

isCOBOL Evolve: Veryant isCOBOL Extension

Veryant isCOBOL Extension for Visual Studio Code



Copyrights

Copyright (c) 2023 Veryant
6390 Greenwich Drive, #225, San Diego, CA 92122, USA

All rights reserved.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution and recompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Veryant and its licensors, if any.

Key Topics

- [Configuration](#)
- [Project management](#)
- [Code Editing](#)
- [Compiling](#)
- [Running](#)
- [Debugging](#)

Overview

The Veryant isCOBOL extension provides support for compiling, editing, and debugging COBOL source code in Visual Studio Code.

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux.

Getting Started

The setup of the development environment requires the following steps:

1. [Download and install the Java Development Kit \(JDK\)](#)
2. [Download and install the isCOBOL Evolve SDK](#)
3. [Download and install Visual Studio Code](#)
4. [Download and install the Veryant isCOBOL extension](#)
5. [Activate the License](#)

In order to activate your isCOBOL Evolve products, you will need the e-mail you received from Veryant containing your license key. Contact your Veryant representative for details.

Download and install the Java Development Kit (JDK)

A JDK must be installed on your machine in order to use the Veryant isCOBOL extension. For best results and performance, install the latest JDK version available for your platform. isCOBOL is certified to work correctly with both Oracle JDK and OpenJDK from version 8 to version 17.

Self-extracting setups are provided for the Windows platform.

On Unix/Linux platforms Java may be already installed. If it's not the case, you can install it using the appropriate system commands (e.g. yum, or apt-get).

Download and install the isCOBOL Evolve SDK

A minimal installation of the isCOBOL Evolve SDK is required.

The Veryant isCOBOL extension uses the compiler and runtime available in the isCOBOL Evolve SDK.

Refer to [Download and install isCOBOL Evolve SDK](#) in the User Guide for information about downloading and installing the isCOBOL SDK.

Download and install Visual Studio Code

Visual Studio Code is a third party product.

Setups and installation instructions are available at <https://code.visualstudio.com/>.

Download and install the Veryant isCOBOL extension

1. If you haven't already done so, [Download and install the Java Development Kit \(JDK\)](#).
2. Go to "<https://support.veryant.com>".
3. Sign in with your User ID and Password.
4. Click on the "Download Software" link.
5. Scroll down to the list of Platform Independent files and download "veryant-extension.vsix".
6. Open Visual Studio Code.
7. Open the "extensions" sidebar (you can use "Ctrl+Shift+X").
8. Click on the ellipsis icon in the top right corner of the menu.
9. Select "Install from VSIX..."
10. Browse for "veryant-extension.vsix" that you previously downloaded.

Activate the License

The Veryant isCOBOL extension looks for the following configuration properties for the license key:

```
iscobol.compiler.license.2023=<license_key>  
iscobol.license.2023=<license_key>
```

The keys should be stored in one of the following files (if they exist):

Windows

1. \etc\iscobol.properties in the drive where the working directory is
2. C:\Users\<username>\iscobol.properties (the setup wizard saves licenses here, if you don't skip activation)
3. iscobol.properties found in the Java Classpath
4. a custom configuration file passed on the command line
5. %ISCOBOL%\iscobol.properties

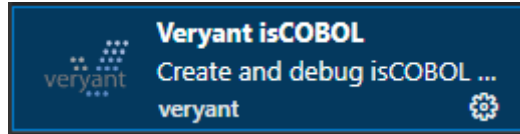
Unix/Linux

1. /etc/iscobol.properties
2. \$HOME/iscobol.properties
3. iscobol.properties found in the Java Classpath
4. a custom configuration file passed on the command line
5. \$ISCOBOL/iscobol.properties

NOTE - Files are listed in the order they're processed. If the license key appears in more than one of the above files, then the last occurrence is considered.

Configuration

After a successful installation, the following item appears in the "extensions" sidebar:



Click on the gear icon (or right click) and choose "extension Settings" to configure the Veryant isCOBOL extension.

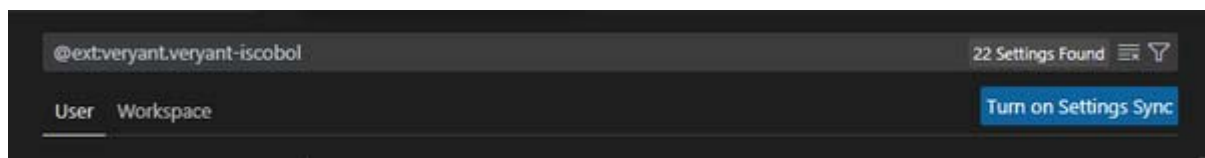
The following settings are available:

Setting	Description
Veryant > Compiler: Options	List of compiler options to be used. Multiple options are separated by space.
Veryant > Debugger: External Debug Configs	Path to external configuration file, which provides specific command names and regular expressions to parse and interact with external debug process.
Veryant > Debugger: Params	Parameters that will be accepted before running the debugger.
Veryant > Debugger: Trace File	Path to the file where trace will be stored. When specified, every interaction with the external command-line debugger will be logged on this file.
Veryant > Editor: Diagnose Copy	Controls whether Visual Studio Code should diagnose copy files.
Veryant > Editor: Special Colors	Special color settings for terms interpreted by the Veryant isCOBOL extension.
Veryant > Editor: Folding	Controls whether folding is allowed.
Veryant > Editor: Formatter: Location	COBOL external formatter location.
Veryant > Editor: Invert Special Colors in Light Theme	Controls whether special colors should be reversed when using a light theme
Veryant > Editor: Log	Controls whether Veryant isCOBOL extension logging is active.
Veryant > Editor: Max Cache Time For Expanded Source	Maximum time to keep expanded source cache in milliseconds.
Veryant > Editor: Returns Last Cache From Expanded Source	Indicates whether to return the last cache when reaching the maximum cache time.
Veryant > Editor: Snippets Repositories	Repositories where JSON snippets are located.
Veryant > Editor: Special Auto Documentation	Controls whether special auto documentation of some language objects is active.

Setting	Description
Veryant > Editor: Tabstops	COBOL tabstops.
Veryant > Editor: Variable Suggestion	Controls whether COBOL variable suggestion is allowed.
Veryant > Jdk: Root	Root folder of the Java JDK.
Veryant > Jre: Root	Root folder of the Java JRE.
Veryant > Main: Program	Program to start.
Veryant > Program: Arguments	Arguments to pass to the isCOBOL program.
Veryant > Runtime: Options	Options passed to the iscrun command when running the isCOBOL program.
Veryant > Sdk: Folder	Root folder of the isCOBOL Evolve SDK.

The minimal configuration for a correct editing and compiling requires *Veryant > Jdk: Root* and *Veryant > Sdk: Folder* to be set.

The same settings are available at User level and Workspace level:



Settings made at User level are used for every project.

Settings made at Workspace level are used for the current project and override settings at User level.

Project management

Visual Studio Code doesn't use the usual "File > New Project" dialog method, which IDEs commonly use to start developing new projects. The basic design of Visual Studio Code is that of a text editor, which uses your file system to browse for existing files to edit and compile.

Although you can use the Veryant isCOBOL extension just to view and edit COBOL source files, you need a project with a specific directory structure to be able to compile, run and debug COBOL programs.

To create a new project

1. click on the gear icon in the bottom left corner and choose "Command Palette..." or alternatively press Ctrl-Shift-P to open the command palette,
2. select the command "isCOBOL: create new COBOL project",
3. browse for the directory where your project will be stored.

The following folders will be created in the chosen directory:

Folder	Content
cpy	COBOL copybooks
list	list files
output	compiled class files
source	COBOL source files

To open an existing project

1. click on *File* in the menu bar and choose "Open Folder..." or alternatively press Ctrl-K followed by Ctrl-O,
2. select the directory where your project is stored.

Code Editing

The Veryant isCOBOL extension is designed for COBOL code developed with FREE FORMAT style, and is not fully prepared for FIXED FORMAT. Comment lines are treated as `*>` and not as `*` which is typically used in FIXED mode.

The Veryant isCOBOL extension offers the following features:

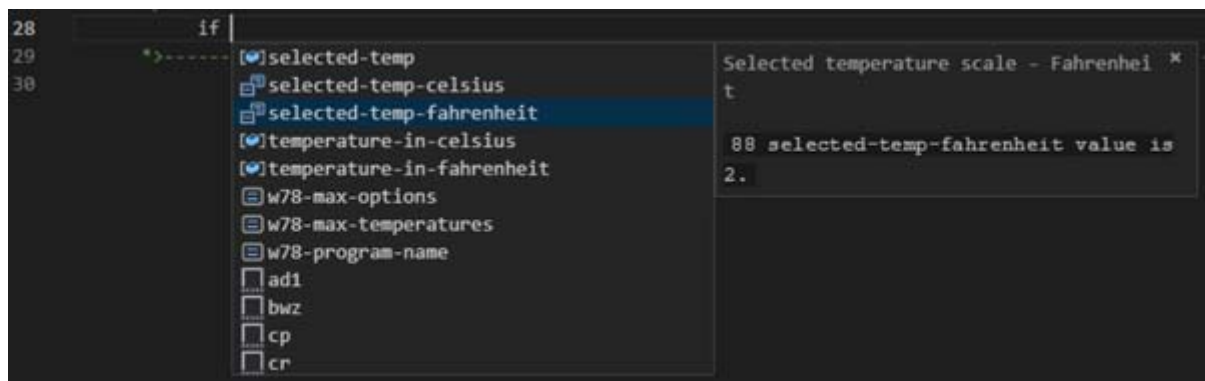
Syntax coloring

COBOL keywords, strings and identifiers are colored with different colors so you can easily notice typo errors like misspelled keywords or unclosed strings. The syntax coloring also makes the source code easier to view.

```
7 paragraph-1.  
8 *> this line is a comment  
9     move "string" to data-item-x.  
10    move 123 to data-item-9.
```

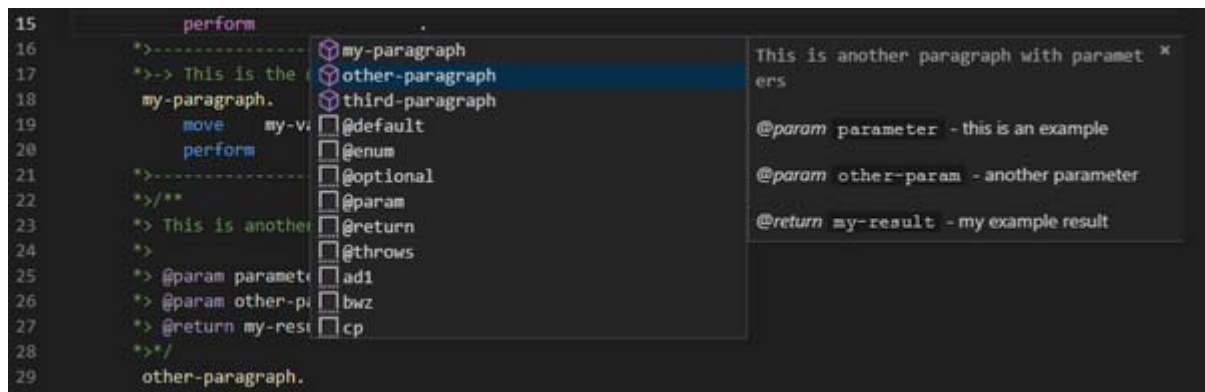
Variable suggestion

Variables are suggested on several COBOL commands, such as MOVE, IF, SUBTRACT, ADD and others. Constants are also considered, each one appearing with a different icon on IntelliSense.



Paragraph suggestion and documentation

This extension suggests paragraphs based on what is typed on perform clause.



Code highlight

When you click over some COBOL keywords, the related keywords are highlighted for better understanding.

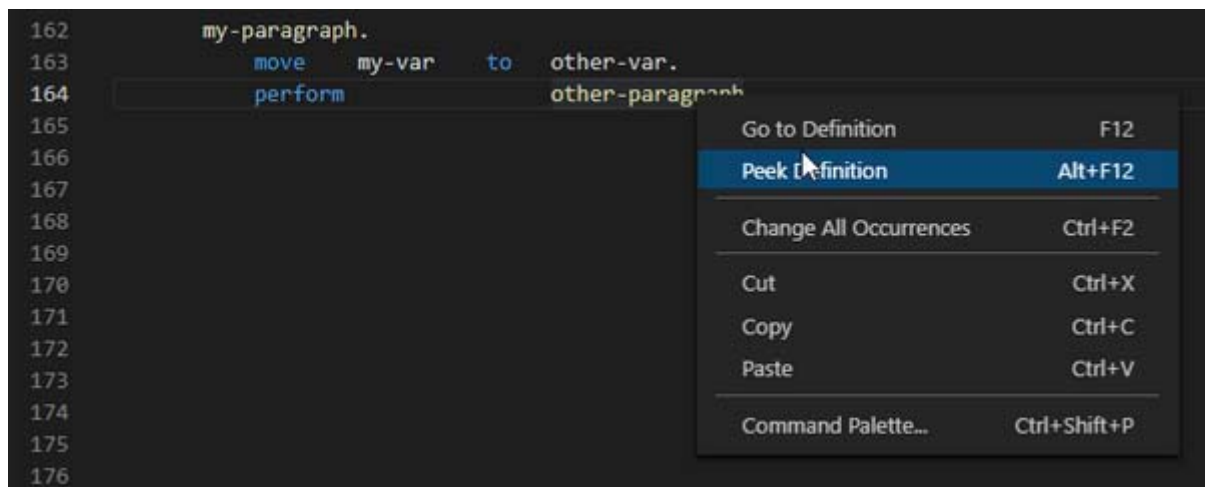

```

29      if condition
30          ...
31      else,
32          evaluate          true,
33          when condition
34              ...
35          when condition-2
36              ...
37      end-evaluate,
38      end-if.
39
40      perform          until exit
41          ...
42      end-perform.

```

Peek definition

You can also peek at COBOL variable and paragraph definitions.



Formatter and loop completion

The Language Server provides automatic formatting for several COBOL clauses.

IntelliSense for variable declaration

The variable declaration is done in two steps so that the extension parses the picture and generates the most appropriate value in the VALUE clause.

Compiling

Before compiling, be sure to save the last modification by pressing Ctrl-S.

To compile the current source file, select "isCOBOL: Compile current source file" from the command palette or press Ctrl-Shift-F9.

To compile all the source files in the project, select "isCOBOL: Compile source code" from the command palette or press Ctrl-F9.

Running

To run the current program, select "isCOBOL: run current file" from the command palette or press Ctrl-Shift-F5.

To run the whole application, set [Veryant > Main: Program](#) in the settings, then select "isCOBOL: run project" from the command palette or click the green play button in the tool-bar.

The runtime configuration properties should be stored in one of the following files (if they exist):

Windows

1. \etc\iscobol.properties in the drive where the working directory is
2. C:\Users\<username>\iscobol.properties (the setup wizard saves licenses here, if you don't skip activation)
3. iscobol.properties found in the Java Classpath (the output directory of the project is included in the Classpath)
4. a custom configuration file passed with the `-c <configuration_file>` option in [Veryant > Runtime: Options](#)
5. %ISCOBOL%\iscobol.properties

Unix/Linux

1. /etc/iscobol.properties
2. \$HOME/iscobol.properties
3. iscobol.properties found in the Java Classpath (the output directory of the project is included in the Classpath)
4. a custom configuration file passed with the `-c <configuration_file>` option in [Veryant > Runtime: Options](#)
5. \$ISCOBOL/iscobol.properties

NOTE - Files are listed in the order they're processed. If the license key appears in more than one of the above files, then the last occurrence is considered.

Debugging








Before debugging, be sure that the programs are compiled either with the `-d` option or the `-dx` option.

One of these options should appear in [Veryant > Compiler: Options](#) in the settings.

To debug the current program click the *Run and Debug* button in the Activity Bar or press F5.

To run the whole application, set [Veryant > Main: Program](#) in the settings, then click the *Run and Debug* button in the Activity Bar or press F5.

The following debug actions are available:

Function	Button	Keyboard shortcut
Continue		F5
Pause		F6
Step Over		F10
Step Into		F11
Step Out		Shift-F11
Restart		Ctrl-Shift-F5
Stop		Shift-F5

It's possible to set breakpoints by clicking in the area before the line number or by pressing F9 when you're positioned on the desired line. A breakpoint is represented with a red circle at the beginning of the line:

```

94  set exception value 1 to item-help
95  set environment "HELP-PROGRAM" to "helpprg"

```

If you hover the mouse over a data-item name, the data-item content is shown in a tool-tip:

```

80  accept terminal-abilities from terminal-info.
81  1
82  call "C$GETRUNENV" giving env-code
83
84  set environment "gui.justify_num_fields" to "1"

```

It's possible to monitor the changes of the value of a data item by adding it to the *Watched variables* list. To do so:

1. select the data item in the editor
2. right click with the mouse
3. choose "Add to Watch" from the context menu

The runtime configuration properties should be stored in one of the following files (if they exist):

Windows

1. \etc\iscobol.properties in the drive where the working directory is
2. C:\Users\<username>\iscobol.properties (the setup wizard saves licenses here, if you don't skip activation)
3. iscobol.properties found in the Java Classpath (the output directory of the project is included in the Classpath)
4. a custom configuration file passed with the `-c <configuration_file>` option in [Veryant > Runtime: Options](#)
5. %ISCOBOL%\iscobol.properties

Unix/Linux

1. /etc/iscobol.properties
2. \$HOME/iscobol.properties
3. iscobol.properties found in the Java Classpath (the output directory of the project is included in the Classpath)
4. a custom configuration file passed with the `-c <configuration_file>` option in [Veryant > Runtime: Options](#)
5. \$ISCOBOL/iscobol.properties

NOTE - Files are listed in the order they're processed. If the license key appears in more than one of the above files, then the last occurrence is considered.