



TIBCO EBX®

Container Edition Guide

Version 6.2.3
April 2026

Table of contents

1. Building the image.....	4
2. Running the image.....	9
3. Customizing the image.....	29
4. Documentation and Support.....	35
5. Legal and Third-Party.....	37

CHAPTER 1

Building the image

This chapter contains the following topics:

1. [Overview](#)
2. [Requirement](#)
3. [Building the image](#)

1.1 Overview

TIBCO EBX® Container Edition image is Linux OS based for several architectures and includes the following:

- Application Server Apache Tomcat® 10.1
- Java JDK 25 LTS

1.2 Requirement

The image can be built on a POSIX native system or using a compatible layer:

- **Operating system** : Linux, macOS, Windows (10 version 2004, Server 2022 or greater) using WSL2,
- **Processor type** : AMD64 (Intel 64 bits), ARM64,
- **Docker Engine** installed and running,
- Access to **Internet** .

1.3 Building the image

Download installers

To build an EBX® image, one needs to download file **TIB_ebx_6.2.3_addon_6.2.X_container_edition.zip** from TIBCO eDelivery.

Running the interactive installer

To start installer on **Linux** or **macOS** :

- Unzip the **TIB_ebx_6.2.3_addon_6.2.X_container_edition.zip** ,

- Open a terminal in folder where file **ebx-ce-installer.sh** file is located,
- Execute command **./ebx-ce-installer.sh** .

To start installer on **Windows** :

- Unzip the **TIB_ebx_6.2.3_addon_6.2.X_container_edition.zip** ,
- Open a Windows PowerShell in folder where file **ebx-ce-installer.bat** file is located,
- Execute command **ebx-ce-installer.bat** .

Follow the instructions and select the optional components to be added to the image. If Metadata is selected, the interactive installer automatically adds the following addons: DAMA, DINT, DMDV, DPRA and TESE.

The installer will then build the image and print a summary similar to:

```
*****
The TIBCO EBX Container Edition image was successfully created with following names:
  ebx:latest
  ebx:6.2.3-dama-dint-dmdv-dpra-mame-metadata-tese-6.2.X
This image includes the following optional component(s):
Metadata
Digital Asset Manager (DAMA)
Data Exchange (New) (DINT)
Data Model and Data Visualization (DMDV)
Insight (New) (DPRA)
Match and Merge (MAME)
EBX GO (MODA)
Information Search (TESE)
Data Exchange (ADIX, legacy)
Insight (DQID, legacy)
To run this image with the default configuration and an embedded database, use command:
  docker run -p 8080:8080 -d ebx:latest
You can test EBX by visiting http://localhost:8080 in a browser.
To run this image with other configurations, for example with an external database, see
documentation.
*****
```

If you select the **Generate image only for Linux on current architecture?** option, the install prompts for additional architectures that you want to build EBX® Container Edition for. In that case, Docker image tags will contain the architecture it was built for. For instance, **ebx:6.2.3-dama-dint-dmdv-dpra-mame-metadata-tese-6.2.X-amd64** .

Running the batch installer

To start batch installer on **Linux** or **macOS** :

- Unzip the **TIB_ebx_6.2.3_addon_6.2.X_container_edition.zip** .
- Open a terminal in the folder where the **ebx-ce-installer-batch.sh** file is located.
- Execute the following command: **./ebx-ce-installer-batch.sh [options] [metadata] [<addon(s)>]**

```
Batch script to build image with optionally Metadata and/or addons.

Options:
-a          add Metadata and all addons excluding legacy addons
-al         add Metadata and all addons including legacy addons
-p <arch,...> architecture(s) to build on (i.e. \"amd64,arm64/v8,ppc64le\")
            possible values: amd64, arm64/v8, ppc64le, s390x
            using this parameter the default image tag is suffixed
-t <tag_file> create a file named tag_file including the image tag built
--help      display this help and exit

[metadata]  install Metadata
[<addon(s)>] install one or more addons (in lowercase and separated by a white space)
            for example: mame tese
```

To start the batch installer on **Windows** :

- Unzip the **TIB_ebx_6.2.3_addon_6.2.X_container_edition.zip** .
- Open a Windows PowerShell in the folder where the **ebx-ce-installer-batch.bat** file is located.
- Execute the following command: **ebx-ce-installer-batch.bat [options] [metadata] [<addon(s)>]**

The batch installer will then build the image and print a summary identical to the interactive installer.

Testing the image

The image can be run locally using command:

```
docker run -p 8080:8080 -d ebx:latest
```

In **production** , it is recommended to **not** use the **latest** tag. The installer always generates another tag, depending on the selected optional components.

Using the previous example, the tag is **6.2.3-dama-dint-dmdv-dpra-mame-metadata-tese-6.2.X** because optional components where selected:

- Metadata,
- Digital Asset Manager (DAMA),
- Data Exchange (New) (DINT),
- Data Model and Data Visualization (DMDV),
- Insight (New) (DPRA),
- Match and Merge (MAME),
- Information Search (TESE).

The image can then be run using the following command:

```
docker run -p 8080:8080 -d ebx:6.2.3-dama-dint-dmdv-dpra-mame-metadata-tese-6.2.X
```

If no optional components are selected, the tag is **6.2.3** .

The image can be run using the following command:

```
docker run -p 8080:8080 -d ebx:6.2.3
```

Sharing the image

The steps to share an image depend on your company's infrastructure.

In the following example, the image is pushed to a Docker private registry named **myregistry** :

```
docker tag ebx:6.2.3 myregistry:5000/ebx:6.2.3
docker push myregistry:5000/ebx:6.2.3
```

In the event that the image is built for several architectures, detailed instructions are provided in the **README-MULTIARCH.md** file located in the **files/** subdirectory.

Other architectures than AMD64

The installers are tested on AMD64 (Intel 64 bits). On these architectures the generated image's architecture will be **linux/amd64** .

The installers may succeed when running on a system using another architecture. In that case the generated image platform will be linux for that architecture type.

For example, if the installer runs on a macOS M1 workstation, the generated image's platform will be **linux/arm64** .

Attention

Generating an Linux image for AMD64 (Intel 64 bits) or ARM64 architecture is fully supported. The other platforms are **experimental** and are **not supported in production** .

CHAPTER 2

Running the image

This chapter contains the following topics:

1. [Support Policy](#)
2. [Starting EBX®](#)
3. [Container access](#)
4. [Environment variables](#)
5. [Configuration files](#)
6. [Volumes](#)
7. [Linux user and group](#)
8. [Host configuration](#)
9. [Logs access](#)

2.1 Support Policy

Docker

TIBCO EBX® Container Edition is tested with the Docker Engine version 20.10.

Kubernetes

Current EBX® Container Edition release was tested with a certified Kubernetes implementation version 1.26 and Red Hat® OpenShift® 4.13.1.

We provides Helm chart samples for EBX® Container Edition that are available on GitHub.

See <https://github.com/TIBCOSoftware/ebx-container-edition> for more information.

2.2 Starting EBX®

First-launch assistant

To start EBX® with default configuration that includes an embedded H2 database, execute command:

```
docker run -p 8080:8080 -d ebx:6.2.3
```

Using a browser, you can connect to EBX® with URL <http://localhost:8080> . This will display the **first-launch assistant** that will help you configure EBX®.

For more information on the first launch assistant, see chapter *Initialization and first-launch assistant* .

Automatic initialization

To start EBX® with automatic initialisation on first startup and an embedded H2 database, execute command:

```
docker run -d -p 8080:8080 \
-e "EBX_FLA_DISABLED=true" \
-e "EBX_INSTALL_ADMIN_PASSWORD=<password>" \
-e "EBX_INSTALL_LICENSE=XXXXX-XXXXX-XXXXX-XXXXX" \
ebx:6.2.3
```

The EBX® repository will be automatically created on first startup.

Using a browser, you can connect to EBX® with URL <http://localhost:8080> . This will display the EBX® login screen. The username for administrator is **admin** and the password is the one specified in previous command.

Note

It's possible to specify another username for the administrator. For more information see [Automatic repository installation on first launch](#) [p 11] .

Supported browsers

For details a supported browsers see: *Supported Web Browsers* .

2.3 Container access

The following command will start a bash shell inside the EBX® container using default user:

```
docker exec -it <container-id> bash
```

To connect as root, use command:

```
docker exec -it --user root <container-id> bash
```

2.4 Environment variables

This chapter describes the environment variables supported by **EBX® Container Edition** .

All are optional.

Disabling First-launch assistant

For security reasons, one might want to disable the first-launch assistant in all circumstances.

This is achieved by setting environment variable to **EBX_FLA_DISABLED** to **true** .

Automatic repository installation on first launch

If the repository is not yet initialized and first-launch assistant is disabled, EBX® will automatically trigger its installation if following mandatory variables are provided:

Name	Default	Description
EBX_INSTALL_ADMIN_LOGIN	admin	Sets the EBX® administrator login name. This parameter is ignored if repository variable EBX_FL_A_DISABLED value is not true or if repository is already initialized.
EBX_INSTALL_ADMIN_PASSWORD		Sets the EBX® administrator password. This parameter is mandatory if EBX_FL_A_DISABLED value is true and is ignored if repository variable EBX_FL_A_DISABLED value is not true or if repository is already initialized.
EBX_INSTALL_LICENSE		Sets the EBX® license key if not already defined.

Note

If mandatory variables are not provided, EBX® will display an error message.

Example

To automatically install repository launch EBX® using following command:

```
docker run -d -p 8080:8080 \
-e "EBX_FL_A_DISABLED=true" \
-e "EBX_INSTALL_ADMIN_LOGIN=<login-name>" \
-e "EBX_INSTALL_ADMIN_PASSWORD=<password>" \
-e "EBX_INSTALL_LICENSE=XXXXX-XXXXX-XXXXX-XXXXX" \
ebx:6.2.3
```

Define EBX® license key

To start EBX® with a dedicated license key, execute command:

```
docker run -d -p 8080:8080 \
-e "EBX_INSTALL_LICENSE=XXXXX-XXXXX-XXXXX-XXXXX" \
ebx:6.2.3
```

Note: **EBX_INSTALL_LICENSE** is ignored if license key is already defined at persisted location.

The license key is persisted at location **/ebx/data/ebx-license.properties** .

See [Volumes](#) [p 26] for more information.

EBX® run mode

This property defines how EBX® runs.

Name	Default	Description
EBX_BACKEND_MODE	development	Three run modes are available: <i>development</i> , <i>integration</i> and <i>production</i> .

```
docker run -d -p 8080:8080 \
```

```
-e "EBX_BACKEND_MODE=production" \  
ebx:6.2.3
```

See *EBX® run mode* for more information.

URL configuration

Some EBX® features require generating URLs. Specific configuration may be required to achieve this, for example if EBX® is running behind a reverse proxy or on a Kubernetes cluster.

Name	Default	Description
EBX_IS_SECURED	If incoming request is HTTPS, "true" is assumed or else "false" is assumed.	If "true", the protocol "HTTPS" is always assumed. This value can be useful if the container is behind a reverse proxy, a firewall or an ingress that takes care of HTTPS encryption. This will also set the EBX® authentication cookie to be secure. This means that a user using a modern browser will not be able to log in using HTTP. If "false", the protocol "HTTP" is assumed.
EBX_AUTHENTICATION_REDIRECT_TO_HTTPS	Default is "false".	If "true" and the incoming request to the login form is HTTP, a redirect occurs to force HTTPS. This property it should be set to false if the container is behind a reverse proxy, a firewall or an ingress that takes care of HTTPS encryption.
EBX_HOSTNAME	The host name specified by the incoming HTTP(S) request.	The EBX® server host name.
EBX_PORT	The port number specified by the incoming HTTP request.	The EBX® server HTTP port number.
EBX_PORT_SECURED	The port number specified by the incoming HTTPS request.	The EBX® server HTTPS port number. Note HTTPS support must be provided by a reverse proxy or an ingress that takes care of encryption.
EBX_ROOT_PATH	By default, the context path is empty.	If set, all EBX® urls will be prefixed by this value. The value must have a leading / and must not have a trailing / except if value is /. For example a valid value is /mdm/sales. Setting this variable is useful when running more than one instance of EBX® with the same host name.

Name	Default	Description
EBX_URL_DEFAULT		This environment variable is used when a background task needs to calculate a URL to EBX®. It should be set to a full URL without the path component (EBX_ROOT_PATH applies for the path component). For example a valid value is: https://host_name. If EBX_URL_DEFAULT is not specified and EBX_HOSTNAME is specified, a default is calculated with following assumptions: - If EBX_IS_SECURED is true, then HTTPS is assumed with port number equal to EBX_PORT_SECURED (443 is the default). - If EBX_IS_SECURED is false or not set, then HTTP is assumed with port number equal to EBX_PORT (80 is the default).

EBX® Database connectivity

For information on supported databases see chapter *Supported databases*.

By default, an embedded H2 database is used. Data for this H2 database is persisted at location **/ebx/data/h2**.

An external database may be configured using the following variables:

Name	TIBCO EBX® main configuration file equivalent	Description
EBX_DB_FACTORY	ebx.persistence.factory	Specifies the type of database server.
EBX_DB_URL	ebx.persistence.url	The JDBC URL. Its format is: jdbc:<dialect>://<database_host>:<database_port_number>/<database_name> .
EBX_DB_USER	ebx.persistence.user	The database user id.
EBX_DB_PASSWORD	ebx.persistence.password	The database user password.

For more information on these variables see their TIBCO EBX® main configuration file equivalent in chapter *Configuring the EBX® repository*.

Note

The container includes JDBC drivers only for H2, PostgreSQL and Microsoft SQL Server. Using other databases that are supported by EBX® requires adding the driver.

For instructions on how to add a driver, see [Adding a new JDBC driver](#) [p 31] for more information.

You can configure EBX® to use a JNDI data source. See [Configuring the database in JNDI](#) [p 32] for more information.

Example

To start an instance of EBX® that connects to a PostgreSQL database, execute the following command:

```
docker run -d -p 8080:8080 \
-e "EBX_DB_FACTORY=postgresql" \
-e "EBX_DB_URL=jdbc:postgresql://<database_host>:<database_port>/<database_name>" \
-e "EBX_DB_USER=<user_name>" \
-e "EBX_DB_PASSWORD=<user_password>" \
ebx:6.2.3
```

Metadata classifier database connectivity

Metadata classifier uses the SQLAlchemy toolkit to connect to its database. The SQLAlchemy toolkit supports most common databases.

By default, an embedded SQLite database is used. Data for this SQLite database is persisted at this location: **/ebx/data/sqlite/ebx-metadata-classifier.db** .

An external database may be configured using the following variables:

Name	Description
EBX_METADATA_CLASSIFIER_DB_URL	The SQLAlchemy URL without credentials. Its format is: <dialect>://<database_host>:<database_port>/<database_name> .
EBX_METADATA_CLASSIFIER_DB_USER	The database user id.
EBX_METADATA_CLASSIFIER_DB_PASSWORD	The database user password.

The container includes only the driver for the default database (SQLite). Using other databases that are supported by SQLAlchemy requires adding the driver.

For instructions on how to add a driver, see [Adding a driver for the Metadata classifier database](#) [p 33] .

Example

To start an instance of Metadata that connects to a PostgreSQL database, execute the following command:

```
docker run -d -p 8080:8080 \
-e "EBX_METADATA_CLASSIFIER_DB_URL=postgresql://<database_host>:<database_port>/<database_name>" \
-e "EBX_METADATA_CLASSIFIER_DB_USER=<user_name>" \
-e "EBX_METADATA_CLASSIFIER_DB_PASSWORD=<user_password>" \
ebx:6.2.3
```

Notes

- The Metadata classifier database name (**database_name**) can be the same as [Database connectivity](#) [p 14] .
- For information about how to use a specific database driver and how to escape special characters in the connection URL, please refer to the SQLAlchemy's documentation.
- The H2 database is not supported by SQLAlchemy.

Directory connectivity

The EBX® directory can be configured through the following environment variables:

Name	TIBCO EBX® main configuration file equivalent	Description
EBX_DIRECTORY_FACTORY	ebx.directory.factory	Specifies the Java directory factory class name (optional)
EBX_DIRECTORY_SYNCHRONIZED_SOURCE	ebx.directory.synchronized.source	Enables and specifies the synchronized directory implementation (LDAP, SCIM or OnLogin)
EBX_SSO_PROTOCOL	ebx.directory.sso.protocol	Enables and specifies the SSO protocol (OIDC or SAML2)

See *Configuring the user and roles directory* for more information.

Synchronized directory

The EBX® synchronized directory can be configured through the following environment variables:

Name	TIBCO EBX® main configuration file equivalent (spaces must be removed)
EBX_DIRECTORY_SYNCHRONIZED_ROLE_MAPPER	ebx.directory.synchronized.role.mapper
EBX_DIRECTORY_SYNCHRONIZED_ROLE_MAPPER_DEFAULT_ADMINISTRATOR	ebx.directory.synchronized.role.mapper.default.administrator
EBX_DIRECTORY_SYNCHRONIZED_ROLE_MAPPER_DEFAULT_READ_ONLY	ebx.directory.synchronized.role.mapper.default.read_oly
EBX_DIRECTORY_SYNCHRONIZED_MAPPING_LDAP_ATTRIBUTES	ebx.directory.synchronized.mapping.ldap.attributes
EBX_DIRECTORY_SYNCHRONIZED_SSO_MAPPING_SAML2_ATTRIBUTES_ROLE_NAME	ebx.directory.synchronized.sso.mapping.saml2.attributes.role.name
EBX_DIRECTORY_SYNCHRONIZED_SSO_MAPPING_SAML2_ATTRIBUTES	ebx.directory.synchronized.sso.mapping.saml2.attributes
EBX_DIRECTORY_SYNCHRONIZED_SSO_MAPPING_OIDC_CLAIMS_ROLE_NAME	ebx.directory.synchronized.sso.mapping.oidc.claims.role.name
EBX_DIRECTORY_SYNCHRONIZED_SSO_MAPPING_OIDC_CLAIMS	ebx.directory.synchronized.sso.mapping.oidc.claims
EBX_DIRECTORY_SYNCHRONIZED_SSO_MAPPING_BEARER_CLAIMS_ROLE_NAME	ebx.directory.synchronized.sso.mapping.bearer.claims.role.name
EBX_DIRECTORY_SYNCHRONIZED_SSO_MAPPING_BEARER_CLAIMS	ebx.directory.synchronized.sso.mapping.bearer.claims
EBX_DIRECTORY_SYNCHRONIZED_SSO_MAPPING_ATTRIBUTE.MAIL	ebx.directory.synchronized.sso.mapping.attribute.mail
EBX_DIRECTORY_SYNCHRONIZED_SSO_DISPLAY_USER	ebx.directory.synchronized.sso.display.user
EBX_DIRECTORY_SYNCHRONIZED_SSO_LISTENERS	ebx.directory.synchronized.sso.listeners

LDAP connectivity

The EBX® integration to an existing LDAP directory can be configured through the following environment variables:

Name	TIBCO EBX® main configuration file equivalent (spaces must be removed)
EBX_LDAP_DEFAULT_HOSTNAME	ebx.directory.ldap.default.hostName
EBX_LDAP_DEFAULT_PORT	ebx.directory.ldap.default.port
EBX_LDAP_DEFAULT_CONNECTION_TIMEOUT	ebx.directory.ldap.default.connectionTimeOutInSeconds
EBX_LDAP_DEFAULT_BIND_DN_OR_USER	ebx.directory.ldap.default.bindDnOrUser
EBX_LDAP_DEFAULT_BIND_PASSWORD	ebx.directory.ldap.default.bindPassword
EBX_LDAP_DEFAULT_AUTHENTICATION_METHOD	ebx.directory.ldap.default.authenticationMethod
EBX_LDAP_DEFAULT_ENCRYPTION_METHOD	ebx.directory.ldap.default. encryptionMethod
EBX_LDAP_DEFAULT_MAPPING_ROLE_REFERENCE_ATTRIBUTE	ebx.directory.ldap.default.mapping.role.referenceAttribute
EBX_LDAP_DEFAULT_MAPPING_ROLE_MEMBER_ATTRIBUTE	ebx.directory.ldap.default.mapping.role.memberAttribute
EBX_LDAP_DEFAULT_MAPPING_USER_REFERENCE_ATTRIBUTE	ebx.directory.ldap.default.mapping.user.referenceAttribute
EBX_LDAP_DEFAULT_MAPPING_USER_REFERENCE_COMPONENT_DN	ebx.directory.ldap.default.mapping.user.referenceComponentInDN
EBX_LDAP_DEFAULT_MAPPING_ROLE_MAIL_ATTRIBUTE	ebx.directory.ldap.default.mapping.role.mailAttribute
EBX_LDAP_DEFAULT_MAPPING_USER_MAIL_ATTRIBUTE	ebx.directory.ldap.default.mapping.user.mailAttribute
EBX_LDAP_DEFAULT_DISPLAY_ROLE	ebx.directory.ldap.default.display.role
EBX_LDAP_DEFAULT_DISPLAY_USER	ebx.directory.ldap.default.display.user
EBX_LDAP_DEFAULT_DISPLAY_USER_ALL	ebx.directory.ldap.default.display.user.all

Each request from USER_SEARCH, USER_GROUPS_SEARCH, ALL_USERS_SEARCH, ALL_GROUPS_SEARCH, can be configured through the following environment variables:

For example, the USER_SEARCH request would be:

Name	TIBCO EBX® main configuration file equivalent (spaces must be removed)
EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_BASEDN	ebx.directory.ldap.default.request.userSearch.baseDN
EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_FILTER	ebx.directory.ldap.default.request.userSearch.filter
EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_SCOPE	ebx.directory.ldap.default.request.userSearch.scope
EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_CACHE_EXPIRATION	ebx.directory.ldap.default.request.userSearch.cache.expirationInSeconds
EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_CACHE_REFRESH	ebx.directory.ldap.default.request.userSearch.cache.refreshInSeconds
EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_CACHE_MAX_SIZE	ebx.directory.ldap.default.request.userSearch.cache.maxSize
EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_PAGE_SIZE	ebx.directory.ldap.default.request.userSearch.pageSize
EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_TIME_LIMIT	ebx.directory.ldap.default.request.userSearch.timeLimitInSeconds

See *Built-in LDAP directory* for more information.

Example

To start an instance with LDAP connectivity, execute the following command:

```
docker run -d -p 8080:8080 \
-e "EBX_DIRECTORY_FACTORY=com.orchestranetworks.service.directory.ldap.LdapDirectoryFactory" \
-e "EBX_LDAP_DEFAULT_HOSTNAME=localhost" \
-e "EBX_LDAP_DEFAULT_PORT=389" \
-e "EBX_LDAP_DEFAULT_MAPPING_ROLE_REFERENCE_ATTRIBUTE=cn" \
-e "EBX_LDAP_DEFAULT_MAPPING_ROLE_MEMBER_ATTRIBUTE=member" \
-e "EBX_LDAP_DEFAULT_MAPPING_USER_REFERENCE_ATTRIBUTE=uid" \
-e "EBX_LDAP_DEFAULT_DISPLAY_ROLE={{cn}}" \
-e "EBX_LDAP_DEFAULT_DISPLAY_USER={{cn}} ({{mail}})" \
-e "EBX_LDAP_DEFAULT_DISPLAY_USER_ALL=true" \
-e "EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_BASEDN=dc=example,dc=org" \
-e "EBX_LDAP_DEFAULT_REQUEST_USER_SEARCH_FILTER=(uid={{login}})" \
-e "EBX_LDAP_DEFAULT_REQUEST_USER_GROUPS_SEARCH_BASEDN=dc=example,dc=org" \
-e "EBX_LDAP_DEFAULT_REQUEST_USER_GROUPS_SEARCH_FILTER=(amp(objectClass=groupOfNames)(member={{dn}}))" \
-e "EBX_LDAP_DEFAULT_REQUEST_ALL_GROUPS_SEARCH_BASEDN=dc=example,dc=org" \
-e "EBX_LDAP_DEFAULT_REQUEST_ALL_GROUPS_SEARCH_FILTER=(objectClass=groupOfNames)" \
-e "EBX_LDAP_DEFAULT_REQUEST_ALL_USERS_SEARCH_BASEDN=dc=example,dc=org" \
-e "EBX_LDAP_DEFAULT_REQUEST_ALL_USERS_SEARCH_FILTER=(objectClass=inetOrgPerson)" \
ebx:6.2.3
```

Note

To enable LDAP directory, **EBX_DIRECTORY_FACTORY** must be defined to **com.orchestranetworks.service.directory.ldap.LdapDirectoryFactory**.

SSO SAML 2.0 connectivity

The EBX® Single Sign-On (SSO) SAML 2.0 built-in can be configured through the following environment variables:

Name	TIBCO EBX® main configuration file equivalent
EBX_SAML2_DEFAULT_SP_BASE_URL	ebx.directory.saml2.default.sp.base.url
EBX_SAML2_DEFAULT_SP_ENTITY_ID	ebx.directory.saml2.default.sp.entity.id
EBX_SAML2_DEFAULT_KEYSTORE_URI	ebx.directory.saml2.default.keystore.uri
EBX_SAML2_DEFAULT_KEYSTORE_TYPE	ebx.directory.saml2.default.keystore.type
EBX_SAML2_DEFAULT_KEYSTORE_PATH	ebx.directory.saml2.default.keystore.path
EBX_SAML2_DEFAULT_KEYSTORE_PASSWORD	ebx.directory.saml2.default.keystore.password
EBX_SAML2_DEFAULT_KEYSTORE_KEY_ALIAS	ebx.directory.saml2.default.keystore.key.alias
EBX_SAML2_DEFAULT_KEYSTORE_KEY_PASSWORD	ebx.directory.saml2.default.keystore.key.password
EBX_SAML2_DEFAULT_NAME_ID_ATTRIBUTE	ebx.directory.saml2.default.name.id.attribute
EBX_SAML2_DEFAULT_NAME_ID_FORMAT	ebx.directory.saml2.default.name.id.format
EBX_SAML2_DEFAULT_AUTHN_REQUESTS_SIGNED	ebx.directory.saml2.default.authn.requests.signed
EBX_SAML2_DEFAULT_WANT_ASSERTIONS_SIGNED	ebx.directory.saml2.default.want.assertions.signed
EBX_SAML2_DEFAULT_WANT_RESPONSES_SIGNED	ebx.directory.saml2.default.want.responses.signed
EBX_SAML2_DEFAULT_SP_LOGOUT_REQUEST_SIGNED	ebx.directory.saml2.default.sp.logout.request.signed

See *SAML 2.0* for more information.

SSO OpenID Connect connectivity

The EBX® Single Sign-On (SSO) OpenID Connect built-in can be configured through the following environment variables:

Name	TIBCO EBX® main configuration file equivalent
EBX_OIDC_DEFAULT_CLIENT_BASE_URL	ebx.directory.oidc.default.client.base.url
EBX_OIDC_DEFAULT_CLIENT_ID	ebx.directory.oidc.default.client.id
EBX_OIDC_DEFAULT_CLIENT_SECRET	ebx.directory.oidc.default.client.secret
EBX_OIDC_DEFAULT_DISCOVERY_URI	ebx.directory.oidc.default.discovery.uri
EBX_OIDC_DEFAULT_CLAIM_USERNAME	ebx.directory.oidc.default.claim.username
EBX_OIDC_DEFAULT_HTTP_CONNECT_TIMEOUT	ebx.directory.oidc.default.http.connect.timeout
EBX_OIDC_DEFAULT_HTTP_READ_TIMEOUT	ebx.directory.oidc.default.http.read.timeout

See *OpenID Connect* for more information.

Bearer Authentication

The EBX® Bearer Authentication can be configured through the following environment variables:

Name	TIBCO EBX® main configuration file equivalent (spaces must be removed)
EBX_BEARER_DEFAULT_ENABLED	ebx.directory.bearer.default.enabled
EBX_BEARER_DEFAULT_JWT_JWKS_URI	ebx.directory.bearer.default.jwt.jwks.uri
EBX_BEARER_DEFAULT_JWT_ISSUERS	ebx.directory.bearer.default.jwt.issuers
EBX_BEARER_DEFAULT_SCOPE_DATASERVICES	ebx.directory.bearer.default.scope.dataservices
EBX_BEARER_DEFAULT_JWT_CLAIM_USERNAME	ebx.directory.bearer.default.jwt.claim.username
EBX_BEARER_DEFAULT_JWT_ENCRYPTION_ENABLED	ebx.directory.bearer.default.jwt.encryption.enabled
EBX_BEARER_DEFAULT_JWT_ENCRYPTION_KEY_URI	ebx.directory.bearer.default.jwt.encryption.key.uri
EBX_BEARER_DEFAULT_JWT_ENCRYPTION_KEYSTORE_TYPE	ebx.directory.bearer.default.jwt.encryption.keystore.type
EBX_BEARER_DEFAULT_JWT_ENCRYPTION_KEYSTORE_PATH	ebx.directory.bearer.default.jwt.encryption.keystore.path
EBX_BEARER_DEFAULT_JWT_ENCRYPTION_KEYSTORE_PASSWORD	ebx.directory.bearer.default.jwt.encryption.keystore.password
EBX_BEARER_DEFAULT_JWT_ENCRYPTION_KEYSTORE_KEY_ALIAS	ebx.directory.bearer.default.jwt.encryption.keystore.key.alias
EBX_BEARER_DEFAULT_JWT_ENCRYPTION_KEYSTORE_KEY_PASSWORD	ebx.directory.bearer.default.jwt.encryption.keystore.key.password
EBX_BEARER_DEFAULT_INTROSPECTION_REQUEST	ebx.directory.bearer.default.introspection.request

See *Bearer Authentication* properties for more information.

SCIM directory

The EBX® SCIM directory can be configured through the following environment variables:

Name	TIBCO EBX® main configuration file equivalent (spaces must be removed)
EBX_SCIM_DEFAULT_REQUEST_USERS_URL	ebx.directory.scim.default.request.users.url
EBX_SCIM_DEFAULT_REQUEST_USERS_HEADERS	ebx.directory.scim.default.request.users.headers
EBX_SCIM_DEFAULT_REQUEST_USERS_PAGE_SIZE	ebx.directory.scim.default.request.users.page.size
EBX_SCIM_DEFAULT_REQUEST_USERS_CACHE_MAX_SIZE	ebx.directory.scim.default.request.users.cache. max.size
EBX_SCIM_DEFAULT_REQUEST_USERS_CACHE_EXPIRATION	ebx.directory.scim.default.request.users.cache. expiration
EBX_SCIM_DEFAULT_REQUEST_GROUPS_URL	ebx.directory.scim.default.request.groups.url
EBX_SCIM_DEFAULT_REQUEST_GROUPS_HEADERS	ebx.directory.scim.default.request.groups.headers
EBX_SCIM_DEFAULT_REQUEST_GROUPS_PAGE_SIZE	ebx.directory.scim.default.request.groups.page.size
EBX_SCIM_DEFAULT_REQUEST_GROUPS_CACHE_MAX_SIZE	ebx.directory.scim.default.request.groups.cache. max.size
EBX_SCIM_DEFAULT_REQUEST_GROUPS_CACHE_EXPIRATION	ebx.directory.scim.default.request.groups.cache. expiration

See *SCIM feature* properties for more information.

Email connectivity

The EBX® Mail service can be configured through the following environment variables :

Name	TIBCO EBX® main configuration file equivalent	Default	Description
EBX_SMTP_HOST	ebx.mail.smtp.host		SMTP server host name.
EBX_SMTP_PORT	ebx.mail.smtp.port		SMTP server port number.
EBX_SMTP_CONNECTION_TIMEOUT	ebx.mail.smtp.connectionTimeout	600000	SMTP socket connection timeout value in milliseconds.
EBX_SMTP_TIMEOUT	ebx.mail.smtp.timeout	600000	SMTP socket read timeout value in milliseconds.
EBX_SMTP_WRITE_TIMEOUT	ebx.mail.smtp.writeTimeout	600000	SMTP socket write timeout value in milliseconds.
EBX_SMTP_LOGIN	ebx.mail.smtp.login		SMTP server login id.
EBX_SMTP_PASSWORD	ebx.mail.smtp.password		SMTP server login password.
EBX_SMTP_SSL_ENABLED	ebx.mail.smtp.ssl.activate	true	Enables SSL. Value can be 'true' or 'false'.
EBX_WORKFLOW_MAIL_SENDER	ebx.manager.workflow.mail.sender		The workflow sender email. If not set, Workflows cannot send notifications.

More information on the used properties can be found in chapter *Activating and configuring SMTP and emails* .

Memory configuration

Environment variables **JAVA_MEMORY_PERCENT** may be used to configure the percentage of the container memory that is assigned to the JVM the runs EBX®. It must be an integer value between 0 and 100.

If not set, a default value is used at startup.

Note

This variable is for advanced usage. Setting it too low or too high may cause runtime issues.

Authentication for REST services

Basic authentication for REST services is not enabled by default.

To enable this feature, set environment variable **EBX_REST_AUTHENTICATION_BASIC** to **true** .

Staging

The environment variable **EBX_STAGING_ACTIVATED** activates, if value is **true** , or deactivates, if value is **false** , the staging feature.

By default, staging is activated.

2.5 Configuration files

Two Java property files are currently used to configure EBX®.

On startup EBX® reads property files in the following order:

- **/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties**
- **/my_custom/conf/ebx-container.properties**

File ebx-default.properties

The file **/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties** sets default EBX® configuration properties for the container.

It should **never** be modified at runtime as this may prevent easily updating EBX® to a next version, instead use **/opt/ebx/conf/ebx-container.properties** .

File ebx-container.properties

The file **/opt/ebx/conf/ebx-container.properties** is by default empty. Any property value specified here will override the value set by **ebx-default.properties** .

This file is useful to change a property at runtime. To change a property at run time, create a new file, for example **/my_custom/conf/ebx-container.properties** , containing the new property values and mount de parent folder from the host to the container:

```
docker run -v /my-custom/conf:/opt/ebx/conf -p 8080:8080 -d ebx:6.2.3
```

For the list of properties supported by EBX® see chapter *TIBCO EBX® main configuration file* .

2.6 Volumes

This image defines the following volumes:

Location	Description
/ebx/data	The EBX® root directory is located in this volume. It contains EBX® indexes and, when H2 embedded database is used, persisted data. It is recommended to use a high performance volume, for example an SSD. Using a shared network volume, for example a NFS mount, is not recommended and may cause performance and stability issues.
/ebx/logs	This volume is used for log files.
/ebx/temp	This volume is used for temporary files.

To start EBX® with all dedicated volumes, execute command:

```
docker run -d -p 8080:8080 \
-v /my_custom/data:/ebx/data \
-v /my_custom/logs:/ebx/logs \
-v /my_custom/temp:/ebx/temp \
ebx:6.2.3
```

Note

The volume **/ebx/data** should be mapped to a persistent volume even when an external database is used. If not, EBX® will have to rebuild its indexes on startup which may considerably increase boot time.

2.7 Linux user and group

The container is started using user **ebx** (uid 1500). User ebx's primary group is **root** (guid 0).

Note

Red Hat® OpenShift® may use another UID than 1500 when starting the container. For details see Red Hat® OpenShift® documentation.

2.8 Host configuration

It may be necessary to configure the Host so that the EBX® Container can reserve the resources required by numerous memory-mapped files.

On a Linux OS:

- Command **ulimit -n** should return a value equal or greater than **512000** .
- Command **sysctl vm.max_map_count** should return a value equal or greater than **262144** .

2.9 Logs access

Logs are sent to the stdout and stderr output streams and can be viewed using the following command:

```
docker logs <container-id>
```

Logs for both EBX® and Tomcat will be displayed.

Log files are also available under folder **/ebx/logs** :

- EBX® logs files are in folder **/ebx/logs/ebx**
- Tomcat logs files are in folder **/ebx/logs/tomcat** .

CHAPTER 3

Customizing the image

The EBX® Container Edition image can be used as a parent image to create a customized image.

This chapter contains the following topics:

1. [User specified in Docker file](#)
2. [Setting default configuration](#)
3. [Adding a custom module](#)
4. [Adding a new locale](#)
5. [Adding a new JDBC driver](#)
6. [Configuring the database in JNDI](#)
7. [Configuring the mail in JNDI](#)
8. [Adding a driver for the Metadata Classifier database](#)

3.1 User specified in Docker file

The EBX® Container Edition image's docker file specifies user **ebx** .

A Docker file extending the image may need to set current user to root and later switch back to **ebx** as in following sample:

```
FROM ebx:6.2.3
USER root
# Do something requiring being root...
USER ebx
```

3.2 Setting default configuration

Setting default EBX® configuration should not be based on **/opt/ebx/conf/ebx-container.properties** as this file may be overridden at runtime.

Instead, proceed as following in the Docker file:

- Rename file **/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties** , for example to **ebx-default-original.properties** .
- Create a new file **/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties** container new property values. This file must define property **ebx.file.previous** set to the original property file new name, for example **ebx-default-original.properties** .

For the list of properties supported by EBX®, see *TIBCO EBX® main configuration file* .

Here is a sample Docker file that set the locale to "en-US"

```
FROM ebx:6.2.3
USER root

RUN mv /opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties \
/opt/ebx/webapps/ebx/WEB-INF/ebx-default-original.properties

RUN echo "ebx.file.previous=ebx-default-original.properties" >> \
/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties
RUN echo "ebx.locales.available=en-US" >> /opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties

USER ebx
```

3.3 Adding a custom module

One can extend EBX® by developing custom modules. An EBX® module is a standard Jakarta EE web application, packaging various resources such as XML Schema documents, Java classes and static resources.

For more information on EBX® module, see *Packaging TIBCO EBX® modules*.

With EBX® Container Edition, it is recommended to deploy modules as "unpacked" (exploded) WARs. This allows a faster startup and avoids unnecessarily increasing container size because Tomcat will not need to unpack WAR files.

The recommended way to add a module to an image is to:

- Copy the WAR to folder **/opt/ebx/webapps**. As stated previously, exploded format is recommended.
- Create an associated **Tomcat context XML file** named **module_war_name.xml** and copy it to folder **/opt/ebx/contexts**.
- Optionally, copy shared JARs to folder **/opt/ebx/lib**.

The **Tomcat context XML file** should have the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<Context docBase="${ebx.container.base}/webapps/module_war_name"/>
```

Using variable **\${ebx.container.base}** is required for correct support of environment variable **EBX_ROOT_PATH**.

For more information on Tomcat contexts see https://tomcat.apache.org/tomcat-10.1-doc/config/context.html#Defining_a_context.

Here is a sample Docker file:

```
FROM ebx:6.2.3
USER root
COPY "/module_name.xml" "/opt/ebx/contexts"
COPY "/module_name" "/opt/ebx/webapps"
USER ebx
```

3.4 Adding a new locale

To add one or more locales, you need the *language packs software* zip file(s). Here is a shell script that creates a Docker file and builds the target image. This script can be modified to suit your needs.

```
#!/bin/bash
#
# Build a docker image tag including language packs for
# a specified version available from the current folder.
#
# Usage: build.sh <lp-version> <from-tag> <build-tag>
#
```

```

set -e

LP_VERSION="$1"
FROM_TAG="$2"
BUILD_TAG="$3"

if (( $# != 3 )); then
    echo "Syntax $0 <lp-version> <from-tag> <build-tag>" 1>&2
    exit 1
fi

EBX_LOCALES=""

if [ -d files ]; then
    rm -rf files
fi

mkdir -p "files/lib"

for zip in TIB_ebx-lp_${LP_VERSION}_addon_*_languagepack*.zip ; do
    locale=$(echo ${zip} | sed -e 's/.*languagepack-\([a-zA-Z]*\).zip/\1/g')
    EBX_LOCALES="${EBX_LOCALES},${locale}"
    echo unzip locale=${locale}
    unzip -oq ${zip}

    if [ -d ${locale} ] ; then
        mkdir -p "files/webapps/ebx-manager/www/${locale}"
        mv "${locale}/" "files/webapps/ebx-manager/www/"
    fi
done

mv *.jar "files/lib"

cat > Dockerfile << EOF
FROM ${FROM_TAG}

COPY "files" "/opt/ebx"

USER root

RUN mv "/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties" \
    "/opt/ebx/webapps/ebx/WEB-INF/ebx-default-original.properties" && \
    echo "ebx.file.previous=ebx-default-original.properties" >> \
    "/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties" && \
    echo "ebx.locales.available=en-US,fr-FR${EBX_LOCALES}" >> \
    "/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties"

USER ebx
EOF

docker build -t ${BUILD_TAG} .

echo "To run a container using this image, use command:"
echo " EBX_LOCALES=${EBX_LOCALES}"
echo " docker run -p 8080:8080 -d ${BUILD_TAG}"

```

In this example, libs and optional documentation are copied, then locales are set using the method described in the [setting default configuration](#) [p 29] section.

3.5 Adding a new JDBC driver

Adding a new JDBC driver is similar to adding a new library. You simply have to copy the jar file in the "/opt/ebx/lib" folder with the correct permission. Here is an example with the Oracle JDBC driver:

```

FROM ebx:6.2.3
USER root

ADD \
    https://repo1.maven.org/maven2/com/oracle/database/jdbc/ojdbc11/23.8.0.25.04/ojdbc11-23.8.0.25.04.jar \
    "/opt/ebx/lib/"

RUN chmod +r "/opt/ebx/lib/ojdbc11-23.8.0.25.04.jar"

USER ebx

```

See *Database drivers* for more information.

3.6 Configuring the database in JNDI

To configure a JNDI data source for EBX®, you'll need to customize your Docker image. The following sample Dockerfile demonstrates the necessary steps to integrate your **ebx.xml** configuration.

```
FROM ebx:6.2.3
USER root

RUN mv "/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties" \
    "/opt/ebx/webapps/ebx/WEB-INF/ebx-default-original.properties" ; \
    mv "/opt/ebx/context/ebx.xml" \
    "/opt/ebx/context/ebx.original" ; \
    echo "ebx.file.previous=ebx-default-original.properties" >> \
    "/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties" ; \
    echo "ebx.persistence.url=" >> /opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties

COPY ". /ebx.xml" "/opt/ebx/context"

USER ebx
```

The **ebx.xml** file manages the web application context and includes essential configuration information for the JNDI data source used by EBX®.

Critical point : each property listed below must be carefully adjusted, added, or removed depending on your database (e.g., Oracle, PostgreSQL, SQL Server, etc.) or your cloud provider.

Important : This sample file must be adapted to your specific environment before being integrated into your deployment image. UTF-8 encoding is required for the **ebx.xml** file.

```
<?xml version="1.0" encoding="UTF-8"?>
<Context docBase="{ebx.container.base}/webapps/ebx" crossContext="true">
  <Resource
    name="jdbc/EBX_REPOSITORY"
    auth="Container"
    type="javax.sql.DataSource"
    factory="org.apache.tomcat.jdbc.pool.DataSourceFactory"
    username="<username>"
    password="<password>"
    driverClassName="<driverClassName>"
    url="<url>"
    maxTotal="100"
    minIdle="10"
    maxIdle="25"
    initialSize="10"
    validationQuery="<validationQuery>"
    rollbackOnReturn="false"
    enableAutoCommitOnReturn="false"
  />
</Context>
```

Note

By configuring the JNDI data source, you're instructing EBX® to use that JDBC connection method. Consequently, EBX® will no longer consider environment variables beginning with 'EBX_DB_'.

3.7 Configuring the mail in JNDI

To configure a JNDI Mail session for EBX®, you'll need to customize your Docker image. The following sample Dockerfile demonstrates the necessary steps to integrate your **ebx.xml** configuration.

```
FROM ebx:6.2.3
USER root

RUN mv "/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties" \
    "/opt/ebx/webapps/ebx/WEB-INF/ebx-default-original.properties" ; \
    mv "/opt/ebx/context/ebx.xml" \
    "/opt/ebx/context/ebx.original" ; \
    echo "ebx.file.previous=ebx-default-original.properties" >> \
```

```

"/opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties" ; \
echo "ebx.mail.activate=true" >> /opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties ; \
echo "ebx.mail.smtp.host=" >> /opt/ebx/webapps/ebx/WEB-INF/ebx-default.properties

COPY "./ebx.xml" "/opt/ebx/contexts"

USER ebx

```

The **ebx.xml** file manages the web application context and includes essential configuration information for the JNDI mail session used by EBX®.

Critical point : each property listed below must be carefully adjusted, added, or removed depending on your mail manager or your cloud provider.

Important : This sample file must be adapted to your specific environment before being integrated into your deployment image. UTF-8 encoding is required for the **ebx.xml** file.

```

<?xml version="1.0" encoding="UTF-8"?>
<Context docBase="{ebx.container.base}/webapps/ebx" crossContext="true">
  <Resource
    name="mail/EBX_MAIL_SESSION"
    auth="Container"
    type="jakarta.mail.Session"
    factory="org.apache.naming.factory.MailSessionFactory"
    mail.debug="false"
    mail.smtp.host="<host>"
    mail.smtp.port="<port>"
    mail.smtp.auth="true"
    mail.smtp.user="<user>"
    password="<password>"
    mail.smtp.ssl.enable="true"
    mail.smtp.starttls.enable="false"
    mail.smtp.connectiontimeout="600000"
    mail.smtp.timeout="600000"
    mail.smtp.writetimeout="600000"
    mail.transport.protocol="smtp"
    description="Mail session for Tomcat 10.x resource" />
  />
</Context>

```

Note

By configuring the JNDI mail session, you're instructing EBX® to use that mail connection method. Consequently, EBX® will no longer consider environment variables beginning with 'EBX_SMTP_'.

3.8 Adding a driver for the Metadata Classifier database

Adding a driver for the Metadata Classifier database requires installing the corresponding Python module. The Python module must be installed within the Python virtual environment. The following examples show how to customize the Docker image to add the driver.

For **PostgreSQL database** :

```

FROM ebx:6.2.3

RUN pipenv install psycopg2-binary

```

For **Oracle database** :

```

FROM ebx:6.2.3

RUN pipenv install oracledb

```

For **Microsoft SQL Server database** :

```

FROM ebx:6.2.3

USER root

# Install Microsoft SQL ODBC driver
RUN UBUNTU_VERSION=$(grep VERSION_ID /etc/os-release | cut -d '"' -f 2) && \
echo "Detected Ubuntu version: $UBUNTU_VERSION" && \
curl -sSL -O https://packages.microsoft.com/config/ubuntu/$UBUNTU_VERSION/packages-microsoft-prod.deb && \
dpkg -i packages-microsoft-prod.deb && \

```

```
rm packages-microsoft-prod.deb && \  
apt-get update && \  
ACCEPT_EULA=Y apt-get install -y msodbcsql18 && \  
apt-get clean && rm -rf /var/lib/apt/lists/*  
  
USER ebx  
RUN pipenv install pyodbc
```

See [SQLAlchemy dialects](#) for more information about the database drivers that can be used with SQLAlchemy.

CHAPTER 4

Documentation and Support

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

This chapter contains the following topics:

1. [How to Access TIBCO Documentation](#)
2. [Product-Specific Documentation](#)
3. [How to Contact Support for TIBCO Products](#)
4. [How to Join TIBCO Community](#)

4.1 How to Access TIBCO Documentation

Documentation for TIBCO products is available on the [Product Documentation website](#) website, mainly in HTML and PDF formats.

The [Product Documentation website](#) website is updated frequently and is more current than any other documentation included with the product.

4.2 Product-Specific Documentation

The documentation for the TIBCO EBX® is available on the [TIBCO EBX® Product Documentation](#) page. This page contains the latest version of each document.

The documentation for the TIBCO EBX® Add-ons is available on the [TIBCO EBX® Add-ons Product Documentation](#) page. This page contains the latest version of each document.

To view the documents for Add-on Bundles that are compatible with other versions of TIBCO EBX®, use the **Bundle version** menu to select the desired release.

4.3 How to Contact Support for TIBCO Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our [product Support website](#).
- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the [product Support website](#). If you do not have a username, you can request one by clicking **Register** on the website.

- To submit ideas, suggestions, or feedback for TIBCO products, visit our [product Support website](#).

4.4 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](#).

CHAPTER 5

Legal and Third-Party

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, TIBCO EBX®, TIBCO EBX® Data Exchange Add-on, TIBCO EBX® Add-on's Root Module, TIBCO EBX® Digital Asset Manager Add-on, TIBCO EBX® Match and Merge Add-on, TIBCO EBX® Data Model and Data Visualization Add-on, TIBCO EBX® Insight Add-on, TIBCO EBX® Graph View Add-on, TIBCO EBX® Add-on for Oracle Hyperion EPM, TIBCO EBX® Information Governance Add-on, TIBCO EBX® GO Add-on, TIBCO EBX® Activity Monitoring Add-on, TIBCO EBX® Rule Portfolio Add-on, and TIBCO EBX® Information Search Add-on 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© 2006-2026. Cloud Software Group, Inc. All Rights Reserved