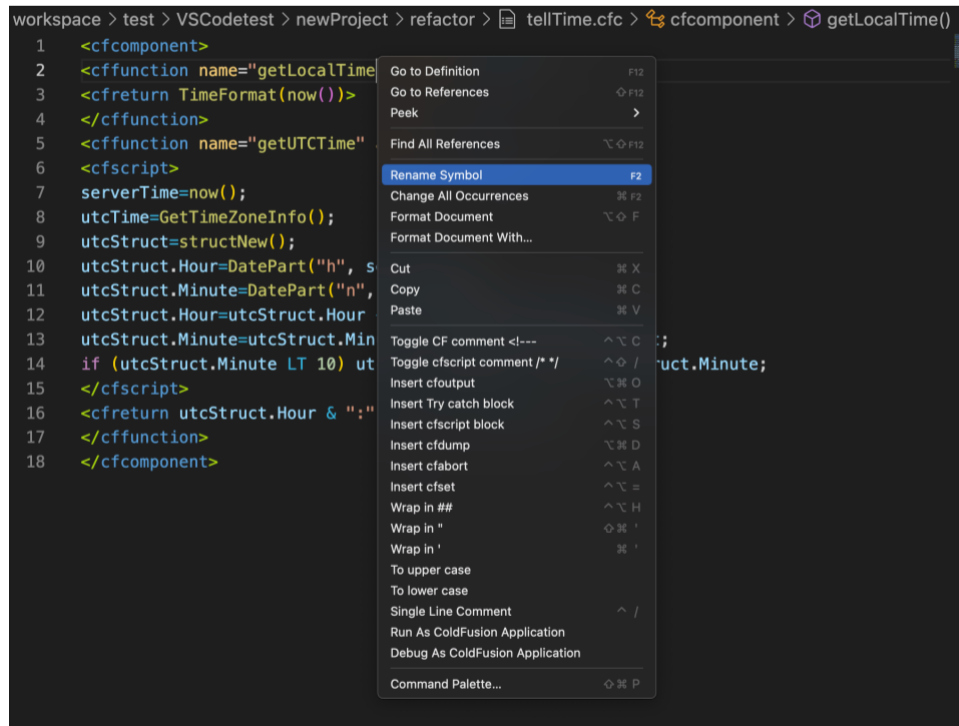
Right-click the file and select Find File References.



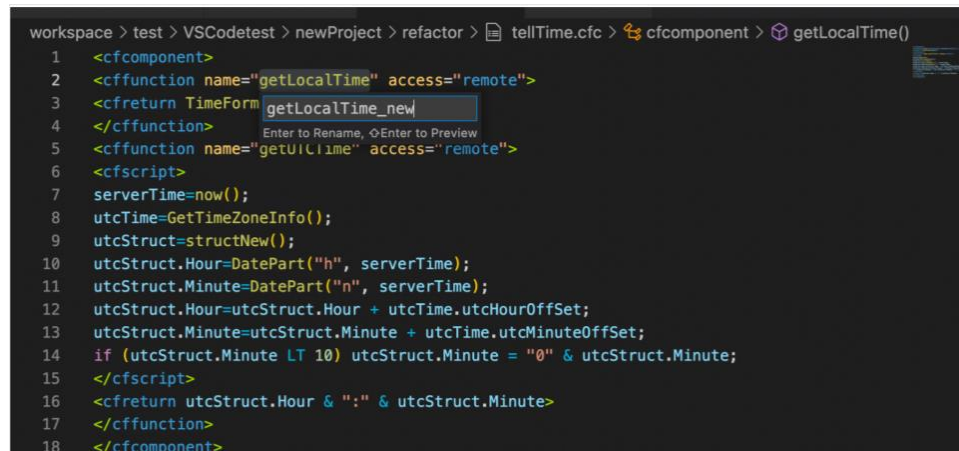
Refactor function or variable

When you rename a function or variable, you can refactor all valid references of the function or variable.

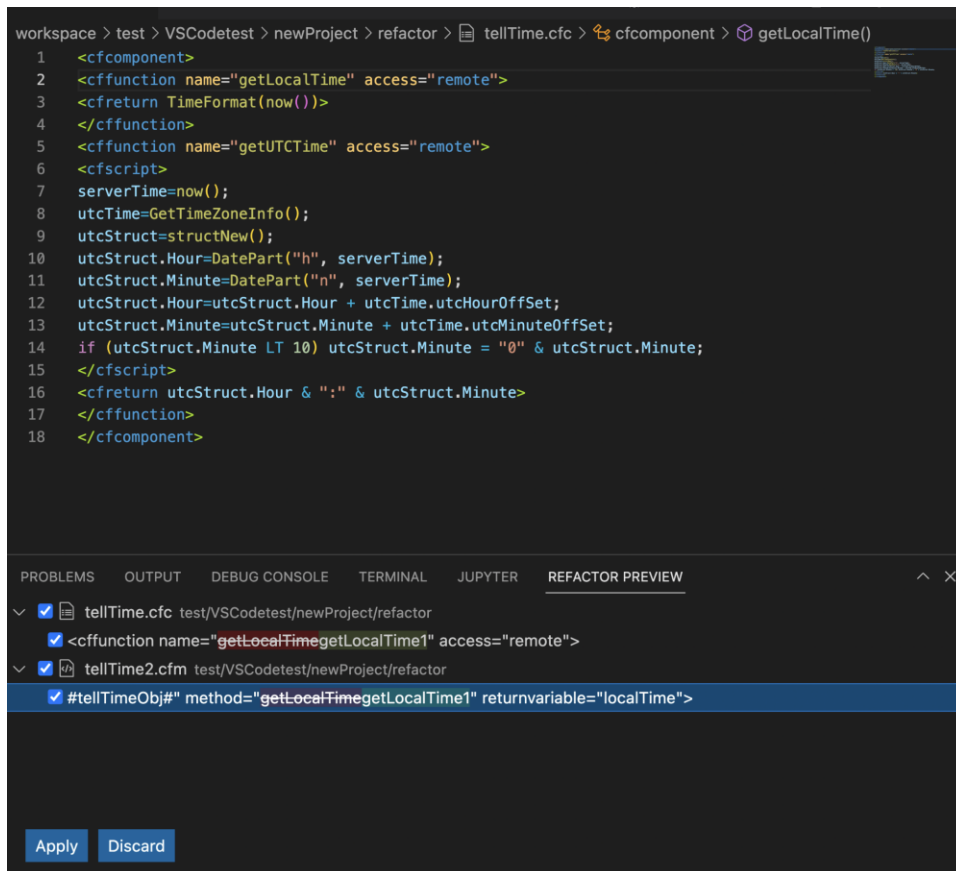
1. Right-click the function or variable.



2. Click **Rename Symbol**.



3. Enter the new name and hit Shift + Enter.
4. The Refactor preview appears at the bottom.



Known issues in this release

- If you right-click the Problems panel, the Context menu options, such as, Delete, Select All, etc. do not work as expected.
- The text that appears upon hovering on member functions is currently not supported.
- If you copy a project from one workspace to another, the project displays, but does not run.
- Reference search does not work as expected on linked folders.
- There is limited code assist support for cfjava (tag and script versions).
- Variable mappings are not supported in this release.
- In the Project Manager, you are unable to multi-select the project files.
- When creating a project, you are unable to create a template, as the option is not present.
- The debugger does not work as expected in this release.
- Security Analyzer may not work as expected when the project is attached to a Remote server in VS Code.
- Code Assist for function names does not work in writeDump.
- Code navigation does not work for try-catch block.
- Copying a project from one workspace to another in a Project Manager displays the project, but the project will not run.
- Help text on hovering on a few tags may not appear.

- As part of Code Refactoring, if a CFC or cfm file is renamed, the change is made in all the referenced files, but the files are not saved automatically.
- On macOS, autocomplete for SQL does not display any table that is a part of a datasource.