



TIBCO Flogo® Extension for Visual Studio Code

Installation

Version 1.3.2 | July 2025

Contents

Contents	2
Installing the Extension	3
Before you Begin	3
Installing through the CLI	3
Installing through the Visual Studio Code GUI	3
After Installing the Extension	4
Uninstalling the Extension	4
Upgrading the Extension	4
Product Overview	6
Using the Extension with Visual Studio Code Dev Containers	7
Docker Desktop Configuration	7
Configuration 1: WSL-2-Disabled, Hyper-V-Enabled, File-Sharing-Enabled	7
Configuration 2: WSL-2-Enabled	8
Prerequisites	8
Approach 1: Using a Predefined Microsoft Base Image	8
Approach 2: Configuring the .json File with a Custom Image	10
Approach 3: Creating and Using a Custom Dev Container Image	11
TIBCO Documentation and Support Services	13
Legal and Third-Party Notices	15

Installing the Extension

You can install Flogo® Extension for Visual Studio Code (VS Code) using the following ways:

- Command-Line Interface
- Visual Studio Code Graphical User Interface

Before you Begin

- Familiarity with Visual Studio Code
- Familiarity with TIBCO Flogo®
- Familiarity with TIBCO® Control Plane
- Familiarity with TIBCO Cloud™ Integration

Installing through the CLI

Use the following command:

```
code --install-extension <filename>.vsix
```

Installing through the Visual Studio Code GUI

Procedure

1. Open the **Extensions view** (Ctrl+Shift+X) from the activity bar of Visual Studio Code.
2. Click the overflow button available on the upper-right of the **extension view**.
3. Select **Install from VSIX** from the bottom of the overflow menu.
4. Browse to the location of the .vsix file and select **Install** to complete the installation.

After Installing the Extension

Procedure

1. Before creating a Flogo application, you must create a new workspace. For more information, see the "Creating a Flogo App section" in the *TIBCO Flogo® Extension for Visual Studio Code* user guide.
2. Go to **File** and enable the **Auto-save** option.
3. Install go version=go1.24.4 or above to build and run Flogo apps locally. For more information, see "Building the App locally" section in the *TIBCO Flogo® Extension for Visual Studio Code* user guide.



Note: Some Flogo Connectors require local driver installation. For more information, see the "Prerequisites for Connectors" section in the *TIBCO Flogo® Extension for Visual Studio Code* user guide.

Uninstalling the Extension

Procedure

1. Open the **Extensions** view (Ctrl+Shift+X) from the activity bar of Visual Studio Code.
2. Under the **INSTALLED** dropdown menu, click the extension you want to uninstall.
3. Click **Uninstall** on the Extension Detail page.

Upgrading the Extension

To upgrade Flogo Extension for Visual Studio Code, you can perform the following steps:

Procedure


1. Uninstall the current version of TIBCO Flogo® Extension for Visual Studio Code by performing the steps in [Uninstalling the Extension](#).

2. Install the latest version by performing the steps in [Installing the Extension](#).

If you do not want to uninstall the existing version, you can perform the following steps:

Procedure

1. To install the latest version of extension without uninstalling the existing one, perform the steps in [Installing the extension](#).
2. Restart the extension.

 **Note:** Ensure you close all the **app.flogo** tabs in the Webview before installing the latest version.

Product Overview

Using Flogo® Extension for Visual Studio Code (VS Code), you can accelerate the development of TIBCO Flogo® apps by tapping into the robust Microsoft Visual Studio Code ecosystem. You can:

- Design, build, and test Flogo® apps locally in Flogo Extension for Visual Studio Code.
- Deploy the apps anywhere, including on-premises and private or public cloud services, such as AWS Lambda, or devices.

Flogo Extension for Visual Studio Code is powered by Project Flogo®, a lightweight integration engine. You can use Flogo Extension for Visual Studio Code to interconnect APIs, systems, and data and use the TIBCO Platform™ as well to deploy and monitor your apps.

Using the Extension with Visual Studio Code Dev Containers

Visual Studio Code Dev Containers offer self-contained, reproducible development environments. These Docker containers are configured for a full development experience, containing the necessary tools, runtimes, and libraries. For more information, see [Create a Dev Container](#).

This section provides three approaches to configure a Dev Container for Flogo app development. These approaches apply to various Docker Desktop configurations. You can configure the foundational Docker image and preserve the environment within a personalized image.

Docker Desktop Configuration

Before proceeding, ensure that your Docker Desktop configuration aligns with your requirements, as it affects file access and performance.

Configuration 1: WSL-2-Disabled, Hyper-V-Enabled, File-Sharing-Enabled

- Hyper-V is used as the virtualization backend.
- Windows Subsystem for Linux (WSL) is disabled.
- File sharing is enabled to provide container access to Windows files.

i Note: This configuration offers better performance in some environments but requires a file-sharing setup.

Configuration 2: WSL-2-Enabled

- WSL 2 is used as the virtualization backend.
- File system integration between Windows and Linux-based WSL 2 environment.

i **Note:** Both these configurations support the approaches in the following sections, but file paths and performance might vary.

Prerequisites

- Microsoft Visual Studio Code (VS Code) with the **Dev - Containers** extension
- Docker Desktop
- Linux build (.vsix) of TIBCO Flogo® Extension for Visual Studio Code

i **Note:** The Dev Container mode is not compatible with Mac or Windows builds.

- A project folder (can be empty) to initialize the Dev Container.

Approach 1: Using a Predefined Microsoft Base Image

To set up a Dev Container using a predefined Microsoft base image available in VS Code, perform the following steps:

1. Open your Flogo project (or an empty folder) in Visual Studio Code.
2. Open the Command Palette (Ctrl+Shift+P or Cmd+Shift+P) and select **Dev Containers: Open Folder in Container**.

i **Note:** If a .devcontainer folder with a devcontainer.json file does not exist, you are prompted to create one.

3. Choose **From a definition** to select a configuration. Options include **Docker in Docker devcontainers**, **Docker outside of Docker devcontainers**, or **Docker outside of Docker Compose devcontainers**.
4. **Optional:** Select the **Install OSS Moby build instead of Docker CE** option.
5. Select and enable features, such as Go devcontainers, to install the Go language and the related utilities inside the container. For example, Microsoft's Dev Container [.mcr.microsoft.com/devcontainers/base:bullseye](https://mcr.microsoft.com/devcontainers/base:bullseye).

Based on the configuration and your selections, a `.devcontainer` folder with a `devcontainer.json` file and a `Dockerfile` is built. You are connected to the container environment in Visual Studio Code.

6. Ensure the Linux `.vsix` installer file of the extension is accessible to the container.

i Note: It is a best practice to copy the `.vsix` file into a location within the container, such as a shared folder or the container's `/tmp` directory. If you are using Docker Desktop with file-sharing mode, ensure that the folder is shared with Docker.

7. Open the Extensions view (Ctrl+Shift+X), click the ellipsis icon, and select **Install from VSIX**.
8. Navigate to the `.vsix` file and install it.


i Note: If you want to use connectors, such as EMS or Oracle, install the required client libraries and configure the `LD_LIBRARY_PATH` environment variable. For more information, see [Prerequisites for Connectors](#).

9. Build, run, and test your applications.
10. **Optional:** You can also use the Maven plug-in within the Dev Container to build, run, and test your flows. For more information, see the [Maven Documentation](#).
11. To stop the container, close the Visual Studio Code window or use the Command Palette and select **Remote-Containers: Close Remote Connection**.


Approach 2: Configuring the .json File with a Custom Image

To manually configure the devcontainer.json file to use a custom Docker image, perform the following steps:

1. Open Visual Studio Code and open the folder for your Flogo project.
2. Open the Command Palette (Ctrl+Shift+P) and select **Dev Containers:Open Folder in Container**.

 **Note:** If a .devcontainer folder with a devcontainer.json file does not exist, you are prompted to create one.

3. Choose a basic container definition, such as Debian or Ubuntu.


 **Note:** Do not select a specific Go image.

4. In the devcontainer.json file, modify the image property to specify your custom image:

```
"image": "golang:1.24-alpine"
```

Based on the configuration and your selections, a .devcontainer folder with a devcontainer.json file and Dockerfile is built. You are connected to the container environment in Visual Studio Code.

5. Ensure the Linux .vsix installer file of the extension is accessible to the container.

 **Note:** It is a best practice to copy the .vsix file into a location within the container, such as a shared folder or the container's /tmp directory. If you are using Docker Desktop with file-sharing mode, ensure that the folder is shared with Docker.

6. Open the Extensions view (Ctrl+Shift+X), click the ellipsis icon, and select **Install from VSIX**.
7. Navigate to the .vsix file and install it.

i Note: If you want to use connectors, such as EMS or Oracle, install the required client libraries and configure the `LD_LIBRARY_PATH` environment variable. For more information, see [Prerequisites for Connectors](#).

8. Build, run, and test your applications.
9. **Optional:** You can also use the Maven plug-in within the Dev Container for building, running, and testing your flows. For more information, see the [Maven Documentation](#).
10. To stop the container, close the VS Code window or use the Command Palette and select **Remote-Containers: Close Remote Connection**.

⚠ Caution: Custom Docker images have the following limitations:

- Alpine Linux-based images might have compatibility issues with glibc libraries or applications. Enterprise Message Service, which is built on glibc, encounters relocation errors.
- Debian or Ubuntu-based images might require installation of the `pkg-config`, `libssl-dev`, and `zlib1g-dev` packages. You might also need to install the Go language.

Approach 3: Creating and Using a Custom Dev Container Image

To create a custom Docker image from a configured Dev Container for consistent setups that support sharing and reproducibility, perform the following steps:

1. Perform the steps in Approach 1 or Approach 2 to configure a Dev Container.
2. Open a terminal where Docker Desktop is installed.
3. To list the running containers, run the following command:

```
docker ps
```

4. Identify the container ID or the name of the running Dev Container.
5. To create an image from the running container, run the following command:

```
docker commit <container_id_or_name> my-flogo-dev:latest
```

- a. Replace <container_id_or_name> with the actual container ID or name
 - b. Replace my-flogo-dev:latest with a suitable name and tag for your custom image
6. **Optional:** If you want to share the image with other developers, push it to a Docker registry by performing the following steps:
 - a. Log in to a Docker registry: `docker login`
 - b. Push the image: `docker push my-flogo-dev:latest`. Adjust the image name or the tag
7. In your project, open the `.devcontainer` or the `devcontainer.json` file.
8. To specify the custom image, run the following command:

```
{  
  "image": "my-flogo-dev:latest",  
  // ... other devcontainer.json settings  
}
```

9. Use the Command Palette and select **Dev Containers:Open Folder in Container** to open the folder in the container.
10. Build, develop, and test your applications.

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

Product-Specific Documentation

The following documentation for this product is available on the [TIBCO Flogo® Extension for Visual Studio Code Product Documentation](#) page:

- *TIBCO Flogo® Extension for Visual Studio Code Release Notes*
- *TIBCO Flogo® Extension for Visual Studio Code Installation*
- *TIBCO Flogo® Extension for Visual Studio Code User Guide*
- *TIBCO Flogo® Extension for Visual Studio Code Mapper Functions Guide*

Other TIBCO Product Documentation

When working with TIBCO Flogo® Extension for Visual Studio Code, you may find it useful to read the documentation of the following TIBCO products:

- [TIBCO® Control Plane](#)
- [TIBCO Flogo® Enterprise](#)
- [TIBCO Cloud™ Integration](#)

How to Access Related Third-Party Documentation

When working with TIBCO Flogo® Extension for Visual Studio Code, you may find it useful to read the documentation of Visual Studio Code: <https://code.visualstudio.com/>.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to

gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the [TIBCO Ideas Portal](#). For a free registration, go to [TIBCO Community](#).

Legal and Third-Party Notices

SOME CLOUD SOFTWARE GROUP, INC. (“CLOUD SG”) SOFTWARE AND CLOUD SERVICES EMBED, BUNDLE, OR OTHERWISE INCLUDE OTHER SOFTWARE, INCLUDING OTHER CLOUD SG SOFTWARE (COLLECTIVELY, “INCLUDED SOFTWARE”). USE OF INCLUDED SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED CLOUD SG SOFTWARE AND/OR CLOUD SERVICES. THE INCLUDED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER CLOUD SG SOFTWARE AND/OR CLOUD SERVICES OR FOR ANY OTHER PURPOSE.

USE OF CLOUD SG SOFTWARE AND CLOUD SERVICES IS SUBJECT TO THE TERMS AND CONDITIONS OF AN AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER AGREEMENT WHICH IS DISPLAYED WHEN ACCESSING, DOWNLOADING, OR INSTALLING THE SOFTWARE OR CLOUD SERVICES (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH LICENSE AGREEMENT OR CLICKWRAP END USER AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE SAME TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, Flogo Enterprise, TIBCO Cloud Integration, and TIBCO Control Plane are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only. You acknowledge that all rights to these third party marks are the exclusive property of their respective owners. Please refer to Cloud SG’s Third Party Trademark Notices (<https://www.cloud.com/legal>) for more information.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file for the availability of a specific version of Cloud SG software on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at <https://www.cloud.com/legal>.

Copyright © 2025. Cloud Software Group, Inc. All Rights Reserved.